

EMBER
INFORMÁCIÓ
RENDSZER

Avagy mit kell tudni az információs rendszerekről?

HALASSY BÉLA

Budapest, 1996.

A szerkesztést a szerző végezte.

Az ábrák és a fedőlap megrajzolásában közreműködött *Sajben Gábor*.

© Dr. Halassy Béla, 1996.

Mottó: „Édes fiam, a fejedben legyen világosság...!”

(Soma tanár úr, piarista nevelő)

Barátaimnak, tanítványaimnak és kollégáimnak

TARTALOMJEGYZÉK

AJÁNLÁS.....	5
HÁTTÉR	5
SZÁNDÉK.....	7
SZELLEM	8
ÖSSZEFOGLALÓ.....	9
TARTALOM.....	9
FORMA.....	9
1. RENDSZER ÉS RENDSZERSZEMLELET	11
1.1 A RENDSZEREKRŐL ÁLTALÁBAN	11
1.2 A RENDSZER HATÁRAI ÉS KÖRNYEZETE	15
1.3 A RENDSZER TAGOLÁSA	17
1.4 KETTŐS RENDSZERNÉZET	21
1.5 TÉVES SZEMLELETEK.....	25
1.6 AZ EGYETLEN HELYES SZEMLELET	28
1.7 RENDSZER ÖSSZEFOGLALÓ	31
ELLENŐRZŐ KÉRDÉSEK - 1.....	33
2. AZ INFORMÁCIÓS RENDSZER FOGALMA	35
2.1 ADAT ÉS INFORMÁCIÓ	37
2.2 TEVÉKENYSÉG ÉS ESEMÉNY	44
2.3 AZ INFORMÁCIÓS ERŐFORRÁSOKRÓL.....	51
2.4 A FELHASZNÁLÓKRÓL.....	56
2.5 A SZABVÁNYOKRÓL	60
2.6 IR ÖSSZEFOGLALÓ.....	64
ELLENŐRZŐ KÉRDÉSEK - 2	66
3. A RENDSZER SZEMLELETE ÉS FEJLESZTÉSE	68
3.1 RENDSZERANALITIKA	68
3.1.1 A rendszer vizsgálati szempontjai.....	68
3.1.2 A rendszerstruktúrálás elvei	73
3.1.3 Az elvek harmonizálása.....	77
3.1.4 Az IR vetületei	79
3.1.5 Az IR szintjei.....	83
3.2 RENDSZERFEJLESZTÉS	89
3.2.1 A fejlesztésről általában	89
3.2.2 Háromszoros terv	90
3.2.3 Tervezés és elemzés, dokumentáció és specifikáció	93
3.2.4 Tervezés, szervezés, modellezés.....	97
3.3 SZEMLELETI ÖSSZEFOGLALÓ	100
ELLENŐRZŐ KÉRDÉSEK - 3	101
4. FEJLESZTÉSI MÓDSZER.....	103
4.1 SZEMLELETI ZAVAROK	103
4.2 ALAPKONCEPCIÓ	105
4.3 STRUKTÚRA ÉS TERMINOLÓGIA.....	108
4.4 MEGFOGALMAZÁSI MÓD	111
4.5 KRITÉRIUMRENDSZER	115
4.5.1 Valóságghűség	116
4.5.2 Érthetőség	117

4.5.3 Egyértelműség.....	117
4.5.4 Minimalitás	118
4.5.5 Teljesség.....	119
4.5.6 Harmónia	120
4.6 TERVEZÉSI ALGORITMUS	121
4.7 DOKUMENTÁCIÓ-KEZELÉS	124
4.8 TECHNIKAI TÁMOGATÁS	126
4.9 METASZABVÁNYOK	127
4.10 MÓDSZERTANI ÖSSZEFOGLALÓ	128
ELLENŐRZŐ KÉRDÉSEK - 4	130
5. AZ IR ADATRENDSZERE	132
5.1 AZ ISMERET DIMENZIÓI	132
5.2 AZ EGYED ÉS A TULAJDONSÁG FOGALMA	135
5.3 A KAPCSOLATTÍPUS FOGALMA	139
5.4 ADATBÁZIS, ADATMODELL, ALMODELL	144
5.5 ISMERETKEZELÉSI MÓDOK.....	149
5.6 ADATRENDSZER ÖSSZEFOGLALÓ	155
ELLENŐRZŐ KÉRDÉSEK - 5	157
6. AZ IR FELDOLGOZÁSRENDSZERE	159
6.1 AZ ISMERET ASPEKTUSAI.....	159
6.2 BEMENET, KIMENET, ÁTMENET.....	163
6.3 ADATBÁZIS ÉS FELDOLGOZÁS	167
6.4 AZ SQL ÉS A NAVIGÁLÁS KONCEPCIÓJA.....	173
6.5 FELDOLGOZÁSTERVEZÉS.....	177
6.6 MAKRO- ÉS MIKROTERVEZÉS.....	183
6.6.1 Makrotervezés	183
6.6.2 Mikrotervezés.....	186
6.6.3 Feldolgozás-konzisztencia	189
6.7 FELDOLGOZÁS ÖSSZEFOGLALÓ.....	190
ELLENŐRZŐ KÉRDÉSEK - 6	192
7. AZ IR KÖRNYEZETRENDSZERE	194
7.1 AZ EMBERI KÖRNYEZET.....	194
7.1.1 Pszichológiai mozzanatok	194
7.1.2 Munkaszervezési mozzanatok	196
7.1.3 A rugalmasság és a szilárdság mozzanatai.....	198
7.2 A SZABVÁNYRENDSZER	200
7.2.1 A szabványokról általában	200
7.2.2 Milyen szabványokra van szükség?	201
7.3 RENDSZERDOKUMENTÁCIÓ.....	204
7.4 KÖRNYEZETI ÖSSZEFOGLALÓ.....	206
FELADATMEGOLDÁSOK.....	207
MEGOLDÁSOK - 1.....	207
MEGOLDÁSOK - 2	208
MEGOLDÁSOK - 3	209
MEGOLDÁSOK - 4	211
MEGOLDÁSOK - 5.....	212
MEGOLDÁSOK - 6	213
FOGALOMJEGYZÉK.....	216

AJÁNLÁS

A komoly szakmai művet általában vagy bevezetéssel, vagy előszóval szokták kezdeni. A magára egy csöppceskét is adó szerző pedig egyiket sem felejtí el a könyve elejéről. Mi most egy picit másként fogunk eljárni, aminek két oka van.

Az egyik az, hogy a műnek a célját, szerkezetét, tartalmát, formáját etc. *kvázi-objektíven* megvilágítani kívánó - a fejezetszámozásba amúgy sehogyan sem illő - bevezetések sohasem sikerülnek igazán jól. Elvégre lehetetlen jól összefoglalni a könyvben lévő tengernyi gondolatot. Ráadásul szárazon, ridegen, mintha pont a szerzőnek nem lenne semmi köze munkája lelkületéhez.

A másik az, hogy a szerkesztők mindig roppantul kevés teret engedélyeznek a szerintük kizárólag csak a szerző úgymond lelkivilágának a kitergetésére való, a *kvázi-szubjektíven* megfogalmazott gondolatok átadására alkalmas - amúgy a fejezetszámozásba sehogyan sem illő - előszónak. Egy szakmai munkában ne legyen érzés, a regények pedig úgymond magukért beszélnek.

Ám ez alkalommal kivételes szerencsének örvendhetünk: a szerkesztő (H. B.) speciális engedélyt tett a szerzőnek (H. B.). Eltúrte, hogy előszó és bevezetés helyett objektív és szubjektív alapgondolatait egyetlen passzusban foglalja össze még akkor is, ha az a szokásosnál egy picit terjedelmesebb. Végül is nem döntött rosszul, mert *a mű alapvető tárgya az ismeret*, ez a furcsa és érthetetlen testből és lélekből álló jelenség, amely kvázi-objektíven és kvázi-szubjektíven kötődik a testből és lélekből álló emberhez.

Bocsássa meg tehát az olvasó a szokatlan kezdést annál is inkább, mert maga a könyv sem a szokott módon közelíti meg a témáját. Fogadja tehát az ajánlásnak szánt alábbi sorokat ugyanazzal a derűs lélekkel és türelemmel, mint amivel az készült. Túrje el, hogy a szerző azonnal rátér a személyes hangnemre, amelytől ugyan a későbbiekben tartózkodni próbál, ám - ki lát egy szerző lelkébe? - nem biztos, hogy mindig sikerrel.

HÁTTÉR

Barátaim, kollégáim, tanítványaim - ezek nem kizáró kategóriák - igen régóta nyaggattak azzal, hogy miért nem írok már végre egy könyvet az információs rendszerekről úgy általában is. (Értsd: Egyesek ilyen módon óhajtották közölni velem, hogy unják már az egyéb dolgaimat.) Nem is sejtették, hogy már régóta élt bennem egy ilyesfajta vágy. Ám húztam-halasztottam a dolgot, amit senki sem kényszerített, kényszeríthetett rám. Kétvalakit kivéve...

Szóval hivatkoztam én mindenre, főleg az időhiányra - minden idők minden normális informatikusának e legnagyobb menedékére. Pedig az igazság sokkal egyszerűbb: gyáva voltam. Az információs rendszer félelmetes ellenség. Ravasz, álnok, becsapós valaki. Igenis valaki, mert lelke van. Még egy emberét is nehéz kiismerni, akkor miképpen lehet felvállalni a közös szellem magyarázatát?

„Ha valamitől félsz, azt tedd meg!” - cseng a fülemben tanítóim tanácsa. Nem tehettem mást: végül is kezembe kellett vennem a számítógépet. Kellett. Miért?

Egyrészt látom barátaim tehetetlenségét. Ők nem szakemberek. Csak haladni akarnak a korról, tehát számítógépet vesznek és becsülettel megtanulják annak kezelését. A tipikus séma szerint a kezdeti lelkesedést kis idő után csüggedés követi. Nem értik, hogy a dolgok miért nem alakulnak a kedvükre. Nem látják ugyanis a feladat, a megoldás és az eszköz harmóniáját. Minél többet költenek az utóbbira, annál kevesebbet törődnek a másik kettővel. Nem értik, hogy

nem a technika teszi az informatikát!

Az informatika az ismeretek megismerésének, elrendezésének és kezelésének a tudománya. Ehhez ma már nélkülözhetetlen eszköz a számítógép, de aki ért az utóbbihoz, az még távol áll attól, hogy informatikus legyen.

Másrészt tapasztalom kollégáim értetlenségét. Csak nem akarják felfogni azt, hogy „az ismeret a szellemi élet kenyere” - ahogyan mondogatni szoktam. Ők mindig napi feladatokban, napi kötöttségekben gondolkodnak. Eszükbe sem jut, hogy

a számítástechnika szakma, az informatika pedig hivatás!

Miért, mi a különbség? A diák nem tudja, hogy *miért* tanul - ám a valódi tanár igen, hogy miért tanít. A beteg csak érzi a kórt - a jó orvos ismeri annak az okát és a gyógy módját. A hívó ösztönösen hisz - a pap látja az okokat is. A hivatást az jellemzi, hogy a profi tudás és tudatosság mellé szándék is párosul. Emberre irányuló szándék. A profi számítástechnikus jól akar megírni egy programot. A hivatásos informatikus jó ismeretekkel akarja szolgálni az embereket. És itt van az alapvető különbség: a szakember célja saját maga: kapni akar, az elkötelezett vágya más: adni szeretne.

Harmadrészt itt vannak a tanítványok, a fiatalok. Utálom a „bezzeg a mai...” kezdetű szövegeket. Azt sem hiszem, hogy el kellene marasztalni őket azért, mert kedvelik a zenét, a tévét, a kólát és olykor még a másik nemet is. Az élet szeretete és megélése egyáltalán nem zárja ki a komoly szakmai törődést. Csak egy bajom van velük: nem látják a dolgok árát, ami viszont a korukkal jár. Mire is gondolok?

Az információ az ember és az ismeret viszonya. Ezért annak, aki informatikai hivatással rendelkezik és ezt a pályát választja, nemcsak a számítógépeket kell megértenie, hanem az embert és az ismeretet is. Attól még nem lesz senki sem informatikus, hogy képes a jó eszközt technikailag jól használni. Ehhez képest megdöbbenő, hogy

a fiatalok tanulás nélkül akarnak az informatika mestereivé válni.

Az öntelt mosolyokat le kell törölni. A szemeket és füleket ki kell nyitni. A jó informatikus a technika ismerete által még jobbá válhat. Ám a silányból a világ legragyogóbb számítógépe vagy szoftvere sem lesz képes mestert faragni.

Akinek egyetlen célja a pénz, aki a széles és egyenes utat akarja választani a keskeny és kanyargós helyett, az jobban teszi, ha itt abbahagyja az olvasást. Én ugyanis nem ígérhetek 'informatikai szupersztrádát'. Csak verejtékes munkát.

Gondolatban végignézek a magyar és a nemzetközi 'könyvespolcok' szakmai traktusain. Látok drága és holnap a verzióváltás miatt kidobható kézikönyveket. Az X bizgentyű lelkivilágát - olykor tévesen - kifejtő, a lényegét soha meg nem világító eszközleírásokat. Az elemi iskoláktól kezdve az egyetemekig bezárólag olyan tankönyveket, amelyekből csak éppen az alapok hiányoznak. Tantervet, amelyben az információ lelkülete nem kap helyet. Kapcsold be a gépet...! - ez ma a jelszó, a sarokkő.

Honnan, miből ismerje fel a barát, hogy rossz nyomon jár? Ki világosítja fel a profit, akit 'lefelé' - a gép irányában - továbbképeznek, 'felfelé' - az ember felé - viszont nem? És miért korholom a tanítványt, ha nincs is miből tanulnia?

Nos, ezért kellett 'elkövetnem' ezt a könyvet.

SZÁNDÉK

Az említett hiányt látva néhány ismerősöm arra ösztönzött, hogy kézikönyvet, fogalomtárat, jegyzetet, oktatási anyagot, vagy nem is tudom mit írjak. Szigorút, formálisat, meggyőzőt és bebiztosítót. 'Egy definíció - két osztályozás' alapon. Fiatalnak és öregnek, amatőrnek és profinak. 'Tudományosan', de érthetően.

Nem tehetek róla, hogy egy tucat könyvet nem tudok egyetlen műben megírni. A sok 'jó-tanács' persze újabb mentséget adhatott volna a feladat elnapolására. Ha így, ha úgy... Saját elképzeléseim ezzel a könyvvel a következők:

Ad 1) **Tisztázás.** Manapság annyi 'informatikai' fogalom kavarg a fejekben, hogy az döbbenetes. Nem elég, hogy az informatika amúgy is nagyon összetett dolog. Ma az 'informatikai fogyasztói társadalom' korszakát éljük. Ez egyrészt azzal jár, hogy a dolgokat kódósítani kell. A minap történt, hogy egy fiatal már tizedszer rágta a számba, hogy ők 'objektum-orientáltak' dolgoznak. Elkövettem azt a gyalázatot, hogy megkérdeztem tőle, mi az az 'objektum'? Hát ő azt nem is tudja, de hát most ez a módi, meg azután úgyhogy, ugye... Másrészt az ínyeket csiklandozni kell a legújabb 'informatikai' ingyencségekkal. Ezért a réges-régi és ismert dolgoknak (pl. számítógép-kapcsolatok) új neveket találnak ki (pl. client-server), és ha van egy gyenge szoftverecskénk, arra ráragasztjuk az új címkét.

Engem nem érdekel, hogy mit minek neveznek: lássuk a medvét! Az nem baj, ha valaki más - 'modern' - szavakat használ. Csak a tartalomban egyezzünk meg. És - vessenek meg ezért - szerintem a tartalom az előbbre való. Tehát először a fogalmakat kell tisztáznunk. Például sokáig azt hittük, hogy a demokrácia az 'népi'; ma azt gondoljuk, hogy 'nyugati'; közben pedig a demokrácia teljesen más... Tegnap azt mondták nekünk, hogy az adatbázis az ...; ma már egészen mást dúdolnak a fülünkbe; holott az adatbázis történetesen egyik sem...

Ad 2) **Elrendezés.** Az emberek döntő többsége egy adott szemszögből nézi az informatika dolgait: a feladatának, funkciójának és igényének megfelelő módon. Ma így, holnap úgy. Szerepkörtől függően. A saját vállalatában mások számára tökéletes rendszert készítő számítástechnikus van a legjobban felháborodva, ha egy másik cégnél őt ügyfélként nem jól szolgálják ki.

A világ telis-tele van számítástechnikai skizofrénekkel. Ennek pedig a legfőbb oka az, hogy a szemléleteket - világméretű hálózat ide vagy oda - mindig is az **individualizmus** uralja. Az én adatbázisom, az én programom, az én gépem, az én információigényem. A **közösség** tudata elveszőben van. A 'mi' éppen úgy nem számít, mint a 'jövő'. A közös élet helyett az egyéni túlélés lett a jelszó. Számít az, hogy egy ország milliárdokat dob ki egy rossz és kényelmetlen adókárttyára? A lényeg az, hogy valakik ebből pénzt csináljanak.

A valódi információs rendszerben minden mindennel összefügg: az ismeret, annak feldolgozása, az erőforrás, az ember, a törvény. Mikro- és makroszinten egyaránt jellemző, hogy az emberek nem látják, vagy éppen nem akarják látni ezen tényezők általános elvi elrendezését. Ezért azután aligha csoda, hogy egy-egy konkrét feladatnál csődöt mondanak.

Ad 3) **Nevelés.** A minap megkérdeztem egy programozót, hogy kinek és miért köszönhetjük azt, hogy a számítógépet programozni lehet. Hanyagul vállalt vont: „Kit érdekel? A munkám a lényeg.” Neumann Jánosról az életében nem hallott. Nyeglesége nem háborított fel: nem a programozás atyját, nem engem sért vele. Magának fog ártani előbb-utóbb, mert a munkája velejét nem ismeri.

Bevallom, hogy néha ragaszkodom az 'ósvi' megoldásaimhoz. Nem rohantom például, hogy megvegyem a Win '95-öt. A fiatalok olykor jól le is tolnak, hogy „Béla(bácsi), még azt sem tudod...”. Az viszont érdekes, hogy amikor kikapad a gép vagy egy valóban informatikai célú megoldás nem válik be, akkor rögtön a „Béla(bácsi), ugye segítesz...” névre hallgatok. Azért, mert az új seprő jól seprő, de a régi tudja, hogy hol van a piszok.

A fiatalok jó része észre sem veszi, hogy - rab. Marionett báb. Nem is annyira láthatatlan madzagokon ide-oda rángatott valaki. Aki még ráadásul azt is hiszi, hogy úgy jó neki, ha minden eszközforgalmazó helyette gondolkodik.

Téved, aki azt véli, hogy a korszerű technika ellen papolok. Nemigen hiszem, hogy nálam bárki jobban szeretné a számítógépet. Több, mint negyedszázada az informatika a hivatásom, a számítástechnika a szakmám. Akik ismernek, tudják, hogy ami kevés sikereset alkottam ennyi idő alatt, az nem a technikának, hanem a szemléletnek volt köszönhető.

A gépkezelésre ki lehet bárkit képezni. A számítástechnika oktatható. Viszont a szemlélet nem tanítható. A nevelés - bármennyire is célomul tűztem ki ebben a könyvben - nem egy ember és egy mű feladata. A szülő, a tanár, a társadalom - a példa - a siker kulcsa. És jobb, ha ezt az elmélkedést most gyorsan abbahagyom. Mert mit lát maga körül a leendő informatikus?

SZELLEM

Az embereket sokszor megtéveszti az abszolút és a relatív igazságról hirdetett hazug maszlag. Miként? Ha az igazság relatív, akkor nyilván az én igazságom a fontos és nem a tiéd, nem a társadalomé. Következésképpen nekem csak akkor szükséges mások igazságára is figyel-nem, ha a saját érdekem azt diktálja. A mai 'informatikát' is pontosan ez az önáltató szemlélet jellemzi. Jó az, ami nekem jó.

A számítástechnika állandóan változik. Ezért kétségtelen, hogy vannak időben és térben relatív igazságai, amelyek azonban egyáltalán nem törvényszerűek. Az informatika törvényei viszont abszolútak. Félreértés ne essék: ez egyáltalán nem azt jelenti, hogy nem bővülnek a megismerés során. Az egyszeregy még ma is az annak dacára, hogy az euklideszit felváltotta a görbült tér geometriája, amitől az előbbi egyáltalán nem veszítette el érvényét.

Az olvasóra bízom a relatív és az abszolút megítélését, a ma és a holnap közti választást. Azt, hogy neki az eszközben vagy a rendszerben való gondolkodás-e a fontosabb.

Csak egy dolgot kérek: ne tessék görcsölni! A vizsgákon, a napi munkában, az életben általában vannak örök olyan vizsgázók és vizsgáztatók, akik merevek és gondterheltek, állandó kölcsönös félszben vagy bizonyítási vágyban élnek. Az élet szép, az élet jó, azt humorosan - nem viccként - kell felfogni. Komolyan és saját képtelenségeinken nevetve.

Már csak az ajánlás maradt hátra. Köszönöm ezt a könyvet annak a huszonöt kiló pontynak, amit nem foghattam ki, mert nem volt időm elmenni pecázni. A gondos feleségnek, akinek a - nem pontyból készült - ebédje rám várva kihűlt. Két lányomnak és barátaimnak, akik szórakozottságom 'vértanúi'. A megígért, de elmaradt telefonok/levelek címzettjeinek. Sokat segítettek, de...

Ez a könyv valahonnan és valamiért megszületett. Ihletője a pénzecskéjét nem a várt időben megkapó kisnyugdíjas; a bűnözés ellen valóban küzdő zsaru; a piackutató kínlódása; a cégbíróság előtti hosszú sor; a téves számla; az elrontott nyaralás áldozata... Folytassam?

Reményeim amúgy nem rózsásak. Megírtam ezt a könyvet, de a végén úgyis én leszek letolva. Sokak által, de most csak a 'Háttérben' emlegetett két valakire gondolok. Az egyik, mint sejtik, én magam vagyok. A másik Valaki pedig majd így szól hozzám: 'Végül is megírtad? No, és jól? Ugyan édes fiam, ezt nevezed te - könyvnek? No nem baj, majd talán legközelebb...'

Halassy Béla

ÖSSZEFOGLALÓ

TARTALOM

A könyv hét fejezetből áll. Ezek a rendszer általános fogalmából kiindulva az információs rendszer környezeti aspektusáig vezetik el az olvasót. A felépítést megalapozó gondolatmenet a következő:

- Az 'információs rendszer' fogalom két lényegyet kapcsol egymáshoz. Ahhoz, hogy értelmesen beszélhessünk róla, tisztában kell lennünk egyrészt a **rendszer** általános vonásaival, másrészt az **információ** általános természetével, végül eme két dolog specifikus **viszony**ának a mibenlétével. Az első fejezetben a rendszert módszertani kategóriaként vizsgáljuk. Kitérünk a **határ**, a **környezet** továbbá a **rendszertagolás** kérdéseire. A továbbiak előkészítéseképpen külön hangsúlyt fektetünk arra, hogy bemutassuk az információs rendszerek **téves szemléleteit**.

- A második fejezetben az információs rendszerek alapjait különböző tényezők szervezett együtteseként mutatjuk be. Ismertetjük az **adat** illetve az információ; az információs **esemény** és **tevékenység**; az **erőforrás** és a **felhasználó**; végül az információs **szabvány** mibenlétét. A 'szervezett együttes' kifejezés két egymással összefüggő dolgot takar. Az információs rendszer kialakításához az szükséges, hogy megteremtsük az említett tényezőcsoportokon **belüli** összhangot mégpedig úgy, hogy a tényezőcsoportok **közötti** bonyolult viszonyokra is figyelünk.

- Az 'információs rendszer kialakítása' három dolgot feltételez. Tudnunk kell azt, hogy ez a lényeg milyen **aspektusokból** vizsgálható. Az adat, a feldolgozás (ide tartozik az esemény és a tevékenység) illetve a többi tényezőt felölelő ún. környezet az ilyen rendszerek három vizsgálati **vetülete**. Ezek közül az első kettő emberi absztrakció, amit éppen ezért több **szinten** lehet szemlélni. A kialakítás feltételeként tisztában kell lennünk a rendszer **fejlesztésének** a lényegével. Ezt a két feltételt a harmadik fejezetben tárgyaljuk. Megfelelő **módszer** nélkül nincs remény a fejlesztés sikerére, ezért a negyedik fejezetben a fejlesztés módszertani kellékeit foglaljuk össze.

- A előző, a megalapozást célzó részekkel szemben a továbbiakban a mélyebb összefüggésekkel foglalkozunk. Azaz részletesebben is kifejtjük az információs rendszer fentebb említett három - **adat**, **feldolgozás** és **környezet** - vetületének a tényezőit, azok összefüggéseit, valamint tervezési és elemzési szempontjait.

FORMA

A mű fejezetei a **cél** meghatározásával kezdődnek és a lényegi mondanivalót tömörítő, olykor egy kis többletet is adó **összefoglalóval** zárulnak. Pontosabban a szemlélet elsajátítását szolgáló összegzést még **ellenőrző kérdések** is követik. Ezek célja kettős. Egyrészt segítik az olvasót abban, hogy tükröt tartson maga elé. Másrészt olykor a mondanivaló kiegészítésére is szolgálnak. Éppen ezért a feladatok néha nem könnyűek. Remélhetőleg meggondolkodtatják az olvasót. Ez azonban nem jelenthet problémát, mert a **megoldások** a mű végén találhatók.

A könyvben nem kerülhettük el a **meghatározások** és **alapelvek** alkalmazását sem, bár - az alább leírt szellem jegyében - igyekeztünk minél kevesebb formális definícióval élni. Az alapelvek kvázi-axiómákként az információs rendszer egy-egy eredendő törvényét fogalmazzák meg.

Ezeket lehetne ugyan bizonyítani is, de mi inkább az olvasó józan belátására hagyatkozunk. A meghatározások és az alapelvek fejezetenként újra számozottak (D x.y és E z.q) csakúgy, mint a ábrák és a történetek (T v.w). A definíciókban szereplő fogalmakon kívül vannak más fontos tényezők is, amelyekre nem kívántunk formális meghatározást adni. Azt, hogy az író mit tekint 'fontosnak' az olvasó megtudhatja a könyv végén található **fogalomjegyzékből**. Az idegen szakkifejezéseket olykor nem korrektül fordítják magyarra. Ezért a legfontosabb fogalmak nevét első előfordulásukkor angolul is megadtuk például így: elosztott [distributed].

A mondanivaló megvilágítására **történeteket** alkalmaztunk. Ezekről két dolgot kell tudni. Egyrészt azt, hogy mind valós esetek. Másrészt azt, hogy éppen ezért 'esettanulmányként' szolgálnak. Az ember furcsa lény. Egy-egy negatív példából olykor többet tanul, mint több sornyi pozitív magyarázatból.

Az **ábrák** a szerző agyában születtek vagy most, vagy már jóval korábban. A 'Halassy-féle szőrös emberek' - ahogyan a régi tanítványok nevezték e rajzokat - nagy része saját kivitelezésű. Ez meg is látszik a minőségen, bár Sajben Gábor sokat tett javításuk érdekében. Nem volt célunk hivatásos rajzoló megbízása e feladattal, mert az nem illett volna könyvünk szelleméhez.

A szerzőnek már más művével is előfordult, hogy **tankönyvként** használták. Ez végül is örömdetes dolog. Ha netán ennek a könyvnek is ez lenne a sorsa, a szerző nagyon kéri a tanárt és a diákot egyaránt, hogy ne a betűhöz, hanem a szellemhez ragaszkodjanak! Nem befűlázni, nem kikérdezni/elmondani kell azt, amit ez a munka tartalmaz. Beszélgetni kell róla, meg kell érteni a lényegét és a hangulatát. Az információs rendszer nem hétfejű, tüzet okádó, gonosz sárkány. Egyrészt sokkal több feje van. Másrészt tényleg tüzet okád. Harmadrészt viszont egészen szertetre méltó és megszéldíthető kreatúra...

1. RENDSZER ÉS RENDSZERSZEMLÉLET

Amióta ember az ember, azóta jellemző rá, hogy az általa használt szavakon át próbál meg uralkodni a világon. Azáltal veszi birtokba a valóságnak valamilyen részét, hogy azt néven nevezi, vagyis *fogalmat* alkot róla. Az ember társas lény. Ha több ember érintkezni akar egymással, akkor közös *nyelvet* kell beszélniük, vagyis azonos fogalmakban kell gondolkodniuk. Akkor is a fogalom és az azon alapuló nyelv a közlés eszköze, ha az nem a kimondott vagy a leírt szó, hanem - például - a mozdulat vagy a kép formájában testesül meg.

A „hajtsd uralmad alá a Földet” parancsnak vagy kihívásnak - most mindegy, hogy melyikben gondolkodunk - csak lassan-lassan teszünk eleget. Ennek egyik alapvető oka az, hogy szavaink szűkösek és nyelvünk összezavarodott. Például ha azt mondjuk: ‘információs rendszer’, akkor biztosra vehetjük, ez a fogalom 99 ember közül 99-ben teljesen más képzeteket kelt. Azért, mert már magukon az egyes szavakon - ‘információs’ és ‘rendszer’ - is mást-mást értenek. Éppen ez az oka annak, hogy ennyire - minden ellenkező híresztelés dacára - lassacskán halad előre a világ. Eszközünk ugyan egyre jobban tökéletesednek, ámde megismerési képességünk messze nem tart lépést ezzel a technikai fejlődéssel.

Egyáltalán nem a számítógép a világ megismerésének a kulcsa. Nem arról van szó, hogy egyelőre ez az eszköz - az álhiedelmekkel szemben - sokakat inkább szétválaszt, semmint összeköt. Hanem arról, hogy az emberi érintkezés alapvető eszköze mindig is a nyelv marad. Ezért csak a közös emberi nyelv megtalálása, a fogalmak pontosítása és azok egyértelműbb használata által jutunk közelebb a végcélhoz - a valóság megismeréséhez és annak átalakításához.

Jelen fejezetnek az a célja, hogy tisztázza a rendszer fogalmát és rámutasson az információs rendszerek szemléleti hiányosságaira.

Nincs semmi kivetendő abban, hogy a ‘rendszer’ és az ‘információ’ fogalmát a *mindennapi életben* oly gyakran és mindannyiszor annyira lazán használjuk. Ha például az ‘átlagos testalkatú személy’ fordulattal élünk, akkor ugyancsak zokon vennénk, ha bárki is számon kérné tőlünk az ‘átlagos’ matematikai vagy éppen a ‘testalkat’ biológiai definícióját. Ezzel szemben roppantul meglepődnénk azon, ha a matematikus vagy a biológus saját *szakmája gyakorlása* közben sem tudná azt, hogy miről is beszél. Márpedig az a sajnálatos helyzet, hogy az információ, a rendszer és az információs rendszer alapvető fogalmait az ebben a szakmában dolgozók is féltudatosan, nem érzékelik, csakis hétköznapi értelemben használják.

1.1 A RENDSZEREKRŐL ÁLTALÁBAN

A **rendszer** kifejezéssel igen gyakran találkozunk a mindennapi életben. Az emberek ugyanis önkéntelenül ebben a kategóriában gondolkodnak, miheyst valamilyen összetettebb - vagyis több tényezőt felölelő - dologgal találkoznak. Ebből a szempontból szinte közömbös, hogy alkalmazák-e vagy sem magát a ‘rendszer’ szót. Néha igen, máskor nem. Ha az atomról vagy a világ-egyetemről; a lakásról vagy a városról; a fáról vagy az erdőről - bármiről is - diskurálunk, mélytudatunkban mindig ott rejtőzik a rendszer képzelete akkor is, ha magát ezt a fogalmat ki sem ejtjük a számon. Mások viszont különféle jelzőkkel illetjük vitáink, megbeszéléseink, közléseink

tárgyát. Például: most a gazdasági, a politikai, a pénzügyi, a hírközlési, a számítási, az ökológiai, a mezőgazdasági, az olajfinomító stb. rendszer a témánk.

A világon minden egyes dolog a maga szintjén összetett. Ezért nem meglepő, hogy a rendszer kifejezést ennyire sűrűn használjuk akár explicit, akár implicit módon. Mivel pedig ezt a kategóriát szinte bármire alkalmazhatjuk, magának a rendszer fogalomnak a meghatározása szinte lehetetlen. Mert ugyan mi közöset lehet találni - például - a tízes számrendszerben, a Mengyelejev-féle periódusos rendszerben és egy gépkocsi elektromos rendszerében?

Azért, hogy semmi esetleg odatartozót ki ne zárjunk ebből a kategóriából, az alábbiakban egy nagyon egyszerű meghatározást fogunk adni:

D 1.1 A rendszer egymással összefüggő tényezők együttese.

Az egyik oldalon igaz, hogy ez a definíció tényleg nagyon átfogó és nem túl sokatmondó. Csak annyit tudunk meg belőle, hogy a rendszer olyan **elem**ekből épül fel, amelyek között **viszonyok** léteznek. Ám sem az elemeknek, sem azok viszonyainak a természetére nézve nem kapunk semmiféle eligazítást. Mi több, még a rendszer térbeli kiterjedéséről vagy időbeli érvényéről sincs fogalmunk. Viszont a másik oldalon éppen ezért bátran alkalmazhatjuk ezt a kategóriát a tízes számrendszerre, a Mengyelejev-féle periódusos rendszerre és a gépkocsi elektromos rendszerre egyaránt, mivelhogy mindezek esetében 'egymással összefüggő tényezők együtteséről' van szó.

A szólás szerint „a világon minden mindennel összefügg”. Ezért úgy tűnik, hogy valójában csak egyetlen egy végtelen nagy rendszer létezik: ez pedig maga a világ. Ám - a költő szavaival szólva - „az egész világ nem a mi birtokunk”. Ezért a **rendszer-szervező**nek (egyelőre így titulálunk minden olyan személyt, aki rendszereket vizsgál abból a célból, hogy azokat tökéletesítse) meg kell találnia saját érdeklődési körének a szűkebb tárgyát. Eközben azonban nem szabad elfeledkeznie a 'minden mindennel összefügg' elvéről. Ez nem közhely; számos információs rendszer éppen azért tökéletlen, mert megalkotói nem ügyeltek eléggé a kitétel mögötti mély és valódi tartalomra.

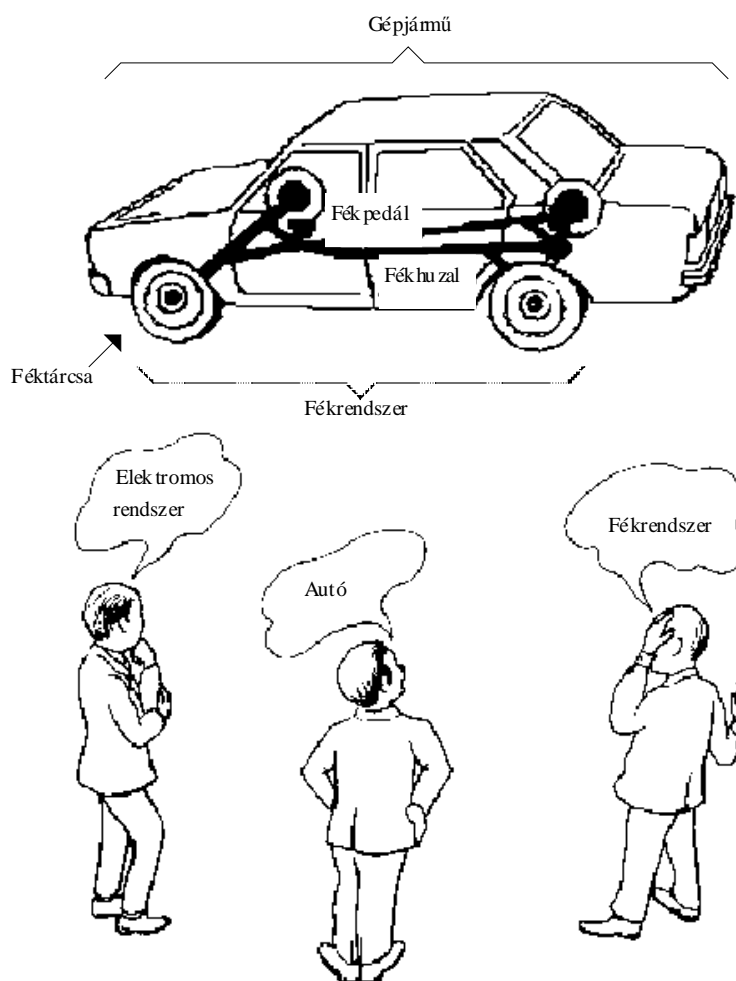
Ahhoz, hogy majd sikeresen foglalkozhassunk egy-egy általunk vizsgálandó és tökéletesítendő **konkrét** rendszerrel, tisztában kell lennünk a rendszernek, mint **absztrakt** fogalomnak a lényegével. Ezért ezt az első fejezetet arra kell szánnunk, hogy a rendszerek legfontosabb vonásait feltárjuk.

A rendszerek az azokat alkotó tényezők jellemző vonásai, viselkedése, jellege - egyszóval: természete - szerint nagyon sokfélék lehetnek. Ez a felismerés a nyolcvanas éveket megelőző időkben sok rendszertudóst arra csábított, hogy magukat a rendszereket is rendszerezze, vagyis ilyen-olyan módon osztályokba sorolja. A hatvanas évek végén és a hetvenesek elején az akkori szervezőknek egy olyan tankönyvből kellett vizsgáznuk, amely nem volt más, mint egyetlen egy terjedelmes taxonómia. A könyv több száz oldalon, több tucatnyi ismérv szerint sorolta fel, hogy van élettelen és élő; homogén és inhomogén; elsődleges és másodlagos stb. stb. rendszer. Mondanivalónkat tekintve ezt a teljesen elvi osztályozást nem tartjuk alapvetően fontosnak. Így az olvasónak nem kell félnie attól, hogy ebben a műben is taxatív felsorolásokkal fogjuk untatni. Azt viszont nem tudjuk majd elkerülni, hogy kitérjünk a bennünket érdeklő rendszerek egy-egy lényeges sajátosságára.

A gondolatmenetet folytassuk azzal, hogy a rendszer az egyik legáltalánosabb **módszertani-technikai** kategória. Ez azt jelenti, hogy a dolgok vizsgálatára és tökéletesítésére törekvő módszeres ember mindig rendszert lát, vagyis tudatosan tényezőkben és azok összefüggéseiben gondolkodik. Azonban a 'módszertani-technikai' kifejezés megtévesztő is lehet, és ezért sokan az objektivitás leple alatt már az első vizsgálati lépést elhibázzák azáltal, hogy szubjektíven fogalmazzák meg a rendszer lényegét.

Mindez arra vezethető vissza, hogy elvileg mindenkinek szíve-joga bármilyen olyan tényező-együttest rendszernek tekinteni, amit ekként - rendszerként - akar vizsgálni. Ha tehát például az órák azt állítja, hogy a reparálendő darab rendszer, akkor nem téved, sőt, az egyetlen helyes megközelítést alkalmazza. Ám arról sem illenék elfeledkezni, hogy „a szabadság = felismert szükségyszerűség”. (Senkit se vezessen félre az idézőjeles kitétel, amely sokkal régebbi eredetű, mint azt esetleg gondolnák és semmi köze sincs a materializmushoz.) Ezért a konkrét, az általunk vizsgálandó rendszereket nem teljesen szabadon határozzuk meg és fogjuk fel.

Az 1.1 ábra azt illusztrálja, hogy mindenki a maga szemléletének megfelelően határozza meg vizsgálata tárgyát. Van, akinek számára a teljes gépjármű jelenti ‘a’ rendszert, mások részére viszont csak a fékberendezés tényezőinek és azok összefüggéseinek az együttese ‘a’ rendszer.



1.1 ábra: A rendszer mint technikai kategória

A bennünket érdeklő rendszerek mindig kollektív gondolkodás termékei. Az órát az órák egyedül javítja (bár már ez sem teljesen igaz, hiszen lehet, hogy van segédje is). Ezzel szemben még sohasem készített senki sem egyedül, egy személyben **információs rendszert**. Az ilyen

rendszer mindig többé-kevésbé meghatározott megegyezéseken - idegen szóval: **konvenciókon** - alapul. Ezért semmit sem tud kezdeni az ilyen rendszerekkel (így valójában nem is igazi rendszerszervező) az, aki nem ismeri ezeket a konvenciókat.

Tehát szabadságfokunk meglehetősen korlátos. Elvileg ugyan jogomban áll, hogy én másként értelmezsem a tízes számrendszert, mint az szokásos, vagyis eltérjek a bevett és közismert konvencióktól. Csak éppen az én értelmezésemet senki sem fogja átvenni, nem fogják megérteni, ebből nekem származik bajom, és látszólagos szabadságom a káromra válik. Valóban szabad akkor leszek, ha felismerem: a tízes számrendszert szükségszerűen úgy kell alkalmazni, hogy...

A bennünket érdeklő rendszerekre kétszeresen is vonatkozik az előbbi kitétel. Ugyanis az információs rendszerek - szemben például a tízes számrendszerrel - mindig meglehetősen **összetettek**. Ez már önmagában véve is azt sejteti, hogy kollektív megismerésükhöz és fejlesztésükhöz nagyon sok konvencióra van - illetve lenne - szükség. A mi rendszereink - szemben például az órával - mindig közvetlenül **személyes ismeretekhez** fűződnek. Ez pedig roppantul megnehezíti a megismeréshez és fejlesztéshez szükséges megegyezések kialakítását. (NB.: Megegyezés alatt itt nem alkut kell érteni. A konvenció a rendszer tényezőire és az azok összefüggéseire vonatkozó közös nyelv. Ha én 'egyet' mondok, akkor azon te is 'egyet' értesz, és mindkettőnk számára az egyet a kettő, nem pedig a három követi.)

A konkrét információs rendszerek lényegének a meghatározása nem könnyű feladat. Ha például azt mondjuk: 'anyaggyártási rendszer', akkor igen sokan tudni vélik, hogy miről is van szó. Azért, mert bizonyos körökben ismertek az erre a tényezőegyüttesre vonatkozó általános és alapvető konvenciók. Azonban az összetettség és a személyhez kötöttség miatt ezek a megegyezések távolról sem annyira pontosan meghatározottak és széleskörűen elterjedtek, mint az egzakt tudományokban - például a matematikában - alkalmazottak. Ezért száz ember közül nem fog akadni kettő sem, aki teljesen pontosan ugyanúgy fogalmazná meg az anyaggyártási rendszer mélyebb velejét annak dacára, hogy látszólag valamennyien ugyanarról a dologról beszélnek.

Ez pedig baj. Sőt, a **rendszerszervezés** gyakorlatában ez az egyik legnagyobb gond. Sajnos sokszor nem azzal kezdik a rendszerszervezési munkákat, hogy kialakítanak egy kollektív gondolkodásmódot, nyelvet, konvencióhalmazt. Erre úgymond nincs idő. Mivel azonban megegyezések nélkül nem létezik rendszer, így azokat a résztvevők során kell 'megtalálni', ez pedig együttesen jóval több időt és energiát igényel, mint amennyibe a jó alapozás kerülne.

Ezen pont lezárásaként felhívjuk a figyelmet a bennünket érdeklő rendszerek egy lényeges vonására. A régi könyvek némelyike különbséget tett **szervezett** - tudatos emberi munkával kialakított - és **szervezetlen** - az embertől függetlenül létrejött - rendszerek között. Számos elvi ok miatt nem érthetünk egyet ezzel az osztályozással. Egyrészt azért nem, mert a 'szervezetlen rendszer' szó páros a mi fülünknek nagyon furcsán hat. Másrészt azért nem, mert néhány 'tudatos emberi munkára' igen kevésbé illik a 'szervezett' jelző. Harmadrészt azért nem, mert mi aligha neveznénk a világegyetemet, az óceánt és a kismadarat 'szervezetlennek'. Végül pedig azért nem, mert a jelzett párosítás egyféle filozófiai világnézetet is sugall, amellyel nem mindenki ért egyet.

Mindettől függetlenül a szervezett jelzőnél mi sem találunk megfelelőbbet. A könyv további részeiben mi is csak olyan rendszerekkel foglalkozunk, amelyek tudatos és kollektív emberi munka eredményei. Ezért korábbi definíciónkat az alábbi módon egészítjük ki:

D 1.2 A rendszer egymással összefüggő tényezők szervezett együttese.

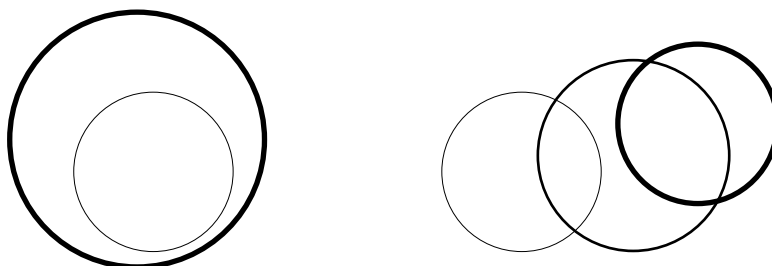
Azt pedig az olvasóra bízuk, hogy netán ezt a meghatározást tekinti-e az általánosan is érvényesnek és minden rendszert tudatosan kialakított lényegként fog-e fel...

Intermezzo

Ha az olvasó az eddigi passzusokban a kelletténél több filozófiai beütést vél felfedezni, akkor csak félig téved. Az eddig elmondottakban valóban akadtak filozófiai momentumok, nem is kevesek, de a kelletténél egyáltalán nem többek. Fel kell hívnunk a figyelmet arra, hogy a rendszer, az információ, a megismerés a legáltalánosabb filozófiai kategóriák. Ezért aligha válhat jó rendszerszervező abból, aki egy kicsinykét sem jártas a filozófia tudományában vagy elhanyagolja annak általános elveit a konkrét munkájában. A helyes **elvek** elsajátítása sokkal fontosabb, mint a **technikák** megismerése. Sajnos, ma ez a tétel egyáltalán nem érvényesül. A mai információs rendszerek némelyike nem annyira sikeres, mint amennyire lehetne. Ennek okai pedig elsősorban nem a technika, hanem sokkal inkább a szemlélet hiányosságaiban keresendők. Éppen ezért ennek a könyvnek az egyik alapvető feladata a szemlélet formálása, ami elkerülhetetlenül filozófiai jellegű gondolatok felvetésével is jár.

1.2 A RENDSZER HATÁRAI ÉS KÖRNYEZETE

„Like a wheel within a wheel...”. Mint kör a körben. A rendszerek is pontosan ilyenek. Nem tudjuk - nem is lenne célszerű - a teljeset megragadni. Éppen ezért kiemeljük a bennünket érdeklő részletet és rá azt mondjuk: ez az én rendszerem. Azt azonban pontosan tudjuk, hogy mesterségesen húztuk meg a vonalakat: az én rendszerem nem független a többitől, számos másikkal körkörösén átfed vagy pedig az egyik kör a másikat tartalmazza. Ezért a rendszer globális vizsgálatánál ki kell térnünk az **átfedésnek** és a **tartalmazásnak** az elvi kérdéseire, amelyek egyébként egymástól sem függetlenek.



1.2 ábra: A rendszerek egymás közötti viszonyai: tartalmazás és átfedés

Minden fejlesztést, átalakítást **elemzésnek** kell megelőznie. Az elemzés első, alapvető feltétele a **megismerés**. A megismeréshez viszont az szükséges, hogy **kijelöljünk** a fejlesztés tárgyát, vagyis meghatározzuk rendszerünk lényegét.

Ez a meghatározás úgy történik, hogy meghúzzuk a **rendszer határait**. Mit kell ezen a kitételen érteni? Nos, a rendszer egymással összefüggő tényezők együttese. Ezért a határmeghúzás nem más, mint egy céltudatos igenlés és tagadás páros. Kijelentjük, hogy rendszerünkbe ezek meg ezek a tényezők és azok viszonyai tartoznak. Ez az **igenlés** momentuma. Mivel azonban minden mindennel összefügg, az igenlés egyben **tagadás** is: rendszerünkben kimaradnak azok meg azok a tényezők és azok viszonyai. A papíron nem lehet úgy kört rajzolni, hogy az szét ne válassza a papír mezőit: a körön kívüli már nem tartozik a **körbe**, de a **körhöz** tartozik, mert annak meghúzása által lett kívüli.

Miért van már megint szükség ilyen filozófikus kijelentésekre? Azért, mert a fenti logikus és pofonegyszerű elvi tételről a gyakorlatban szinte nap mint nap elfeledkeznek. Az információs rendszerek 'szereplői' - egyelőre nevezzük így a vezetők, az alkalmazók és a fejlesztők csoportjait - több hibát is elkövetnek.

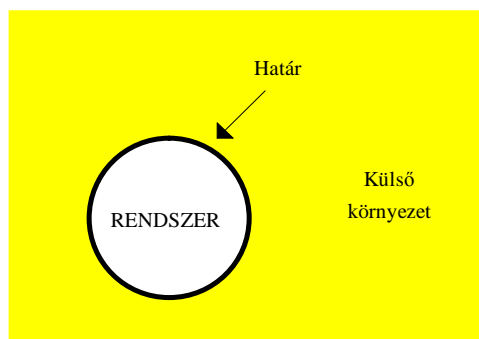
Az egyik az, hogy egyáltalán nem húznak kört. Magyarul: nem határozzák meg pontosan a rendszer határait. Most úgymond anyaggazdálkodási rendszert fejlesztenek, de senki sem rögzítette, hogy azon mit is kell érteni, vagyis milyen tényezők és viszonyok tartoznak az anyaggazdálkodási rendszerbe.

Ebből fakad azután a második probléma, amit 'luftballon effektusnak' lehetne nevezni. Jellemzőbb a „jé!-még-azt-is-lehetne-hogy” tágulás. Kezdetben - és persze titokban, mert a határokat senki sem rögzítette - az anyaggazdálkodás valójában csak anyagnyilvántartást jelentett. Azután a kör elkezd tágulni - „még azt is lehetne” - és egy idő után már senki sem tudja, hogy a kezdetek kezdetén mihez is rendelték az erőforrásokat? Ritkábban előfordul a „jajj!-istenem-ez-túl-nagy-béka” effektus is, és a rendszer-lufi fokozatosan lepad a vállalkozókedv csökkenésével párhuzamosan.

A harmadik - legtipikusabb - jelenség a 'kör kivágása'. Ez és csakis ez tartozik a 'mi' rendszerünkbe; a vonalon túli dolgok bennünket nem érdekelnek; azok a 'ti' rendszereitekbe tartoznak, ha tartoznak - de az nem a mi dolgunk. Egészen komolyan megdöbbentő az a mindennapi jelenség, amely szerint az információs rendszerek szereplői - vezetők, felhasználók, fejlesztők - sáncnak, várnak érzik az 'ő' rendszerüket. „Az én házam az én váram” alapon a határ végletes - mi több: sokszor végzetes - elhatárolódást jelent a számukra.

Ezzel a sajnálatos és zavaros gyakorlattal szemben a tiszta és logikus elmélet azt mondja, hogy a határok meghúzása nem jelent sem többet, sem kevesebbet, mint az ún. **rendszerkörnyezetek** szétválasztását. Attól a pillanattól fogva és azt feltételezve, hogy meghúztuk a rendszer határait, az azokon belül eső tényezők szervezett együttesét a rendszer **belső környezetének** tekintjük. Ez persze nem jelenheti azt, hogy elfeledkezhetünk a vonalon kívül maradt dolgokról, azaz a rendszer **külső környezetéről**. Azért nem, mert a kört nem üres lapra rajzoltuk és a körvonal meghúzásával mesterségesen szakítottunk meg tényezők közötti viszonyokat.

Egy példával szemléltetjük a fenti kitételt. Tegyük fel, hogy cégünk cikkek kereskedelmével foglalkozik. Az értékesítési információs rendszer körvonalait úgy határozták meg, hogy a számla és a számlázás azon kívül maradt. Ez az elhatárolás önmagában véve nem nevezhető sem helyesnek, sem helytelennek. A rendszer határait jogunk van céljainknak megfelelően kijelölni. Az viszont hiba, ha nem vagyunk tekintettel a számlára, mint a rendszer külső környezetébe tartozó olyan tényezőre, amely a belső környezet elemeivel - például magával az értékesített cikkel - közvetlenül is összefügg. Hiba, mert a rendszer, a határ, a külső és a belső környezet egymással szorosan kapcsolódó fogalmak: egyikről sem lehet beszélni, egyikkel sem lehet foglalkozni a másik nélkül.



1.3 ábra: Rendszer, határ és külső környezet

A határkijelöléssel kapcsolatosan tudni illik, hogy számos olyan kérdést érint, ami már a **rendszersz pszichológia** körébe tartozik. Vegyük alapul az X vállalatot, amely mondjuk A, B és C információs rendszert készít. Ezek egyazon dolognak a részei. Ennek ellenére e három rendszer alkotói nincsenek tekintettel a másik rendszerre, mint **közvetlen** külső környezetre. Az A-ban is lesz - mondjuk - egy személy-állomány, meg a B-ben is úgy, hogy a kettőben vezetett ismeretek még távolról sem hasonlítanak egymásra. Ezzel szemben mindenki kényszerben ügyel arra, hogy a **közvetett** külső környezetet jelentő Y hatáság igényeit a maga A, B és C rendszerében érvényesítse.

Mindezek után jogosan vetődik fel a kérdés, hogy szép-szép a határokról és környezetekről szóló elmélet, de egy konkrét helyzetben, a valós gyakorlatban hol célszerű meghúzni a körvonalat? Létezik-e valamilyen objektív módszer a rendszer behatárolására, ha szubjektíven mindenki azt tekinti rendszernek, amit akként akar vizsgálni? Férfiasan bevalljuk, hogy erre a felvetésre csak dodonai választ tudunk adni: a kör az kör. Mindössze csak két dolgot lehet tudni róla. Az egyik az, hogy kerek. A másik az, hogy a kör szétválaszt (határ) és összekapcsol (környezet), mert a körök egymásba érnek. Viszont kör a kicsi is, meg a nagy is; a körzővel meg a számítógéppel rajzolt is. Annyi tehát biztos, hogy nem a méret és nem az eszköz a mérvadó.

A szemléltetés kedvéért ismét forduljunk az értékesítési rendszer példájához. Azt, hogy ennek része-e a számlázás vagy azt külön rendszernek tekintik, nem tudjuk és nem is akarjuk eldönteni. Az viszont bizonyos, hogy amennyiben két körben gondolkodunk, úgy azokat csak átfedékként szemlélhetjük határokat és belső/külső környezeteket tartva a szemünk előtt. A másik megfontolandó dolog az, hogy az általunk megrajzolt valami lehet hibás is. Ha az értékesítési rendszer tényezői közül kimaradna - mondjuk - a termékadatok karbantartási funkciója, akkor ez a kör nem lenne kerek. (Az ilyesmi persze - az olvasó szerint - nemigen fordulhat elő. Pedig jobb, ha tudja: hasonló eset konkrétan is megtörtént. Egy eléggé neves intézményünkben a 'rendszert' úgy telepítették, hogy elfelejtették megírni a karbantartási funkciók programjait...)

Intermezzo

Könyvünkben több hasonló történettel fogjuk untatni az olvasót. Ezekről két dolgot kell tudni. Az egyik az, hogy nem kitalált mesékről, hanem a valóságban tapasztalt eseményekről van szó. (Természetesen a konkrét neveket és helyeket számos ok miatt nem közölhetjük.) A másik az, hogy e történetekkel egyáltalán nem az adomázás, sőt még csak nem is valaki(k)nek az elmarasztalása a célunk. Mások kárán tanul az okos: talán a velük megtörténtek segítenek abban, hogy elkerüljük a tipikus hibákat.

A fentiek szerint nincs kézzelfogható, objektív, mindig használható módszer a rendszer határának és környezeteinek a kijelölésére. Az informatika még nagyon fiatal tudomány (egyesek még annak sem tartják) és tárgya - az ismeret - nagyon szubjektív lényeg. Éppen ezért arra sincs még mód, hogy mások tapasztalatait átvegyük. Azt tudjuk, hogy mi az óra és mi a tízes számrendszer. Azt viszont nem tudjuk, hogy mi az anyaggazdálkodási rendszer. Az információs rendszerek köreit - egyelőre - mindig magunknak kell megrajzolnunk. Ma a rendszerelmélet még csak általános fogódzókat tud nyújtani az egymással összefüggő tényezők szervezett együttesének a megragadásához. Az egyik ilyen támpont a rendszer tagolása, annak részekre bontása, amiről már legalább annyit tudunk, hogy nem végezhetjük teljesen önkényesen.

1.3 A RENDSZER TAGOLÁSA

A kenyérnek a morzsája is - kenyér. A rendszernek a része is - rendszer. Csak a bolondos vagy neveletlen fiatal esik neki a kenyér egészének ahelyett, hogy abból szeleteket vágna ilyen vagy olyan módon. Az információs rendszerek sem emészthetők meg egyben: mindig szükség van

a szeletelésükre. Ámde a legfőbb baj az, hogy sokan nem vagy helytelenül látják a kenyér és a szelet viszonyát.

Tegyük fel, hogy éppen a veseköveket operálják ki. Érthető okokból engem roppantul zavarna, ha ezt a műtétet mondjuk egy fogorvos végezné. Nekem első megközelítésben olyan sebészre van szükségem, aki rendszernek a vesét és nem pedig a fogsoromat tekinti. Viszont attól, hogy az operatőr ért a veséhez, én még egyáltalán nem leszek teljesen boldog. Mert mi van akkor, ha a műtét közben elkezd a vérnyomásom, a szívem rendetlenkedni? Mit gondol az olvasó: a műtét során én mit tekintek rendszernek? Csak a vesét?

Más. Szomszédom autóján a féket birizgálták. Hazahozta: hát a fékolaj szintjét visszajelző lámpa nem működik. Szentigaz, hogy a fékszerelő nem elektromos szakember, de ettől barátom még nem boldog. Számára ugyanis nem a fék, nem az elektromos bizgentyű, hanem a teljes autó a rendszer, hiába beszélünk amúgy külön-külön is a gépkocsi fék- és elektromos rendszeréről (ld. 1.1 ábra).

Értik már a lényegét? Tagolásról csak akkor lehet beszélni, ha eleve ismerjük azt a valamit, azt a teljeset, ami számunkra, személyesen, egységes egyet jelent. Ha a műtőasztalon fekszem, számomra én vagyok a rendszer. Eközben persze mások számára a kórház is egy rendszer; a műszerek is rendszerek; a sebészt és az asszisztenseit adott bérrendszerben fizetik (alul); egész társadalmunk a rohadt csúszópénzre épült rendszer; minden mindennel összefügg; de azért ott és akkor számomra csakis én vagyok a rendszer. Ha pedig a féket javítják, akkor aligha érdekli barátomat, hogy milyen általában a magyarországi szervízrendszer (bár arról is van véleménye), elvileg a kocsikban milyen fékrendszerek létezhetnek stb. Számára a saját kocsija jelenti a rendszert, ami más dolgoktól ugyan nem független, de mindez őt nem érdekli, csak az, hogy működjön már.

Tagolni csak az egyszer már átlátott egészet lehet. A vállalatokban, cégekben, intézményekben - általánosan: a szervezetekben - ez az alapvető egység adott és az nem más, mint a szervezet teljes egésze.

E 1.1 Az informatikus számára maga a teljes szervezet 'a' rendszer.

Az első megközelítésben nem lehet ennél sem nagyobb, sem kisebb körben gondolkodni. Ez az alapelv annyira logikus, annyira kézzelfogható, annyira egyszerű, hogy sokan talán nem is értik: miért van szükség az ilyen erőteljes hangsúlyozására? Elmondjuk.

A földrajzi térség, az állam, a társadalom, az egyéb szervezetek a rendszerünk tágabb környezetét jelentik. Nem hagyhatjuk figyelmen kívül ezeket a dolgokat. Természetesen még arra is törekszünk, hogy mindezeknek a tényezőit a saját elképzeléseinkhez igazítsuk. Azonban tudjuk, hogy e közvetett környezetnek a vizsgálata és átalakítása nem a mi feladatunk. Nem akarunk túl sokat fogni, mert az ilyen kísérlet eleve kudarcra van ítélve. A diktátoroktól eltekintve ezt a tényt mindenki belátja. A fordítottját viszont nagyon kevesen ismerik fel és el: hiba az is, ha túl kicsiben gondolkodunk. Bár az értékesítési rendszer nem tartalmazza a számlázást, de az előbbi vizsgálata és átalakítása ugyanazon a szervezeten belül sohasem lehet független az utóbbiétól - és fordítva. Nem mondhatjuk azt: csak a vese tartozik ránk, a szív pedig nem érdekli: a kettő egy szervezetnek a része.

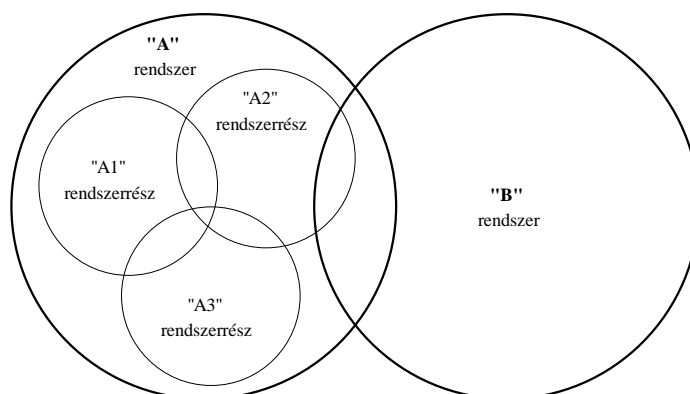
Az információs rendszerek bonyolultak. A hatvanas években még voltak olyan elképzelések, amelyek szerint az információs rendszert egyben, egyetlen monolit dologként kell fejleszteni. Ma már tudjuk, hogy ez a megközelítés nem vezethet sikerre. A rendszereket kezelhetőségük, megfoghatóságuk érdekében feltétlenül tagolni kell. A **rendszer-tagolás** nem szétszabdalást, nem egymástól független dolgokra való darabolást jelent.

Ennek a felismerésnek a jegyében a régi elméleti szakkönyvek bevezették a **rész-** és az **alrendszer** fogalmát. Mi több, komoly disszertációk jelentek meg e két valaminek az egyezőségéről, különbözőségéről és viszonyáról. Például az anyaggazdálkodáson belül részrendszer a

készletnyilvántartás, a számlázás, a vevőnyilvántartás stb. Ezzel szemben két alrendszer az X- és az Y-féle anyagok készletének a vezetése. Mindebben volt valami ráció. A jelzett felosztás szerint a rendszer funkciók szerint tagolt egységeit rész-, az ismeretek szerint tagoltakat pedig alrendszereknek titulálták. Azonban már az elméleti szakkönyvek sem alkalmazták következetesen ezt a két fogalmat. A gyakorlatban pedig végleg senki sem tudta, hogy mikor kell használni az 'al', mikor a 'rész' megjelölést. Ez pedig oda vezetett, hogy a jelzőket elhagyva és az összefüggésekre nem figyelve ma már mindig egyszerűen csak rendszerekről beszélünk. A gépjárműnek és a lakásnak a biztosítása egy-egy rendszer; nem 'al' vagy 'rész'; semmi közük sincs egymáshoz, mert nem tudunk - vagy talán még inkább: nem óhajtunk - egyben és azon belül többen gondolkodni.

A rendszert tagolni kell és ezt tudatosan kell tenni. Az 'al' és 'rész' elvi jelzőket illetően gordiuszi megoldást fogunk kínálni. A rendszer részekből áll - és passz. Éppen ezért tetszik nekünk az egyik gyakorlati szakember fordulata, aki nemes egyszerűséggel **rendszerresz**ekről beszélt. Nem is tudta, hogy e szóhasználat mögött milyen mély filozófia rejlik. A szavakat ízlelgetve gondolkodjon el az olvasó azon, hogy a részrendszer kifejezés önkéntelenül is elválaszt, míg a rendszerresz inkább az összekötést sugallja.

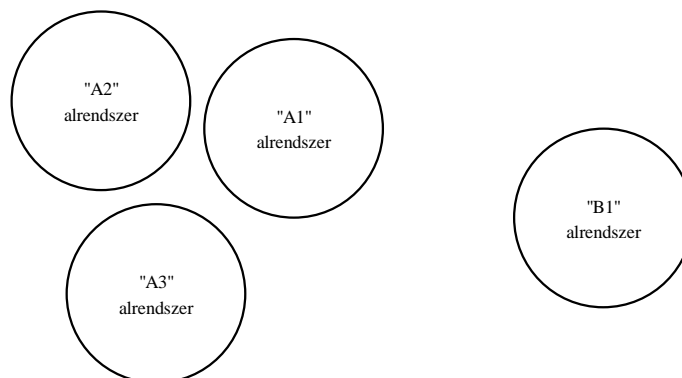
Természetesen a tagolásban sem a szóhasználat, hanem a mögöttes tartalom a fontos. Megismételjük a korábbi kitétel: tagolni csak azt lehet, ami egyszer már megfogalmazásra került egészként is. Ehhez képest döbbenetes, hogy manapság mennyien beszélnek a vállalatuk ilyen-olyan al- vagy részrendszeréről anélkül, hogy valaha is egységes egészként látták, vizsgálták, megfogalmazták volna azt egyetlen és mindent átfogó informatikai lényegként.



1.4 ábra: A helyes rendszertagolás

A kenyérnek a része is kenyér. A rendszernek a része is az. Így tehát magukra a rendszerreszekre nézve is mindaz igaz, amit az előző alponthan elmondottunk. Az 'A' rendszert alkotó 'A1' és 'A2' rendszerreszek viszonya elvileg nem sokban különbözik az 'A' és a 'B' rendszerek kapcsolatától. Ezért a figyelmetlen olvasó számára úgy tűnhet, hogy egy kicsit feleslegesen ismételjük önmagunkat.

Éppen ellenkezőleg: a gyakorlatról lehetne ugyanezt elmondani. A szavak igen furcsa dolgok. Visszahatnak ránk. Amit sokszor mondunk, azt előbb-utóbb úgy is gondoljuk. Ha az X vállalatban naponta elhangzik, hogy az értékesítés az egy rendszer (A), míg a könyvelés az egy másik rendszer (B), akkor hamarosan a legértelmesebb szervező is arra kényszerül, hogy összefüggő rendszerreszek helyett két teljesen különböző dolgot lásson. A nagy körön belüli kisebbek ill. az egymással átfedők helyett papírból kivágottakat, élesen elkülönítetteteket. Nem egymástól elhatároltakat, hanem egymástól elzártakat.



1.5 ábra: A helytelen rendszertagolás

Az igazi szervező számára a rendszerek úgy kapcsolódnak egymásba, mint mondjuk az olimpiai karikák. A rendszerrészek pedig egyetlen egy nagy tükör darabjaiként ugyanannak a tükrözött valaminek a részeit mutatják. Ugyan nem a teljeset, de mégis az egészet. Rajtunk múlik, hogy barbár módon - kalapáccsal - széttördelt, vagy tudatosan, finom metszéssel egymáshoz illesztett darabokban akarjuk-e látni a valóságot. Mindkettő tükörkép ad, no de mégis ...

A rendszereket tagolni kell. Nem létezik olyan objektív és mindig használható módszer, amely a tagolás mikéntjére nézve eligazítana bennünket. A rendszerek ugyanis nagyon különböző természetűek, és ezért a *'hogyan'*-ra ma még nem tudunk érdemben válaszolni. Viszont a *'hogyan nem'*-ről egyre többet tudunk. Ez nem is csoda, mivel az elmélet próbaköve a gyakorlat. Ma még a pozitív elveket nem tudjuk pontosan megfogalmazni, de a negatív tapasztalatokból leszűrteket már össze tudjuk foglalni.

T 1.1 Egy kórházi információs környezetben 'A rendszernek' tekintették az X-féle lemezkéken, 'B rendszernek' pedig az Y-féle lemezkéken beérkező - tartalmilag azonos jellegű - ismeretek kezelését, amelyek egy-egy adatszolgáltatóhoz kapcsolódtak.

Ehhez nem szükséges különös kommentár. A kórház informatikusainak igazat lehet adni akkor, ha ők is elismerik: a vese más rendszer, ha azt az X- vagy az Y-féle szikével operálják. Ez pedig egyáltalán nem mulatságos. Ma a rendszerek kijelölésében világszerte dívik a lénységnek (vese), a feladatnak (kő eltávolítása) és az eszköznek (sziké) ez a tipikus összekeverése.

T 1.2 Számos cégnél külön 'A' és 'B' 'rendszernek' hívják a Q szoftver ikszedik illetve ipszilonadik változatában megírt programokat.

Vajon mi a gondunk ezzel a megközelítéssel? Hiszen fentebb egyértelműen kijelentettük, hogy mindenkinek jogában áll rendszernek tekinteni azoknak a tényezőknek az együttesét, amelyet ekként - rendszerként - kíván vizsgálni és tökéletesíteni! Nem következetlenség tehát, ha ódzkodunk a hardver- és szoftver-verziókhoz kapcsolt rendszertagolástól?

A szoftverekkel ugyan sok bajunk akad, de ez a problémakör most egyáltalán nem ide tartozik. Az viszont igen, hogy a szoftver éppen úgy csak eszköz, mint a hardver. Így tehát balgaság a rendszer vagy akár a rendszerrész fogalmát hozzá kötni. A számítástechnikán kívüli világban nem túl sok példa akad arra, hogy az eszköz és nem a tartalom, a cél szolgál a rendszer-tagolás elvi alapjául.

A T 1.1 és a T 1.2 történetet illetően közösek és kettős jellegűek a gondjaink. Az első felvetésünk *tartalmi* jellegű. A munkaügyi és az eszközgazdálkodási 'A' illetve 'B' rendszerrész kétségtelenül csak igen távolról kapcsolódik egymáshoz. Tartalmuk annyira eltérő, hogy - persze a közvetett összefüggéseikre figyelve - esetükben jogosan beszélhetünk akár két rendszerről is. (Bár a teljes szervezetet tekintve csak részéről van szó, hiszen az eszközökért felelős személyek a másik rendszerben is szerepelnek.) Viszont a rendszerek mélyebb szintű bontásaiban a részek egyre jobban kötődnek tartalmilag is egymáshoz. Éppen ezért a rendszer- és a rendszerrész-szintű tagolás csak technikai értelemben analóg. Mivel pedig a tartalmat nem a hordozó (T 1.1) és nem a kezelő (T 1.2) eszköz határozza meg, a két történetben közös, hogy helytelen alapon történt volna a rendszertagolás.

Volna, mert második felvetésünk *formai* jellegű. Szerintünk a rendszer illetve a rendszerrész fogalmát csak akkor lenne szabad használni, amikor a tartalom, a határok, a belső és külső környezet viszonya már meghatározásra került az előre rögzített céloknak megfelelően. Ha a cél, a tartalom, a határ és a környezet nem tudatosan meghatározott, zavaros vagy nem tisztázott, nincs is leírva - pontosan dokumentálva -, akkor a 'rendszer' kifejezés használata túlzásnak tűnik. Vegyük csak például a 'szociális hálót'! Itt meg ott számítógépes eszközöket vásárolnak; ebben az országban már több tucat - ha nem több száz - számítógépes 'rendszert' készítettek a szociális ellátás támogatására. Ám ezt az egész dolgot senki sem látja valódi rendszerként; nincs meghatározott tartalma, tagolása, környezete; a végcélra - az ellátandó emberre - pedig szinte senki sem ügyel (tisztelőt a ritka kivételnek).

Most pedig váltsunk témát. Ez a könyv nem általában a rendszerekről, hanem speciálisan az **információs rendszerekről** szól. Sajnálatos, hogy a cél, a mód és az eszköz összhangja nem túlzottan jellemző a mai ilyen rendszerekre. Ennek a ténynek az a legfőbb oka, hogy az emberi agyakban összekeveredik a valóság, annak képe és a mindkettő manipulálásához szükséges illetve rendelkezésre álló segédletek hada. Ezért az információs rendszerek lényegének a taglalása előtt rá kell mutatnunk a többszörös látásmód szükségességére illetve zavaraira.

1.4 KETTŐS RENDSZERNÉZET

A régi szakkönyvek helyesen mutattak rá arra (bár ezt sokszor csakis taxatíván tették), hogy vannak egymáshoz képest (!) elsődleges és másodlagos rendszerek. A jelzők itt nem a *fontosságra*, hanem az *eredendőségre* utalnak. Pl. a termelő vállalatban nyilván maga a termelés, nem pedig a róla szóló statisztika a fontos, viszont a statisztikai hivatalban éppen fordított a helyzet. Ezen két szemlélettől függetlenül annyi biztos, hogy a két dolog viszonyában a termelés az eredendő, mert nélküle nem létezhetne statisztika.

Az ismeret - ez itt nem meghatározás, csak eligazító tájékoztatás - a valóság tükörképe. Éppen ezért első megközelítésben a valóság *elsődleges*, a róla szóló ismeret pedig *másodlagos* momentumnak tekinthető. Ez többnyire így van már amiatt is, hogy általában a másodlagos az elsődlegeshez képest akár többszörös is lehet. Például Icáról X azt tudja, hogy..., viszont Y számára úgy tűnik, hogy... Közben pedig a valóságban Ica teljesen más: nem felel meg sem az X, sem az Y róla alkotott másodlagos képének. Ehhez képest elég meglepő, hogy az emberek a mindennapok során a másodlagos képek alapján határolják be az elsődlegeseket. Pl. nem az a lényeg, hogy a gépkocsi valójában piros színű, hanem az, hogy egy papírra valaki egykor azt írta róla, hogy 'kék'.

Éppen ezért az elsődleges és másodlagos valami igen furcsa viszonyban állhat egymással. Először a 'nadráglopás' talán közismert példájára hivatkozunk. X-től elcsenték a gatyáját (elsődleges valóság), majd X-ről elterjed a hír, hogy részese volt a nadráglopásnak (másodlagos valóság), ami ugyan igaz, de... Komolyabb esetre térve tegyük fel, hogy a raktárban fizikailag ott heverészik öt darab dűzni - az elsődleges valóságnak megfelelően. Ám a készletkarton illetve a számítógép - most ne firtassuk, hogy milyen emberi hibák miatt - nulla darabot mutat ki. Ez tehát a másodlagos valóság. Az egyszerű kérdés már csak az, hogy a dűznikből vajon hányat fogunk eladni? Ha még Ali bácsi, a raktáros, emlékszik és veszi a fáradságot arra, hogy a tárolóhelyhez baktasson, akkor a dűzni elkel. Egyébként olyan, mintha nem is létezne.

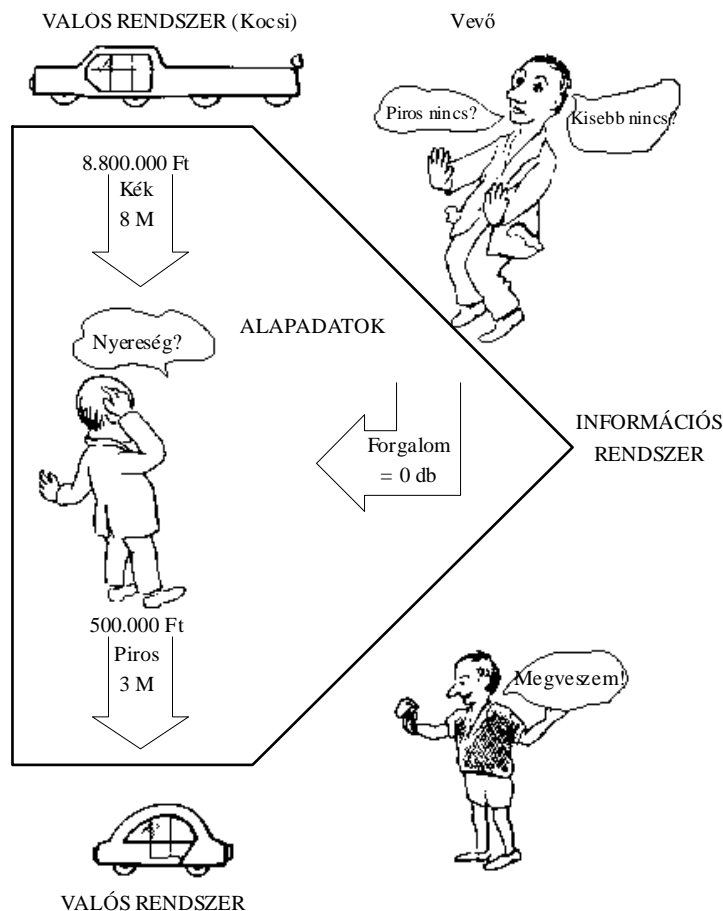
Az első tanulság az, hogy nem az van, ami valóban van, hanem az van, amiről azt hisszük, hogy van. A gyakorlat naponta igazolja ezt a kitétel. Egyszer e mű szerzőjének a lakcímét elírták a telefonkönyvben - ami kész csoda, hiszen régen abban a helyes lakcíme szerepelt. Azóta nem a valódi címén próbálják keresni. A másodlagos valóság, a leírt valami vált meghatározóvá az elsődleges rovására.

A kétféle dolog meglehetősen - hogy is mondjuk - 'dinamikus' viszonya dacára mégis elsődleges és másodlagos rendszerben kell gondolkodnunk. Éppen ezért az informatikus-tudósok hosszú időközön keresztül a **valós rendszer** [real system] kifejezéssel illették az ismeretekkel kiszolgálni kívánt elsődleges rendszert, míg az azt kiszolgáló másodlagosat **információs rendszernek** [information system] titulálták. Tehát mindig két egymásba fonódó rendszerről van szó. Ha a tartalom az anyaggazdálkodás, akkor van egy érdemi, tényleges, valós anyaggazdálkodási rendszerünk, amely ilyen-olyan 'logisztikai' (ma ez a divatos kifejezés) alapokon arra az utánrendelési algoritmusra épül, hogy... Ezt az elsődleges rendszert nem is lehet ma már elképzelni - egyébként sohasem lehetett - az anyaggazdálkodási információs rendszer, mint másodlagos lényeg nélkül. Dűzni pedig nincs akkor sem, ha fizikailag ott heverészik a raktárban, de nincs róla kimutatás.

A korábbi múltidős fogalmazást ('illették') az indokolja, hogy időközben egy picit megváltozott a terminológia. A 'valós rendszer' fogalom kegyvesztett lett, aminek két oka is van.

Az egyik az, hogy az informatikus-tudósok végre rádöbrentek arra, hogy - jé! - hiszen **maga az információ is valóság**. Nemcsak az anyag, hanem a róla szóló ismeret is él, megfogható, kezelendő. Nem csak abban merül ki a valóság, hogy az anyagot talyigára rakjuk, majd ide-oda hurcoljuk. Az is valóság, hogy papírt - anyagigénylést - kell kitölteni; az ismereteket kézzel-fogható számítógépekre kell vinni; az ismereteket is 'ide-oda kell hurcolni'. Tehát nemcsak az alkatrész, nem is csak a rá vonatkozó anyagigénylési bizonylat, hanem az utóbbinak a tartalma is 'matéria', valóság.

A másik ok az, hogy **végleg kudarcot vallott az a** vegytisztán materialista **szemléletmód**, amely szerint a fizikai anyag az elsődleges, a róla szóló szellemi ismeret pedig csak másodlagos lényeg. Már említettük a statisztikai hivatal példáját. Itt ahhoz tehetnénk a piackutató, a 'public relations' és a különféle egyéb szolgáltató cégek példáit. Ezek esetében mindig maga az ismeret számít elsődleges valóságnak (az ismeret a tevékenység alapvető tárgya). Azt pedig mindannyian tudjuk, hogy az ismeret önmagában véve is manipulálható - e szónak nem feltétlenül a pejoratív értelmében.



1.6 ábra: Kettős nézet

A fentiek miatt a pedáns informatikus-tudorok hamarosan meghátráltak. Nem kifejező, mert túlzottan is földhözragadt a 'valós rendszer' fogalom. Igazándiból nem is valós dolog az, amiről szó van, mert az netán csak megalapozatlan vágy, vagy csak olyan elképzelés, amely ködös - mi több: zavaros - gondolatokban él. Erre az úgymond korszerű és alapvető felismerésre alapozottan néhány elméleti informatikus a szerintük csúnya, nem-kifejező, sőt: félrevezető - 'valós rendszer' kifejezés helyett bevezette az '**amiről-szó-van**' fogalmát. Erre rögtön egy szép új rövidítést is alkalmaztak: **UoD** [University of Discourse]. Az anyaggazdálkodás ma már nem 'valós rendszer', hanem pusztán egy olyan dolog, amiről szó van - azaz UoD. A statisztika sem valóság, hanem csak éppen aktuálisan szóbanforgó tárgykör - vagyis egy másik UoD.

Intermezzo

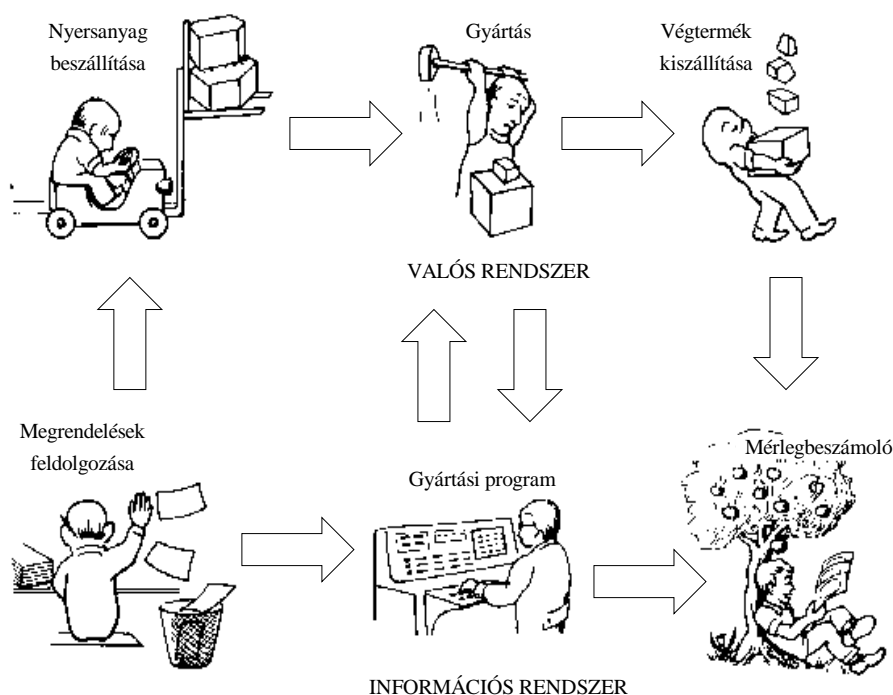
Érdeemes megfigyelni, hogy valahányszor egy áthidalhatatlannak tűnő elméleti vagy gyakorlati informatikai probléma merül fel, akkor az informatikus-tudósok ill. az informatikai terméket előállító cégek mindig egy új varázsszó-készlettel állnak elő. Nem tudunk megbirkózni az elsődleges és a másodlagos kérdésével? Nosza, vezessük be az 'UoD' fogalmát! Képtelenek vagyunk arra, hogy valóban hasznos adatkezelővel lássuk el a felhasználót? Tereljük el a figyel-

mét a valódi lényegről a 'relációs' vagy az 'objektum-orientált' - most mindegy, hogy ezek a dolgok mit is jelentenek - elkápráztató fogalmakkal. Vagy mondjuk például a számítógépek régóta ismert kapcsolati módjára azt, hogy az a 'legmodernebb és legjobb client-server' megoldás! Aki csak most ismerkedik a számítógépekkel, az hinni is fog e jelzőknek.

A gyakorlatban az új elméleti terminológia nem old meg semmit, és főleg nem ad választ a réges-régi keletű problémákra. (Ez a kitétel nem mond ellent annak, hogy a későbbiekben éppen a pontosan használt terminológia fontosságára is fel kell hívnunk a figyelmet.) Egyelőre nem a „minek nevezzetek” a lényeg, hanem maga a tartalom. Az 'UoD' új kulcsszóként semmivel sem mond többet, mint a 'valós rendszer'. E könyv szerzője a vágyat, az ötletet, a tervet, az elképzelést is valóságosnak tekinti. Ezért a finomkodó és ravaszkodó 'amiről szó van' helyett e műben továbbra is a földhözragadt valós rendszer kifejezést fogja alkalmazni.

E döntésének nemcsak a szóhasználati puritánság az oka. Az 'UoD' óhatatlanul egybemos két összefüggő, ámde eltérő természetű dolgot, mivel ködbe burkolja az elsődleges illetve a másodlagos viszonyát. Ha például az anyaggyártás tökéletesítése a cél, akkor ez egyszerre, egyidejűleg jelent egy közgazdasági és egy informatikai nézetű feladatot. Nem az egyikről és nem a másikról, hanem a kettő együtteséről van szó. Az 'UoD' éppen ezt a lényeget takarja el.

Ezért e műben maradunk a valós és az információs rendszer megnevezéseknél, amiket - a rövidség kedvéért - helyenként a VR és az IR betűpárosokkal fogunk jelölni. Például a banki ügyintéző számára az ügyfélkiszolgálás az elsődleges, a valós teendő, ám az elválaszthatatlan a másodlagos, az információs megoldástól. Az utóbbi azért másodlagos, mert ma lehet ilyen, holnap pedig olyan akkor is, ha közben maga a valós teendő (számlanyitás, kamatvezetés stb.) időközben tartalmilag semmit sem változik.



1.7 ábra: A valós és az információs rendszer viszonya

A VR és az IR nem független egymástól, de ugyanakkor mindkettő viszonylag önálló életet is él. A kétféle rendszer összefüggésének a magyarázatát illetően az olvasó talán még el nem feledett biológiai ismereteire kell apellálnunk. A régen tanultak szerint a testben van ún. **nagy-** és **kisvérkör**. Az előbbi lefelé irányul és a törzs éltetése a feladata. Az utóbbi felfelé irányul és az agyműködtetés a célja. Melyik a kettő közül a fontosabb? Ez így csacska kérdés, mert a test nélkül nem működik az agy - és megfordítva. A test és a szellem nem azonos lényeg, idővel mindkettő önállóan is változik, de egymást mindig feltételezi.

Nagyjából és egészéből a két vérkőre hasonlít a VR és az IR viszonya is. Két egymásba fonódó körről, a testről és a szellemről van szó úgy, hogy egyik sem életképes a másik nélkül. Pl. nem létezhet anyaggazdálkodás anyaggazdálkodási információk nélkül, az utóbbiaknak viszont nincs értelmük az előbbi nélkül. (A roppantul jártasak erre felvethetik, hogy a test igenis életben tartható a szívhalál után is, illetve egy szervezet 'halála' után még továbbélhetnek az arra vonatkozó ismeretek. Ez ugyan igaz, viszont ezek az ismeretek már éppen úgy nem az adott - valaha élt - szervezetet szolgálják, mint ahogy a mesterségesen éltetett agy sem volt tulajdonosának használnak.)

Jóllehet az elméletben a VR és az IR fenti analógiával szemléltetett viszonya teljesen tisztázott, a mindennapos gyakorlatban a kettő csak nem akar egyetlen lényeggé kovácsolódni, csak nem akar egységes testet-lelket alkotni. Ennek a ténynek két oka van. Az egyik az értetlenség, a másik a helytelen szemlélet. Az előbbi abban nyilvánul meg, hogy az IR fejlesztésének a résztvevői nem értenek szót egymással. A fejlesztő nem beszéli (jól) a felhasználó nyelvét, mert magát a VR-t nem eléggé ismeri. A felhasználó pedig nem érti meg (teljesen) a fejlesztő gondolatmenetét, mert számára meg az IR lényege tűnik titokzatosan távolinak.

A helytelen szemléletből származó problémák annyira súlyosak, hogy ennek az oknak külön pontot kell szentelnünk. Ezt a szakaszt azzal zárjuk le, hogy a bajok közös gyökere egy helyen - ti. a képzetlenségben keresendő.

1.5 TÉVES SZEMLÉLETEK

Tegyük fel, hogy valaki éppen lakást építtet. Beleszól a fürdőszoba általános elrendezésébe? Felhasználóként ezt bizonyára megteszi, de természetesen erre csak a ház és a többi helyiség általános elrendezése által korlátozott módon van lehetősége. Mármint belebeszél-e az átlagos állampolgár a vízvezeték-rendszer 'A' vagy 'B' módú elrendezésébe? Ezt aligha teszi meg, mert hiszen nem is ért hozzá: a technikai részletek meghatározásához szakemberre van szükség.

Érdemes megfigyelni, hogy a mindennapos 'valós' rendszereinkkel szemben az információs rendszereket illetően mennyire más a beállítottságunk.

Egyrészt ma még a szervezetek egyáltalán nem 'információs házakban' (vagyis többreszes rendszerekben), hanem eleve csak szeparált 'információs lakásokban' (tehát külön rendszerekben) gondolkodnak. Értsd: gyakran esik szó az integrált IR-ről, de azt senkinek sem áll szándékában megvalósítani. A szándék ugyanis nem vágy, hanem tudatos akarat. Az integrált IR akarásához, a rendszerrészek összehangolásához viszont **generáltervezőre** lenne szükség. A ház ugyanis sokkal több, mint a benne lévő szeparált lakások halmaza. Szervezett együttes - és kell lennie valakinek, aki ezt a rendszert a maga teljes egészében látja. Márpedig ma még elég nehéz találni olyan vállalatot, amelyben - mondjuk - a humánpolitikai és az eszkozzgazdálkodási rendszerrésznek közös a tervezője.

Másképp az 'információs lakások' maguk is rosszul tervezettek. Igen sokszor előfordul, hogy az 'információs helyiségek' önmagukban is célszerűtlen módon kerülnek kialakításra és nincsenek összehangolva a többi 'helyiséggel'. Értsd: az információs funkciók elrendezése nem cél-

szerű. Olyan ez, mintha az adott lakás fürdőszobáján belül (!) több vizesblokk is lenne ill. azok nem harmonizálnának a konyha vizesblokkjával. Az ilyesmi nem ritkaság a mai gyakorlatban. Például az egyazon ügyféllel kapcsolatos társadalombiztosítási funkciókat külön-külön csatornákon intézik egymással összehangolatlan módon akkor is, ha azok igen szoros elvi kapcsolatban állnak egymással. Ennek pedig az az oka, hogy a lakás kialakításában *több részlettervező* működik közre. Vagyis pl. más-más személy tervezi az X- és az Y-féle foglalkoztatáspolitikai eszköz információs rendszerét.

A harmadik momentum a legérdekesebb. Mindennapos valós rendszereinkben szinte elvesztettek vagyunk. Sokan - a szerző nem tartozik ebbe a sorba - még a vízcsapot sem tudják megjavítani. Ámde az IR-t illetően *mindenki szakértőnek* tekinti magát. Tessék csak megfigyelni, hogy az IR-rel foglalkozó értekezleten a felhasználók milyen nagy garral szólnak bele az elrendezésekbe, az eszközökbe, a módszerekbe - az IR minden vonatkozásába. A technikai részletekhez valóban értő igazi szakember ilyenkor szinte felesleges kolonc. Helyette - de persze az ő felelősségére - állnak elő a felhasználók a legképtelenebb ötletekkel.

A fürdőszoba személyes elrendezése nagyon fontos az emberek számára, mert a test is megkívánja a maga kényelmét. Azonban az információk szellemi világa még a fürdőszobákénál is intímebb. Az információ az emberek legszemélyesebb kincse. Ezért aligha találhatunk kivétnelet abban, hogy a felhasználók nagyon is bele akarnak szólni az IR fejlesztésébe. Mi nem is az aktív részvételi szándék ellen szoltunk feljebb: csak a szakértővé való önelöléptetés ellen tiltakoztunk.

A hamis szemléletek számtalan válfaja közül talán a legeslegveszélyesebb ez a *szereptévesztés*. Érdekes módon ennek a negatív jelenségnek egy nagyon pozitív tény ad egyre erőteljesebb alapot: mára már a számítógép és annak alkalmazása - akarva-akaratlanul - szinte mindenkit megfertőzött. A személyi számítógépek elterjedése tette lehetővé ezt a jelentős fordulatot. Manapság akadnak olyanok is, akik egyelőre még ódzkodnak a számítógépes technikától, de már nekik is be kell látniuk azt, hogy aki ezen a szinten marad, az a jövő század elején a szó szoros értelmében analfabétának fog számítani.

A számítástechnika örvendetes áttörése azonban igen veszélyes jelenségekkel is párosult. Amint a felhasználó pötyögni kezdett a billentyűzeten, alkalmazásba vette ezt meg azt a szoftvert, kezelni tudta a hardvert, azonnal egy tévhitbe esett. Úgy vélte, hogy ezzel a *technikai rész-tudással* ő már informatikussá vált. Nem, dehogyis van szüksége szakemberre, hiszen ő maga is annyi mindent megold. Az alábbi történet tipikus. A szerző szinte nap mint nap találkozik a kezdeti számítógépes lelkesedéseket követő kudarc-hangulattal.

T 1.3 Egy kisvállalkozó barátom 386-os gépen Excel-ben készítette és kezelte a számláit és azzal vezette a különböző nyilvántartásait is. Mert ahhoz értett. Máról-holnapra az egész dolog összeomlott. Nem is történhetett másként: az Excel ugyanis nem adatbáziskezelő rendszer. A nyilvántartások kezeléséhez más eszközre, annak alkalmazásához pedig szakemberre lenne szükség. Ma már barátom belátja volt szereptévesztését.

A picit felületes *számítástechnikai* részismeret nem azonos a roppantul alapos *informatikai* tudással. Téves és veszélyes az a szemlélet, amely ezt a két dolgot összemosza. Az utóbbi birtokának a hiányában a felhasználó sohasem veheti át a fejlesztő szerepét - és ennek a kitételnek az igazságát előbb-utóbb a saját bőrén fogja megérezni, amint azt a fentebbi történet is mutatta.

A következő gond az, hogy egyre inkább terjed egy furcsa számítástechnikai betegség, egy számítógépes AIDS.

T 1.4 'Te még mindig ezzel a vacak szövegszerkesztővel dolgozol? El vagy te maradva!' - jelentette ki egy másik barátom, látva, hogy éppen egy irományt fogalmazgatók.

A számítástechnika hallatlan iramú fejlődése sokakat megszedt. Megfordul a fejekben **a cél és az eszköz viszonya**: az utóbbi válik dominánssá. A szerző nem röstelli, hogy egy mindössze 170 K-s programcskával írogatja piszkozatait - és nem veti be erre a célra a legcsodálatosabb szövegszerkesztőket. A szerzőt nem fertőzte meg a számítógépes kivagyiság vírusa.

Ami pedig egyre jobban terjed. A felhasználók a feladat iránt immunná válva mindig az újabb meg újabb 'csodás' eszközökben keresik a megoldásokat. Ha valami nem működik tökéletesen, nem a feladat megoldásán gondolkodnak el, hanem új 'verziót' vásárolnak. Amitől persze semmi sem változik. Mi több: az IR egyre és egyre kötöttebbé, immunhiányossá, verziófüggővé válik. Alig kerül ki a cég az egyik 'fejlesztésből', máris a másik 'fejlesztésben' találja magát. No, nem érdemben: energiájának 70-80 százalékát a verzióváltás köti le.

Intermezzo

Érdemes elgondolkodni a fejlesztések költségeinek az arány-megoszlásán. A hetvenes években a tipikus minta a következő volt: 60-30-10%. Vagyis 60%-ot költöttek hardverre, 30%-ot szoftverre, 10%-ot a szellemi munkára. Mára már az első két faktor aránya megfordult: a szoftver teszi ki a költségek nagyobb részét. Ám ama bizonyos harmadik tényezőnek a súlya a régi szinten maradt...

Az IR fejlesztési költségeinek ez a - finoman szólva - furcsa megoszlása igen jól mutatja a téves szemléletek eluralkodását. Azonban erre utal a mindennapos szóhasználat is. Az emberi fejekben valami megfordult - és tótágast áll. Legyen itt szabad néhány különös fordulatra felhívunk a figyelmet.

Itt van elsőként maga az **alkalmazás** [application] kifejezés. Megfigyelhetjük, hogy a **szoftverek** kereskedői mindig magát az eszközt nevezik alkalmazásnak. Természetesen a Word, az Office, az Excel a **hardver** oldaláról nézve valóban 'alkalmazás'. Azonban a normális felhasználó nem a hardver, hanem a **feladat** oldaláról szemléli a dolgokat. Számára a konkrét levél megírása, az adatbázis konkrét kezelése, a konkrét táblázat összeállítása az alkalmazás. Csakhogy ma minden a feje tetejére állt. Már régen nem arról van szó, hogy barátunknak kell egy levelet írunk, ügyfeleink adatait kell adatbázisban kezelni, táblázatban kell rögzíteni a hegy-mászó kötelek paramétereit. Hanem arról, hogy csakis akkor vagyunk korszerűek, ha ilyen-olyan eszközt alkalmazunk. Ekképpen a 'mit' helyét végleg átveszi a 'mivel' - ami a legnagyobb szemléleti tévedések egyike (vö. T 1.1 és T 1.2).

A másik - tudatosan? - félrehasznált kulcsszó maga a **szoftver**. Érdekes módon nemcsak az úgymond laikus felhasználók, hanem a kimondottan profi fejlesztők is hamis szemléletben élik sajátos világukat. Számukra is a szoftver az alkalmazás.

A hetvenes években vált divattá a 'szoftver-technológia' kifejezés, amely mind a mai napig megőrizte belső lényegét. A hetvenes években nem az volt a végcél, hogy megtaláljuk az információs rendszerek tökéletesítésének a módjait, hanem az, hogy a megfelelő 'szoftver-technológiát' alkalmazzuk. A kilencvenes évekre ez a hamis szemlélet mit sem változott. Ma is ezer meg ezer olyan konferencián lehet résztvenni, olyan könyvet ill. cikket olvasni, aminek a címe - megtevesztő módon - az 'információs rendszer' fejlesztésére utal, miközben maga az anyag csakis a szoftverrel bíbelődik. Ez a jelenség képtelen méreteket ölt.

T 1.5 Idegenbe szakadt félkollégám kérdezte az első találkozáskor: 'Mivel foglalkozol?' Mondom neki, hogy informatikus vagyok. 'Áhá, tehát szoffört fejlesztesz?' (Ez nem elírás: idegenbe szakadt kollégáink ez így mondják.)

A cél, a feladat, a megoldás, az alkalmazás, a szoftver, az eszköz - és ekként maga a rendszer - fogalma a rengeteg vágy és az ezernyi lehetőség miatt nem is kicsit összezavarodik a fejekben. A számítógéppel frissen megismerkedő kvázi-laikusában éppen úgy, mint a technikákat régen ismerő kvázi-profiban. Vajon mi ennek a jelenségnek az oka? Mi a gyökere? Mi a helyes szemlélet kulcsa?

Megnyugtadjuk az olvasót, hogy nem újkeletű problémáról van szó.

T 1.6 1975-ben egy szerszámpari cég számítástechnikai szervezési (!) főosztályvezetője az anyaggazdálkodási információs rendszer fejlesztése kapcsán elemzési és tervezési kísérleteinket a következő kézlegyintéssel intézte el: „Rendszerszervezésre itt nincs szükség. Majd kreálunk egy indexelt-szekvenciális anyagfájlt és kész”.

Majd kreálunk... Az egész világot az ilyen azonnali és kézenfekvő megoldások ringatják el. Az eszközök, a könnyűnek ígérkező alkalmazások, a szoftverek. Ma sokan nem abban látják a feladatot, hogy javítsák például az anyaggazdálkodást. Hanem abban, hogy szoftvert alkalmazzanak vagy fejlesszenek. A hetvenes évek során mindez feketén-fehéren kiderült akkor, amikor az IR fejlesztése helyett a 'gépesítés', a 'géprevitel' volt a lózung, a jelszó. Nem az volt a célkitűzés, hogy az ismereteket elrendezzék, hanem az, hogy azokat 'gépre ültessék'.

Mármost húsz év alatt a lényeg semmit sem változott. A számítógépek jobbak; a szoftverek változatosabbak lettek. Az ósdinak vélt pl. COBOL nyelv helyett ma az egyik korszerűnek tartott relációsat használjuk. A képernyők színesebbek, nagyobbak és ma modern módszerekkel - szoftver-technológiákkal - dolgozunk. Pontosan olyan rosszul, mint régen. Mert még mindig csak gépesítünk ahelyett, hogy információs rendszert alkotnánk.

1.6 AZ EGYETLEN HELYES SZEMLÉLET

Valami rettenetesen hiányzik a rendelkezésre álló technika és maga a feladat között. A felhasználó ennek a hiánynak csak a tényét érzi - és tapasztalja nap mint nap a saját bőrén. Már tudja, hogy technikai részismeretei nem elegendőek a problémáinak a megoldására, de nem látja a kiutat. A hagyományos fejlesztő pedig - valljuk be férfiasan - egy picit 'szakbarbár'. Ért a technológiához és talán a mindig megújuló technikához is. Valami miatt mégsem képes jó információs rendszereket alkotni. Vajon mi az oka ennek a kétoldali képtelenségnek? Igaz lenne az, hogy sem a felhasználó, sem a fejlesztő, sem a kettő együttesen nem képes jó információs rendszert alkotni?

Sajnos, ezt a kitéltet a gyakorlat nap mint nap igazolja. Tökéletesen mindegy, hogy egy-egy kisvállalkozó pár száz vagy pár ezer ismeret-tételéről, vagy pedig az országos jelentőségű cégek százezres-milliószámú információs rendszereiről van-e szó. Nem a nagyságrend, hanem az IR elemeinek az összetettsége a döntő tényező. Száz számla kiállítása semmivel sem nehezebb, mint millióé: az eltérés csupán csak *technikai* jellegű. Ám az első, a legeslegelső számla *informatikai* megfogalmazása mindent eldönt.

Az előző pont legvégén utaltunk a felhasználó és a fejlesztő közötti kölcsönös megértés hiányára. A számszerűen kis méretű rendszereknél - például egy-egy magánvállalkozónál - sokszor úgy vélik, hogy fejlesztőre nincs is igazi szükség (ld. T 1.3). Másutt pedig úgy gondolják, hogy a profi számítástechnikus feladata a célszerű megoldás kitalálása. Mindkét esetben tévednek. A számítástechnika ugyanis nem azonos az informatikával.

D 1.3 Az informatika az ismeretek megismerésének, azok célszerű elrendezésének és kezelésének a tudománya. Az informatikus az a szakember, aki ebben a tudományban járatos.

Remélhetőleg egyetlen felhasználó sem sértődik meg azon, ha kijelentjük: a mikroépes eszközök kezelésében adott fokú jártasságra szert tevő személyeket még nem tekinthetjük informatikusoknak. Ámde a számítástechnikusoknak egy jelentős hányadára sem jellemző, hogy eligazodik az ismeretek megismerésében, célszerű elrendezésében és kezelésében. A számítástechnikusok jártasak például a programozásban, a számítógép javításában, a billentyűk kezelésében, az X és az Y szoftver használatában, a hálózatkezelésben stb.

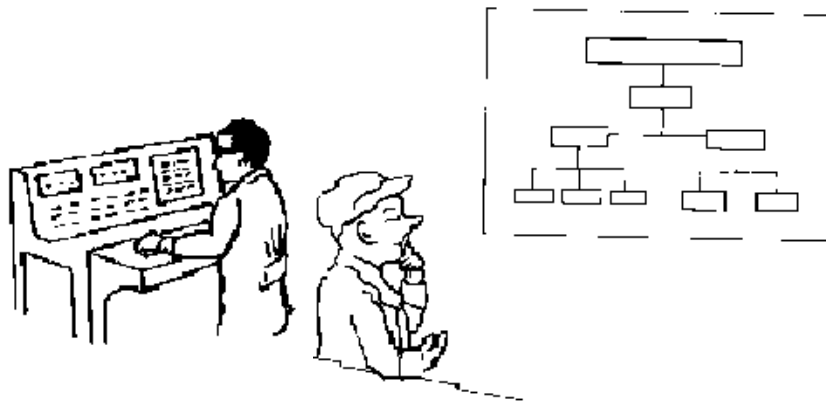
Ma már minden informatikus egyben számítástechnikus is. A fenti passzusban sem kevesebbet, sem többet nem akartunk mondani, mint azt, hogy ez a viszony nem kölcsönös. Vannak olyan számítástechnikusok, akik egyben informatikusok is, és akadnak olyanok, akik viszont nem azok.

Intermezzo

Az egyik cég - mondjuk - kereskedelmi számítógépes programcsomagot akar eladni nekem. A kereskedő ötvenszer elmondja, hogy az ő rendszere mennyit de mennyit tud. Nem igazán örül annak, amikor ötvenszer kiigazítom: Drága uram, szoftver, nem rendszer. Érti? Szoftver. Nem rendszer! Végül felháborodottan így támad nekem: Mi, informatikusok, miért nem jutunk dűlőre? Ekkor jön a végső kiigazítás: Azért nem, mert én informatikus vagyok, Ön pedig nem az.

Ez részről nem volt sértegetés; csak éppen én sohasem keverem össze az informatikust például a szoftveressel. (Az szinte mellékes, hogy az eladni kívánt termék mögötti 'adatbázisterv' a szakértelem teljes távollétéről tanúskodott.)

Hát igen. Az informatika és az informatikus a két legelcsépeltebb és leginkább tévesen használt szavunk. A számítógéppel valamilyen fokon bíbelődők ma az előkelőnek ható 'informatikus' címet öltötték magukra. Mára már az elektronikus ill. az írott sajtó is elfelejtette a 'számítástechnika' szót, és azt végleg felváltotta a mutatósabb 'informatika' kifejezéssel. Persze mindenki behódol ennek az újabb divathullámnak is. A szavak - mint a sajtóban mindig - játékszerekké váltak. Az informatikának és szolgárolányának, a számítástechnikának a lényege, így szép csendeskén összemosódott.



1.8 ábra: A számítástechnikus és az informatikus
(Az egyik lefelé néz a számítógépre, a másik felfelé a feladatra.)

Mivel feltételezzük, hogy az informatika és a számítástechnika viszonya a fentebbi elvi magyarázatok dacára még továbbra sem világos a felhasználó, a fejlesztő és a teljesen külső szemlélő számára, egy viszonylag gyakorlatias példával próbáljuk jobban megvilágítani ezt a

lényeges kérdést. Közelebbről a szoftvervásárlás manapság nem ritka gyakorlatára fogunk hivatkozni. (Előre le kell szögeznünk a félreértések elkerülése végett, hogy elvileg egyáltalán nem vagyunk semmilyen jó szoftvertétel ellenségei: nem a beszerzés ténye, hanem annak módja szolgáltat ürügyet a következő gondolatsornak.)

A vezető szoftvert - nem pedig rendszert! - vesz, vetet. Információs rendszert ugyanis egyáltalán nem lehet venni. Most csak egyetlen tényezőt említünk, mert nem akarunk az információs rendszer definíciója elé futni. Ennek a rendszernek része az ember is. Azt pedig senki sem vélheti komolyan, hogy rendszert vesz - emberekkel együtt. (A pontosság kedvéért említjük, hogy a cégvásárlás valóban információs rendszervételt is jelent. Ha valaki megveszi az X vállalatot, akkor 'hozományként' kapja annak információs rendszerét is...)

A vezető tehát szoftvert vásárol. Megnézi, hogy szépek-e a képernyők (mindig azok). Megvizsgálja, hogy helyesen dolgoznak-e a beígért műveletek (általában elboldogulnak). Rábök az Fx funkciógombra (az is működik). Figyelembe veszi az egyéb általános számítástechnikai szempontokat (pl. biztonság). Majd ezzel a 'rendszervizsgálat' be is fejeződik.

A valódi informatikus gondolkodásmódja egészen más. Azért, mert pontosan tudja, hogy pl. a Brémában készült szoftver csak a brémai muzsikások kottájára működik. Ezért annak valóban sikeres alkalmazásához az lenne szükséges, hogy a magyar muzsikások is ugyanabból a kottából zenéljenek. Csak pár apróságot említve: az itthoni papírok megegyezzenek az ottaniakkal; ugyanazok legyenek itt is az azonosítók, a kódok, a részismeretek értelmezései, mint ott; a papírokat ugyanolyan fokon képzett és ugyanolyan fokon kultúrált emberek töltsék ki; a brémaival azonos eljárásokkal; legyenek ugyanazok a technikai segédeszközök - sokszorosítás, telefon stb. - által nyújtott körülmények; a termék biztosítsa az egyéb rendszerrészek felé mutató külső kapcsolatokat; etc. etc. Amint látjuk, a szoftver az egy dolog, a rendszer pedig annál százszorososan több.

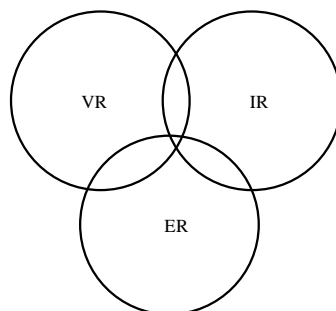
Itt és most nem arról van szó, nem az a lényeg, hogy a brémai fuvarlevelet a pesti ügyintéző sohasem fogja tudni kitölteni a brémai módjára, jóllehet a vétel ezt (is) feltételezné. Még csak azt sem említjük, hogy a saját rendszerünknek az idegen szoftver szerinti átalakítása éppen úgy nagy nehézségekbe ütközik, mint a szoftvernek az eladóval történő átalakíttatása a mi igényeinknek megfelelően.

Nem a szoftvertételi problémák ecsetelése, hanem a szemléleti hiányosságok feltárása a feladatunk. A valódi informatikus ugyanis információs rendszerben, nem pedig szoftverben gondolkodik. Nemcsak számítógépekben, eszközökben, hanem elsősorban feladatokban.

Ez az egyetlen és helyes szemlélet kulcsa: a feladat megértése és informatikai elrendezése. Az eszközök, az eljárások, az azonosítók, a papírok, a kódok és a miegymások kicserélése szoftvertétel esetén, feltalálása saját fejlesztéskor nem olyan nagy 'kunszt'. A 'fejek', a gondolatok - a vezetői igények és a felhasználói nézetek - átformálása az igazi informatikai kihívás.

Az egyetlen helyes és célszerű informatikai szemléletnek megfelelően a valódi informatikus három lényegben és azok viszonyaiban gondolkodik. Ez a három dolog pedig nem más, mint maga a **valóság** (valós rendszer, VR), kiegészítve a róla alkotott **ismereti képpel** (információs rendszer, IR), továbbá a rendelkezésre álló **technikai erőforrások** halmazával (**eszközrendszer**, ER). Így például maga a közgazdasági értelemben vett anyaggazdálkodás egy olyan feladat (VR), amit úgy lehet kiszolgálni ismeretekkel (IR), hogy rendelkezésre állnak a szükséges számítógépes eszközök is (ER).

Három egymással átfedő, egymást feltételező, de egymástól részben független rendszerről (VR-IR-ER) van szó. Az eszközök karbantartása külön feladat akkor is, ha a másik két lényeg nem változik. Rengeteg lehetőség adódik az ismeretek kezelésének a tökéletesítésére akkor is, ha az eszköz a régi marad és a feladat maga sem változik.



1.9 ábra: A három rendszer viszonya

Azt azonban el kell ismernünk, hogy „est modus in rebus”, tehát mindig akad valamiféle módbeli változás a valós-, az információs- és az eszközrendszerben. Éppen emiatt használjuk a **környezet** kifejezést egy másik értelemben is. Azon nemcsak a rendszer határain kívül és belül eső dolgokat értjük, hanem sokszor a körülményeket - az alkalmazási és az eszközbeli belső feltételek együttesét - is. Mert nem közömbös például, hogy egy vagy több, netán eltérő feladattal bíró felhasználó alkalmazza-e a rendszert (**alkalmazási környezet**) illetve egy- vagy többféle eszköz szolgálja-e a célokat (**eszköz-környezet**).

Ezt a gondolatsort lassan be kell fejeznünk. Az információs rendszer helyes megközelítéséhez az szükséges, hogy egymással harmonizált módon lássuk a valós feladat, az informatikai megoldás és az alkalmazott eszköz lényegét és viszonyát. A számítástechnikus akkor téved, amikor csak az eszközökre próbál koncentrálni. A felhasználó akkor hibázik, amikor kizárólag a feladat lebeg a szeme előtt. Végeredményben pedig mindig a vezető a bűnös. Mindegy, hogy egy picinyke vagy egy óriási vállalkozást vezet-e. Fel kellene ismernie, hogy a cél és a megoldás között valami 'hézag' van. Olyan, amelynek a kitöltésére sem az úgymond laikus felhasználó, sem az úgymond profi számítástechnikus nem alkalmas. Ők nem beszélnek egymás nyelvén. Szükség lenne olyan valakire, aki a két nyelvet egyaránt érti: a felhasználóit és a számítógépeset is. Vagyis szükség lenne igazi - informatikusra. Csakhogy ilyen kevés adatik - és akkor sem látják ma még nagy szükségét. Inkább az újabb szoftverziókban bíznak...

1.7 RENDSZER ÖSSZEFOGLALÓ

A tudatos ember mindig rendszerben gondolkodik. Többé-kevésbé pontosan meghatározza vizsgálatát tárgyát. Az órásnak és a vízvezeték-szerelőnek e téren könnyű a dolga, mert kézzelfogható dolgokkal foglalkozik. Ezzel szemben az információs rendszereket illetően a fejekben káosz uralkodik. Elsősorban azért, mert az információ nem annyira kézzelfogható - nagyon is személyekhez kötött - dolog. Erről majd a következő fejezetben szólnunk bővebben.

Csak annyi bizonyos, hogy a valamennyire is összetett rendszereket mindig részekre kell tagolni ahhoz, hogy megérthessük a lényegüket. Ez a tagolás nem szétdarabolást jelent. A rendszerrészek behatárolása nagyon is tudatos eljárás. Eldöntjük, hogy mi és mi nem tartozik vizsgálatunk körébe. Ezt úgy tesszük, hogy a tagadást ('ez nem') az igenléssel ('ez igen') párosítjuk. Azért, mert meg kell vonnunk a határokat, ámde a határok nemcsak szétválasztanak, hanem egyben össze is kötnek.

A rendszer és az al/részrendszer (ill. rendszerrész) fogalmak ekként szorosan összetartoznak. Elvileg mindenki azt tekinti rendszernek, amit ekként vizsgálni akar. Ámde a pusztán technikai - eszközbeli - elhatárolások nem adhatnak kellő útmutatást. Elvégre a hosszúságot sem minősíti az, hogy fa- vagy fémcölstokkal mérjük...

Roppantul nehéz megérteni a 'kettőben is egy' elvét. Az ember - egy. Akkor is az, ha teste és lelke eltérő módon működik. Az információ a valósághoz képest másodlagos dolog; az információ a valóságtól eltérően, másként működik; ámde a valóság sem létezhet a róla alkotott információs kép nélkül. Ezért bár igaz az a kitétel, hogy a valóság elsődleges, a róla alkotott információ pedig másodlagos, ezt a megállapítást nem szabad akarnok módon túlhajszolni. Nemcsak a fizikai valóság létezik; a róla alkotott szellemi kép is valós dolog.

A számítástechnika csodálatos fejlődése általános értetlenségre talált. A célok lassan-lassan elszürkülnek a technikai adottságok árnyékában. Az 'alkalmazás' és a 'szoftver' jelszavak mögött üzleti célok bújnak meg. Időközben az informatikát - elkődösítő módon - a számítógép-alkalmazással vették azonos kalap alá. Ezért manapság nem a célok átgondolása, hanem az eszközök közötti válogatás vált az elsődleges feladattá. A valós kihívás (VR), az informatikai megoldás (IR) és az eszközök (ER) harmonizálása ma nem látszik alapvető célként. Az ismereteket 'gépre kell ültetni' - mint a hetvenes években -, mindig a legkorszerűbbnek vélt eszközök kényszere mellett.

Pontunk zárásaként óvnunk kell az olvasót az elkövethető hibáktól. Manapság egyre nagyobb divat az 'informatika rohamos fejlődéséről' papolni. Aki ezt teszi, az nem valódi informatikus. Régóta ismert tények ugyanis az alábbiak:

A **hardver** fejlődése rohamos. Számátalan beszámolót olvashatunk az újabb meg újabb áttörésekről; meghertzekekről, bájtokról, eszközökről. Már a hetvenes években is tudtuk, hogy az alkalmazások 2-3 százalékához kevés a hardver, ám a 'vas' 97-98 százalékát akkor sem tudtuk kihasználni. Mindazonáltal a hardver fejlődése - egyes becslések szerint - tíz-tizenöt évvel haladja meg a szoftverét.

A **szoftver** fejlődése az első ránézésre még látványosabb. Csak az a baj, hogy a mai szoftverek némelyike még kipróbálatlan, hibás, a céljának meg nem felelő. Egyébként az újabb programeszközök ragyognak, százféle kényelmet nyújtanak a felülettel és nem a feladattal törődők számára - persze nem kis többlet-hardver árán. Mivel pedig nem a feladat áll a figyelem központjában, nem is csoda, hogy a szoftver fejlődése cirka húsz évvel előzi meg az alkalmazását.

Ma már akadnak egészen jó adatbázis-kezelők és egyéb kellemes - az IR-t jól kiszolgáló - szoftverek. E programcsomagok némelyike annyira összetett, hogy a laikus felhasználó hozzájuk sem tud nyúlni, de még a legprofibb szervező ill. programozó is csak a képességek pár százalékát képes megismerni és értelmesen alkalmazni. Mindennek az az oka, hogy az **orgver** - az ismeretek szervezésének a módszere - a legújabb álhírek dacára is messzire elmarad az előző két tényező rohamos fejlődésétől. Magyarul: egészen egyszerűen nem tudunk jó információs rendszereket és jól készíteni.

A hetvenes években még az eszközök hiánya miatt maradt hátra az orgver. A nyolcvanas években (a más szempontokból pozitív hatású) PC-mánia közepette senki sem gondolt arra, hogy szervezési módszerre is szükség van. Manapság a fejlesztési eszközök csodálatos változatossága és fejlett képességei tévesztenek meg egyeseket. Az X vagy az Y fejlesztőrendszer birtokában nagyon sokan úgy érzik, hogy majd az eszköz leveszi a vállukról a feladat átgondolásának a terhet.

Mindennek egyetlen egy és végső oka van: a megfelelő szemlélet hiánya. Ez pedig abból fakad, hogy az iskolákban, a szakkönyvekben, a folyóiratokban, a konferenciákon, a médiában stb. nem az informatikai profik *nevelésére*, hanem a számítógép-technikusok *kiképzésére* irányulnak az erőfeszítések. Csak az a cél, hogy az embereket betanítsák a legújabb eszközök mindenféle bűvöletére. Az egyáltalán nem feladat, hogy megismertessék velük az információs rendszer csodálatos belső rejtelseit és hogy belőlük valódi informatikusokat neveljenek. Ez nem is csoda, hiszen az IR fejlesztési módszerének, az orgvernek az oktatása lényegesen nehezebb feladat, mint a technikai kiképzés.

Az IR helyes szemléletének és a megfelelő módszereknek a hiányában nincs és soha nem is lesz lehetőség jól működő információs rendszerek kialakítására. Ezért annak, aki nemcsak programokat akar írni és nemcsak eszközöket óhajt alkalmazni, hanem az IR értelmes megfor-

málására is törekszik, annak feltétlenül meg kell ismerkednie az információs rendszereknek és fejlesztési módszereiknek a lényegével. Ezek bemutatására fogunk sort keríteni a következő fejezetekben.

ELLENŐRZŐ KÉRDÉSEK - 1

Az olvasó tegye fel magának a kérdést, hogy érti-e a rendszer, a határ és a környezetek lényegét. Világos-e számára a helyes és a helytelen tagolás elve. Érti-e az alkalmazási-, az információs- és az eszközrendszer viszonyát. Ezért az alábbi kérdésekre ne pusztán választással válaszoljon, hanem érveljen is.

- 101 Állítson fel egy sorrendet! Mi az oka annak, hogy az információs rendszerek nem annyira sikeresek, mint azt elvárnánk?
- 1 - kevés a hardver
 - 2 - tökéletlen a szoftver
 - 3 - nincsenek egységes fogalmaink
 - 4 - fejletlen a szervezési módszer.
- 102 Tekinthető-e rendszernek (I) vagy sem (N) egyetlen kis darab árusítási szövetminta?
- 103 Tegyük fel, hogy egy vállalat számlázójaként dolgozik. Ön mit tekintene rendszernek?
- S - a számlázási funkciót
 - É - az értékesítési részleget
 - V - a vállalatot.
- 104 Ön megbíz egy festőt azzal, hogy tegye rendbe kizárólag a hálósobáját. Ez az előszobából nyílik és az erkélyre van kijárata. Válaszoljon arra a kérdésre, hogy ebben az esetben mi a rendszer? Vigyázat, csalunk!
- L - a teljes lakás
 - H - a hálósoba
 - E - a három nevezett egység együttese (hálósoba, erkély, előszoba).
- 105 Az előző kérdést folytatva: helyesen gondolkodik-e a festő, ha csakis a hálóval törődik (I), vagy sem (N)? Ezután mondja meg, hogy az erkély és az előszoba minek tekinthető a hálósobához képest? Mi a rendszer határa és az miért fontos?
- 106 Egy kisvállalkozó számítógépen akarja vezetni a számláit is és az általa felhasznált anyagokra vonatkozó ismereteket is. Az alábbi állítások közül melyik igaz (I), melyik hamis (H) és melyik lehetséges (L)?
- a számla- és az anyagnyilvántartás két rendszer
 - a két rendszerrész átfed
 - két rendszerrészről van szó, de ezek viszonyát még nem ismerjük
 - a számlázási rendszerrésznek alkotóeleme az anyagnyilvántartás.

- 107 Vannak import és hazai megrendeléseink. Az alábbi megoldások közül melyik jó (J) és melyik rossz (R)?
- egyetlen monolit (tagolatlan) rendelési rendszert készítettünk
 - az importokat fontosságuk miatt külön gépen külön rendszerként kezeljük
 - két átfedő rendszerrészben gondolkodunk.
- 108 Most egy logikai feladat következik. Keresse ki az 1.4 ábrát. Nézze meg az „A” rendszert. Próbálja meg szavakba önteni azt, amit az ábra kapcsán mi nem mondtunk el. Kitöltik-e a rendszerrészek a rendszert? Mivel jár a rendszerrészekre való tagolás?
- 109 Barátomnak három ‘rendszere’ van. Egy Excel-ben írt számlázási, egy Clipper-es anyagnyilvántartási és egy Word-ben tárolt vevőkatalógus (a levelek miatt). Hány valódi rendszere van a barátomnak? Érveljen! Mi az alapvető probléma?
- 110 Az alábbi feladatok közül melyik tartozik az információs rendszer hatáskörébe (I), melyik az alkalmazásnak is nevezett valóséba (V), melyik kizárólag az eszközrendszer belügye (E) és melyik tartozik e három rendszer metszetébe (M)?
- a számítógépek karbantartása
 - az elsődleges bizonylatok szerkesztése
 - adott szervezet felépítésének átszervezése
 - egy újabb ‘Foglalkozáskód’ bevezetése.
- 111 A 109-es feladatra visszatérve saját szavaival mondja el, hogy miben látja a mai gyakorlati számítógép-alkalmazások mögötti elméleti bajokat?

2. AZ INFORMÁCIÓS RENDSZER FOGALMA

Az információs rendszer is - rendszer. Tehát minden olyan általános vonás, amit a rendszerekről korábban elmondottunk (ld. 1. fejezet), az rá is érvényes. Ezért:

Ennek a fejezetnek az a célja, hogy tisztázza az 'információs' jelző tartalmát és első közelítésben bemutassa az IR-t alkotó tényezőket.

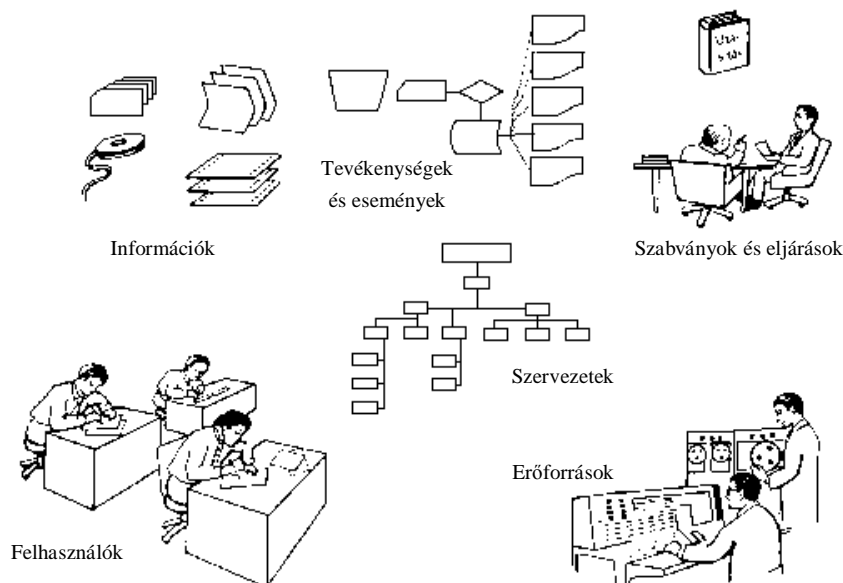
Ebben a könyvben igen tudatosan alkalmazzuk az *információs* megjelölést, és szándékosan nem *információ*-rendszerrel beszélünk. Az utóbbin értelemszerűen csakis az ismeretek elrendezését lehet érteni, márpedig az információs rendszer nem csak információkból áll.

D 2.1 Az információs rendszer

- adatoknak (információknak);
- a velük kapcsolatos információs eseményeknek;
- a rajtuk végrehajtott információs tevékenységeknek;
- az előzőekkel kapcsolatos erőforrásoknak;
- az információk felhasználóinak;
- ill. a fentieket szabályozó szabványoknak és eljárásoknak

a szervezett együttese.

A definícióban említett lényegeket majd a következő pontokban egyenként is ismertetjük. Itt elsősorban az általános vonásokra akarjuk felhívni a figyelmet. Ezek közül a legelső az, hogy az IR az ún. *inhomogén* rendszerek családjába tartozik. Inhomogénnek azokat a rendszereket nevezzük, amelyeknek tényezői illetve az azok közötti viszonyok nem egyneműek. Azzal persze nem mondunk semmi különlegeset, hogy kijelentjük: az IR inhomogén. Nem is pusztán erre, hanem inkább az IR tényezőinek és viszonyainak az igen eltérő jellegére akarjuk irányítani a figyelmet. Nem árt például észrevenni, hogy mi a felhasználókat is az IR szerves részének tekintjük éppen úgy, mint például a felhasználóknak és - mondjuk - a gépeknek (ld. erőforrás) a viszonyait! A tevékenységet szabályozó eljárások nemcsak a VR, hanem egyben az IR részei is. Ezért meghatározásunk messze nem csak adatokról, szoftverekről, *nem csak számítástechnikáról* szól.



2.1 ábra: Az információs rendszer

A hetvenes években nap mint nap találkoztunk az információs rendszereknek a különböző jelzős változataival. Ezek a tartalmatlan, célszerűtlen, pusztán csak a nagyzási kedvet tükröző jelzők még ma is lépten-nyomon előfordulnak. Ezért - egyáltalán nem kellemes - kötelességünk egy picit elelmélkedni róluk.

Az egyik ilyen jelző a **komplex**. Soha senki sem árulja el, hogy vajon mit is kell ezen érteni, de ugyancsak divat komplex információs rendszerről diskurálni. Pedig a 'komplex információs rendszer' nem más, mint rejtett tautológia. Azért az, mert minden IR összetett inhomogén rendszer, tehát eleve komplex. Nincs mód arra, hogy ezt a rosszízü kifejezést mással váltsuk fel. Ha valaki ugyanis 'nagyon bonyolultat' ért a komplex alatt, akkor azonnal fel kellene tennünk azt a kérdést, hogy mit is jelent a 'nagyon' és vajon kinek jelenti azt? Ha viszont a komplex a 'sok mindent felölelő' szinonimája, akkor ismét csak zavar támad, mert a legegyszerűbb IR is sok mindent felölel. Legjobb tehát elkerülni ezt a jelzőt, mert használatával csak a tudálékosággal leplezett tudatlanságunkra fog fény derülni.

A másik varázsszó az **integrált** (amit így kell használni: komplex és integrált.) Ezt illetően az előző bekezdést akár macskakörömbe is tehetnénk. Tautológia, mert minden rendszer integrált, a nem-integrált dolog pedig nem rendszer. A rendszer a tényezők szervezett - vagyis integrált - együttese. Az integrálás szó azt a mindennapos tényt leplezi, hogy korábban valamilyen tényezőegyüttest valójában nem rendszerként fogtunk fel, most pedig arra kísérletet teszünk. Ez a szándék ugyan örömmel üdvözlendő, ám a valódi informatikus eleve csakis integrált - azaz: rendszerezett - rendszerben képes gondolkodni.

Még az előzőeknél is jóval kínosabb fordulat a **vezetői információs rendszer** [management information system - MIS]. Mindenki erről szónokol, holott ilyen dolog önmagában nem létezik. Az informatikus számára világos, hogy a vezető éppen úgy egy felhasználó, mint a nem-vezető. A vezető részére adott információ csakis a nem-vezetők által alkalmazottakból születhet. Például: Ugyan miképpen tájékoztathatnánk a vezetőt az anyagokra vonatkozó legfontosabb adatokról, ha nem létezne anyaggazdálkodási információs rendszerünk? Az ő ismeretei tehát ebbe a rendszer(rész)be tartoznak, attól elválaszthatatlanok. Ugyanez a helyzet a tárgyi eszközökre, a bérekre, a beruházásokra stb. vonatkozó információkkal. A vezetői szintű ismeretek mindig az

alapismeretekből származtatottak, tehát azok természetéhez kötődnek, azok nélkül nem is létezhetnek. A 'mennyi prémiumot adtunk az első félévben' adatnak pedig semmi köze sincs a 'miképpen áll az X-féle anyag készlete a ma napon' ismerethez. Ezekből az ismeretdarabokból nem faragható egy olyan külön rendszer, amit 'vezetőinek' titulálunk.

A fenti meghatározásra alapozva most sorra vesszük az információs rendszer különböző tényezőit. Az információkról, eseményekről, tevékenységekről stb. rengeteg mindent kellene tudnunk. Ráadásul ezek a dolgok - nagyon bonyolult módon - még össze is függenek egymással. Így nem csoda, hogy egyelőre csak első megközelítésben szólhatunk róluk. A részletes kifejtéseket majd könyvünk későbbi fejezetei tartalmazzák.

2.1 ADAT ÉS INFORMÁCIÓ

Ezt a két fogalmat a mindennapi életben felcserélhető módon, teljesen azonos értékű szinonimákként használjuk. Ebben a tényben nincs semmi kivetendő. Az sem kizárt, hogy könyvünk további részeiben mi is ezt a gyakorlatot követjük. A további ismertetés előtt azonban határozottan le kell szögeznünk, hogy az adat és az információ messze nem azonos lényegek. Ezért nem válhat valóban kiváló informatikus abból, aki nem érti e két kategória tartalmát és viszonyát.

Intermezzo

A könyv szerzője az informatikusokkal való ismerkedését sokszor azzal kezdi, hogy nem is pontos, hanem csak kvázi-meghatározást kér tőlük az adatra nézve. Számára megdöbbentő az a tapasztalat, amely szerint a megkérdezett - gyakorló! - számítástechnikusok kilencvenkilenc százaléka (!) képtelen volt arra, hogy egy úgy-ahogy elfogadható definíciót adjon az adat lényegére. Most képzeljen el az olvasó egy olyan kőművest (vö. számítástechnikus), aki méghozzá magát nem is csak annak, hanem építésznek nevezi (vö. informatikus) és aki nem ismeri pl. a téglát vagy a beton lényegét!

A fentiek miatt az adattal ill. az információval - mint az információs rendszer jelzőjéhez a legszorosabban kötődő tényezőkkel - a többinél egy picit többet kell foglalkoznunk. Legelőször is álljon itt az adat meghatározása:

D 2.2 Az adat értelmezhető, de nem értelmezett ismeret.

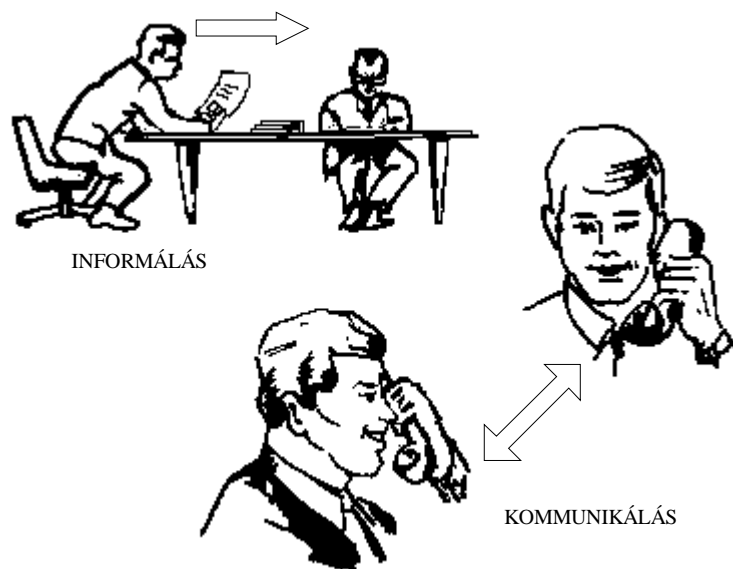
Ennek a definíciónak - első ránézésre - az a baja, hogy komplikáltnak, viszont egyszersmind valamiképpen nagyon egyszerűnek is tűnik. Magyarázatát kezdjük talán azzal, ami belőle - látszólag - kimaradt. Ez egyben alkalmat ad nekünk arra is, hogy két másik meglehetősen félrehasznált fogalmat megvilágítsunk.

Az ember tanult és társas lény. A *tanult* alatt itt most egyáltalán nem valami magasfokú iskolát végzettet, műveltet stb. kell érteni. Születésünk percétől - sőt, szerintünk a fogantatás pillanatától - fogva tanulunk, azaz ismereteket szerzünk. Már a pici baba feje is ismeretekkel tömített. Mivel *társas* lények vagyunk, az ismereteket közlések útján szerezzük meg. Ezért szellemi létünk fenntartásához a másoktól nyert információk éppen annyira fontosak, mint testi létezésünkhöz a mindennapi élelem.

E 2.1 „Az ismeret a szellemi élet kenyere.” (Kiemelés: a szerzőtől.)

A közlések két csoportba sorolhatóak. Mivel ahol ezt megteheti, ott az ember mindig téved, ezért a mindennapi életben ezt a két kategóriát sem pontosan, nem az igazi tartalomnak megfelelően használjuk. Valódi természete szerint ugyanis a közlés vagy egy-, vagy többirányú lehet. Az előbbi esetben **informálás**ról, az utóbbiban pedig **kommunikálás**ról van szó.

Megint csak nem a szavakkal való játék a célunk. Viszont vegyük észre, hogy a mindennapi életben 'tömegkommunikációs' valaminek nevezik azt a dolgot, amivel egyáltalán nem lehet 'kommunikálni', vagyis többoldalú eszmecserét folytatni. Ki kellett nőnünk a kisgyermek-korból ahhoz, hogy rádöbbenjünk: a rádióknak nem lehet visszabeszélni...



2.2. ábra: Informálás és kommunikálás

Mármost mi köze mindennek az információs rendszerhez? Rengeteg. Számos IR-ben ugyanis nem a kommunikációra, hanem kizárólag az egyirányú közlésre, az úgymond informálásra törekednek. Visszajelzés - kommunikáció - hiányában viszont aligha tudhatják a 'kommunikálók', hogy az ismeretek valóban elérték-e bennünket. És ezen a ponton visszakanyarodhatunk az adat lényegéhez. Mert pl. egyes általunk kapott csekkiken megjelennek olyan rejtélyes jelsorok, amelyek - legalábbis a számunkra - nem értelmezhető ismeretek, tehát nem is adatok!

Az ember testből és lélekből álló, egyidejűleg és elválaszthatatlanul fizikai és szellemi természetű lény. Ezért teljesen világos, hogy minden egyes közlésnek két momentuma van: egy formai és egy tartalmi. A **formai** mozzanat a közlés módja, ideértve annak eszközét is. A **tartalmi** aspektus maga a közlés által átvitt ismeret. Egy egészen egyszerű példa jól szemlélteti a kettő viszonyát. Ha valakit megkérdezik, hogy szereti-e a mákos tésztát, akkor az bólinthat ill. csóválhatja a fejét és (egyidejűleg) azt is mondhatja, hogy 'igen', vagy 'nem'. A forma tehát a fejmozdulat vagy a beszéd, a tartalom pedig a formával kifejezett lényeg.

Az adat - értelmezhető ismeret. Az értelmezhetőség feltételezi a tartalom és a forma egységét, jöllehet az előbbi az utóbbitól bizonyos mértékig független. Ha nemet intek, leírom ezt a szót: 'nem', ha a szemem nem úgy villan, mint várnád, már értelmezhető ismeretet kapsz, a formától függetlenül. Ha viszont egy bulgár a magyar módjára nemet int, akkor jó tudni: valójában 'igen'-t mondott! Tehát a tartalom nem abszolút független a formától.

Intermezzo

Az információs rendszerek egyéb tényezőinek a magyarázata messze nem lesz ennyire hosszadalmas. Itt ismételt elnézést kérünk a terjedelmesebbnek látszó 'filozófiai fejtegetésért'. Azonban az adattól az információig vezető út hosszú, és - a költő szavaival szólva - „ezt ha nem érted, hagyj fel...”. Mármost ama csábos gondolattal, hogy valaha is jó informatikus válik belőled.

Most pedig nézzük meg az értelmezhetőség *feltételeit* ill. vizsgáljuk meg azt is, hogy milyen *lépések* során jutunk el a tényleges értelmezésig. Ugyanis az ismeret fogadása után több ilyen fázison keresztül érünk el az információig, noha ezek a szakaszok a pillanat töredékei alatt játszódhatnak le bennünk. Ezen lépések természetét egyenként is ismernie kell a vérbeli informatikusnak, ha az a célja, hogy értelmezhető ismeretet szolgáltatson másoknak.

Az ismeret megszerzésének három momentuma van. Sorrendben: az első az érzékelés, a második az észlelés, a harmadik pedig a felfogás. Talán nem árt, ha egy-egy kis gondolatfoslány keretében kitérünk e komoly lényegekre.

Hiába írjuk fel egy papírra vagy egy táblára azt, hogy 'igen'. Hiába mondjuk ki ezt a szót. Vannak olyan embertársaink, akik az *érzékelés* szempontjából nálunk egy picit kevésbé szerencsések. Így a csökkentlátók vagy -hallók számára a leírt ill. a kimondott közlés fizikailag nem értelmezhető. (Az efféle testi bajokat nem szabad lebecsülni, de - az informatika terén - nem is illő túldimenzionálni. E mű szerzője meglepetéssel tapasztalta, hogy - a csúnyább szavakat használva - a vakok és süketek mennyivel fogékonyabbak az úgymond átlagos emberekénél a mélyismeretekre. Kár, hogy erre a tényre nem figyelnek fel az illetékesek...)

Az informatikusnak törekednie kell az érzékelhető ismeret-megjelenítésre. Az érzékelés legelső és legalapvetőbb feltétele az ismeret *fizikai* jelenléte, amin idő- és térbeli érzékelhetőséget kell érteni. Egészen csacska történetek igazolják, hogy ezt a követelményt a mindennapi életben százszor is megsértjük.

T 2.1 „Bélusom, *tegnap az X-közértben olcsóbb volt a kolbász.*” (Sokat érek ezzel az 'információval', amit a szomszéd közöl velem ma). „Olvastad az újságban... De hiszen drágám bevitted a *hivatalba!*”

Mindannyian bosszankodunk azon, ha csekket kapunk a szerződésünk lejártá után akár több hónappal is (idő). Senkinek sem kellemes, ha hivatalos papírjait elcímezik (tér); a neki szóló máshová kerül, a megkapott iraton viszont másnak a neve áll. Az adat - értelmezhető ismeret. Érdemi, használható információ csak akkor válhat belőle, ha az informatikus az időre és a térre, az érzékelhetőség eme két fizikai feltételére is gondol.

T 2.2 „Tudod Béla, hogy mit csinálunk a *hétfőn reggel talicskán betolt tablókkal? Elvisszük a műhely sarkába, majd reggel kidobjuk.*”

Mondta ezt 1975-ben egy számítógépet (!) is gyártó híradástechnikai vállalat műszakvezető mérnöke. Igaza volt. A tablók ugyanis azt tartalmazták, hogy az előző napon (!) milyen gépek álltak le és emiatt miképpen kellett átprogramozni az aznapi termelést. Persze mire ezek az 'információk' megérkeztek, addigra már az átprogramozás réges-régen meg is történt.

Talán az *észlelést*, mint második momentumot a legnehezebb elmagyarázni. Azonban a mindennapos példák talán itt is kiségitenek bennünket. Hányszor, de hányszor fordul elő velünk, hogy rákérdezzünk partnerünkre: „Mit is mondtál?”. A közlést érzékeltük, de nem észleltük. E ponton gondolhatunk arra a számtalan szerződésre is, amelyet aláírtunk anélkül, hogy a záradékokat igen figyelmesen elolvastuk volna. Az információk természetét jól ismerő vállalatok - így

például a hírhedt csomagküldő cégek - roppant nagy százaléka éppen arra játszik, hogy valamit érzékelhessünk, de ne észleljünk.

Az egyik ravasz megoldás az, hogy a papír egy eldugott helyén bolhabetűkkel közlik a lényegét. (Persze az értelmes ember ezt már régen tudja; ezért számára e trükk nem jelent újdonságot.) Egy másféle kísérletezgetés a kódosításra az, hogy a papírt - pl. egy csekket - túlsúfoljuk olyan száz 'adattal', amelyek dzsungelébe elrejtjük a valóban fontos ismeretet.

Vegye észre az olvasó, hogy az ismeret fizikai **elrendezése és megjelenítése** az észlelés alapvető feltétele. Hiába érzékelhető a fontos adat, ha lényegessége nem tűnik fel sem a helyéből, sem az írott formájából.

T 2.3 Egy pénzügyi intézményben a hibásan feladott tranzakciókat - ma is - egyetlen közös hibalistán küldik vissza a rögzítőknek, az ügyintézőknek. A hibák jelentős százaléka megismétlődik. Miért?

Az adott esetben az ügyintézőnek ezernyi hibátételből kell kikeresnie azokat, amelyek csak rá tartoznak. Az adat fizikailag rendelkezésre áll, csak hogy nem eléggé észlelhető: el van rejtve a lényegtelen dolgok közé. Az ügyintéző 'elnézi' a tételt - és megismétli a hibás feladást.

A számítástechnikus - éppen technikai okokra hivatkozva - megteheti, hogy az ismereteket a képernyőkön, a papírokon - a számítógépes kimeneteken - illetve a bizonylatokon érzékelhető, de nem észlelhető módon rendezze el. Mert ugyebár a képernyő mérete, meg a papír alakja - halljuk a kifogásokat. Az informatikus viszont tudja, hogy nem a papír drága, hanem az észlelés miatt elvesztegetett idő a valóban jelentős költségtényező. Ezért törekszik az észlelhetőség javítására.

T 2.4 Amerikai barátom szerint az ottani újságok nem manipuláltak. Korrekten közlik az ismereteket. Roppantul megsértődött, amikor ellent mertem mondani neki.

Persze az amerikai újságok általában valóban roppantul 'objektívek'. Amit leírnak, az mindig úgy is van. Csak éppen olykor fontos híreket egyáltalán nem közölnek (érzékelhetőség), máskor pedig az első oldalon öles betűkkel írják meg azt, ami nem is fontos, a huszadik oldalra dugva el a lényegeset (észlelhetőség). Amúgy egyébként egyáltalán nem manipuláltak...

Könyvünknek egyáltalán nem az informatikai manipuláció - mi több: bűnözés - a tárgya. Azt azonban nem szabad elfeledni, hogy éppen az informatikai latrok a legjártasabbak az ismereteknek az észlelhetőségen alapuló manipulálásában. A becsületes informatikusnak nem ártana elsajátítani tőlük egy pár trükköt.

Az ismeretszerzés harmadik momentuma a **felfogás**, a megértés. Hányszor, de hányszor mondjuk a mindennapos életben, hogy „értem, értem csak éppen fel nem foghatom”. Ez a kiszólás ugyan egy pici ellentmondást tartalmaz, mivel a felfogás és a megértés ugyanazt jelenti, de ez egyelőre ne zavarjon bennünket. Most tessék a következő ábrát, jelet, valamit jól megfigyelni, és eldönteni azt, hogy mit is jelent:



A fentebbi 'rajz' érzékelhető és észlelhető. Érzékelhető, mert hiszen látható. Észlelhető, mivelhogy külön is felhívtuk rá a figyelmet. Valószínűleg kevesen akadnak, akik tudják, hogy valójában mit is mond. (Még az is megtörténhetett, hogy netán a szerző nem pontosan 'rajzolta'

le a lényegét, mert jó huszonöt éve annak, hogy japánul próbált tanulni.) A habozáson, kételyeken csak egyeseket vezet át, ha ide latin betűkkel is leírjuk: ‘nyippon’ vagy ‘nyihon’. Még további segítségként eláruljuk, hogy ‘a felkelő nap országáról’ van szó. Ugye, hogy így már ismert, így már felfogható? A rajz és a két kifejezés egyaránt Japánt jelent.

Most vegyünk elő egy másik példát:

‘A dunaújbelai Boldogság térszben a kukorica hozama 10 mázsa/hektár.’

Az olvasó nyilván érzékelte és észlelte ezt a közlést. De vajon felfogta-e? Nem használtunk olyan kifejezéseket, amelyek az átlagos képzettségű felnőtt, magyar ember számára ismeretlenek lennének. A példamondat felfogható. Éppen ezért sokan hajlanak arra, hogy kijelentsék: a mondatban közölt ismeret - információ. Pedig nem az, csak adat. Érzékelhető, észlelhető, felfogható - azaz értelmezhető - ismeret. Értelmezhető, de nem értelmezett. Mert lássunk csak egy ellenpéldát. A rádióban bemondják a következő hírt:

‘Újabb információink szerint ödémban meghalt Pál Pali okarinaművész.’

A médiában oly gyakran alkalmazott ‘királyi többséssel’ (információink) most ne foglalkozzon az olvasó - az csak egy rossz szokás gyümölcse. A lényeg az, hogy a közlés egyáltalán nem információt, hanem csak adatot takar. Miért?

Péter Peti a rádiót figyelve - a hozzá nem értők szerint - információt nyer. Az persze letagadhatatlan, hogy ő eddig még az életében nem hallott Pál Paliról. Az sem vitás, hogy az okarinát ezen mai zenés tornák - na, anyjuk, hogy is hívják őket? - egyikének tekinti. És csak a biztonság okáért kérdezi meg a szomszédját, hogy vajon Ödéma Észak- vagy Dél-Amerikában található-e?

Nesze neked információ, nesze neked informálás! Péter Peti érti, érti, csak épp fel nem foghatja a közlést. A felfogásnak ugyanis az a *szellemi* feltétele, hogy a közlő (esetünkben ez a rádió) a befogadóval (példánkban: Péter Peti) azonos nyelvet beszéljen. A szónak a valódi és az átvitt értelmében is. A példamondat Péter Peti részére ‘japánul’ hangzik, mert nem ismeri annak alapvető fogalmait.

A számítástechnikus megteheti, hogy - technikai korlátokra hivatkozva - nem-érthető, csúnya, kódolt, jelképes stb. ismeretdarabkákkal kínozza a felhasználót. Az informatikus viszont arra törekszik, hogy érthető - azaz felfogható - legyen a közlése. Ehhez képest meglepő, hogy az átlagos állampolgár nap mint nap kap olyan számítógépes papírokat, amelyekben hemzsegnek az ismeretlen fogalmak.

Kizárólag a ‘németesek’ kedvéért ideírunk egy egészen hétköznapi fogalmat, amit a számítógépes termelésirányításban használnak. Visszajelzést nem kérünk, de azért kíváncsiak vagyunk, hogy hányan képesek megfejtetni ezt a rébuszt:

MERGERÜBERSICHTSTEJLEVERWERDUNGSRACHWEIS.

A számítástechnikusok egy része a ‘laikus’ előtt azzal akar villogni-csillogni, hogy érthetetlen kifejezéseket használ. ‘Schwyterdütsch’ nyelven beszél. Az informatikus a világosságra törekszik, mert csak a felfogható ismeretből válhat értékes információ. Amelynek a definícióját már könnyű megadni:

D 2.3 Az információ értelmezett adat.

Csak ennyi lenne a meghatározás? Mindössze négy szó? Hiszen könyveket írnak az információ lényegéről! Nincs itt valahol tévedés?

Nincs. Ez a négy apró kis szó ugyanis roppantul súlyos mondanivalót takar. Az ismeret-szerzésnek a negyedik - végső - momentuma az **értelmezés**. Ennek a mozzanatnak az elvi megértése nem jelent gondot. Viszont a gyakorlatban való érvényesítése nem könnyű feladat.

Az értelmezés az a szellemi tevékenység, amelynek során az újonnan kapott ismereteket a korábbiakhoz kötjük. Mindez egy pillanat alatt, nem is tudatosan zajlódik le bennünk. Szinte ösztönösen tudjuk, hogy mi az a 'hektár', és nagyon fárasztó lenne, ha e szó hallatán mindig felélesztenénk magunkban eme területi mértékegység pontos meghatározását. A hektár - az hektár és kész.

Mármint a szerencsétlen Pál Pali sorsára vonatkozó közlés mindenki számára adat, vagyis értelmezhető, de nem értelmezett ismeret. Péter Peti csak annyit tud a 'hírből', hogy valaki meghalt. Számára ez egyáltalán nem információ. Több ok miatt sem az. Egyrészt azért nem, mert ezt a közlést nem tudja más ismereteihez kötni. Másrészt azért nem, mert erre nézve nincs is benne semmi vágy. Egyes meghatározások szerint (ld. alább) az információ valamiféle 'bizonytalanságot szüntet meg'. Az adott esetben bizonyosak lehetünk abban, hogy Péter Petiben Pál Pali halála semmilyen bizonytalanságot nem szüntetett meg.

Megismételjük: az információ mindig az aktuális adatnak - közlésnek - a régi ismeretekhez való kötését jelenti. Ez több dolgot jelent. Ha nincs mihez kötni az adatot, mert nincs régi ismeret (nem tudjuk, hogy ki Pál Pali, ill. nem ismerjük a mázsa vagy a hektár fogalmát), akkor információ sem születhet. Akkor sem, ha az összekapcsolás egyszer már megtörtént. Az információ mindig **új** ismeret. Ha itt most elmondjuk, hogy e könyv szerzője HB, akkor az senkinek sem jelent új ismeretet, információt, mert hiszen ezt a dolgot már régen tudja.

Az információ akkor születik, amikor gondolkodni kezdünk a közlés, az adat tartalmán. Most térjünk vissza a téesz példájára! Hm, nem ismerek 'Dunaújbéla' nevű helyet. Unokáink már azt sem fogják tudni, hogy mi is volt az a 'téesz'. És vajon mi az a 'mázsa', ez a már nem-hivatalos mértékegység? A 'tíz' az roppant kevés, akkor is, ha...

Na látják! 'A tíz az roppant kevés akkor is, ha...' - ez az információ. A leszűrt ismeret, a következtetés, az emberi agyon áteresztett gondolat. Mármint annak, akiben élnek (halvány) elképzelések arról, hogy Magyarországon illetve amúgy általában milyen szokott lenni a kukorica átlagtermése. Nem maga a 'tíz', hanem a 'kevés' az újszerű tartalom. Azt pedig nem az adat, hanem az értelmezett adat árulta el.

A fentiekből le kell vonnunk egy következtetést. Az adat - az értelmezhető, de nem értelmezett ismeret - lehet **mennyiségi** természetű is. Viszont az információ valamiképpen mindig **minőséget** is jelent. A 'tíz mázsa' - mennyiség. A 'kevés' - minőség. A 'Pál Pali meghalt' ismeret - száraz tény. A 'de kár', a 'végre', a 'na és' reflexió - a tény interpretációja.

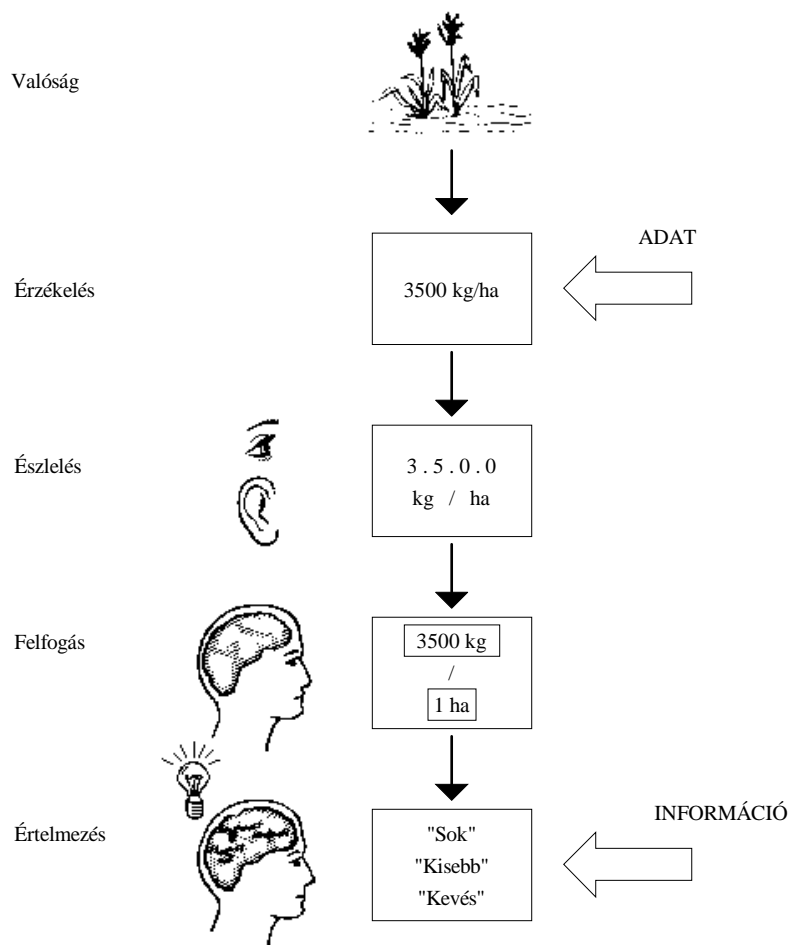
Intermezzo

E könyv szerzője rettentően sokat tanult *Shannon*-tól, az informatika egyik - ha nem az egyik legnagyobb - mesterétől. Most mégis meg kell picit tagadnia. A mester az információt matematikai - mennyiségi - módon akarta behatárolni és mérni. A felvetett kérdésre adott igen vagy nem válasz szerint egy bitnyi (a **bit** a **binary digit** - magyarul: bináris szám - rövidítése) határozatlanságot szüntet meg a kérdezőben. Szerinte az információ = megszüntetett határozatlanság.

Nézeteit - sajnos - nem tudjuk osztani. Nem csak azért nem, mert sokszor úgy is nyerünk információt, hogy korábban nem is volt bennünk határozatlanság. A fene volt kíváncsi a dunaújbéla téesz gabonahozamára, most mégis látjuk, hogy ugyancsak silány volt a termés. Bajunk van a „fekete-fehér, igen-nem” játékkal is. Az ember nem bitekben gondolkozik. Hogy szeretem-e a tökfőzeléket? Nem is tudom. Lehet, hogy jövő szerdán, no meg, ha a nejem főzi és sok-sok tejföllel. Hol van itt a bit, a mérhetőség, a mennyiség - a matematika?

A matematikai információ-elmélet - mondjuk ki kerek perez - embertelen. Ha megkérdezik Jucikát, hogy szereti-e a tökfőzeléket és szereti-e Lajost, akkor a matematikusok szerint a két kérdésre adott igen ill. nem válasz azonos mértékű határozatlanságot fog megszüntetni - Lajosban. Ítéljen el az olvasó, ha mi azt merjük mondani, hogy ez hamis, embertelen, téves felfogás.

Mindez nem a matematika elleni kirohanás. A mennyiségi vizsgálatoknak is nagy szerepük van az információs rendszerek kialakításában, sőt. Azonban azt is tudomásul kell venni, hogy az információ - elsősorban minőség.



2.3 ábra: Az adat információvá való értelmezésének a mozzanatai

T 2.5 Hirdetés: „Az információs rendszer fejlesztéséhez matematikus-programozókat keresünk.”

E könyv szerzője az információs rendszer fejlesztéséhez nem matematikust és nem programozót, hanem informatikust keresne. De vajon hol találna?

Ezt a pontot lassan le kell zárunk; amit nem tehetünk meg anélkül, hogy ne utaljunk a mindennapi szóhasználatból fakadó további káros következményekre. Az ember lépten-nyomon azt hallja, hogy a számítógépen információt tárolnak, a számítógépen információt kezelnek, a számítógép információt állít elő stb. Ez az elvileg szakszerűtlen megfogalmazás nem kell, hogy bántson bárkit is. Már tudjuk, hogy az információ olyan roppantul személyes dolog, ami kizárólagosan csak az ember fejében születhet az értelmezés által. A számítógép nem ember, sohasem lesz az, sohasem fog tudni gondolkodni, sohasem lesz lelke, szelleme. Sohasem fog információt tárolni, kezelni, előállítani. Legfeljebb csak adatot.

Kedveljük a számítógépet. Sokkal kevésbé a számítástechnikusoknak azt a bizonyos csoportját, amelynek tagjai valóban azt hiszik, hogy a számítógépen információkat tárolnak és kezelnek. És éppen emiatt a hamis tudat miatt nem is törekednek arra, hogy a számítógépeken olyan adatokat tároljanak és kezeljenek, amelyek az ember által könnyen és gyorsan információvá értelmezhetők.

T 2.6 1974-ben egy műszerteknikai cég vezetője így szólt hozzám: „Béla, én nem akarok adatfeldolgozást: én vezetői információs rendszert akarok”.

Akkor még nem tudtam elmagyarázni a számítógépes eszközöket (!) is gyártó vállalat második emberének, hogy a számítógépen csakis adatok lehetnek, éppen ezért azon csakis adatfeldolgozás végezhető. Mivel az információ az ember által értelmezett adat, a számítógépes adatfeldolgozás akkor tölti be jól a szerepét, ha éppen ezt az értelmezési folyamatot könnyíti meg a számunkra. Magyarul: arra kell törekedni, hogy a számítógépen értelmes adatfeldolgozással minden egyes felhasználó részére személyesen, jól előkészítetten, megfelelően tált és éppen ezért könnyen emészthető - információvá értelmezhető - adatsorokat készítsünk.

Ne feledjük, hogy az információ a szellemi élet tápláléka. Tehát semmi furcsa sincs abban, hogy fentebb az 'emészthető' kifejezést használtuk. Az ismeretek ugyanis többé-kevésbé 'elő-emésztett' módon kerülnek a számítógépre illetve jutnak el a számítógéptől hozzánk. A szerző már korábban is a **feldolgozási fok** kifejezéssel illette az ismeretek adott szintű elrendezésének a mértékét. Ennek a dolognak a lényegét már a következő pontban kell megvilágítanunk. Azért, mert a feldolgozás - az ismeretek előkészítése és tállása - már átvezet bennünket az információs rendszer tényezőinek a második csoportjához.

2.2 TEVÉKENYSÉG ÉS ESEMÉNY

Az alcímhez két dolgot kell előzetesen megjegyeznünk. Az egyik az, hogy itt mi természetesen az **információs** tevékenységekben és eseményekben vagyunk érdekeltek, azonban a rövidség kedvéért ezt a jelzőt a továbbiakban elhagyjuk. A másik az, hogy a két faktort nem lehet egymástól függetlenül szemlélni, mert az esemény és a tevékenység egymáshoz képest **relatív** fogalmak. Ezért is kell azokat egy alcímszó alatt tárgyalnunk.

Az információkkal kapcsolatos **tevékenységek** roppantul sokfélék lehetnek. Ide tartozik a tömeg-tájékoztatás és a távközlés éppen úgy, mint a sokszorosítás. Nem lehet feladatunk ebben a könyvben az, hogy a széles társadalmat érintő vagy éppen az aprólékos technikai munkát jelentő ténykedéseket mind-mind áttekintsük. Ezért a tevékenységnek egy szűkebb fogalmát kell használnunk:

D 2.4 **Tevékenységnek az adatok kezelését és előállítását célzó illetve az előbbieket vezérlő műveletek szervezett egységét tekintjük.**

Nem túl könnyen érthető fogalomról van szó. Ezért egy megrendelési példán keresztül próbáljuk megvilágítani a meghatározás elemeit és azok viszonyait. A kezelés és az előállítás együttese nem más, mint a hétköznapi értelemben vett **adattfeldolgozás**. Pl. a megrendelési rendszerben megrendelések - pontosabban: a megrendelési adatok - feldolgozása folyik.

Intermezzo

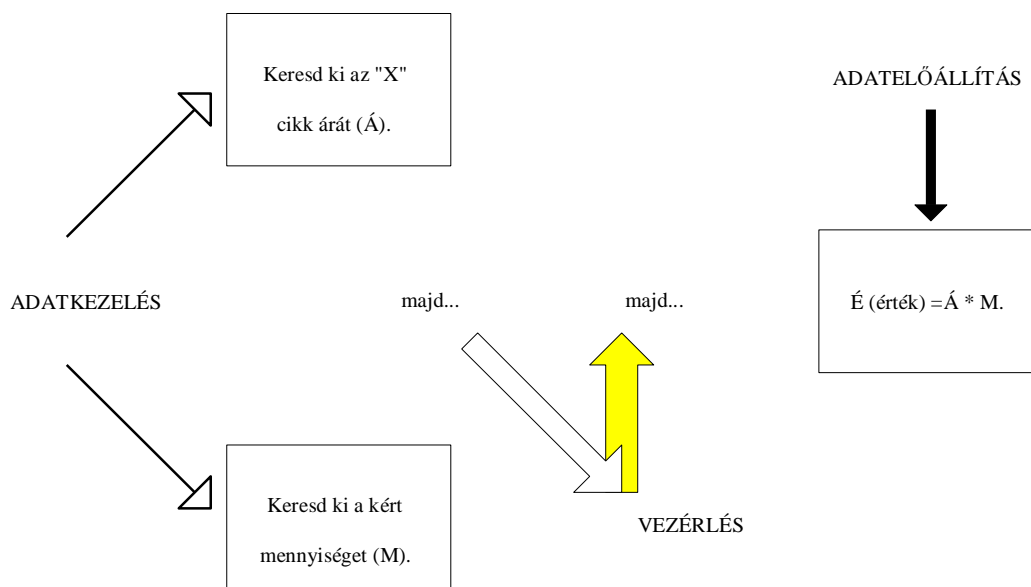
Huszonöt évvel ezelőtt az elméleti szakemberek a számítógépeket 'műszaki', 'tudományos' illetve 'adattfeldolgozó' csoportokba sorolták. Viszont ma már tudjuk, hogy ez az osztályozás értelmetlen és célszerűtlen. A folyamatirányítást vezérlő gépek éppen úgy adatokat dolgoznak fel, mint pl. az űrhajózási számításokra alkalmazottak vagy azok, amelyeket 'csak' könyvelésre használnak.

Az adatok kezelési és előállítási műveleteinek az elvi szétválasztása számos ok miatt nem teljesen értelmetlen. Akkor sem az, ha tudjuk, hogy a tevékenységben ezek gyakorlatilag elválaszthatatlanok, egymást váltják. (Egy kicsit a dolgok elé futva utalnunk kell például arra, hogy léteznek olyan általánosított adatkezelő rendszerek, amelyek az adatkezelési műveleteket támogatják, ámde nem céljuk az adatelőállítási műveletek általánosítása.) Ezért nem árt egy picit közelebbről megismerkedni a kétféle tevékenységrész lényegével.

Az **adatkezelési műveletek** során nem születnek új ismeretek. Adatkezelési művelet például a rendezés, a másolás, a mentés, az ismeret bevitele, törlése, módosítása, kikeresése, kijelzése, kiírása stb. Talán egyesek furcsállják, hogy szerintünk az adatbevétel és -módosítás közben nem születik új ismeret. Be kell azonban látniuk, hogy csak azt lehet bevinni és módosítani, ami már van. Ezért maga a beviteli/módosítási művelet önmagában valóban nem vezet új ismeretre, bár - mint majd látni fogjuk - előállítási művelettel párosulhat. A fentiek szerint egy-egy rendelés adatainak a bevitele, a rendeléstétel törlése, a rendelések adott szempont szerinti átrendezése stb. mind-mind adatkezelést jelent.

(A 'bevétel', 'kikeresés' stb. szavak senkit se tévesszenek meg. Bár azt sejtetik, hogy itt számítógépes adatkezelésről van szó, tágabb értelemben pl. a bevétel nem jelent mást, mint egy új tételnek a rögzítését, ami akár papírra is történhet.)

Az **adatelőállítási műveletek** során új ismeretek születnek. Ilyen részművelet például az, amikor a beszerzett cikk mennyiségét megszorozzuk az egységárral, hogy megkapjuk a rendeléstétel értékét. Így az 'érték = mennyiség * egységár' művelet új ismeretet eredményez. Ha viszont magát az értéket is tároljuk valahol és előkeressük ezt a tárolt értéket, akkor nem állítunk elő adatot, hanem csak a már meglévőt kezeljük. Itt vegyük észre azt is, hogy míg a kezelési műveletek mindig szükségszerűen technikához kötöttek, az előállításiak nem azok, viszont sokkal inkább a felhasználó ismeretigényeihez kapcsolódnak.



2.4 ábra: Adatkezelő, -előállító és vezérlési művelet

Természetesen egy-egy tevékenység során a kétféle művelet valamilyen adott sorrendben változtatja egymást, vagyis - szervezett együttest képez. Éppen ezért figyelniünk kell arra, hogy a tevékenységben vannak **vezérlési műveletek** is. Ezeken múlik - többek közt - a résztevékenységek összehangolása. Vezérlési részművelet(sor) indítja - például - a rendelés ellenőrzése után annak bevitelét, azt követően a diszponálást, majd a készletkarbantartást stb. Úgy is mondhatjuk, hogy a másik kétféle műveletek sora a vezérlésiek által lesz szervezett.

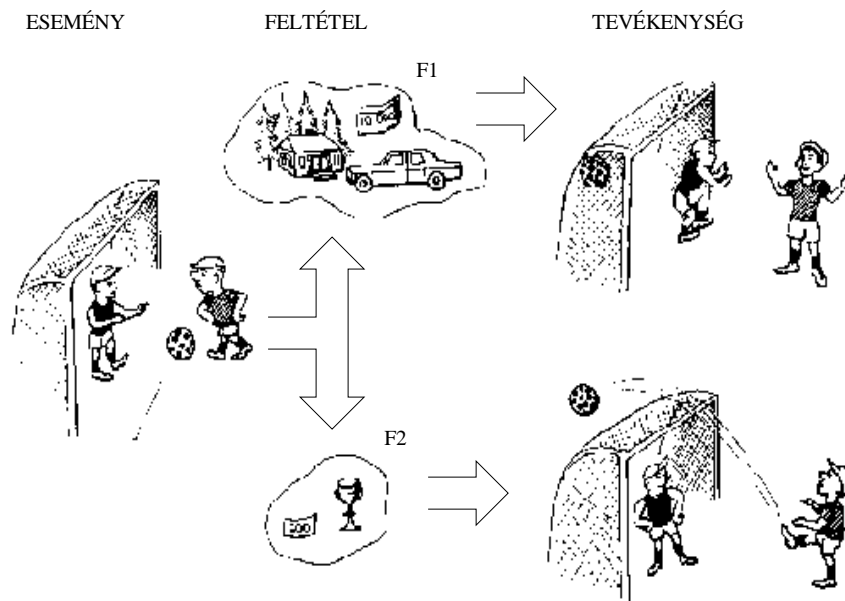
Itt érkezünk el a másik alapvető fogalomhoz, az **esemény**hez. Az eseményt a legtöbbször a **változás** fogalmával azonosítják, és ebben nem is nagyon tévednek. Maga a változás váltja ki az eseményt. Már csak az a kérdés, hogy mit is kell az előbbin érteni. Ugyanis többféle - szűkebb és tágabb - értelmezésre van mód.

A bennünket érdeklő jelenségeknek van egy ún. **életciklusa**. (Ezt a dolgot nem illik összevetésztetni magának a rendszerfejlesztésnek az életciklusával, amiről majd másutt ejtünk szót.) Az egyedek életciklusa [entity life cycle] a születés, a változás és az elmúlás fázisait foglalja magába. Például bennünket foglalkoztat a 'vevő' jelenségcsoport (más kifejezéssel: egyedtípus). A konkrét vevőismeretek (egyedek) létrejönnek, átalakulnak és megszűnnek, természetesen a már ismert beviteli, módosítási és törlési adatkezelési műveletek által érvényesített módon. Ha új vevőnk akad, ha egy régebbinek az adatai módosulnak vagy pedig a volt vevőnk számunkra megszűnik ezt a szerepet játszani, akkor ez mind-mind olyan változást jelent, amely információs eseményként jelentkezik. Ezért információs tevékenységet vált ki, amely ekképpen az eseménytől elválaszthatatlan.

Itt fel kell hívnunk a figyelmet arra, hogy a számítástechnikus mindig és eleve darabszámokban gondolkodik. Ezer és egy oka lehet annak, hogy az aktuálisan fellépő ismeretváltozást nem érvényesítik azonnal a számítógépen, hanem - egy célszerűen korlátos logika szerint - a változásokat összegyűjtik, tehát **kötegelik**. A változások átvezetésének ez az elnapolása praktikus okokból megengedhető, bár tudjuk, hogy ez gyakorlati gondokkal jár. (Lekérdezéskor a felhasználó nem a legfrissebb ismeret birtokába jut.) Ámde az informatika elméleti szintjén nincs ilyen kötegelési megfontolás. Nem a kötegelt adatok kezelését szolgáló program elindítása, hanem maga a változás jelenti az információs eseményt.

Mindenesetre a kötegelés (itt nem részletezendő) megoldása máris a roppantul fontos **idő** tényezőjére irányítja a figyelmünket. Mert az biztos, hogy ha minden más változatlan is marad (ami melleleg eléggé nehezen képzelhető el), maga az idő mindig változik. Mármost vannak olyan tevékenységek, amelyek tipikusan az időhöz kötöttek. Pl. a bérszámfejtést minden X időszak Y-dik napján indítani kell, minden egyébtől függetlenül. „Tempora mutantur et nos mutamus in illis.” Az idők során, az idő változásával mindannyian változunk. Ez pedig azt jelenti, hogy állapotunk a korábbihoz képest módosul. Maga a változás önmagában még nem mindig jelent állapot-módosulást. Pillanatról-pillanatra másokká válunk, de csak a ‘lényeges változás’ az érdekes számunkra. A raktárkészlet fogy (változás), de ez csak akkortól fogva érdekel bennünket, amikor az a minimálszintet is eléri (állapotváltozás). Ezért az **állapot** változása különleges esemény.

Vannak olyan módon kialakított rendszerek is, amelyekben valamilyen adott ismérvhez kötötten (ez lehet időbeli, méretbeli vagy egyéb kategória) úgy mond ‘automatikusan elindul’ egy olyan segédtevékenység, amely bizonyos állapotokat ellenőriz. Ha elértük a minimális készlet-szintet, akkor a segédtevékenység ezt az állapotot felismerve kikényszeríti pl. a készlet-utánrendelés tevékenységét. Az állapotváltozási jelzőkhöz kapcsolt ellenőrzési eljárásokat **‘ravasznak’** [trigger], ‘begyújtó szerkezetnek’ tekinthetjük, és a ravasz meghúzása - esemény.



2.5 ábra: A tevékenység és az esemény viszonya

Mindeddig az eseményt viszonylag objektív dolognak tekintettük. Ámde maga a tudatos ember nemcsak szemléli és nyomon követi a tényleges változásokat, hanem azokat maga is előidézi, hogy formálja a világot. Például leül a géphez és valamit búvárkodik az adatokban. Talán a bekapcsoláskor még azt sem tudja, hogy mit is keres, csak úgy tallózik. Felfogásunkban a ‘leül a géphez’ mozzanat is esemény, mivel tevékenységet vált ki.

A fentiekben csak arról volt szó, hogy az esemény valamilyen tevékenységet **indít el**. Azonban a ténykedésnek valamikor vége szakad. Eközben pedig valami megváltozott: ha más nem, az idő biztosan. Következik - illetve következhet - az újabb tevékenység, amelyet - adott esetben - éppen az előző befejezése, az általa előidézett új állapot vált ki. Éppen ezért a tevékenységet **lezáró** momentumot is eseménynek célszerű tekintenünk.

Az információs rendszer - ilyen értelemben - nem más, mint az események és a tevékenységek végtelen láncolata. Felesleges lenne azon elmélkedni, hogy az információs tevékenység vagy az információs esemény az elsődleges a másikkal szemben. Az ... E-T-E-T-... esemény-tevékenység sorozatnak ugyanis elvileg nincs sem eleje, sem vége. Éppen ezért az esemény meghatározása relatív:

D 2.5 Eseménynek az információs tevékenységet kiváltó illetve az azt lezáró momentumot nevezzük.

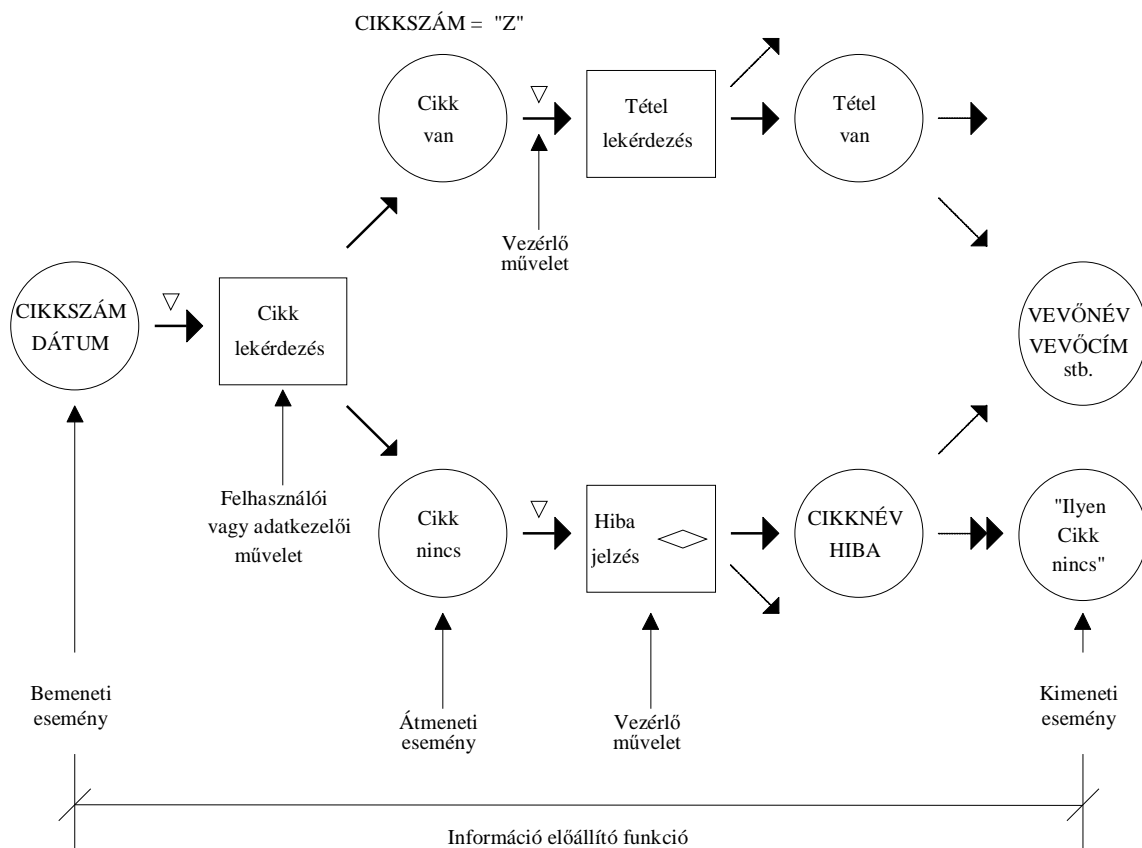
Persze ugyanilyen erővel azt is mondhatnánk - a D 2.5 meghatározás újabb változataként -, hogy a tevékenység az az adatfeldolgozási egység, amelyet a *felhasználói igény* indító és lezáró eseményei határolnak. Mivel minden egyes tevékenységnek legalább két kimenete lehetséges - sikeres/nem az -, a lezáró események esetében mindenképpen a többes szám a helyes megfogalmazás.

Miért kötjük ennyire az eseményt a felhasználói igényhez? Nos, az IR-ben az adatok rendszere mindig kvázi-objektívan meghatározható és elrendezhető. Bár maguk az ismeretek persze kvázi-szubjektívek, hiszen szintén igények tárgyai, az utóbbiak meghatározása után mindig megtalálható a 'jó' adatszerkezet. (Az adatbázisstervezés tudománya foglalkozik ezzel a kérdéskörrel.) Ugyanakkor a tevékenységek elrendezésének nincs ennyire megfogható módszere. Álljon itt szemléltetésként a következő példa:

P 2.1 Az A, B és C cégek mindegyike folytat értékesítést. A rendelés beérkezése után az A-ban a 'rendelésfeldolgozás' keretében csak rögzítik a rendelés ismereteit, minden más már új tevékenységnek számít. A B-ben a 'rendelésfeldolgozásba' tartozik az is, hogy a kielégíthetőséget ellenőrzik és kiállítják a diszpozíciót. A C-ben pedig automatikusan új utánrendelést is kibocsátanak, amikor a készlet a minimális szintet eléri.

Melyik vállalat gyakorlata a helyes? Erre a kérdésre nem lehet igazán jól válaszolni. Azért nem, mert az események és tevékenysége láncolata szinte végtelen; azt nem lehet egyként felfogni (mindaddig, ameddig tökéletesen automatizált szervezet nem létezik); a láncot valahol meg kell törni azért, hogy a tevékenység-egységeket meg lehessen fogni (pl. programozni); ez a tagolás pedig nem attól jó vagy rossz, hogy hol húztuk meg a vonalat, hanem attól az, hogy ezt minőségileg hogyan tesszük: rendszer marad-e a rendszer, vagy sem?

Ezek után már csak egy pár dologra kell figyelni. Az első a fentebb említett felhasználói igényhez kapcsolódik. Az események és tevékenységek láncolata nem egy végtelen *sor*, hanem egy végtelen *háló*. Azért az, mert több kiváltó és lezáró eseménye lehet egy-egy tevékenységnek úgy, hogy a lezáró események eltérő újabb tevékenységeket generálhatnak.



2.6 ábra: Esemény-tevékenység háló

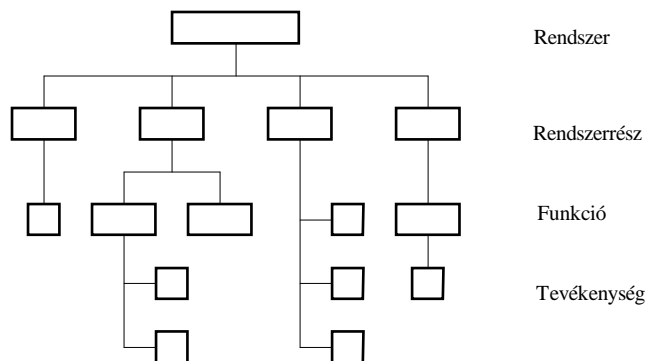
Ha ezt a hálót a síkban vízszintesen képzeljük el, akkor annak valahol meg kell találnunk az elejét és a végét abban a tudatban, hogy egyik sem feltétlenül egyszeres. Az, hogy esemény- vagy tevékenység-orientáltan nézi-e valaki ezt a hálót, számunkra magánügyet jelent. (A matematikai hálótervezésben jártasak tudják, hogy az esemény- és tevékenységorientált PERT illetve CPM technika egyike sem a szemlélet miatt jobb vagy rosszabb a másiknál...)

A háló vízszintes megbontása - képletesen szólva: függőleges vonalakkal való elvágása - azon tevékenységek végénél, azon eseményeknél történik, ahol a felhasználó a tevékenység által szolgáltatott ismeretekre kíváncsi azért, mert azok alapján be akar avatkozni a folyamatba. Bár nekünk a P 2.1 példában a C vállalat gyakorlata tetszik a leginkább, de véleményünk szubjektív, mert számos - általunk nem ismert - ok miatt az A és a B vállalat is helyesen gondolkozhat. Ha a készlet a minimális szint alá süllyed, akkor ez a tény a C vállalatnál a tevékenységen ('rendelés-feldolgozás') belüli **belső eseményt** jelent, viszont az A vállalatnál új tevékenységet indító **külső esemény**. A háló megszakítása tehát azt jelenti, hogy a belsőből emberi beavatkozást igénylő külső eseményt kreálunk; a hálók összevonása viszont éppen a fordított hatással jár.

Mivel 'minden mindennel összefügg', nincs jogunk egyetlen cég gyakorlatát sem bírálni. Az viszont világos, hogy a nevezett háló megszakításával olyan részhálókat kreálunk, amelyeknek továbbra is szerves módon össze kellene függeniük egymással. A tevékenységeket - képletes értelemben - vertikális egységekbe, körökbe kell összefoglalnunk és látnunk kell e körök átfedéseit.

A rendelésvétel/ellenőrzés, diszponálás, utánrendelés tartozhat egyetlen vagy három körbe. Csak az a baj, ha az utóbbi esetben a körök nem találkoznak.

E téren nem szabad lebecsülni a 'minden mindennel összefügg' tételét. A kis körök - a tevékenységek - nagyobbakat alkotnak. Ezeket **funkcióknak** hívjuk. A rendelésvétel és a diszponálás lehet két külön tevékenység, azonban csakis a megrendelés-feldolgozás funkción belül. Ha a rendszer összetett, akkor esetleg az is szükséges lehet, hogy a funkciókat **funkciócsoportokba** soroljuk. Ezek - vagy maguk a funkciók - alkotják a rendszerrészeket. Más **rendszerrészt** jelent a megrendelés-feldolgozás, mint a számlázás, ámde a két kör egymásba ér. A rendszeren belüli tevékenység-egységek rendjét hierarchikus ábra mutatja:



2.7 ábra: A tevékenységek vertikális absztrakciója

A tapasztalatlanok semmi közöset sem látnak például a termelésirányítási és a munkaügyi rendszerrészben, éppen ezért mindkettőt 'rendszernek' is titulálják. Szerintünk viszont csak a munkás tud termelni. A munkás éppen úgy tényezője a termelésirányítási, mint a munkaügyi rendszerrésznek. A tevékenységeknek a funkciókba, funkciócsoportokba, rendszerrészekbe való besorolása nem más, mint a teljes szervezet áttekintését segítő absztrakció. És amint azt már korábban elmondottuk a határok és a környezetek kapcsán, a tudatos szétválasztás egyben mindig összekötést is jelent.

A harmadik - és az alponban utolsó - tagolási témához érkeztünk el. Most már nem hallgathatunk arról a tényről, miszerint egyáltalán nem közömbös az, hogy számítógép segítségével végezzük-e az információs tevékenységet, vagy sem. Mindez alapjaiban összefügg az információ meghatározásával (ld. D 2.3).

Az információ az egyéni ember fejében születik abban a pillanatban, amikor a saját agyában lévő ismereteket összegyűrja az újonnan kapott adatokkal, vagyis az utóbbiakat értelmezi. Az információ tehát sohasem egyetlen ismeret, hanem mindig többféle ismeret összetételének a gyümölcse. Hiába mondjuk valakinek azt, hogy 'Japánban ...' (egy ismeretdarab), ha az nem tudja, hogy a 'Japán' az egy országot jelent (másik ismeretdarab).

Az informatika egyik legnagyobb, legkihívóbb kérdése az, hogy mekkora erőfeszítést követel meg személy szerint bárkitől az ismeretek értelmezése, vagy - fenti csúnyább szavunkkal - összegyűrése. Maga az értelmezés pedig mindig kétféle dolgot jelent: egyrészt szelekciót, szétválasztást (ez rám tartozik, az meg nem), másrészt integrálást, összekapcsolást (jé, ez a dolog azzal összefügg).

A fentiek miatt egyáltalán nem közömbös, hogy az információs tevékenység által befogadott adatkezelési és adatelőállító műveleteket miként fogalmazzák meg. Pontosabban szólva: a számítógépes feldolgozás hol kezdődik, azaz hol lép be a tevékenységek láncolatába és hol végződik, tehát hol lép ki a tevékenységi hálóból. Magyarul: a számítógépes feldolgozás milyen mértékben könnyíti meg az ismeretek kezelését.

T 2.7 Számítástechnikát oktató, szoftvereket forgalmazó cégben még nemrég is úgy történt a bérszámfejtés, hogy az adminisztrátorok tucatjai kézzel kiszámolták a pótlékokat és a levonásokat. Majd az így ‘előemésztett’ adatokat a számítógépre vitték, és azt mondták, hogy van egy bérszámfejtési számítógépes rendszerük.

E mű szerzője egy régebbi cikkében már kifejtette a **feldolgozási fokról** vallott nézetét. Úgy gondolja most is, hogy erre a lényegre ma sem figyelnek eléggé. A feldolgozottsági - más szóval: az automatizáltsági - fok ugyanis azt mutatja, hogy a teljes végrehajtandó tevékenységi láncolatot alkotó adatkezelési és adatelőállító műveleteket milyen arányban automatizáljuk. A fogalom jobb megértéséhez a fentebbi és az alábbi történet szolgál. A kettő jól szemlélteti, hogy eszközeinket ma még nem aknázzuk ki a megfelelő szinten.

A feldolgozási lánc a 2.7 történetben **előlről nyitott**. A számítógépet nem a célszerű módon használják, és ezért alacsony a feldolgozottsági fok. Ugyanis a pótlékokat és levonásokat nyugodtan ki lehetne számoltatni a géppel is. Az igazi megoldás természetesen az lenne, ha a pótlékokat és levonásokat fellépésük után (vö. esemény) azonnal a számítógépen rögzítenék. Ezzel nemcsak a számítást automatizálnák, hanem az ismeretek lekérdezhetőségét is növelnék. Ráadásul még a bérekkel való igazi gazdálkodás lehetősége is valóságossá válna.

T 2.8 Még a hetvenes években történt, hogy egy neves közlekedési vállalatunkban a készleteket csak mennyiségben vezették, vagyis értékben nem. A hónap végén elkészült hatalmas táblóhalmazon az adminisztrátorok tucatjai kézzel szorozták fel a mennyiségeket értékekké.

A feldolgozás láncában ebben az esetben **hátról nyitott**, azaz be nem fejezett. A történet ebből a szempontból nem igényel különösebb magyarázatot. Manapság ennyire durva hibákat már nem követünk el. Azonban ma is jellemző a hátról nyitott feldolgozások uralma. Gondoljunk csak arra, hogy az értelmezéshez két momentum szükséges: a szelekció és az összekapcsolás. Márpedig még ma is rengetegszer találkozunk nem nekünk szóló - vagyis a mi szempontunkból nem eléggé feldolgozott - tételekkel ill. tételeken belüli adatokkal (szelekcióhiány). Az sem ritka, hogy kézzel kell összesítéseket végeznünk illetve külön eljárással kell kikeresnünk a tételhez a bennünket érdeklő kiegészítő adatokat (a kapcsolás hiánya).

Vegye észre az olvasó, hogy nem mi vagyunk következtelenek, amikor a fenti szakaszokban mintegy váratlanul a számítógépes megoldásokra tértünk ki. Ez a dolog tényleg egy kicsit talán meglepő, mivel egyelőre nem a technikával, hanem az információs rendszerek lényegével óhajtunk foglalkozni. Csakhogy itt éppen arra akartuk felhívni a figyelmet, hogy a fejlesztőknek nem volna szabad eleve szétválasztaniuk a számítógépes és a gépen kívüli tevékenységeket, hanem a teljes - manuális és automatizált - tevékenységkörben kellene gondolkodniuk. Ha nem így tesznek - például azért, mert a kézi és a gépes folyamatokért eltérő részlegek a felelősek -, akkor szükségszerűen előlről és/vagy hátról nyitott, nem teljes feldolgozási láncok fognak születni.

2.3 AZ INFORMÁCIÓS ERŐFORRÁSOKRÓL

Ezt az alponthoz az eddigieknél rövidebbre fogjuk. Ezt azért kell tennünk, mert ennek a könyvnek nem feladata az információs rendszerekkel kapcsolatos összes lehetséges **erőforrásnak**

a részletes áttekintése, magyarázata. Így az alábbiakban csak néhány alapvető dologra szeretnénk felhívni a figyelmet.

Az élet bizonyos területeit illetően az a nézet terjedt el, hogy a rendszer és az erőforrás kapcsolata lényegét tekintve a **cél** és az **eszköz** viszonyának felel meg. Például: Egy társasház építése során maga a ház a rendszerként tekintendő cél; a megalkotáshoz igénybe vett munka, pénz, gép, anyag stb. pedig olyan erőforrás, amelyet a cél eléréséhez eszközként használnak.

Az is közismert, hogy a cél és az eszköz **fontossági sorrendje** néha megfordul az emberi tudatban, vagyis az erőforrás veszi át a rendszer elsődleges szerepét. Ilyesmivel két esetben találkozhatunk: akkor, ha az erőforrások szűkösek, illetve akkor, ha azok drágák vagy bonyolultak. Az utóbbi helyzetre a legékeesebb példa maga a számítógép, az előbbire pedig a szociális ellátás. Bár maga a szerepcsera sokszor megérthető, fel kell hívni a figyelmet annak roppant veszélyeire, mert a két példában közös, hogy az eszközzel való túlzott foglalkozás miatt maga a cél szenved csorbát. Nem is ritkán annyira, hogy magának a rendszernek a lényege veszik el. Pl. mivel nincs elegendő pénzünk, felmentve érezzük magunkat attól, hogy a szociális ellátást teljes rendszerként átgondoljuk. Ezért nem egy meglévő és átgondolt rendszerhez igazítjuk az erőforrásokat, hanem éppen megfordítva: a szűkös erőforrással megvalósított valamire fogjuk rá, hogy az maga a rendszer. A számítógépek esetében is hasonló a helyzet. Nem az eszközgazdálkodás kerül a figyelem középpontjába, mint adott feladatú és célú rendszer, hanem a gépen megvalósított valamit hívjuk eszközgazdálkodási rendszernek.

A cél és az eszköz egybemosisásával ellenkező irányú, ám mindennap tapasztalt jelenség az, hogy e két dolog viszonyát nagyon sokan mereven szemlélik, és így nem látják a **rendszer és az erőforrás dialektikus egységét**. A következő példa jól mutatja a félreértések lényegét és következményeit. A téglát sokan csak a ház felépítéséhez szükséges erőforrásnak tekintik, ami baj. Azért az, mert a téglát az épületbe beépítve a háznak, mint rendszernek az elválaszthatatlan részévé válik. Ha például rossz minőségű, akkor maga a rendszer - a ház - majd korábban fog felújításra szorulni.

Miért kellett mindezt elmondanunk? Azért, mert az információs rendszerekre adott korábbi definíciónk (ld. D 2.1) egy bizonyos értelemben ellentmondásos. A szó **tágabb** értelmében az adat, az esemény, a tevékenység, a felhasználó és a szabvány e rendszernek nemcsak tényezője, hanem egyben erőforrása is. Pl. gondoljunk csak arra, hogy az egyik ismeret (cél) a másiktól születik (eszköz), tehát ha úgy tetszik, az utóbbi az előbbinek erőforrása. Azt is szoktuk mondani, hogy az adat a szervezet egyik legfontosabb erőforrása. Ezért a definícióban az erőforrásnak, mint külön tényezőnek a feltüntetése többletmagyarázatot igényel.

Ebben a könyvben mi az erőforrásnak egy **szűkebb** értelmezését alkalmazzuk. Elismerjük azt, hogy pl. maga az adat is erőforrás, azonban mi ezt a tényezőt e könyvben elsősorban nem ebben a minőségében óhajtjuk majd vizsgálni. Ehhez jogunk van a többi tényező esetében is. Csak akkor lennénk következtetlenség, ha nem árulnánk el, hogy mit is nevezünk a szó szűkebb értelmében erőforrásnak.

D 2.6 Erőforrásnak az IR fejlesztéséhez és működtetéséhez szükséges időt, pénzt és technikai eszközöket - hardver illetve szoftver - tekintjük.

Most nem feladatunk az, hogy az idővel, a pénzzel és a technikai eszközökkel mint önmagukban is vizsgálható rendszertényezőkkel foglalkozzunk. E dolgok taglalása a projektmenedzsment tárgykörébe tartozik, ami viszont kívül esik e műnek a mondanivalóján. Ezért itt csak néhány alapvető dologra - elsősorban az erőforrásokkal kapcsolatosan elkövethető szemléleti hibákra - világítunk rá.

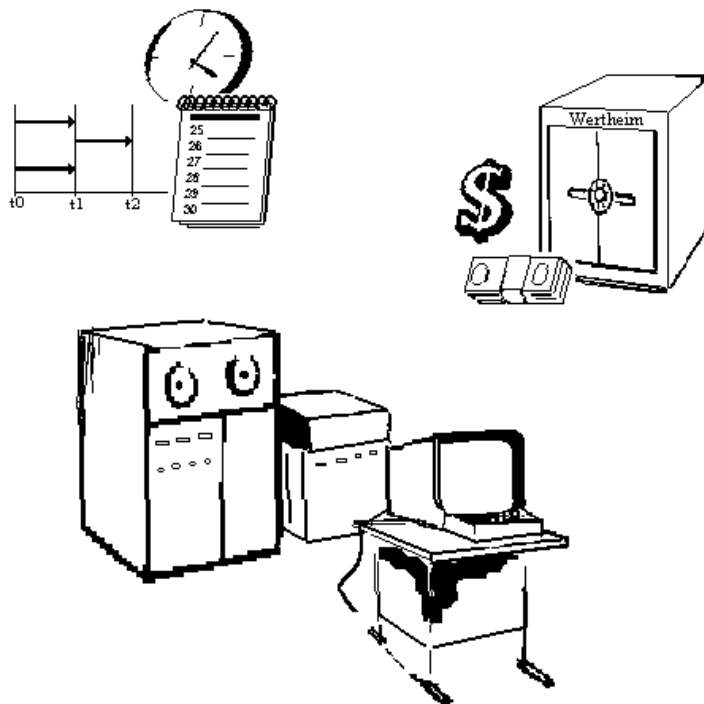
Az IR minden tényezője pénzbe kerül: az adat éppen úgy, mint a tevékenység; mi több, bizonyos értelemben maga az ember is. Ezért nagyon sokak számára az információs rendszerek világában is a **pénz** a kezdet és az a vég. Az anyagiakat általános mérceként tekintők csupán csak

arról feledkeznek meg, hogy éppen az ismeretek az emberek legszemélyesebb kincsei. Ezért számukra nem egyedül a 'gazdaságos' a feltétlen mérték. A 'jó' és a 'szép' legalább annyira fontos mutató. Nem szabad ugyanis elfeledkezni arról, hogy az ismeret nem csupán mennyiség, hanem - mégpedig elsősorban - minőség.

Éppen ebből a minőségi természetből következik az is, hogy az IR erőforrásai nem tetszőlegesen konvertálhatók egymásba. A legtöbb vezető az **idő** tényezőjét illetően él végzetes tévedésekben.

T 2.9 1992 nyarán azzal kerestek meg, hogy két hét alatt (!) készítsek egy rendszertervet egy vállalatok értékesítésével foglalkozó cég részére. Az ajánlatot elutasítottam - akkor is, amikor az eredeti vállalkozási összeg kétszeresét kínálták fel.

Persze a végén akadt egy kókler, aki elvállalta ezt a feladatot, jóllehet két hét alatt nem lehet becsületes módon rendszertervet készíteni; ki van zárva. Akkor is, ha százan, ha ezren dolgoznak rajta. „Ha egy asszony kilenc hónap alatt szül egy gyereket, akkor kilenc asszony...” Sok IR-menedzser abban a tévhitben él, hogy több eszközzel, több emberrel - vagyis több pénzzel - le lehet rövidíteni a fejlesztéshez szükséges időt. Pedig már réges-régen tudjuk, hogy a „mennyiség átcsap minőségbe” tételnél nagyobb csacsiság aligha létezik e földön. Az igaz, hogy a minőséghez mennyiség (is) kell, de ez a képlet nem megfordítható. Az információs rendszer feltárása, megértése, elemzése olyan feladat, amely ugyan időt (mennyiség) is igényel, de elsősorban átlátást (minőség) követel. És senkire sem lehet ráparancsolni, hogy rövidebb idő alatt lássa át a dolgokat.



2.8 ábra: Az információs erőforrások

Azután arra is gondolni kell, hogy „Az idő - pénz.” szólás mennyire féloldalas és mennyire félrevezető. Ugyanis az egykor elvesztett pénz talán még valamikor pótolható; ám az elveszte-

getett idő sohasem az. A késedelem megfizethető, de a késés miatti változások sohasem visszafordíthatóak. Avagy ki tudja pénzzel azt kompenzálni, hogy egy megkésett levél miatt nem voltam ott a legjobb barátom esküvőjén? A hamis információ miatti kár anyagilag ugyan némileg pótolható, ám az erkölcsi veszteség korrekciójára soha nincs mód. Az ismeret az emberek legféltettebb, legdrágább kincse. A börtönökben megnyomorítottaktól elvehették az anyagi javaikat, a tudásukat viszont nem. Ám sohasem adhatták nekik vissza úgymond kárpótlásként az elveszített időt. Ezért a valódi informatikus sohasem dől be az idő és a pénz látszólagos konvertálhatóságának.

Az 'erőforrások egymásba nem konvertálhatók' alapvető tétel az eszközökre is vonatkozik. Hiába adnának ide nekem most egy sokkal gyorsabb számítógépet; attól én még nem kínlódnék kevesebbet e könyv megírásával. És megfordítva: vajon mit is szólna az olvasó akkor, ha kijelenteném: ez a könyv azért ennyire gyengéske, mert rossz volt a számítógépem? (Ami mellesleg nem is lenne igaz, mert a hozzám paterolt gép mindent tud, amire szükségem van.)

A gyenge informatikus sokszor az **eszköz** úgymond elégtelenségével próbálja meg leplezni a saját képtelenségét. Tanulságos esetként álljon itt ez a történet:

T 2.10 Az egyik legnagyobb - banki funkciókat is ellátó - pénzügyi intézményünkben az egyik napról a másikra több tíz millió forintért lecserélték a korábbi számítógépeket. Kérdésemre, hogy mi váltotta ki ezt a váratlan és költséges akciót, a lakonikus válasz a következő volt: „A régiek lassúak voltak.” Viszont arra a második kérdésemre, hogy „Miben nyilvánult meg a lassúság?” senki sem tudott válaszolni.

Ez a történet több problémát vet fel. Az első szerint az illető cégnél senki sem volt tisztában azzal, hogy a 'lassú' az nem abszolút, hanem relatív fogalom. Ha 10 másodperc alatt van szükségem egy ismeretre és a válasz csak 11 másodperc alatt érkezik, akkor a feldolgozás valóban lassú. Ha viszont a válaszügy igény egy óra és az ismeret 59 perc alatt jelenik meg, akkor nem beszélhetünk lassúságról.

T 2.11 Barátom át akart térni a Word 6.0-ra. 386-os kis gépén két teljes másodpercbe került a kurzor visszaléptetése. Hja, két megában! Új gépre nem volt pénze - maradt a régi Word változatnál. Ami szintén lelassult, mert barátom 200 fonttal, tapétákkal stb. tömte ki a szoftvert. Neki ez így is megéri...

Második észrevételünk szintén általánosítható, mert hiszen nem csak a történet főszereplőjére, hanem a legtöbb magyar felhasználóra jellemző az eszközökben való balga reménykedés. Nevezetesen az, hogy az időt és az eszközt egymásba konvertálható erőforrásoknak tekinti. Ha tehát a *szubjektíven* érzett válaszügy nem megfelelő, akkor gyorsabbnak vélt gép és szoftver vásárlásába fog ahelyett, hogy előbb megvizsgálná az *objektív* miérteket.

Barátomnak elmagyaráztam, hogy csodák nincsenek. Ha neki megéri, hogy a Word két perc alatt jelentkezik be, akkor nincs okom vitatkozni vele. Ő tudja, hogy mit és miért vállalt. Viszont vele szemben a profi számítástechnikusok miért nem keresik a miérteket? Például a számítógép attól is lehet lassú, hogy rosszul - azaz redundánsan - szervezik meg magukat az adatokat. Attól is, hogy célszerűtlenül fogalmazzák meg a tevékenységeket illetve az azokat megvalósító programokat. Elképzelhető az is, hogy a menüpontokként megjelenő események rendszere rosszul meghatározott. Uram bocsá', a felhasználó képzetlensége is lehet a lassúság oka. Az alkalmazó az adott helyzetben nem tudja, hogy mit is kellene tennie, mert nem jól meghatározottak a szabványok. Végeredményben a számítógép lassúsága mögött valójában az IR bármelyik másik

tényezőjének a fogyatékos, célszerűtlen meghatározása is rejtőzhet. Ezért nem a gép leváltása, hanem a problémák elemzése és a bajok kiküszöbölése vezethet csak sikerre.

T 2.12 1994-ben cégünknel egy bérelszámolási programrészlet hat órát futott. Mindenki a gépet szidta a lassúsága miatt. Kolléganóm a külsősök által megírt programot átgondolta - és írt egy új verziót. Az ő programja pár perc alatt lefutott...

Hazánkban - bár nem igazán hisszük, hogy ez a jelenség csak a magyarokra jellemző - a cél és az erőforrás viszonya különösen zűrzavaros. Ezt részben a korszerű eszközöktől való korábbi elzártságunk utáni mai eufórikus hangulat, részben az újabban minden területen jelentkező fellengző kivagyiság (nekem van, neked nincs...), végeredményben pedig a szintén a korábbi elzártságból fakadó kultúrátlanság és képzetlenség magyarázza, ha nem is indokolja. Az alábbi történet jól mutatja a visszasságokat.

T 2.13 Ebben az évben történt, hogy egy számítógép-alkatrészeket is forgalmazó vállalatban barátom keservesen megbukott. A cégnek mindenből a 'legjobb' kellett. Összevásároltak pentiumokat, húszféle ilyen-olyan szoftvert, háromféle (!) hálózatot stb. Hja, arra nincs pénz és idő - nincs szükség -, hogy a készletnyilvántartást megszervezzék. Erről nem nyilatkoznak: tessék a számítógépet használni, ott van, és kész...

Vásárolni - ahhoz nem kell ész. A felhasználó nem mondja el az igényeit. Tessék a számítógépet alkalmazni. Ilyen civilizálatlan megközelítésre azért a fehér Európában nem sok példa akad...

T 2.14 Az egyik biztosítási intézménynél a számítógépeket és szoftvereket az előzőeknél úgymond kétszer hatékonyabbakra cserélték le. Érdekes módon ettől egyetlen felhasználónak sem támadt az az érzete, hogy ő gyorsabban jut ismeretekhez.

A nyolcvanas évek közepén e mű szerzője egy igazi XT-vel dolgozott. 20 Mb háttértár, 512 Kb RAM. Érdekes módon az a kis kütyü akkor minden célunknak megfelelt. Ma?

A fentebbi történet ismét csak a cél és az eszköz, a rendszer és az erőforrás tudatbeli ellentmondásaira akarja irányítani a figyelmet. Amikor a kétszer jobb eszközöket megvásárolták, az intézményben azonnal és önkéntelenül - vagyis nem tudatos és előre elgondolt módon - átdefiniálták a rendszer lényegét. Egy picit, no de egészen csak egy picinykét - a duplájára - bővítették ki a rendszert. Ezek után az lett volna a csoda, ha valami tényleg felgyorsul.

Az erőforrásokra vonatkozó pontot lassan be kell zárunk. Az erőforrás az IR egyik tényezője. Mivel pedig az IR a tényezők *szervezett* együttese, a logikus ember azt gondolhatná, hogy az információs rendszer fejlesztése és működtetése során az IR-ért felelős valakik az erőforrásokat a többi tényezővel összhangban látják. Természetesen erről szó sincs. Az erőforrásokat az ember nagyon is képes *szervezetlen* módon használni. Képes arra is, hogy a pénzt, az időt, az eszközt elvesztesse teljes ellentmondásban azzal, hogy viszont nagyon sokszor éppen ezeket a másodlagos tényezőket helyezi a többiek elébe.

Egyáltalán nem csak a szerző magánbánata az, hogy egy százmillió forintos (!) tender kapcsán szinte pillanatok alatt döntöttek az eszközvásárlásról úgy, hogy erre az aktusra - mármint a gép- és programbeszerzésre - több hónapot szántak. Eközben a rendszertervezési munkára heteket sem hagytak, ámde a szerzőnek a mindössze pár tízezer forintnyi (!) honoráriumán hónapokig vitatkoztak. Ennyit az információs rendszerek céljáról és annak erőforrásairól.

2.4 A FELHASZNÁLÓKRÓL

Az olvasó már megszokhatta, hogy ebben a könyvben számos fogalmat nem a hétköznapi, nem is a számítógépes környezetekben bevett módon alkalmazunk. Ennek az az oka, hogy a dolgokat olykor a szokványostól kissé eltérően, olykor pedig bizonyos tágabb összefüggésekre is figyelve kívánjuk bemutatni. Ezt nem kerülhetjük el a **felhasználó** [user] fogalmának az esetében sem.

Az olvasónak látnia kell, hogy a D 2.1 meghatározásban a 'szervezett együttes' kitétel nem csak azt sugallja, hogy az eltérő természetű rendszertényezőket - így például az adatokat illetve a tevékenységeket - kell összehangolni. Az egyazon kategóriába tartozó rendszerelemek is viszonyokban állnak egymással. Az egyik adat nem független a másiktól, az egyik esemény nem következik be egy másik nélkül stb. Éppen ezért az IR-ek emberi résztvevői, a felhasználók sem foghatók fel csakis az adatokhoz, eseményekhez, tevékenységekhez stb. való viszonyban. A felhasználók a rendszeren belül is sajátos rendszert alkotnak, és ez a tény arra kötelez bennünket, hogy a felhasználó fogalmát nagyon szélesen értelmezzük.

Az informatikában nem ritka dolog, hogy a felhasználó kifejezést viszonyoknak tekintik és azt *élettelen* dolgokra is alkalmazzák. Az egyik gép kapcsolódik a másikhoz; az egyik tevékenység a másik eredményére alapoz; az egyik ismeret a másiktól származik. Ezért logikus, hogy a 'másikat' az 'egyik' felhasználójának tekintsük. Azonban engedje meg az olvasó, hogy ebben a könyvben csak az *élő* embert ill. az általa létrehozott emberi szervezeteket nevezzük felhasználónak.

D 2.7 Felhasználó az ismeretekkel kapcsolatban álló ember(csoport).

Lám ismét egy olyan meghatározást alkalmaztunk, amely nem túl sokatmondó és ugyanakkor mégis nagyon tág. Definíciónk szándékosan ilyen. Egyrészt egy rejtett tagadást, másrészt egy ugyancsak eldugott igénylést von maga után.

Felhasználóknak mi az *ismeretekkel* kapcsolatban álló valakiket nevezzük. Ez rejtett tagadás: a definíció nem a *számítógéppel* vagy a szoftverrel viszonyban álló valaki(k)ről szól. Manapság a számítástechnikai körökben divat a gépet így-úgy kezelgető személyt tekinteni felhasználónak. Ezzel így nem érthetünk egyet. Nem a 'Word-alkalmazó', az 'Oracle-kezelő' vagy a PC-tulajdonos a felhasználó, hanem az, aki a Word-ben írt levéllel, az Oracle-vel kezelt adatbázissal, a PC-n vezetett nyilvántartásokkal, mint ismeretekkel kapcsolatba kerül. Most árulja el őszintén az olvasó, hogy amikor mondjuk bizonyos ismeretek miatt telefonálunk egy üzletfelünknek, akkor számunkra mi a fontos? A megkapott ill. az elmaradt információ, vagy az, hogy 'telefon-felhasználók' vagyunk? Mi nem a telefont, hanem - talán nem meglepő módon - üzletfelünk értesítéseit akarjuk használni.

A *számítástechnikus* gondolkodásmód szerint be kell kapcsolni a gépet azért, hogy például a barátunknak levelet írjunk. Az *informatikus* gondolkodásmódja viszont azt sugallja, hogy valamit meg kell tenni: levelet kell írni. Erre - többek között - a számítógép is alkalmas, bár a nagyon személyes levelet inkább kézzel fogjuk megalkotni.

A mai 'felhasználó' fogalom elleni elvi ódzkodásunknak gyakorlati alapja van, amit alább fogunk megvilágítani. Az *ismeretátadásban* mindig két fél érdekelt. Az egyik az ismeret *küldője*, a másik annak *fogadója*. Ezért az ismeretátadás során mindig két felhasználóra kell tekintettel lenni. A küldőre vár a nehezebb szerep, mivel elsősorban az ő kötelessége, hogy a fogadó fejével

gondolkodjon. Az ismeretkezelés ugyanis sohasem öncélú, hanem valakinek - értelem szerint éppen a fogadónak - az érdekeit szolgálja. Ehhez képest meglepő, hogy nap mint nap tapasztaljuk ennek az elvnek a megsértését. Pl. a hivatalban bennünket nem **felhasználónak**, vagyis az ismeretet fogadó partnernek, hanem **ügyfélnek**, a hivatal működéséhez szükséges rossznak tekintenek. Ezért az IR-t e helyeken nem az ügyfél számára, hanem saját maguk számára készítik. Márpedig az ilyen felhasználói szemlélet semmiképpen sem vezethet jó rendszerre.

A felhasználó hétköznapi/számítástechnikai és informatikai értelmezésének a fenti ellentéte egyáltalán nem végletes és viszonylag könnyen feloldható. Arra kell csak figyelni, hogy az emberek és csoportjaik eltérő **szerepeket** töltenek be az információs rendszerben, vagyis az ismeretekkel való kapcsolatuk (ld. a fenti meghatározást) eltérő jellegű lehet. Ez azt jelenti, hogy a mi felhasználó-képünk a szokásosnál tágabb. Mi ugyanis nem csak a számítógép előtt ülő ill. az arról levett ismereteket közvetlenül hasznosító személyt tekintjük felhasználónak. Az IR működtetésében és kialakításában közreműködő, mi több, az azokban érintett személyek mindegyikét felhasználóként fogjuk fel.

Előzetesen meg kell jegyezni, hogy adott esetben egyazon személy egyszerre több szerepet is betölthet olyannyira, hogy szélsőséges helyzetben valóban ő az egyetlen felhasználó. (Pl. amikor magunknak készítünk könyvnyilvántartást.)

A 'felhasználó vagy ügyfél' dilemmát a **végző-felhasználó** [end-user] fogalom oldja fel. Végző-felhasználó pedig az, akinek az érdekében készül a rendszer. A mai önös szemléletű világban eléggé nehéz megérteni a mögöttes gondolatot. Pl. a hitelintézetnek - a hiedelmekkel ellentétben - nem az az elsődleges célja, hogy a részvényeseknek profitot termeljen, hanem az, hogy személyeket és cégeket lásson el hitelekkel. Ha ezt nem tenné, akkor miből születne a nyeresége? Ezért a hitelintézetnek a legvégző végző-felhasználója a hitelt felvevő, és elvileg az ő számára kell, hogy készüljön az információs rendszer. (Van persze egyéb végző-felhasználó is ebben a rendszerben. A felhasználó az ismerethez kötődik; az éves mérleg a részvényeseknek készül; ezért a mérlegnek ők a végző-felhasználói. A felhasználó szó tehát nem abszolút fogalom, hanem relatív: viszonyt fejez ki.)

Az adott szervezetben a (számítógéppel) kezelt ismeretekkel közvetlen módon kapcsolatban álló ügyintézők, gazdálkodók, adott funkciót ellátó személyek a szó szűkebb értelmében vett **alkalmazási-felhasználók** [application user]. Ismét nem a számítógép kezeléséről, hanem az ismeretek használatáról van szó. Pl. a biztosító társaságnál az ügynök lehet, hogy sohasem ül géphez. Ám a biztosítási ajánlatnak ő mégis alkalmazási-felhasználója. Az 'alkalmazási' jelző azt takarja, hogy az illető adott minőségben kerül kapcsolatba az ismeretekkel. Az ügynök a bérlistát tekintve végző-, a szerződés-nyilvántartásban alkalmazási-felhasználó, a készletgazdálkodáshoz viszont semmi köze sincs (talán).

A fentebbi két szerep az IR működtetéséhez kapcsolódik. Közös bennük, hogy a felhasználó az élő, a működő IR-nek éppen úgy lehet **adatszolgáltatója** (pl. az ügyfél káreseményt jelent be), mint annak **adatfelhasználója** (pl. a partner hírt kap a bonus/malus fokozatáról). Ezzel a közbevetéssel csak arra akartunk utalni, hogy nemcsak az információs esemény/tevékenység lánc végén, hanem annak az elején is szerepet játszanak a felhasználók. Tehát nem szabad elfeledkezni arról, hogy az adatszolgáltatás is a felhasználás egyik - méghozzá igen fontos - módja. Végül rá kell mutatnunk, hogy a 'végző' jelző értelmét a rendszer határai szabják meg. Pl. a KSH-nak átadott statisztikákat ez az intézmény nyilván még tovább is feldolgozza, azonban ez a feldolgozás már a mi rendszerünk határaink kívül esik és így a KSH felhasználóihoz már semmi közünk; számunkra a KSH a 'végző'.

SZOLGÁLTATÓ

FEJLESZTŐ

VÉGSŐ FELHASZNÁLÓ



2.9 ábra: Felhasználók

Az IR rendszer ismereteivel nemcsak a működtetés, hanem a kialakítás során is kapcsolatba lehet kerülni. Ezért ebben a könyvben mi az információs rendszer **fejlesztőit** is felhasználóknak tekintjük. (Fejlesztő a szervező, a programozó, a minőségellenőr, a távadatátviteli szakember stb. Mivel a fejlesztés szó annyiféle szakmát, szaktudást takar, arra nézve nincs is egyetlen kifejező angol fogalom.) Talán furcsa, hogy mi például a rendszerszervezőt is felhasználónak tekintjük. Mi több, azt tartanánk helyesnek, ha ő maga és vezetője is ezt tenné. Ugyanis a szervező ugyanazokkal az ismeretekkel kerül kapcsolatba, mint az előző kétféle szereplő - csak más síkon. Bele kellene élnie magát a végső- és az alkalmazási-felhasználó szerepébe ahhoz, hogy valóban jó rendszert készítsen. Ellenpélda:

T 2.15 1991-ben éppen egy biztosítónál történt meg, hogy a szervező ‘elfeledkezett’ a viszontbiztosítási ismeretek elrendezéséről. Nem tekintette magát felhasználónak, vagyis nem látta át a saját maga kapcsolatát a vonatkozó ismeretekkel.

Sajátos felhasználói szerepet tölt be a **vezető**. Az IR kialakításához éppen úgy, mint annak működtetéséhez nemcsak szakmai tudás, hanem tág értelemben vett hatalom is szükségeltetik. Nem csak a pénzre és nem csak az egyéb erőforrások - emberek és eszközök - fölötti rendelkezés jogára kell itt gondolni. Mi sokkal jobban szeretnénk a ‘hatalom’ szó helyett a ‘gazda’ kifejezést használni. Az igazi vezető jó gazdaként viszonyul a szervezet ismereteihez, mint tág értelemben vett erőforrásokhoz. Arra törekszik, hogy okosan használja fel azokat a vonatkozó szervezet céljaira. Tehát a vezető is - felhasználó. Csak éppen más a viszonya az ismeretekhez, mint a másféle szerepű felhasználóké.

Remélhetőleg az olvasók már kezdik érezni, hogy mennyire nem felesleges a mindennapos értelemben vett felhasználói fogalomnak a fentebbi, informatikai célú kitágítása. Nemcsak az adatokat, a tevékenységeket, hanem az ismeretekkel kapcsolatba kerülő embereket is rendszernek kell tekinteni a rendszeren belül. Ha a szerepek nem világosak, ha azokat nem jól ‘játsszák’, akkor az IR többi része lehet technikailag akármilyen tökéletes, az IR nem fogja elérni a célját.

A pontot egy utolsó gondolatsorozattal kell zárunk. A D 2.7 definícióban az emberek **csoportjairól** is szó esett. Ez a dolog az első látásra talán meglehetősen. Az információt eddig a személyes emberhez kötött valaminek tartottuk. Miként lehetséges az, hogy most nézetet váltunk? Nem tesszük azt.

Az információ ugyan mindig személyes lényeg, mert az egyén képzettségéhez, képességeihez, volt korábbi ismereteihez, gondolkodási módjához, sőt: aktuális érzelmeihez kötődik. (Vö. azzal, hogy az ember nem hall meg valamit, illetve azt hallja meg, amit hallani akar.) Ez azonban egyáltalán nem zárja ki azt az evidens tényt, hogy vannak hasonló képzettségű, háttérű, feladatú stb. emberek. Marika ugyan nem pontosan ugyanúgy látja az anyagokra vonatkozó ismereteket, mint Pannika, de feladatuk közös: a készletnyilvántartás vezetése. Ez pedig mindkettő számára az alapvető ismereteket tekintve egyazon lényeg.

Akkor hát miért nem mondjuk ki, hogy az **egyén** mellett a **szervezeti egység** is felhasználónak tekintendő? Szívünk szerint ezt tennénk, mert első ránézésre nem Marika, a készletnyilvántartó egyedi személy, hanem az anyaggazdálkodási részleg egésze az 'igazi' felhasználó. Gondoljuk csak meg, hogy ha Marika kilép, megbetegszik stb., az ismeretek felhasználása nem állhat meg. Marikát új belépő vagy Pannika fogja felváltani illetve helyettesíteni, de az élet továbbhalad. Ezért könyveljük el, hogy a szervezeti egységek valóban felhasználónak tekintendők.

Két oka is van annak, hogy e ponton egy picit mégis haboztunk. Egyrészt az, hogy a cégek nem mindig célszerűen építik fel a szervezeteiket ill. vannak olyan ismeretek is, amelyek nem egységhez kötöttek. Az előbbi esetre példa az, hogy a cég az anyagok származási körzetei szerint szervezi az anyaggazdálkodás belső szervezetét. (Vö. a régi 'szocialista' és 'kapitalista' exporttal.) A dűzni az dűzni, a rá vonatkozó ismeretekkel viszont nem egy szervezet áll kapcsolatban. A másik esetre példa, hogy az ügyfél ismereteinek a kezelésére nem alakul külön részleg, ámde azokkal szinte minden szervezeti egység kapcsolatba kerül. Mármost az információk rendszerek sokszor azért nem sikeresek, mert - a közös ismeretekre való tekintet nélkül - a rendszerrészeket szervezeti egységek szerint alakítják ki.

Habozásunk másik okát egy történettel fogjuk megvilágítani:

T 2.16 1976-ban egy szállítmányozási cégnél 'számítógépesítésbe' fogtak. Két szervező dolgozott a feladaton. A tapasztalatlan kikérdezte az illetékes csoport vezetőjét, majd részletes kérdéseivel gyötörte a csoport egyik ügyintéző hölgyét. A tapasztalt pedig meghívta az összes ügyintézőt egy kávé melletti fesztelen beszélgetésre.

Mit gondol az olvasó: melyikük tárta fel alaposabban az ismeretigényeket? A kávézás közben kiderült, hogy a szállítóleveleket egyáltalán nem úgy kezelik, mint ahogyan azt a csoportvezető gondolja, szeretné, elrendelte stb. („Hja, a Jenő csak mondja a magáét, de ne tessék ám figyelni...!”) Azután azt is lassan-lassan ki lehetett hámozni, hogy Klárka - a régi 'bútordarab' - mindig szépen elboldogul a többiek ügyeivel, de Jucikát - a friss munkaerőt - még csak az igen egyszerű, sablonos dolgok intézésével merték megbízni.

És mi a végkövetkeztetés? Az, hogy szervezeti egység ide vagy oda, közös feladat így meg úgy, a legvégén mindig kiderül, hogy az elvileg azonos háttér, az elvileg megegyező feladat gyakorlatilag 'mindig azonos, de egy picit más'. Másképpen szólva: csak a lélektelen ügyintéző lát rutinfeladatot az ismeretekkel való viszonyában. Aki a saját munkáját emberinek tartja, az mindig személyesen viszonyul az ismeretekhez is. „In pluribus unum.” - a jelszót picit átértelmezve a sokban is az egyet, a sajátot keresi.

Így szerintünk nem az a helyes felfogás, ha vagy az egyént, vagy a szervezeti egységet tekintjük felhasználónak. Az ismeretnek vannak általános (vö. ügyfél), speciális (vö. készletnyilvántartás) ill. egyedi (vö. különös tartalmú szállítólevél) felhasználói.

Már tényleg csak egy gondolatfoslány maradt hátra. A hetvenes évek során a szervezetek mindegyike saját-bejáratú - mondjuk - 'számlázási rendszert' képzelt el, fejlesztett, programoztatott. Úgy, mintha az A és a B vállalatnál a számlázási ismeretek teljesen eltérőek lettek volna. Jóllehet már akkoriban is világos volt, hogy a számlázás egy olyan kvázi-általános ismeret-halmazt feltételez, amelynek jelentős része nem vállalatfüggő. Mára már ez a helyzet megfordult:

manapság ritka kivételnek számít a számlázás saját fejlesztése - inkább megvásárolunk egy céljainknak megfelelő kész **programcsomagot**.

Tekintettel a számlázás 'kvázi-általános' ismerethalmazára, úgy tűnhet, hogy a szoftvertétel a valóban célszerű megoldás. Azonban a feltételes mód indokolt. Azt ugyanis ne feledjük, hogy a szoftvereladó nem ismeri a mi felhasználóinkat. Csak feltételezett kép van benne a nálunk előforduló adatokról, eseményekről - a mi információs rendszerünk részleteiről. Nemigen lesz tekintettel a számunkra fontos szabványokra sem. Ez a kérdés viszont már átvezet a következő ponthoz.

2.5 A SZABVÁNYOKRÓL

A **szabvány** fogalmát meglehetősen sokan félreértik. Vannak, akik e kifejezés hallatán valamiféle nagyon aprólékos, skrupulózus, a szabad kezét megkötő - és ekképpen a jó megoldást korlátozó - szabályozásra gondolnak. Mások viszont a lényegét az átfogó természetű elvekben, betartandó regulákban, törvényekben, rendeletekben látják. Továbbá már maga a Szabványügyi Hivatal megnevezés is azt sugallja, hogy valamiféle 'hivatalos' dologról, szükséges rosszról van szó.

Mindezekben a nézetekben van valami igazság. Kétségtelen, hogy a szabvány lényege adott mértékig magában hordozza a kényszer momentumát is az elvek szintjén éppen úgy, mint a részleteket tekintve. Természetesen a jó informatikus ismeri és betartja a kötelező előírásokat, azonban ebben a könyvben egyáltalán nem a mások által való kényszeríttetés momentumát kívánjuk hangsúlyozni. Így a szabvány szó helyett szívesebben használnánk a **konvenció** megjelölést, amin itt nem annyira *hagyományt* (azt is), mint inkább *megegyezést* kell érteni.

T 2.17 Ezelőtt pár évvel az egyikféle biztosításomon csupa nagybetűvel és ékezet hiányosan, a másikon nem-magyar ékezetekkel, elől a 'Dr.' titulussal, a harmadikon pedig ékezet helyesen és hátul a 'Dr.' címmel ellátva jelent meg becses nevem. Ki is voltam tehát?

Mindhárom megoldással szemben én 'Dr. Halassy Béla' vagyok. Nem tehetek róla: ehhez szoktam. És roppant nagy a gyanúm, hogy az én ismereteimet nem kezelik helyesen, ha a nevem hol így, hol meg úgy jelenik meg a papírokon. A név írásmódja az ezernyi konvenció egyike, ami nem kényszer és nem is korlát, hanem éppen ellenkezőleg - ez teszi kellemmé az életemet. Ebben a tekintetben engem egyáltalán nem a változatosság, hanem az állandóság gyönyörködtet.

Az 'egyirányú utca' - hagyományos jelölésű - táblát csak a buta vagy a bűnöző tekinti korlátnak. Az épeszű ember ezt a megegyezési jelzést sokkal inkább a saját érdekét szolgáló, kényelmes és érthető tájékoztatásként fogja fel. Ezért az információs rendszerek esetében a szabvány úgy is tekinthető, mint az erkölcsi élet terén mondjuk a Tízparancsolat. A 'parancs' kifejezés csak addig és csak azt rémíti meg, ameddig és aki rá nem döbben arra, hogy a saját érdekeiről van szó.

A szabványok kapcsán felmerülő kényszer érzetét egy utolsó kitétellet akarjuk eloszlatni. Korlátozta-e bárki is e könyv szerzőjét arra, hogy művét a latin ÁBC betűivel írja, és ne - mondjuk - a japán katakana ill. hiragana írásmód jeleivel? Igen: volt ilyen **belső** kényszer. Ha ugyanis azt akarta, hogy sokan megértsék, amit ír, akkor nem is választhatott más megoldást.

Ha tetszik, ha nem, a szabványok az IR minden egyéb tényezőjét uralják. A fentebbi (T 2.17) történet ugyan csak az adatokra vonatkozott, ám az események és a tevékenységek is szabványosak kell, hogy legyenek. Aligha kétséges, hogy például a könyvelést adott előírások szerint kell elvégezni. A könyvelésben nem pusztán az ismeret általános elrendezése kötött, hanem az is szabály, hogy mikor és hogyan kell a könyvelési eljárásokat elvégezni. Éppen ezért - szemben e pont alcímével

- az informatikában nem csak a szabványokról, hanem a szabványos eljárásokról is szót kell ejteni. A fejlett informatikájú országokban az ismeretek elrendezésére és kezelésére vonatkozó eligazításokat így nevezik: **szabványok és szabványos eljárások** [standards & procedures].

Ámde az információs rendszer nemcsak az adatoknak és a velük kapcsolatos - események és tevékenységek által alkotott - eljárásoknak a szervezett együttese. Az erőforrás és a felhasználó is a rendszer része. Az ismeretek értelmezhetősége és az azok kezelésére vonatkozó megkötések miatt az emberek nagy nehezen, de mégis elfogadják az adatokra és az eljárásokra vonatkozó konvenciókat. Ám az erőforrások bizonyos tényezőit tekintve, a felhasználót illetően pedig véglegesen zavarba jönnek, ha a szabvány szót említik. Ezért az alábbiakban e két tényezőre kell kitérnünk.

Először foglalkozunk az **erőforrásokkal**: a pénzzel, az idővel és az eszközzel. Kezdjük a talán legkönnyebbrel. Elvileg az informatikai **eszközök** szabványosak. Gyakorlatilag viszont mindenki tudja, hogy a fizikai eszközök, a hardverek nem is kompatibilisek egymással, a szoftverek (verziói) pedig még csak távolról sem hasonlítanak egymásra. Az egyik és a másik billentyűzet, képernyő és nyomtató teljesen másként működik, mint a másik. Az informatikus sem azt nem várhatja el az eszközgyártóktól, hogy 'szabványaikat' egymással összehangolják, sem azt, hogy a Word ikszedik verziója az ipszilonadikkal egy húron pendüljön. Ebben a helyzetben csak a belső szabvány segít.

A lényegét talán az ellenpéldákból lehet igazán megérteni. Manapság jellemző a tehetősebb cégekre, hogy - a külső és a belső szabványok hiányában - egyfajta 'informatikai kiállítást' rendeznek be. Vesznek egy pár - egymással sem teljesen kompatibilis - mikrogépet. (Bizony, még ez is előfordulhat. A szerző e könyvet egy olyan pécén kezdte írni, amely a Word-öt - hogy is mondjuk - nem mindig fogadta el. A gép alaplapjának a kicserélése után a gondok megszűntek.) Majd beszereznek néhány fejlesztési célú 'munkaállomást', hogy az egész kavalkádot megkoronázzák a 'megaminis' kategóriájú középgepek egy-két változatával. A - most csak példálódzunk - félig-klón pécék, a munkaállomások és minigépek mindegyikén más-más adatbázis-kezelő verziót kell installálni - a gépi korlátok függvényében. Manapság ezt a gusztustalan, célszerűtlen, pazarló, nap mint nap változó eszközvásárlást nevezik informatikának, a nem-standard eszközök miatti belső önkorlátozásokra szólító felhívásokat pedig a haladás gátjának tekintik.

A mai legújabb jelszó, a **nyílt-rendszerek** [open systems] mögött a gátlástalan és rendezetlen - szabvány nélküli - kereskedelem húzódik meg.

T 2.18 A 'nyílt rendszer' jelszó jegyében cégünk az egyik gépet leváltotta a korszerűbb másikkal. Mivel a hardver-szállító szabványai nem bizonnyultak azoknak, az átállás több hetes munkánkba került. Vajon mi történt volna, ha a rendszer nem ennyire 'nyílt'?

Mivel az eszközszállítók ennyire gátlástalanok, a cégeknek maguknak kellene magukat korlátozniuk - belső eszközszabványokkal. Csak olyant veszünk, ami... Persze ezt az elvet könnyű kimondani, de a gyakorlatban más a helyzet.

A másik két erőforrást tekintve még nagyobb az elvi zűrzavar. A pénzről és az időről van szó. A kettő összefügg. Az információs rendszer fejlesztése is munka. Olyan feladat, ami pénzbe és időbe kerül. Egyszer volt, hol nem volt... 1975-ben még itthon is úgy mérték a fejlesztési tevékenységet, mint Amerikában. Ennyi meg ennyi - COBOL - programsor leírása annyi meg annyi időbe kerülhet, ergo egy száz-soros COBOL-program ára annyi, hogy...

Természetesen mindenki azonnal láthatja, hogy az időnek és a pénznek ez a fajta 'szabványos' meghatározása mennyire visszás és megfoghatatlan. Mert nem a leírt programsorok száma, hanem a program által szolgáltatott ismereteknek a használhatósága, fontossága stb. a lényeg. Ámde az ismeret értékének a mérése még ma sem megoldott, no meg nemcsak a programon, hanem az alapismeretet adó valakin is múlik, hogy mennyire használható a végső információ.

Az idő és a pénz mint erőforrás hasznosulásának a mérhetetlensége általában odavezetett, hogy a programsor/idő/forint egykori mércéket elvessek. Ezt jól is tették, mert az IR kialakítása - mint tudjuk - nem csak programozást jelent. Ám az sem elfogadható állapot, hogy teljesen elhagyjuk az információs rendszerek fejlesztéséből az erőforrás-szabványokat. Minden rendszert 'tegnapra' kellene elkészíteni - a fejlesztő pedig gebedjen meg. Az IR-nek tényleg vannak szám szerint is megfogható, a bonyolultságot is viszonylagosan jól tükröző mutatói. Ezért nem érzésre, hanem e változó, nem teljesen objektív, de mégis valamiféle kvázi-tényszerű jelzőkre kellene alapozni a fejlesztés erőforrás-szabványait. És nem a hasraütésre - ami a mai IR-fejlesztések egyetlen becslési eszköze.

Mindez már átvezet bennünket ahhoz a felvetéshez, hogy vajon a felhasználói tényezőt tekintve miként kell értelmezni a szabvány lényegét? Szabványosítható maga a felhasználó? Természetesen és remélhetően nem az. Nincs is rosszabb az ismereteket, az azokra vonatkozó eljárásokat, az eszközöket mindig mereven, az úgymond szabályok szerint értelmező alkalmazási-felhasználónál. A roppantul sarkos ügyintézőnél, a feladatáért szabályszerűen lelkesedő vezetőnél, a precíz fejlesztőnél - no meg a toporzékoló kuncsaftnál.

T 2.19 Egyébként igen értelmes barátom, aki a piackutatás területén jártas, pár éve felmérési eredményeit számítógépen akarta látni. A dolog csak nem akart összejönni. A programozó ezt meg azt eltolt, barátom pedig nem értette meg azt, hogy...

E történet kapcsán két mondanivalónk van. Az egyik az, hogy valóban vannak felhasználói szabványok is. Konvenciók, közös értelmezések. Az előző pontban mi a felhasználót nem egy abszolút valakiként, hanem az ismeretekkel létesített viszonyában határoztuk meg. Akasztják a hóhért... Tessék csak megfigyelni azt, hogy a banki ügyleteket programozó személy miként viselkedik akkor, amikor maga esik be ugyanabba a bankba - ezúttal ügyfélként! Távoli emlékként talán ekkor merül fel benne először és igazán az, hogy - felhasználói - szabványokra lenne szükség. Mert nem jutunk sokra akkor, ha a piackutató mást ért 'mintán', mint az, aki a kutatás eredményeit olvassa. Barátom sem jutott dűlőre azzal a szegény szerencsétlen programozóval, mert a kiemelések, a hangsúlyozások, a piackutatói szempontból fontos lényegek a programozónak ismeretlenek voltak.

A felhasználói szabványok lényege a közös nyelv, a közös fogalomkészlet. Ha nem értünk egyet abban, hogy mit is jelent például a minimális kockázat, akkor a biztosítási információs rendszer 'rendetlenkedni' fog. Az egyik felhasználó így tolja, a másik úgy vonja. A fejlesztő pedig egyiket sem szolgálja ki.

Itt térünk rá a másik mondanivalóra. A szabványok is saját rendszert alkotnak magán az információs rendszeren belül. Ha a piackutatói ismeretek kezelésére a számítógépet alkalmazzuk, akkor tudomásul kell vennünk bizonyos dolgokat. A T 2.19 példa kapcsán a piackutatásban zseni barátom nem vette figyelembe azt, hogy a képernyő és a papír adott méretű (szabványos) és a betűkészletnek ill. az elrendezésnek korlátai vannak. Az IR a felsorolt tényezők szervezett együttese: ezért a szabványok rendszerezett szemlélete alól sem vonhatja ki senki magát.

Ezt a pontot egy szomorú, de tanulságos történettel kell zárunk.

T 2.20 1975-ben a Chase Manhattan Bank 22. emeletéről dobta ki az osztályvezető a programozó nem-szabványosan megírt valamijét. „Kezdje előlről és tartsa be az előírásokat,” - hangzott a kemény ítélet - „különbön veheti a sátorfáját!”.

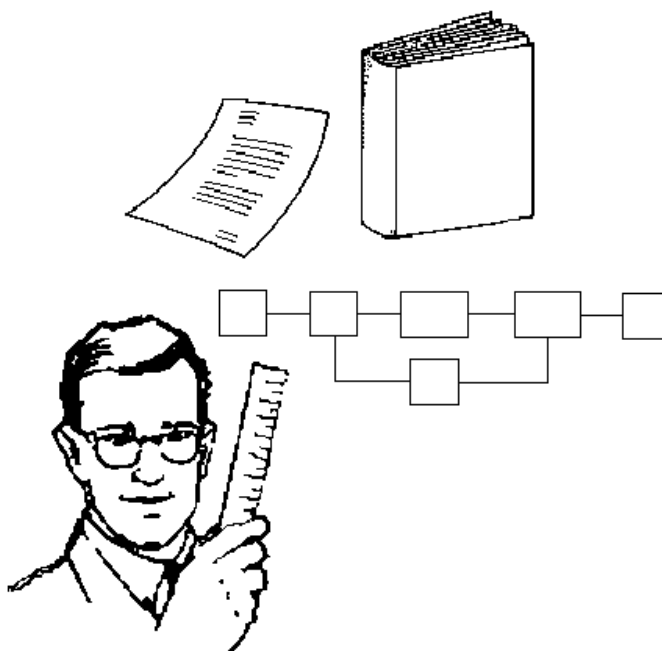
Felesleges szigor? Túlkapás? Félreértelmezés? Nem az. Az adatadminisztráció osztályvezetőjének igazat kell adnunk. A szabványt a végső- és az alkalmazási-felhasználó kellemes, jól-

tájékoztató dolognak tekinti és szinte megköveteli azt, hogy a neki szóló ismeret egyértelmű elrendezést, tartalmat és formát öltjön. Ha viszont a mindennapi polgárnak kell kitöltenie egy papírt, akkor vajon ugyanígy ragaszkodik-e az egyértelműséghez, a szabványokhoz? Természetesen nem, ha erre a fejlesztő által kialakított rendszer nem kényszeríti. Mindannyian örülünk annak, ha például egy rosszul megfogalmazott adóbevallási ívet kapunk - mert ebben az esetben nem mi vagyunk a 'fogók'. A Chase osztályvezetőjének igaza volt akkor, amikor elvágta a „ha akarom vemhes, ha nem akarom nem vemhes” értelmezésre alapot adó megoldásokat.

A szabvány az egyenes beszéd eszköze. Nem mindenki képes és főleg nem mindenki akar tisztán beszélni. A bölcs informatikai vezető ezt tudja. E könyv szerzőjének a Chase-beli tevékenységét azzal kellett kezdenie, hogy el kellett olvasnia a négy méter széles (!) polcon felsorakozó előírások gyűjteményét. Ez azt jelenti, azt úgy kell csinálni, csakis amúgy lehet leírni... Igen fárasztó volt ez a tapasztalat - de megérte. Két lépést sem tehettem volna a Chase-ben ezeknek az alapvető dolgoknak az elsajátítása nélkül. Programomat, jelentésemet a 22. emeletről dobták volna ki...

És idehaza? Nálunk a szabványokat a senki által el nem olvasott, nemhogy a négyméteres polcra világítóan kirakott, hanem az X titkárnő megkörnyékezője által sem elérhető 'szervezési és működési szabályzatok' (SZMSZ-ek) próbálják pótolni. Ezek nem feszes és eligazító tájékoztatások, hanem igen körmönfontan megírt lamenták a 'jó lenne, ha...' szintjén. A változásokkal egyidejűleg sohasem kerülnek karbantartásra - mindig elavultak. Szemben azzal az elvvel, hogy az információs rendszeren belül a szabványok maguk is sajátos rendszert alkotnak, nálunk az egyik 'ukáznak' semmi köze sincs a másikhoz. Nálunk a szabványok egyik feladatukat sem töltik be.

A valódi informatikus által látott információs rendszerben ugyanis a szabvány hármas szerepet lát el. Egyrészt *elgazít* bennünket - az IR minden résztvevőjét - a fejlesztési munka előtt és annak során az 'ezt úgy illik, azt úgy kell' szintjén. Másrészt *korlátként* épül be magába a rendszerbe: például csak a szabványosan írt nevet fogadja el az azt kezelő program. Végül a szabvány *tájékoztatóként* is szolgál a felhasználónak az ismeretek kezelése és kijelzése során. Pl. itt szoktad látni ezt, ott pedig amazt.



2.10 ábra: Szabványok

A pont és a teljes fejezet záró gondolataként utalnunk kell az informatika mai általános helyzetére. Számítógépeink és szoftvereink az emberi alkotóképesség igen kiemelkedő eredményei. Akkor vajon minek tulajdonítható az a tény, hogy manapság az informatika mégsem elismert és önálló tudomány?

A választ éppen a szabványok hiányában kell keresnünk. Még ma is divatos a rendszerfejlesztőt az angol 'systems-engineer' szó alapján 'rendszermérnök'-nek titulálni. Most felejtjük el, hogy az informatikához nem túl sokat értők angolul sem tudnak helyesen, mivelhogy az 'engineering' szó valójában tervezést jelent, és ezért így a 'systems-engineer' helyes fordítása nem 'rendszermérnök', hanem 'rendszertervező'! A szavakon lovaglóknak a saját következtelenségeire kell itt irányítanunk a figyelmet.

Nekünk alapján véve közömbös, hogy a tervezőt mérnöknek fordítják-e. Legyen mérnök. A mérnök egy tapodtat sem tud tenni konvenciók hiányában. Ehhez képest meglepő, hogy a 'rendszermérnök' terminológiát használók hada egy szót sem ejt a megegyezésekről, a szabványokról.

Erről jut eszünkbe, hogy még nem is definiáltuk a szabvány fogalmát:

D 2.8 A szabvány az IR valamilyen tényezőjére vonatkozó megegyezés.

Ez a szűkszavú meghatározás több dolgot eltakar. Például nem utal arra, hogy a megegyezés a rendszer határain belüli vagy kívüli-e, vagyis milyen érvényű. Európai, országos, cégen belüli? A definícióból kimaradt a kényszer mozzanata is. Kötelező, ajánlás, jó lenne? A tárgyról sem esik szó: adatra, tevékenységre, eszközre vonatkozik? A jelleget sem árulja el: tartalmi vagy formai-e az előírás?

Mint már utaltunk rá, a szabványt nem ilyen vagy olyan kényszernek, hanem felismert szükségserűségnek kell tekinteni. Az 'úgy kell' gondolat korlátozó. Az 'úgy jó, mert úgy értik' megközelítés felszabadító. A számítástechnikus szorong, mert korlátos szellemben él. Jajj, ezt lehet-e, szabad-e. Viszont az informatikus felszabadult. Mindent lehet, mindent szabad - a szabványok keretei között.

2.6 IR ÖSSZEFOGLALÓ

A tudatos ember azért él a világban, hogy azt megváltoztassa („ront vagy javít, de nem henyél” - szól a Bánk Bán egyik részlete). Ahhoz, hogy az emberiség az uralma alá hajthassa a Földet, azt rendszeresen meg kell ismernie. Ahhoz pedig, hogy 'rendszeres' ismeretekre tegyünk szert, három dolog szükséges.

Először is kell, hogy legyen valamilyen elhatározott valós **célunk** és arra való akaratunk. Nem légtüres térben élünk, mozgunk, dolgozunk. Vannak feladataink, amelyek ehhez vagy ahhoz a valós rendszerhez (VR) kötnek bennünket, legyen az egy vállalkozás, egy intézmény, vagy bármi más. A szervezetben bennünket egy adott alkalmazási környezet vesz körül a szervezet céljai ill. funkciói által meghatározott módon. Feladatunk az, hogy ezt a környezetet tökéletesítsük.

Az értelmes és közös munkának a második feltétele az, hogy legyen a szó tág értelmében vett, tudatosan kialakított információs **képünk**. Sokszor nem maga a valóság, hanem annak „földi mása” - tehát az, amit tudunk róla - a meghatározó. Ezért van szükségünk az információs rendszerre (IR). Az ismeretek pedig olyan kvázi-szubjektív tükörképek, amelyeket megegyezések uralnak. Ezért külön kell beszélnünk az alkalmazásával együtt létező információs környezetről.

Az emberi megismerőképeség korlátos, az ismeretek sora pedig végtelen. Így a valóság megismerése ma már elképzelhetetlen a harmadik kellék, a megfelelő technikai *megoldás* nélkül. Ezért van szükségünk a jó eszközrendszerre (ER). Az eszközök ma már annyira változatosak, hogy egy-egy szervezetben dönteni kell az alkalmazott hard- és szoftver típusa, jellege felől. Így az alkalmazásnak a környezete szükségszerűen egy eszközkörnyezettel is párosul.

Tehát a valós cél, az információs kép és a technikai megoldás hármasa három rendszer (VR - IR - ER) összehangolását igényli. Ebben a könyvben a középső - az információs - rendszerre fektetjük a hangsúlyt nem feledve sem a cél, sem az eszköz viszonylagos önállóságát. A három nagy 'kör' egymást feltételezi, átlapol egymással, egyik sem képzelhető el a másik nélkül. Ez a tény viszont nem zárja ki azt, hogy az IR-t önmagában is vizsgáljuk.

Az információs rendszer rendkívül eltérő természetű tényezőknek a szervezett együttese. A 'szervezett' jelző arra utal, hogy tudatos munkával kialakított, nem pedig ad-hoc, véletlenszerű együttesről van szó. A tudatosság két dolgot jelent. Egyrészt azt, hogy összhangba kell hoznunk egymással a különböző természetű tényezőket; pl. az adatokat a tevékenységekkel, az eseményekkel stb. Másrészt azt, hogy egy-egy tényezőcsoporton belül is rendet kell teremtenünk; vagyis el kell rendeznünk pl. az adat és a másik adat, az esemény és a többi esemény, a felhasználók stb. egymás közötti viszonyát is.

Már az információs rendszer jelzője is mutatja, hogy annak az alapvető eleme - amihez az összes többinek igazodnia kell - maga az ismeret. Az emberek által nem értelmezett, de érzékelhető, észlelhető és felfogható - vagyis: értelmezhető - ismeret az adat. Az általunk nem értelmezhető közlés számunkra még csak nem is adat, mert nem tudunk vele mit kezdeni. Az adatból a személyes értelmezések során válik személyes ismeret, azaz információ. Ugyanabból az adatból az egyik ember több, a másik kevesebb új ismeretet szűr le a közlés tárgyával kapcsolatos korábbi háttere, képzettsége szerint. A 'több és kevesebb' egyes informatikusokat arra csábított, hogy az információt matematikai úton próbálják meghatározni. Ez az út azonban sikerrel nem járható, mert az ismeret elsősorban minőségi és nem mennyiségi természetű; és a mennyiség nem konvertálható minőségbe.

Új adatok a valóság változásait kísérő információs események során születnek úgy, hogy az ember információs tevékenységeket hajt végre vagy hajtát végre a megfelelő eszközzel. Eseménynek tekintendő a tevékenység indító mozzanata éppen úgy, mint annak lezáró momentum. Ugyanis a tevékenység mindig változást okoz. Ha más nem is, az idő mindig változik és az idők során megváltozhatnak a bennünket érdeklő dolgok állapotai is. Az adatokból az adatkezelő és -előállító, a vezérlések által összehangolt műveletekkel, szelekcióval és összekapcsolással más, immár információvá értelmezhető ismereteket készítnék. Ez a végcél, ami éppen ezért magát a tevékenységet is meghatározza.

Az IR kialakításához és működtetéséhez erőforrásokra van szükségünk. Szűk értelemben az időt, a pénzt és a technikai eszközöket nevezzük erőforrásnak. Az idő, a pénz és az eszköz olyan tényezők, amelyeket önmagukban és egymással való viszonyukban is el kell rendezni. Azt azonban tudnunk kell, hogy e három dolog csak nagyon korlátosan konvertálható egymásba. A 'kevesebb idő - több pénz' illetve a 'több pénz - több gép' jellegű gondolatokkal óvatosan kell bánni. Az IR rendszer sikerét nemcsak az erőforrások szűkössége, hanem a helytelen erőforrás-beosztás is kockáztathatja.

Bár az IR legalapvetőbb eleme az ismeret, legfontosabb tényezője mégis az azzal kapcsolatban álló ember, vagyis a felhasználó. Az emberek és csoportjaik különböző minőségekben kerülnek kapcsolatba az ismeretekkel, vagyis az IR kialakítása és működtetése során eltérő szerepeket töltenek be. Az alkalmazói- és a végső-felhasználón kívül felhasználónak kell tekinteni a fejlesztőt és magát a vezetőt is. Az IR sikere ugyanis azon is múlik, hogy a fejlesztő mennyire azonosul a végső- és alkalmazói-felhasználó gondolataival, illetve a vezető mennyire tekinti magát az ismeretek gondos gazdájának.

Az IR minden tényezőjét szabványok és szabványos eljárások szabályozzák. Kétségtelen, hogy a szigorú apró előírás és a nagy átfogó törvény is szabvány. Azonban ne feledjük, hogy az ismeretek az emberek közötti kommunikációnak a tárgyai. Így e közlésnek a megegyezései, hagyományai maguk is szabványoknak tekintendők, hiszen olykor nagyobb hatással vannak az információs rendszerek sikerére, mint a merev előírások. A szabványokat is szervezett együttesként kell tekinteni. Az például képtelenség, hogy a felhasználóknak előírjuk az ékezetes magyar betűk használatát (ismeret), viszont német pultot és angol szövegkezelőt (eszköz) adunk a kezük alá.

Amint látjuk, az információs rendszert egyáltalán nem csak technikai oldalról lehet és kell megközelíteni. Ezért az IR kialakításához és működtetéséhez nem a szó hétköznapi értelmében vett számítástechnikusokra van csak szükség, hanem a szó nem-hétköznapi értelmében vett informatikusokra is. Az IR igen bonyolult inhomogén rendszer, amely különböző módokon közelíthető meg. A következő fejezetet ezért a megközelítési módok ismertetésére szánjuk.

ELLENŐRZŐ KÉRDÉSEK - 2

Az olvasó tegye fel magának azt a kérdést, hogy érti-e az információs rendszer lényegét. Vegye sorra annak tényezőit, és próbálja meg önállóan meghatározni a tényezők önálló lényegét illetve a tényezőcsoporton belüli és kívüli viszonyát. A legfontosabb az adat és az információ helyes szemlélete. Megértésüket segítheti a gondolati kísérlet. Figyelje meg, hogy barátai miként 'reagálnak' egy adatra.

- 201 Információt (I) vagy adatot (A) takar-e Önnek ez a mondat: 'E mű szerzőjének 64 négyzetméteres lakása van.'?
- 202 A fogpaszta reklámokban naponta elhangzik a 'háromszoros profilaxis' szöveg. Ez Önnek adat (A), információ (I) vagy netán egyik sem (S)?
- 203 Ön az érzékelés (1), az észlelés (2) illetve a felfogás (3) fázisai közül meddig jut el a következő ismeret értelmezésében: 'Az anyagminta desszinszáma 12657-es'?
- 204 Állítson fel egy számszerinti sorrendet! Megmutatjuk az X dűzni eladási számláját az alábbi embereknek. Melyik nyer belőle több információt? Vigyázat, csalunk!
 - főkönyvelő
 - az értékesítési osztály vezetője
 - a számlát kiállító személy
 - a bérosztály egyik dolgozója.
- 205 Fejezze ki egy-egy számmal, hogy Önben mennyi határozatlanságot (0-100%) szüntet meg a következő két közlés: 'A szerző múlt szombaton nem fogott egy darab halat sem.' 'Lajos nem szereti a tökfőzeléket.'! Mondja el saját szavaival, hogy a határozatlanság fogalma miért labilis, tehát miért nem ahhoz kötjük az információ lényegét.
- 206 Ön szerint mi a fontosabb: a közlés tartalma (1), annak formája (2), egyik sem fontosabb a másikonál, mert a kettő csak együtt jelenthet adatot (3). Ne csak válaszoljon, hanem próbáljon meg érvelni is!

- 207 A munka- és bérügyi IR-ben az alábbiak melyike információs esemény (E) és melyike nem az (N)?
- Kovács megnősült
 - Kovács mindig másodikán kap fizetést
 - Kovácsnak betörték az ablakát
 - Kovács elkezd keresgélni a számítógépen.
- 208 Ön szerint melyik információs tevékenység (E) és melyik nem az (N)? Melyik lehet az is, meg nem is (I)?
- matatunk az Internetben
 - sokszorosítjuk a kézikönyvet
 - kikeressük papírjaink (!) közül az X-féle számlákat.
- 209 A projektek emberi erőforrásigényét ‘embernapokban’ szokták kifejezni. Mondja el saját szavaival, hogy ez miért jó és miért rossz! Gondoljon a mennyiség és a minőség kérdésére!
- 210 Mondja el saját szavaival, hogy az embernap, a pénz és az idő miért nem konvertálható egymásba? Adjon egy példát!
- 211 Számlákat kell számítógépre vinni. Van egy tűrhető gyorsaságú 386-os gépe. Vásárol egy kétszer gyorsabb 486-osat. Ön szerint mennyivel fog gyorsulni a számlabevitel: kétszeresére (1), semennyivel (2), lehet, hogy lelassul (3)?
- 212 Ön a könyvtárból egy művet akar kikölcsönözni. Mondja meg, hogy ki a végső- és ki az alkalmazói-felhasználó e példában?
- 213 Magyarországon a csekkszámok számait egységesítették. A rendszer melyik tényezőjéről van szó? Jó-e (J) ez a megoldás, vagy rossz (R), netán ‘is’ (I)? Gondoljon a szabványok többszörös természetére és azok céljára!
- 214 Már jó ideje annak, hogy a szerző két darab százast tett az egyik könyvébe, de már nem tudja, hogy melyikbe. Van 200 forintja (V) vagy nincs (N)?

3. A RENDSZER SZEMLÉLETE ÉS FEJLESZTÉSE

Az IR roppantul összetett inhomogén tényezőegyüttes. Ezért nem csoda, hogy a hétköznapi emberek, akik nincsenek tisztában e tényezők viszonyaival illetve a szakemberek, akiknek pedig éppen ezeket az összefüggéseket kellene ismerniük, annyiféle eltérő módon közelítik meg az IR-t. Vannak, akik az eszköz, mások az eljárás (esemény és tevékenység), ismét mások az ismeret oldaláról szemlélik az információs rendszereket. Ezért felvetődik az a legelső kérdés, hogy az előbbieik közül vajon *melyik a helyes szemlélet* illetve egyáltalán létezik-e egyetlen ilyen.

Azt is észben kell tartani, hogy az IR a valós rendszerhez képest másodlagos. Ez nem fontossági, hanem eredendőségi minősítés. A 'másodlagos' jelző mögött a tükörkép fogalma húzódik meg. Mivel az ismeret tükörkép, az elsődlegeshez képest többszörös is lehet. Sőt: tekintettel arra, hogy eszközöket használunk, az emberi fejben, a papíron és a számítógépen lévő ugyanazon tartalmú ismeretnek mindig több megjelenési formája is van. Kevesen ismerik fel, hogy az esemény és a tevékenység is tükörkép, mert többféleképpen megfogalmazható, többféle formát öltő, választható megoldású tényező (vö. P 2.1). Ezért fel kell tennünk azt a második kérdést is, hogy *milyen absztrakciós szintjei vannak az IR-nek*.

Végül - éppen a szemléleti homályok miatt - a laikus és a szakember egyaránt zavarban van az IR fejlesztésének a lényegét illetően. Általánosan elterjedt pl. az a hamis nézet, hogy a fejlesztés = programozás. Persze a jártasabb körökben ez a sima képlet a fejlesztés = szervezés+programozás (netán: tervezés+programozás) összefüggésre módosul. Azonban úgy, hogy éppen a többletnek, a szervezésnek ill. a tervezésnek a lényege 'opálisan' meghatározott. Ezért választ kell adnunk a harmadik kérdésre, amely így hangzik: *miben áll a fejlesztés lényege*, valamint mi annak a viszonya a szervezéshez, tervezéshez, programozáshoz.

Ennek a fejezetnek az a célja, hogy bemutassa az IR helyes és helytelen megközelítéseit, a célszerű szemléleti módokat, az IR szemléleti szintjeit és vetületeit, valamint a fejlesztési tevékenységek körök lényegét.

Fejezetünk tehát két - egymással összefüggő - tárgykört fog érinteni. Az első pontban a rendszer vizsgálati szempontjairól és az IR makrostruktúrájáról lesz szó. A második pontban a gyakran, de nem tudatosan alkalmazott fejlesztés, elemzés, tervezés, szervezés, specifikáció, dokumentáció stb. fogalmak lényege kerül megvilágításra.

3.1 RENDSZERANALITIKA

3.1.1 A rendszer vizsgálati szempontjai

Az információs rendszerek vizsgálata két szinten történhet. A *konkrét* rendszer feltárása a **rendszerelemzés** [systems analysis] feladata. Ennek egyes kérdéseire ebben a könyvben is kitérünk, ámde behatóbb tárgyalására majd csak egy másik kötetben kerítünk sort. Ahhoz ugyanis, hogy a bennünket adott helyen és adott időben foglalkoztató speciális rendszert jól megismerhessük az szükséges, hogy tisztában legyünk a rendszerek mint *absztrakt* lényegök általános vonásaival. Az X **rendszerelemző** [systems analyst] sohasem lesz igazán képes a - például - Y-féle értékesítési információs rendszer feltárására, ha nem él benne egy kerek kép arról, hogy milyen módon kell általában megközelíteni 'a' rendszert.

Az információs rendszert nagyon sokféle szemszögből lehet vizsgálni. Ezek az aspektusok egymástól sem függetlenek, hanem összetett kapcsolatokban állnak egymással. Ráadásul az emberi tényező nagyon fontos szerepet játszik az ilyen rendszereknél, mivel az ismeretek személyekhez kötöttek. Ezért nincs könnyű dolga az informatikusoknak. Egyidejűleg kell átlátniuk az *objektív* és *szubjektív* résztényezők garmadáját úgy, hogy közben egy pillanatra se veszítsék szemük elől magát az egészet sem.

Vegyünk csak például egy autót! Mennyi mindenre kell figyelniünk a gépkocsi megvételekor, holott egy viszonylag egyszerű rendszerről van szó. Mi az ára? Mennyit fogyaszt? Mekkora az adója? Hogy néz ki? Mekkora a motorja? Mi van és mi nincs benne az alapváltozatban? Mik az extrák? Hol fogjuk tárolni? Ki és hol fogja vezetni? Családi vagy luxus alkalmazású lesz-e? Ha egy gépkocsival kapcsolatosan száz - általános jellegű - kérdésünk merül fel, akkor vajon mennyi mindenre kell figyelniünk egy IR kapcsán, ami sokkal bonyolultabb rendszer?

Intermezzo

Manapság szinte sehol sem oktatnak - és alkalmaznak - valódi IR-vizsgálatot. Talán azért nem, mert egyelőre nem létezik IR-tudomány sem? A mindennapos helyzet az, hogy a konkrét rendszerek különböző aspektusaiért más és más felel és senki sincs, aki átlátná az egészet. Már meg ne haragudjanak, de sem a felső-, sem a számítástechnikai vezető nem alkalmas erre a feladatra. Azért nem, mert ők csak adott - az egész szempontjából korlátozott - aspektusból szemlélik az IR-t. Az egyetlen helyes és kézenfekvő megoldás az lenne, ha alkalmaznának egy 'főinformatikust', pl. a főmérnök analógiájára. Azonban ezt ma sehol sem teszik.

Az első és legalapvetőbb vizsgálati szempont a **cél**. Az embereket a tudatosság jellemzi, ezért a rendszertől mindig valami kézzelfogható eredményt várnak el. A baj az, hogy sokszor már magával a cél-fogalommal sem vagyunk tisztában. Az egyébként roppantul csúnya és magyartalan 'elvárás' szó kiválóan mutatja a félreértéseket. Az akarat és a józan ész párosa által behatárolt cél helyét sokszor a vágy, a 'jó-lenne-ha', az 'úgy-gondoljuk', a 'mi azt szándékozzuk' veszi át. Ez a gondolkodásmód pedig alapot ad az *irreális* célmeghatározásnak. Nem tartozik ugyan szorosan a cél témakörébe az erőforrások kijelölése, azonban az irreális célkitűzéseket jól mutatja a következő történet:

T 3.1 Egy 1980 táján készített ENSZ-felmérés szerint az IR-fejlesztési projektek rész céljai teljesen elrugaszkodnak a valóságtól. Pl. 20 fejlesztésből 19 nem valósult meg a kitűzött időre. A legnagyobb időbeli eltérés a 700 százalékot is meghaladta. 10 fejlesztésből 9 költségesebb volt a tervezettnél, és kilencszerez 'tévedés' is akadt.

Ez a kitérő már átvezet bennünket ahhoz a gondolathoz, hogy természetesen a célt sohasem szabad egyetlen egy homogén valamiként felfogni. Az információs rendszer egy adott alkalmazási környezetben, meghatározott technikai feltételek közepette létezik. Már ebből is következik, hogy a cél többszörös lehet. Mivel pedig a célok általában nem (vagy az erőforrások hiánya miatt nem is lehetnek) egyenrangúak, tisztában kell lenni azok *rangsorával*. Ráadásul a cél részcélokat is felölölhet, ezért figyelni kell a *célhierarchiára* is.

E ponton ismét arra kényszerülünk, hogy megtartsuk nevelési kiselőadásaink egyikét. A célt sokan túl hétköznapiak érzik vagy pedig a lényegét akarják elrejtetni. Tény az, hogy ma a **prioritás** (mi több, az elsődleges, másodlagos stb. prioritás) kifejezés vette át a helyét. „Az a prioritásunk, hogy...” Ne tessék ilyen fordulatokat használni! A prioritás ugyanis elsőbbséget - rangsorbeli helyet - és nem célt jelent. Ezek után miként hangzik a 'másodlagos elsőbbség' fordulat? Az értelmes informatikus **prioritási sorrendet** emleget. A célok ranglistáján az A cél megelőzi a B-t (ahhoz képest előrébb áll), a B a C-t stb. Egy-egy részcélon belül maguknak az

alcélokak a prioritási sorrendjét is meg lehet állapítani. A 'lehet' szót itt komolyan kell venni: két részre osztva egyenlő mértékben fontosnak is tarthatunk.

A rendszer második vizsgálati szempontja a **tartalom**. Itt nem 'másodlagos' aspektusról van szó. A cél és a tartalom a gyakorlatban elválaszthatatlan. Az információs rendszer tartalmát az ismeretek mögötti fogalmak jelentik. Sajnos mindenki tudja, hogy mi is az a 'kórház', ezért még az informatikai szemszögből laikusnak is akad elképzelése arról, hogy mit is jelent a kórházi IR. Betegeket, orvosokat, ápolókat, ágyakat, labort, műszereket, orvosságokat, műtéteket stb.

A tartalom nem más, mint az alkalmazási környezet lényege. A kórházi és a járműjavítói IR ugyan sok tényezőben átfed (itt is, ott is vannak szervezeti egységek, emberek, műszerek, anyagok stb.), azonban az ember és a kocsik 'reparálása' nyilván nem azonos dolog. Az informatikusnak bele kell élnie magát ebbe a tartalomba, azonosulnia kell vele. Bár a rendszerfejlesztés - mint erre majd rámutatunk - alapjában véve módszertani tudomány, szemben a semleges, hasonlóan módszertani jellegű diszciplínákkal (pl. matematika), nem teljesen független az aktuális tárgytól, tartalomtól. A matematikus számára közömbös, hogy az X képletet mire használja. Az informatikusnak sohasem lehet mindegy, hogy saját 'képleteinek' mi a tárgya.

Az IR a tényezők szervezett együttese. A tartalmát képező dolgok egyáltalán nem függetlenek egymástól. Ezért alapvető vizsgálati szempont a **struktúra**, a 'hogyan épül fel' kérdésre adott válaszok összefoglalása. Az informatikusnak tudnia kell, hogy a konkrét, az éppen vizsgált rendszertől függetlenül miként épülnek fel általában az információs rendszerek. Kell, hogy legyen egy általános képe, mintája, modellje a IR szerkezetét illetően. Ahogyan az építésmérnök az éppen épített konkrét háztól függetlenül tudatában van az anyag, az eszköz, a hely, a talaj, az ember, az előírások általános összefüggéseinek, éppen úgy az informatikusnak is mindig az adat, a tevékenység, a felhasználó stb. összképben kell gondolkodnia.

Elszomorító és egyszersmind elgondolkodtató, hogy a számítástechnikusok nem ismerik ezt a modellt. Minden munkájuk kapcsán - és mindig másként, de sohasem teljesen - újra meg újra 'ki kell találniuk' azt, hogy vajon mi minden is tartozik egy rendszerbe. Nem a konkrétumokról van szó! Azt, hogy konkrétan melyek a rendszer alkotóelemei, természetesen mindig újra kell felfedezni. Itt az általános tényezőkről beszélünk. Például a rendszerdokumentációkról. Jé, hiszen a dokumentációban ki kellene térni a kódokra is. Jé, ebben a rendszerben vannak kódok is? Jé ...! Minden információs rendszerben vannak kódok. Akkor miként maradhattak ki az X rendszer dokumentációjából? Csakis úgy, hogy azt olyan valaki készítette, aki nem ismerte az általános strukturális összefüggéseket.

A struktúra a rendszert alkotó tényezőknek a **statikus** elrendezését jelenti. Már maga ez az összefüggéshalmaz is több megközelítésben vizsgálható. Például az első szinten csak annyit tudunk, annyit tárunk fel, annyit közlünk, hogy az X és az Y tényezők kapcsolatban állnak egymással. A második szinten minősíthetjük és feltételekhez köthetjük ezt a viszonyt. Az X és az Y tényező kapcsolatban van úgy és akkor, ha ... Mindez azonban a rendszereknek csak egy nagyon korlátozott szemléletmódja, amelyből hiányzik a mozgás, az élet. Az emberi test felfogható csontokból, erekből, izmokból stb. álló rendszerként úgy, hogy maga a csontváz vagy az izomzat önmagában is és közösen is, adott módon összefüggő tényezők együtteseként szemlélhető. Azonban a test ennél több - a test ugyanis él.

A bennünket érdeklő rendszerek élnek, azaz működnek. Persze az úthálózatra vagy a házra mint rendszerre is szokták azt mondani, hogy 'működik'. Azonban az említett dolgok valójában nem működnek, hanem csak működtetik őket. Az információs rendszer, amelynek része maga az ember is, valóban él és működik. (NB.: Ha a házat nem építményként, hanem emberi közösségként fognánk fel, akkor persze róla is el lehetne mondani ugyanezt.) A tízes számrendszerben csak a statikus összefüggésekre kell figyelni. Ezzel szemben az IR-ben a tényezőknek a **dinamikus** viszonyaira is tekintettel kell lenni. Nemcsak arról van szó, hogy - például - az egyik esemény a másikkal összefügg, hanem arról is, hogy az egyik a másikat időben megelőzi vagy

követi. Az X és az Y adathalmaz nem pusztán a származtatási képletek alapján kapcsolódik egymáshoz. Az adatok között időbeli viszonyok is léteznek.

T 3.2 1976-ban az építésügyi tárca miniszterhelyettese kijelentette, hogy az általánosított építőipari termelésirányítási rendszer fejlesztését a 'durva-programozással' kell kezdeni. E mű szerzőjét - finoman szólva - eltanácsolták a projektből, amikor ellenvéleménnyel élt.

A 'durvaprogramozás' nem csúnyát, nem elnagyoltat jelent. Minden termelési folyamat tervezésénél össze kell vetni az erőforrásokat és az igényeket, figyelve az időtényezőre is. Ebben az időszakban ezt és ezzel, abban pedig azt és azzal fogjuk előállítani. A 'finomprogramozás' feladata az erőforrás és az igény adott időszakon belüli pontos összehangolása. A vágy és a megvalósítás viszonyáról van szó. Van ennyi pénzem és a nyáron el akarok menni Olaszországba - ez a vágy, a 'durvaprogramozás'. Konkrétan X ezer líram van és július huszadikán fogok indulni - ez a megvalósítási terv, a 'finomprogramozás'.

Intermezzo

Ez a könyv nem a termelésirányításról szól. Ennek ellenére egyáltalán nem árt, ha általában is figyelünk az előzetes és a pontosított elképzelések viszonyára. Az információs rendszerek terén ugyanis az a szokás, hogy vagy nincs is előzetes elképzelés, vagy pedig a végső gondolat - a megvalósítás - arra távolról sem hasonlít...

Mi köze mindennek a T 3.2 történethez és a dinamikához? A válasz egyszerű: ha nem tudom, mennyi líram van vagy lesz; ha nem tudom, hogy mi a tavasz és mi a nyár; ha maga Olaszország nem is igazán jöhet számításba, tehát ha maguk az alapelemek ismeretlenek, akkor hogyan készíthetek utazási 'finomprogramot'?

Az ismeretek (is) dinamikusan egymásra épülnek. A durvaprogramozás csak annyit mond, hogy ha egy darab X bizgentyűhöz 2 darab A és 6 darab B valami kell, akkor öt darab X bizgentyűhöz 10 darab A és 30 darab B valami szükséges.

A durvaprogramozás időbeli előfeltétele tehát kettős. Tudni kell azt is, hogy egy darab X miből épül fel és azt is, hogy hány darab X-re van szükség. Ezért tehát a T 3.2 történet akarnok 'főfejese' nagyot tévedett. A 'norma' (egyhez annyi kell) és a 'terv' (összesen ennyi kell) két alaptényezője nélkül ugyanis soha sincs mód durvaprogramozásra. Csakhogy abban az időben képtelenek voltak a normáknak a meghatározására és a tervezés sem állt a helyzet magaslatán. Ezért logikusnak tűnt az adott úr számára, hogy a könnyű sikert ígérő valamihez fogjon hozzá.

Intermezzo

Köteteket lehetne írni az elrontott számítástechnikai fejlesztések történetéről. Az csak egy dolog, hogy a rendszerek tényezőinek a statikus összhangjára nem figyel senki. A dinamika még ennyi figyelmet sem kap. A vélt könnyűség és a saját magunknak bebeszélt pénzügyi mutatók alapján teljesen ad-hoc ponton fogunk a 'számítógépesítés' feladatába. Jellemző például, hogy a fejlesztéseket a könyvelés vagy a statisztika technikailag nagyon könnyű feladatával kezdjük. Viszont megalkotni pl. azt a jó anyagrendszert, amely a könyveléshez a hiteles anyaggazdálkodási alapadatokat biztosítja, már jóval nehezebb.

Az intermezzo után nem térünk át újabb rendszervizsgálati aspektusra. Csak az eddigiekkel kapcsolatos és általánosan tapasztalható félreértéseket tisztázzuk. A felsorolt tényezőket ismerő, az azokra figyelő igazi informatikusoknak az egyik alapvető - szemléleti?, gyakorlatlansági? -

hiányosságáról lesz szó. Nevezetesen arról, hogy az IR fejlesztésének a résztvevői túl hamar ringatják magukat abba a hamis tudatba, hogy ők ismerik a rendszerüket. Ám Murphy ikszedik törvénye szerint „ahol valami rosszul is mehet, ott rosszul is fog menni”. Itt nem az a cél, hogy vészharangokat kongassunk, hanem az, hogy felhívjuk a figyelmet az IR-ek egyik nagyon kellemetlen vonására.

T 3.3 E könyv szerzője egy adatbázis tervezése kapcsán nem figyelt arra, hogy általában biztosítási szerződést csak egyetlen valaki köthet, de abban a sajátos esetben, ha... ez a kitétel nem igaz.

A tapasztalatlan informatikus elkövetheti azt a hibát, hogy merev szabályokat feltételez. Az ‘általában úgy szokták csinálni, hogy...’ szemlélet azonban előbb-utóbb gondokra vezet. A tapasztalt informatikus tudja, hogy könnyű ‘rendszeres’ rendszert alkotni, vagyis a szokásosan feltételezhető dolgokra alapozni, ámde az IR mindig azon a pár tényezőn bukik meg, amelyek kilógnak a sorból. Ezért az értelmes informatikus a bajok elébe megy, vagyis tudatosan keresi a **kivételeket**. Nem várja meg, hogy azok találják meg őt.



3.1 ábra: Az IR néhány vizsgálati szempontja

Természetesen arra nincs mód, hogy minden szélsőséges esetet előre lássunk. A jó informatikus egyik 'titka', hogy nem tekinti egyenrangú fontosságúaknak az IR tényezőit, hanem egyféle **informatikai ABC-analízist** hajt végre magában. Az ABC-analízis itteni interpretációja szerint vannak nagyon fontos [A], nem lényeges [C] és a kettő között lévő [B] tényezők. Általában a 20%-ot kitevő nagyon fontos tényezők 80%-ban járulnak hozzá a rendszer sikeréhez. Tehát a feladat 'csak' annyi, hogy megtaláljuk azt a bizonyos 20 százalékot...

A másik nélkülözhetetlen szemléleti faktor az előrelátás. Itt a **fejlődésről** van szó. Az információs rendszerek nem csak abban az értelemben dinamikusak, hogy bennük események és tevékenységek zajlanak. Ma még igaz az, hogy a szerződések általában egy partnerhez kötődnek (vö. T 3.3), ámde figyelni kell arra is, hogy ez a helyzet idővel megváltozhat. Természetesen az informatikus nem madárjós, hogy előre megláthassa a dinamikus strukturális változásokat. Az azonban súlyos hiba, ha nem figyel a közeljövőben bekövetkezőkre. Például arra, hogy új biztosítási módozat bevezetésére készülnek, amelyben a szerződés több partnerhez is kapcsolódhat. Ilyenkor mindig a tágabb, a kivételesnek tűnő eseteket is számbavevő megoldásokat kell alkalmazni. Gondoljon az olvasó arra, hogy a fiatal kisgyerekek édesanyukája a kellenél egy számmal nagyobb cipőt vesz, mert tudja, hogy a kölök holnapra még azt is kinövi...

Végül itt van egy harmadik - sokszor elhanyagolt - szemléleti faktor. Akadnak olyan számítástechnikusok, akik az első kézenfekvő megoldás megtalálása után örömmel és nyugodtan dőlnek hátra a karosszékekben. Eszükbe sem jut, hogy az első ötletnél akadhat jobb is. Nem látnak **változatokat**, nem mérlegelnek. Pedig az IR részben absztrakció, olyan gondolati termék, aminek ezernyi olyan képlete van, amelyek mindegyike ugyanazt a célt elégti ki, de nem egyformán érthető, hatékony, kényelmes stb. módon.

T 3.4 A minap volt szerencsém áttekinteni egy fizikai adatbázisterv részletét. A tervezők egy jó megoldással álltak elő. Én pedig azonnal megkérdeztem, hogy arra a két másik változatra vajon gondoltak-e? És azóta is jár az agyam, hogy van-e még további...

Az IR annyira összetett, hogy rengeteg aspektusból kell és lehet vizsgálni. A fentiekben nem szóltunk a dokumentációjáról, a menedzseléséről, a földrajzi elosztásáról stb. Az olvasónak meg kell értenie, hogy egyelőre csak arra van módunk, hogy a legfontosabb szempontokra ügyeljünk. Egy több könyvet is igénylő témát nem zsúfolhattunk egyetlen fejezetbe.

3.1.2 A rendszerstrukturálás elvei

A korábbiakban csak általánosságokban szóltunk a rendszerek tagolásáról. A rész- és az alrendszer kifejezés helyett mi a **rendszerész** fogalmat részesítettük előnyben. Ám ezen túlmenően nem árultuk el azt, hogy milyen szempont szerint lehet és melyik szerint célszerű a rendszereket tagolni. Pedig két dolog világos. Egyrészt az összetett dolgok megismeréséhez a felosztás ténye elengedhetetlen. Másrészt a felosztás módja nem közömbös a rendszer sikere szempontjából.

Az alábbi történettel kívánjuk szemléltetni azt, hogy miért kell beszélni helyes és célszerűtlen rendszerstrukturálásról.

T 3.5 Egyre több olyan magyar vállalat/intézmény akad, amely területi értelemben elosztott rendszereket fejleszt. Van központ, regionális központ, végpont. Nem egy és nem két olyan céget ismerhetünk, ahol központi, regionális és végponti ‘rendszerekről’ beszélnek.

A kérdés az, hogy helyes-e ez a szemlélet, vagyis a rendszereknek a *földrajzi elv* szerinti felosztása? A válaszuk egyértelmű: nem. Ezen határozott tagadás jogosságát az olvasó majd csak az alpont végén fogja teljesen megérteni. Itt csak arra utalhatunk, hogy a tagolás mindig szétválasztással jár, és az nem közömbös, hogy ennek a szeparálásnak mi a hatása.

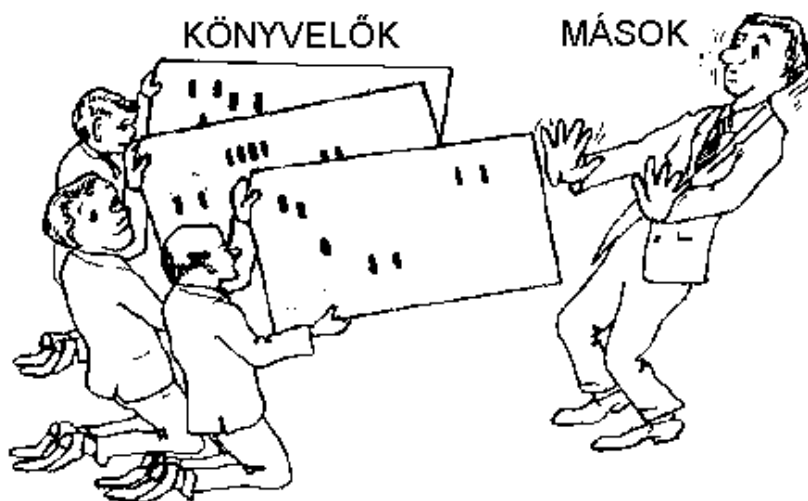
A szerző ügyfélként több országos hálózattal rendelkező intézménnyel, céggel áll kapcsolatban. Ezek nagy része külön-külön központi, regionális és végponti rendszereket alkotott, vagyis nem egyetlen rendszer részeként határozta meg a területi funkcióit. Ezt onnan látja, hogy a rá vonatkozó ismereteket többszörösen és nem összehangolt módon tárolják ill. kezelik. Pl. hol így, hol úgy írottan látja viszont a nevét, címét stb., ami csak szépséghiba. Ámha pénzügyi ügyeit illetően is hasonlóan szétszórta - és nem elosztta - a rendszert, akkor jogos az aggodalma.

A korábbiakban már szót emeltünk az *eszköz elv* ellen. Vagyis az ellen, hogy az információs rendszert nagygépes, munkaállomásos, pécés - mi több: külön 286-os (még ilyen is van!), 386-os stb. - hardver illetve X, Y, Z verziós szoftver szerint osszák fel. Természetesen jól tudjuk, hogy az erőforrások szűkössége és az eszközök kompatibilitásának a hiánya - ami az adatok hordozhatatlanságával jár - miatt a rendszerben kell, hogy legyen ilyen meg olyan módú megvalósítás. Azonban az eszköz-elvet követők nem egyetlen egy, önmagában harmonikusan elképzelt rendszernek a többféle megvalósításában, hanem valóban több olyan rendszerben gondolkodnak, amelyek egymással nem harmonizálnak. Magyarul: nem a fejlesztés *adott fázisában* döntenek el, hogy van ilyen-olyan megvalósítás, hanem a fejlesztés *kezdetén* jelentik ki, hogy van X, Y és Z rendszer.

Csak kitérőképpen említjük meg, hogy a területi vagy az eszköz elv legalább hordoz magában valamiféle rációt. Főleg kisebb és az informatikával viszonylag frissen ismerkedő cégeknél jellemző a *strukturálási elv teljes hiánya*. A hatalmi szó, a megfontolatlan vágy, a szubjektív érzés dominálja az *implicit* tagolást. Ez is felosztás! A vezető sokáig ül a babérain, egyáltalán nem figyelve az ismeretek fontosságára. Majd felébred, az asztalra csap és kijelenti, hogy ő holnapra kér - mit kér, követel! - egy információs rendszert. Az elvek ismeretének a hiányában ad-hoc módon úgy dönt, hogy gépesítsék ezt meg azt. Holnap rádöbben arra, hogy - jó! - még azt is lehetne. Ezért újra az asztalra csap - és kialakítat egy új, egy másik - rendszert. Ettől a pillanattól fogva a tagolás már *explicit*, hiszen a valódi, a teljes információs rendszert már két részre darabolták fel.

De térjünk vissza a (fél)tudatos strukturálási elvekhez! Viszonylag józannak tűnik a *szervezeti elv*. Ha van könyvelési, anyag- és tárgyi-eszköz-gazdálkodási részlegünk, akkor legyen könyvelési, anyag- és tárgyi-eszköz-gazdálkodási IR-részünk is. Ez a gondolatmenet a megtévesztésig gyakorlatiasnak látszik. Ámde több problémát is felvet.

Az egyik az időfaktor. Manapság mindenki ‘gépesíteni’ akar. Vagy azért, mert már anélkül nem megy, vagy azért, mert ez ma így szokás, így divatos, így illik. Ámde már a hetvenes években is általános jelenség volt, hogy akadtak korábban és későbbben ébredő szervezeti egységek, vezetők. A számítógépes eszközökben rejlő fantáziát hamarabb meglátók és az az ellen időlegesen makacsul tiltakozók. Mármost az IR olyan - részben absztrakt - szellemi termék, amire óhatatlanul rá van nyomva a készít(ett)őjének a szemléleti bélyege.



3.2 ábra: Szemléleti nézeteltérések

A vállalat szervezeti egységei elvileg egyenrangú fontosságúak. Gyakorlatilag azonban mindig akad egyenlőbb az egyenlők között. Aki előbb ébred, akinek nagyobb a szava, annak a szemlélete fogja meghatározni az IR természetét. Jól látszott ez a jelenség a hetvenes években. A számítógépeket akkor elsősorban könyvelésekre, kimutatásokra alkalmazták. Később a műszaki szakembereknek ugyancsak nehéz dolguk volt, amikor a már gazdasági szemléletben elkészített rendszerek 'hasznélvezetébe' ők is be akartak szállni. Mivel pedig ez sokszor egyáltalán nem sikerült nekik, a meglévőkkel *párhuzamos*, azokkal alaposan átfedő rendszereket kezdtek kialakítani. (Vö. implicit tagolás.)

T 3.6 Az egyik munkahelyemen alapvető feladat volt bizonyos pénzügyi tranzakciók rögzítése és főkönyvi feladása. A gazdasági igazgató - személyes nagyzolásból - olyan külön könyvelési rendszert csikart ki a maga számára, amelybe csak ő nézhetett bele és amelynek semmi köze sem volt a másik - szintén könyvelési - rendszerhez...

A szervezeti elvnek van két további negatívuma is. Vegyük észre, hogy ha valami nem megy a rendjén egy cégnél, akkor nem a működési zavarok okait keresik, hanem azonnal átszervezésbe fognak. Ez történik akkor is, ha új vezető érkezik. Mármost két eset lehetséges. Ha a rendszer-részek továbbra is a régebbi szervezeti séma szerint tagoltak, akkor nem felelnek meg a valóságnak és ezért óhatatlanul problémák fognak fellépni. A másik lehetőség az, hogy a szervezeti átalakítással párhuzamosan az információs rendszert is újraépítik. Továbbá arra sincs semmi garancia - és ez a második probléma - hogy akár eleve, akár pedig az átszervezés során informatikai értelemben véve is célszerű szervezeti sémát hoznak létre. Ugyanis nemcsak az információs rendszernek kellene igazodnia a valós szervezethez, hanem megfordítva is.

Ezért a fentebbinél egy fokkal célszerűbb a *funkcionális* elv. A funkción most nem valamiféle pozíciót (vö. 'magas funkciót tölt be'), nem is egy matematikai értelemben vett függvényt kell érteni. A funkció adott feladat ellátását, az annak érdekében végrehajtott tevékenységeket jelenti. Ezért akár tevékenységi elvről is beszélhetnénk. A rendszereknek az ilyen alapon való strukturálása már közel áll az informatikailag helyes megoldáshoz.

Egy példával szemléltetjük a mondanivalót. Az összetettebb szervezetekben a könyvelő könyvel, a bérszámfejtő pedig bérszámfejt - vagyis ez a funkciójuk. A kisebb cégekben a két feladatot - és még mást is - egyetlen személy hajtja végre. Mármost változik-e például a könyvelés-

nek az alapvető lényege attól, hogy az előbbi esetben külön szervezeti egység, az utóbbiban pedig egyetlenegy személy látja el a vonatkozó funkciót? Ugye, hogy nem? Tehát nem a szervezeti egység, hanem a feladat az ismeretekkel támogatandó dolog.

A fentiek miatt úgy tűnik, hogy a funkcionális rendszertagolás a jó megoldás. Ámde itt is akad egy kis bökkenő. Mégpedig az, hogy magának a funkciónak a lényegét sokan félreértik. Pl. valamikor úgy alakultak a dolgok, hogy a banknál az X-féle és az Y-féle számlák kezelését másvalakikre bízta, akik kiépítették a saját kis - egymástól részben eltérő - ismeretkezelési birodalmukat. Ezért a bank dolgozóiban rögtön az az érzés támadt, hogy a kétféle számla kezelése tényleg két eltérő funkció, mivel két feladatról van szó. Ezzel a példával nem az akarjuk sugallni, hogy az effajta gondolkodásmód mindig eleve téves. Csak a tévedések lehetőségére akarjuk felhívni a figyelmet. Ha a kétféle számlán lévő ismeretek nagymértékben hasonlóak; a rájuk vonatkozó események illetve a végrehajtandó tevékenységek is rokonok, akkor gyakorlatilag semmi kifogás nem merülhet fel a kétféle funkcióban való gondolkodás ellen, azonban az IR egészének a szintjén elméletileg nem a funkciók szerinti tagolás a megfelelő megoldás. Azért nem az, mert a tárolt és kezelt ismeretben (egy ügyfél érintett lehet mindkét számlában!) és az adatkezelő ill. -előállító műveletekben igen nagy lesz az elvi redundancia. Például többször kell megírni 80-90 százalékban azonos programokat.

Bármennyire is meglepő, az ismeretekkel való teendőket maguk az ismeretek határozzák meg - és nem megfordítva. Nem azért van például vevőállományunk, mert végre kell hajtánunk az új vevő esemény kapcsán a vevőbevitel műveletet, a vevő címének a változása esetében pedig a vevőkarbantartási műveletet. Éppen fordítva: a valóságban vannak vevőink, akiket ismeretekkel írunk le. Az ismeret szüli a bevitelére és a karbantartására vonatkozó igényt.

A fentiek miatt a rendszert legcélszerűbben az *ismereti elv* szerint tagoljuk. Az első pillanatban az olvasónak úgy tűnhet, hogy ez és az előző elv azonos, hiszen az adatfeldolgozásnak - mint neve is mutatja - mindig két oldala van; az ismeret és a rajta végzett tevékenység elválaszthatatlan egymástól. Ámde egy alaposabb megfontolás után be kell látnunk, hogy az adat szerinti tagolás elve más, mint a funkcionális lebontásé és az utóbbiánál magasabb rendű. Ha pl. a kétféle számla adatfésései nagymértékben megfelelnek egymásnak, akkor hiába vannak apró eltérések a feldolgozásukban, a két tevékenységet nem szabad két rendszerrész alapjának tekinteni. Miért nem?

A rendszerrészek nem darabolják a rendszert, hanem úgy választják szét annak tényezőit, hogy azokat egyben össze is kapcsolják (lásd a belső/külső környezet viszonyát). Mármost az X- és az Y-féle számlán is szerepelnek az ügyfél adatai. Funkcionális megbontás esetében két partner állományt fogunk létrehozni úgy, hogy közöttük nincs kapcsolat. A tagolás rossz, mert elveszik a rendszer egésze, az nem lesz 'szervezett együttes', hiszen redundáns és ellentmondó ismeretekkel kell számolnunk. (Kétszer vezetnek a partnerek adatait, ámde nem teljesen azonos aktualitási állapotban.) Ha viszont az ismeretek köre szerinti strukturálási elvet követjük, akkor az ügyfelet tekintjük egy lényegnek és arra koncentrálunk.

Az olvasó vegye észre, hogy a fentiekben ismertetett strukturálási elvek az IR definíciójában (D 2.1) felsorolt tényezőknek felelnek meg. Mivel pedig azokat a fontosság szempontjából nem rangsoroltuk, úgy tűnhet, hogy a tagolási elvek is egyenrangúak. A valóságban azonban nem azok. E ponton ismét hivatkoznunk kell az előző alpontra említett kivételekre - csak éppen fordított értelemben.

A fejlesztők többsége ebben az esetben mindig a különbségre koncentrál, azaz a *nem-közös* adatokra építi a rendszerét. Akkor, ha az X-féle biztosítás ismeretei például 70-80 százalékban eltérnek a Y-féle biztosításától, akkor jogosnak érzi, hogy X és Y rendszerrészt alkosson. Csak-hogy ezzel megfelelkezik a szintén az előző alpontra említett informatikai ABC-analízisről! Ez azért baj, mert éppen a kétféle biztosításban lévő 20-30 százaléknál *közös* adat a legfontosabb. Ezért a struktúrát nem egyoldalúan kellene felépítenie. Az sem helyes, ha csakis X és Y részekben gondolkodik, de az sem, ha a kettőt egyetlen Z-be fogja össze - hiszen annyira eltérőek.

A célszerű tagolás nyilván X1, Y1 és Q, ahol az X1 és az Y1 a nem-közös, a Q pedig a közös tényezőket öleli fel.

Az alpont legelején említett területi elvhez visszatérve ismét figyelmeztetnünk kell arra, hogy ebben a pontban nem a megvalósítás problémáit firtatjuk. Az teljesen világos, hogy a területileg tagolt rendszerekben a földrajzi elosztás elve érvényesülni fog a megvalósításban is. A rendszernek lesznek a központban ill. a régióban és a végpontokon működő részei. Itt azonban egyáltalán nem az ún. *elosztott* [distributed] rendszerek strukturálásáról van szó, ami szintén történhet szervezeti, tevékenységi és ismereti alapon. Persze a magyar valóság az, hogy a terület voltaképpen szervezeti egységet jelent, és ekképpen az alkalmazott tagolás elve nem is területi, hanem a szervezeti elvnek csak egy sajátos válfaja. Tehát a lényegi kérdés nem úgy hangzik, hogy a működő IR-ben elismerjük-e a terület mint sajátos rendszerrész létjogosultságát. Ezt természetesen meg kell tennünk. Számunkra a valódi probléma az, hogy már az IR kialakítása előtt alkalmazzák-e a területi elvet. Tehát az egészet látva a területeteket rendszerrészekként - nem pedig önálló rendszereként - fogják-e fel és alakítják-e ki.

3.1.3 Az elvek harmonizálása

A közös/nem-közös biztosítási ismereteknek a példája önkéntelenül is felveti azt a kérdést, hogy vajon lehet-e és szükséges-e egyetlen rendszer-strukturálási elvet követni? Nem képzelhető-e el olyan kombinált megoldás, amelyben a fenti területi, szervezeti, funkcionális és ismereti elvek jól megférnek egymás mellett? Valóban az ismeretek szerinti rendszertagolás a célszerű? Ha igen, akkor miként érzük el azt, hogy a másik három aspektus azzal harmonizáljon?

Az alábbiakban fel fogunk vázolni egy olyan koncepciót, amely bemutatja azt, hogy a szervezetek *rendeltetésétől* - vagyis alapfeladatától - teljesen függetlenül milyen nagyobb ismeretkörökkel kell számolnunk. Egy bankban éppen úgy, mint egy kórházban, egy termelő vállalatban vagy egy hivatalban.

Minden szervezet alapfeladato(ka)t lát el és ehhez erőforrásokat vesz igénybe. Természetesen az erőforrások az alapfeladatnak megfelelően akár specifikusak is lehetnek. Ettől függetlenül mindig általánosak is. Egyetlen szervezet sem képes működni ember, pénz, anyag, eszköz stb. nélkül. Az erőforrásokat menedzselni kell, és ezt a főtevékenységet nevezik *törzskari* [staff] feladatnak. Mivel pedig az erőforrásokra vonatkozó ismeretek nagymértékben eltérőek - például teljesen más ismereteket vezetünk az emberről, mint az anyagról -, meglehetősen logikus módon a teljes információs rendszer különálló részeinek tekintjük az erőforrások ismereteinek a kezelését. A bér- és munkaügy (nevezzük akár humánpolitikának az utóbbit), a pénzügyi tervezés és a könyvelés, a tárgyi eszközök nyilvántartása valóban különálló rendszerrész. Ezen a tényen mit sem változtat az, hogy például a kórház vagy a mezőgazdasági üzem tárgyi eszközei nem ugyanazok.

Az alapfeladatok a szervezet rendeltetéséhez kötődnek. A biztosító biztosít, a kórház gyógyít, a bank pénzt kezel. Nem csak egyféle módon, mert más az élet- és a vagyonbiztosítás, a belgyógyászat és a sebészet, a kötvény és a számla. Az alapfeladatok képezik a szervezet *funkcionális* [line] tevékenységét. Ezért az is logikusnak látszik, hogy ahányféle funkciót lát el a szervezet, annyiféle módon képezzünk funkcionális rendszerrészeket. Legyen pl. vagyon- és életbiztosítási rendszerrészünk.

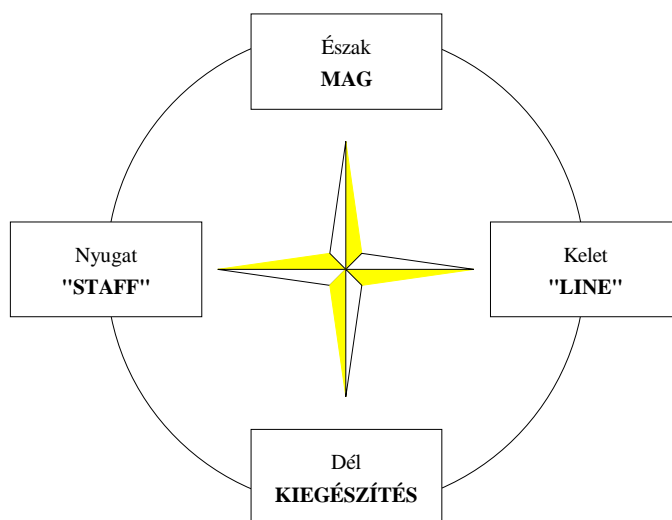
Mind a 'staff', mind a 'line' feladatoknál kiindulásként egy dologra kell nagyon vigyázni. E tevékenységeket természetesen szervezeti egységek látják el, de hiba lenne visszacsempészni a szervezeti elvet. Az X kórházban van egyes és kettes belosztály, ami önmagában még nem jelenti azt, hogy kétféle funkció is van!

A törzskari és funkcionális rendszertagolás szép reményeket támaszt bennünk. Ám van két apró hibája: felülről és alulról sem kielégítő, mert nem kapcsolódik igazán az adatkörökhöz.

Felülről nézve a dolgot rá kell döbbernünk arra, hogy a 'staff' és a 'line' oldal szétválaszthatatlan. Kovács nemcsak erőforrás ('staff'), akinek bért kell fizetni, hanem egyben az X feladatot elvégző ('line') személy is, aki munkájáért felelős. Kovács a bérjegyzéken és Kovács a biztosítási szerződés ügynökeként ugyanaz a valaki. Ha mereven követnénk a törzskari-funkcionális rendszertagolást, akkor Kovács - képletesen szólva - kettészakadna. Sajnos, pontosan ez történik a mai mindennapi gyakorlatban. Kovács a biztosító intézet dolgozójaként, Kovács az intézet biztosítottjaként, Kovács külső partnerként (pl. ügynök) csak nem akar egy Kováccsá kovácsolódni, mert több 'rendszernek' a tényezője.

Felismerve a fenti helyzet fonákságát, a rendszertagolásban érdemes bevezetni a **rendszer-mag** fogalmát. A közös ismeretek és a rájuk vonatkozó tevékenységek alkotják a rendszermagot. Többek között ide tartozik a saját szervezet felépítése; a partner ismeretkör; a tér és az idő közös momentumai (pl. települések); a külső környezet általános tényezői (pl. deviza) és így tovább. A rendszermag tényezői nemcsak az erőforrások, hanem a funkciók köréből is származhatnak. Ezért a mag szervezetenként igen eltérő lehet. Egy példa: a szociálpolitikai ellátásban kérelem vagy pályázat alapján lehet juttatást elnyerni. Az X-féle és az Y-féle szociálpolitikai eszközre (átképzés, előnyugdíjazás stb.) vonatkozó ismeretek ugyan nagyon különbözőek, de a kérelem/pályázat adatai roppantul átfedőek. Ezért lehetséges, hogy ebben a rendszerben a kérelem/pályázat adatai a magban fognak szerepelni annak dacára, hogy más szervezetre egyáltalán nem jellemző funkcionális adatokról van szó. Ezért a rendszermagot úgy kell felfogni, mint az adott szervezetben az erőforrásokkal és/vagy a funkciókkal kapcsolatosan több feladatban is közösen értelmezendő és kezelendő ismeretek részrendszerét. Egy módon értelmezendő és kezelendő a partner ('staff'), ugyanakkor ugyancsak egy módon értelmezendő és kezelendő a szociálpolitikában a partner által benyújtott kérvény/pályázat is ('line').

A felülről közöset így elintéztük volna. Ám **alulról** szemlélve a dolgokat újra csak problémákba ütközünk. Mi a teendő akkor, ha pl. az X-féle és az Y-féle vagyonbiztosításnak a teljes rendszer szintjén közös ismereteit már kiemeltük a magba (felülről), de ezek után a kétféle módozatnak még mindig vannak közös és (alulról) nem-közös ismeretkörei? Nem kell megijedni: csak a józan ész által diktált megoldást kell követni.



3.3 ábra: A rendszer helyes strukturálása

Senki sem írja elő egy rendszer megbontásának a szélességét és a mélységét. Ez a két dimenzió 'egymás ellen dolgozik'. Vagy a szélességet növeljük egyben csökkentve a mélységet,

vagy megfordítva járunk el. Ha az X és az Y adatkör lényegesen eltér egymástól, akkor magasabb szinten két rendszerrészt fogunk meghatározni, a rendszerhierarchia **szélességét** növelve. Ha viszont a két adatkör nagymértékben átfed, akkor egyetlen X rendszerrészben gondolkodunk az első szinten, majd a másodikon - a rendszerhierarchia **mélységét** növelve - kijelöljük az X1 és az X2 rendszerrészen belüli további részeket.

A szervezélmélet némi támpontot nyújt a szervezeti hierarchiák felépítésére vonatkozóan. Sem a túl széles, sem a túl mély, sem az egyenlőtlen eloszlású - itt ilyen, ott olyan mély - felépítés nem kedvező. Ezt a gondolatot támaszként lehet venni az IR tagolásában is. Az emberi áttekinthetőség, az erőforrásokkal történő ellátás - általában a menedzselhetőség - kedvéért szélességben és mélységben kvázi egyensúlyos rendszertagolást kell alkalmaznunk, elsősorban az ismeretek szerinti megosztás elvére alapozva.

Már csak „egy gondolat bánt engemet”. A hetvenes években terjedt el az ún. **strukturált tervezés** ideája, a strukturált programozására alapozva. A fentebbi gondolatokhoz a megtévesztésig hasonlító módon a ‘structured design’ kiemelte a rendszerek célszerű hierarchikus megbontásának a fontosságát. Sajnos, maga a hangsúly a **hierarchia** szóra terelődött. Ez pedig roppantul nagy baj, mert erősíti a szürke-folt effektust. Vegyünk alapul egy példát! Cikkeket értékesítünk. Maga az eladás az értékesítési részleg/funkció feladata. A ‘kasszírozás’ viszont már a számlázási részleg/funkció hatáskörébe tartozik. Baj ez? Önmagában nem az. Az igazi probléma az, hogy az értékesítés és a számlázás egymás mellé rendelésével merev hierarchiát állítanak fel, holott ez a két dolog szorosan összefügg. A valódi informatikus az információs rendszer minden egyes tényezőjét illetően **hálóban**, és nem hierarchiában gondolkodik. Természetesen sokkal, de sokkal nehezebb a hálós, mint a hierarchikus összefüggések megragadása, nyomkövetése. Pedig a valóság mindig hálós. Nemcsak főnököm van (hierarchia), hanem ismereteket cserélek a többi részleg dolgozóival is (háló). Ha valamelyik rendszerrészben (hierarchia) karbantartom az adatokat, az kihat a többire is (háló).

Ezt az alponot egy pregnáns példával fogjuk zárni. Vajon miként lehetséges az - pedig megtörtént! -, hogy Kovács korengedményes nyugdíjasként felvett vállalkozói hitelt? Úgy, hogy a szociális rendszer informatikusai hierarchiában gondolkodtak. Rosszul tagolták a teljes rendszert. Funkcionálisan ill. részlegek szerint. Más funkció - más részleg - felel a nyugdíjazásért és a hitelekért. Nincs kommunikáció a mereven hierarchikusan meghatározott rendszerrészek között. Pontosan erre a hézagra, a szürke foltra játszanak a bűnözők, a szélhámosok. A fejlesztőknél is jobban ismerik a rendszer hézagait. A rossz rendszertagolás még nem bűn, de komoly hiba, amely végzetesen meghatározhatja a rendszer sorsát.

3.1.4 Az IR vetületei

A korábbiakban elmondtuk, hogy az alkalmazási, az információs és az eszköz-rendszer átfed egymással. Ebből következik, hogy az IR-t különböző természetű tényezők alkotják. Az információs rendszerek **konkrét** és **absztrakt**, másképpen szólva fizikailag létező ill. szellemileg elgondolt alkotóelemek együttese. Az élő ember és a felhasznált eszköz (konkrét, fizikailag létező dolgok) illetve az adat, az esemény és a tevékenység (absztrakt, szellemileg elgondolt dolgok) összetett kölcsönhatásban állnak egymással az IR fejlesztése és működtetése során. Ezért nem válhat jó informatikus abból, aki nem ismeri az információs rendszerek elvi szerkezeti felépítésének a konkrét és absztrakt dolgokon alapuló kereteit.

Abból kell kiindulni, hogy a rendszerfejlesztés nem azonos a programkészítés feladatával. Nem szoftvert kell alkotni, hanem az IR szerteágazó tényezőit kell összehangolni, merthogy az IR szervezett együttes. Megszervezéséhez látni kell, hogy e rendszer elemei nem homogének, amennyiben eltérő módon viselkednek. A felhasználó és az erőforrás adottság, olyan fizikailag is létező lényeg, amellyel számolni kell a fejlesztés során. Ezzel szemben az adatokat, az eseménye-

ket és a tevékenységeket ki kell gondolni, meg kell tervezni. Ezek a tényezők mindig tükörképek - másodlagos valóságok. Eme kitétel igazolására álljon itt egy példa.

Tegyük fel, hogy új dolgozót vesz fel a cégünk. Ekkor a megfelelő felhasználó el kell, hogy indítsa a felvételi procedúrát. Ez nyilvánvalóan az eseményeknek és a tevékenységeknek egy olyan láncolata, amelynek adott pontjain ismeretek kezelésére kerül sor. Akár létezik 'szabványos' felvételi eljárás, akár nem, annyi bizonyos, hogy az 'új dolgozó felvétele' folyamatot ezer és egyféle módon lehet végrehajtani. Nincs egyetlen eleve adott, objektíven létező ilyenféle valóság: azt valamikor ki kellett gondolni, meg kellett tervezni. Másképpen akkor, ha gépet alkalmaznak, mintha nem; ha az X szoftvert vették meg az Y helyett; ha a cég a kötelezően vezetendőkönyvön kívül még az A vagy a B ismeretköröket is rögzíteni akarja az új felvételisről stb. Mivel pedig a felvételi egy viszonylag bonyolult eljárás (ide-oda küldözik a szerencsétlen új dolgozót), a tevékenységi lánc több részeseményre és -tevékenységre bomlik nálunk így, másutt pedig amúgy.

Az IR-ben a felhasználó, az erőforrás és a szabvány adottság, feltétel, korlát, sarokkő. Ezzel szemben az adat, az esemény, a tevékenység egymáshoz is és az előzőekhez is igazítottan elgondolandó, megtervezendő, kidolgozandó dolog. Ebből következik, hogy az IR-t legalább két szemléletben kell vizsgálni. Ámde ez a kettő még kevés, mert az adat és az esemény/tevékenység páros természete szintén jelentősen eltér egymástól. Ezért ebben a könyvben három IR-vetületet fogunk megkülönböztetni: az adat, a feldolgozás és a környezet **vetületet**. Most pedig ejtsünk ezekről néhány szót.

Az adat (ill. információ) és a feldolgozás aspektusainak a szétválasztását több körülmény igazolja. Vegyük észre, hogy egy adott szervezetben az alapvető (!) ismeretek lényege és elrendezése viszonylag **stabil**. Ha pl. egy cég értékesítési tevékenységet is folytat, akkor abban mindig szükség lesz a vevőkre, a cikkekre, a megrendelésekre, a szállításokra, a számlákra stb. vonatkozó ismeretekre. Az ismereti kép módosulhat a részletekben, az adatok elrendezése itt-ott változhat, de mindaddig, amíg maga az értékesítési funkció fennmarad, az adatszerkezetet felborító, az azt megrendítő átalakulásokkal nem kell számolni.

A gyakorlati tapasztalat az előzőeknek ellentmondani látszik. Ha az ismeretek rendszere az IR-en belül ennyire stabil, akkor miért van szükség szinte naponta továbbfejlesztésre, kiigazításra - időnként pedig újrakezdetre, integrálásra? A válasz kézenfekvő: rosszul hajtották végre az első fejlesztés(ek)e)t.

Az IR **adatvetületének** a másik jellemző vonása a viszonylagos **objektivitás**. Az emberek többsége abban a tévhitben él, hogy mivel az információ személyes - tehát szubjektív - lényeg, az ismeretek elrendezése is tetszőlegesen történhet. Ez félígazság. Azt valóban a felhasználó dönti el, hogy akar-e vezetni adatokat a vevőkről, a rendelésekről, a cikkekről, és ha igen, akkor milyeneket. (Persze ez a szabadsága is korlátozott a tények által. Mert ha értékesít, akkor számlákat kell kiállítania, részben előírással tartalommal.) Ha viszont adott a vevő, a rendelés, a cikk, a számla, mint ismeretekkel leírandó dolog, akkor az adatok elrendezése már nem történhet önkényes módon. Szinte matematikai pontosságú szabályok igazítanak el arra nézve, hogy mikor jó és mikor nem az az adatszerkezet.

Ezekre a regulákra sok helyen nincsenek tekintettel. Például sok helyütt nem alapadatokat, hanem kézzel már valamilyen szintig előfeldolgozott ismereteket kezelnek a számítógépen (vö. feldolgozottsági fok). Másutt a szervezeti tagolási elv miatt nem látják a teljes ismereti összképet - és így redundáns, ellentmondó adatszerkezetet alkalmaznak. Természetesen a rossz szerkezetről előbb-utóbb kiderül, hogy tarthatatlan - ezért jön a változtatás.

A fejlesztés résztvevői nagyon nehezen értik meg, hogy nem a megvalósított, hanem a megvalósítandó adatvetületben kell gondolkodni. A kettő különbségét majd alább, a szinteknél magyarázzuk meg. A cégek ismereteinek a szerkezete viszonylag stabil, viszonylag objektív - azt csak ki kellene fejteni pl. úgy, mint ahogyan Michelangelo szabadította ki szobrai a márványtömbből. Még azt is hozzá kell tennünk a fentiekhez, hogy az adat a többi IR-tényezőtől viszonylag **független**. Természetesen a gyakorlatban ezt nem fogjuk megtenni, de elvileg a vállalat

fiókjainak, leányvállalatainak, telephelyeinek stb. a rendszere leírható adatstruktúrában anélkül is, hogy azt valamire is használnánk. Anélkül, hogy számítógépes programokat írnanék, hogy eseményeket kötnének az adatokhoz, hogy kijelölnének a felhasználókat.

Az adatvetületnek ez a furcsa vonása egy hatalmas lehetőséget rejt magában. Nevezetesen a valóban integrált fejlesztésről van szó (ha magát ezt a fogalmat nem is kedveljük különösebben). A készletgazdálkodás adatainak a szerkezetét leírhatjuk ma úgy, hogy az összhangban áll az értékesítési ismeretekkel, de csak X idő múlva kerül sor a készletgazdálkodási rendszer részére való fejlesztésére.

Az adat nagymértékben független a feldolgozástól. Ezért van szükség a külön **feldolgozásvetület** szerinti vizsgálatra. A 2.2 pontban már kifejtettük azt, hogy mennyire egymáshoz kapcsolódik az esemény és a tevékenység fogalma. A két tényezőt ezért mi egy vetületbe soroljuk, sok más módszertani koncepcióval ellentétben. A feldolgozásvetület - az adatétól eltérően - viszonylag **instabil** és viszonylag **szubjektív**. Ugyanakkor egyáltalán **nem független** a rendszer többi tényezőjétől. Amint látjuk, pontosan ellentétes vonások jellemzik, mint az előző vetületet. De vajon mire alapozzuk ezt a véleményünket?

Először is a feldolgozás az adathoz képest többszörös lehet. Lapozzunk csak vissza a P 2.1 példához! Három cégben három különböző 'feldolgozási logikát' alkalmaznak a rendelések feldolgozására. Ez megtörténhet akkor is, ha a három vállalatban a rendelési ismeretek felépítése teljesen azonos! Mi több, még egy cégen belül is - például a cikk jelentőségének a függvényében - ugyanazokon az ismereteken hol ezzel, hol pedig azzal a feldolgozási módszerrel dolgozhatnak. Például a kiemelt termékeknél a kézi, a többiekénél az automatikus utánrendelés logikáját óhajtják követni, miközben a kétféle cikk adatsora teljesen azonos.

Talán máris érzékelhető, hogy a feldolgozás mennyivel variábilisabb, mint az adat. Tegyük mindehhez, hogy az előbbi tényező egy picit az utóbbtól függ. Ha van rendelés adatunk, akkor gondoskodni kell annak beviteléről, módosításáról stb. Vannak tehát kötött feldolgozási rész-tevékenységek, amelyek az adatkezelés (ld. 2.2 pont) műveletcsoporthoz kapcsolódnak. Viszont bármelyik felhasználó saját eseményeket találhat ki. Eldöntheti pl., hogy az alapadatokból adatelőállító műveletekkel milyen sajátos ismeretekre óhajt szert tenni. Ma erre, holnap pedig arra kíváncsi. Ezért a feldolgozás viszonylag szubjektív és instabil.

Sokan félreértik a feldolgozásvetületnek ezt a rugalmasságát illetve azt, hogy a feldolgozás és az adat összefügg. Azt hiszik, hogy a feldolgozás változtatásának a szabadsága együtt jár az adatok szerkezetének a tetszőleges (át)definíálásával. Velük szemben a jó informatikus ismeri az ún. **logikai adatfüggetlenség** elvét. (Mivel a vetületeket - képletesen - egymás mellé rendelték-ként képzelhetjük el, ezt az elvet horizontális függetlenségnek is nevezik. Ld. a 3.6 ábrát.)

E 3.1 A logikai adatfüggetlenség elve rögzíti, hogy az adatszerkezetben bekövetkező változásoknak nem szabad kihatniuk a feldolgozásra - és megfordítva.

Gyakorlatilag ez az elv azt sugallja, hogy nem szabadna az adatszerkezetekbe feldolgozás-függő részleteket építeni, az adatstruktúrát a programoktól függően meghatározni. Éppen azért nem, mert a feldolgozás annyira változékony. Az adat-program függetlenség elve régi keletű (a hetvenes évek elejéről származik), ám mind a mai napig nem fordítanak kellő figyelmet rá.

Az adatokat és a feldolgozásokat meg kell tervezni. A rendszer többi tényezője viszont adottság, fizikai tény és nem absztrakció. Mindez nem azt jelenti, hogy pl. a géphasználatot nem kell elgondolni. Ám magát a gépet - szemben például az adattal - nem a fejlesztő alakítja ki. A felhasználó, az eszköz és a szabvány az IR-nek a figyelembe veendő **környezeti vetületébe** tartozik (az alkalmazási- ill. az eszköz-környezet részeként).

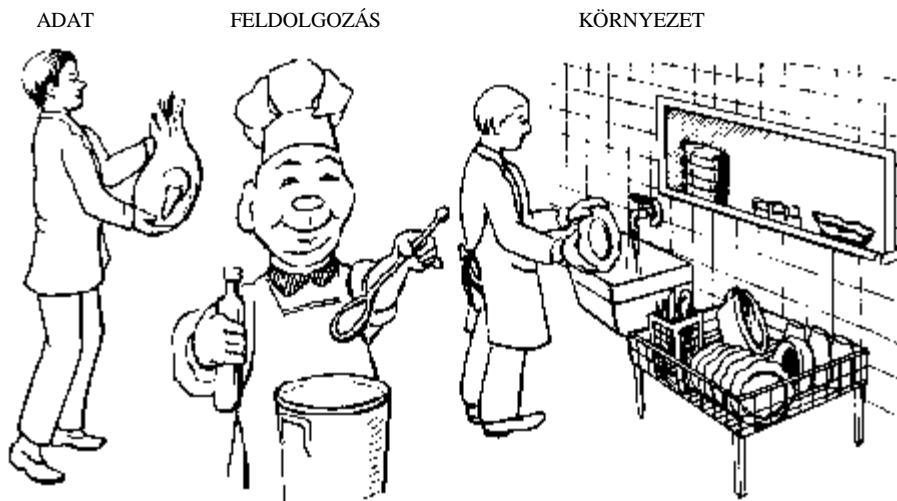
A fejlesztő nem alkothatja meg a tudása és szíve szerinti lehető legjobb IR-t. Számolnia kell a felhasználók szubjektív vágyaival éppen úgy, mint az eszközök objektív képességeivel és a kvázi-

objektív - nem mindig segítő - szabványokkal. Kettős veszéllyel kell szembenéznie. Az egyik az, hogy az X és az Y felhasználó egymásnak ellentmondó követelményeket támaszt. Mindkettő saját szemszögű adatszerkezetet óhajt látni, aminek következményeképpen azonnal megsérül a logikai függetlenség fentebbi elve. A felhasználók nem az egyszeres valósághoz, hanem a saját feldolgozási igényeikhez igazítják az adatstruktúrát. Mivel pedig az igény változik, módosul az adatszerkezet is. A másik veszély az, hogy a gép korlátaira ügyelő fejlesztő azokat beépíti az ismeretstruktúrába, ezzel megsértve a **fizikai** (illetve vertikális vagy eszköz-) **adatifüggetlenség** elvét.

E 3.2 A fizikai adatifüggetlenség elve rögzíti, hogy az eszközökben fellépő változásoknak nem szabad kihatniuk az adatstruktúrára.

A rendelés ismeretsora elvileg nem változik attól, hogy más számítógépet és/vagy más szoftvert használnak. Azonban sok adatkezelő rendszer ma még nem képes az ismeretek természetes elrendezésének a pontos tükrözésére. A lehetőségek ugyan sokrétűek és folyamatosan bővülnek, de informatikailag még távolról sem kielégítőek. A tapasztalatlan fejlesztő 'kiaknázza' a szoftver X-dik verziójának a tökéletesen eszközököt adottságait, miközben 'elhanyagolja' az eszköz-függetlenség elvét. Ezért mind az adat-, mind a feldolgozásszerkezetet változtatnia kell, ha áttér a szoftver Y-dik verziójára. Ezzel persze megsérti a fenti két függetlenségi elv mindegyikét. Az abszolút fizikai függetlenség ugyanis akkor valósulna meg - ez ma még álom -, ha egy adatbázist és az azt kezelő programokat minden módosítás nélkül át lehetne vinni egy másik gépre, egy másik adatkezelő(változat) alá (transzferabilitás).

Az alábbi ábra a háromféle vetület összefüggését kívánja szemléltetni. A sültet különböző eszközökkel és más-más szakácsok is elkészíthetik azonos módon (környezet). Az elkészítés módja nem független sem attól, hogy ki és mivel, sem attól, hogy mit főz (feldolgozás). A nyersanyag ugyanaz akkor is, ha mást készítenek belőle (adat). Ám a végső élvezhetőséghez (cél=információ) a három vetület összhangja szükséges.



3.4 ábra: Az IR vetületei

3.1.5 Az IR szintjei

Az adat, a feldolgozás és a környezet vetület a rendszereknek három általános, egymás mellé rendelt aspektusa. Mivel a rendszer szervezett együttes, az ezekbe tartozó tényezők nem függetlenek egymástól. Ezért az informatikusnak ezeket egymással való viszonyukban kell vizsgálnia. Tehát pl. elemeznie kell az adatok és a tevékenységek, az adatok és a felhasználók stb. összefüggését.

Ugyanakkor az egyes vetületeken belüli elemek is bonyolult módon kötődnek egymáshoz. Pl. az adatok önmagukban is egy igen összetett rendszert alkotnak. Ráadásul az adat- és a környezet-vetület tényezői egy nagyon tudatos, többszintű *absztrakciós folyamat* eredményei. Az alábbiakban ezeket az absztrakt *szinteket* fogjuk bemutatni.

Amikor a mai számítástechnikusok a felhasználói ismereti igényekre alapozva valamilyen számítógépes állományokat képzelnek el, akkor ezt - enyhén szólva - ösztönösen teszik. Ez az ösztönösség abban nyilvánul meg, hogy adattervükben véglegesen egybemosódik a *tartalom* és a *megoldás*. Ezzel pedig a fenti kétféle adatfüggetlenségi elv csorbát szenved. Ez az egybemosódás két dologból fakad. Egyrészt a fejlesztő nem a valóságnak megfelelő tényleges adattartalmat tükrözi a kidolgozott adattervben, hanem az alkalmazó-felhasználó által sugallt korlátozást. Másrészt a fejlesztő az adatterv elkészítésénél az adatkezelő rendszer behatárolt képességeit is eleve számításba veszi. Ezt a két rejtélyesnek tűnő kitéltelt rögtön megmagyarázzuk. Kezdjük a könnyebb tétellel.

D 3.1 Fizikai szintű - röviden: fizikai - adattervnek az ismereteknek az ábrázolását és tárolóeszközökön való elhelyezését rögzítő tervet nevezzük.

Az adatokat számítógépeken, adatkezelő programok segítségével kezeljük. A gépeken az ismeretek a gép technikai felépítésének megfelelő módon kerülnek tárolásra. Bitekben, bájtokban, blokkokban, klaszterekben stb. Most nem ezekre a fizikai tárolási egységekre kell koncentrálnunk, sőt, most az sem érdekes, hogy mi ezeknek a dolgoknak a lényege. A fontos az, hogy a fejlesztőnek a tároláshoz ki kell jelölnie az adat *típusát* és *hosszát*. Ezt a párost hívjuk *ábrázolásnak*. Az adatkezelő rendszerek külön-külön is és együttesen is korlátozzák a típust és a hosszat, továbbá nem azonos kényelmi fokon támogatják a különböző típusú adatok kezelését. Emiatt előfordulhat, hogy a fejlesztő a valós természete szerint numerikus adatot (például: Rendelésszám) karakteres típusúnak definiálja vagy a valóságban keltezés jellegű ismerettel is ugyanezt teszi ahelyett, hogy a dátum ábrázolási formát választaná.

A fejlesztőnek ezek a döntései többnyire jogosak. A baj ott van, hogy csak egy tervet készít, amelyben - a felhasználó nagy megdöbbenésére - a Rendelésszám eleve karakteresnek definiált. De menjünk tovább.

T 3.7 A minap találkoztam egy olyan tervrészlettel, amelyben a tervező az ismerettételek fizikai tárolási sorrendje alapján döntötte el, hogy melyik tétel korábbi keletű. Dátumot nem alkalmazott.

A legtöbb adatkezelő rendszerben a fizikai tárolási sorrendet a fejlesztő nem befolyásolhatja, mi több, sokszor nem is ismeri. Pl. egy meghibásodást követő helyreállítás után a tételek fizikai sorrendje - tehát nem pl. az index által kijelölt logikai egymásutánisága - aligha marad a korábbi. Ezért a fizikai sorrendnek bármilyen elvi tartalmat is tulajdonítani (tétel keltezése) óriási fejlesztői hiba.

Más. Egyes adatkezelők nem kedvelik, ha a tétel - tétel például egy vevőnek a teljes ismeret-sora: azonosítója, neve, címe stb. - túl hosszú, sok mezőt ölel fel, vagy túl sok benne az 'üres', a kitöltetlen mező. Ezért a tervező - elvileg ismét csak jogosan - a vevő tételeket megbontja és kreál

egy Vevő-1 ill. egy Vevő-2 fizikai állományt. A mi problémánk változatlanul az, hogy csak egyetlen tervet készít, ami eleve ezt a megbontott állapotot tükrözi - az eredetit nem.

A 'korszerű' adatkezelők jelentős része nem támogatja a valamilyen okokból 'avított'-nak tartott csoport kezelését. A dátum - csoport, mert évből, hónapból és napból áll. Ezt a valóságos dolgot az adatkezelőnek nem lehet elmondani. Így a fejlesztőnek vagy csak egyetlen adatot (dátum), vagy hármat kell kijelölnie (év, hó, nap). Az előbbi esetben a tényleges dátum részeit, az utóbbiban viszont a teljes dátumot kell műveletekkel - procedurálisan - előállítania vagy a dátumból levágva a megfelelő karaktereket, vagy az év-hó-nap karaktereit összetolva. A fejlesztő a két rossz megoldás közül választ - és tervében azt tükrözi, nem pedig azt a valóságot, hogy a dátum - csoport.

Összegezzük az eddigieket. A fejlesztő nem kerülheti el, hogy a számítógépek természete és az adatkezelők korlátai miatt mesterséges elhelyezési és ábrázolási megoldásokat válasszon illetve kompromisszumokat kössön. Mi több, éppen ez az egyik igen komoly feladata. Informatikai szempontból az a probléma, hogy a fejlesztő csak egyetlen tervet készít. Ebben a megoldandó dolgok és a megoldás maga szétválaszthatatlanul és megmagyarázatlanul összekeverednek. Ez pedig azért baj, mert idővel az eszközöket lecserélik; a korábbi fizikai megoldás ezért érvényét veszíti; ám maga a megoldandó dolog nem dokumentált; tehát a teljes adattervezést kvázi előlről kell kezdeni.

A fizikai terv a fejlesztés harmadik - végső - absztrakciós szintjén készül. A másik kettőnél könnyebb megérteni - azért indítottuk azzal a kifejtéssel. A logikai terv megértése a legnehezebb, ezért most a fogalmi magyarázatára térünk át.

D 3.2 Fogalmi szintű - röviden: fogalmi - adattervnek a valóságnak a kompromisszumoktól mentes képét rögzítő tervet nevezzük.

A mindennapi életben sokszor élünk a „fogalmam sincs” fordulattal. Mintegy jelezzük ezzel, hogy valamilyen ismeret befogadására képtelenek vagyunk. Ha pl. most képtelenek megkérdeznénk az olvasót, hogy mi is az a 'desszin', akkor igen nagy annak a valószínűsége, hogy a „fogalmam sincs” választ kapjuk. (Ez a szó - nagyjából - a textilek anyagát, szövési eljárását, színét, szálúságát és egyéb jellemzőit takarja.) Ha a fejlesztőnek fogalma sincs a valóságról, akkor képtelen a célszerű adatszerkezet megalkotására. Ezért legeslegesítő feladata a felhasználó fogalmainak a megismerése és elrendezése. Elrendezése, mert a fogalmak nem pusztán önálló lényegeket, hanem egymással össze is függenek.

T 3.8 Nem régen láttam egy olyan tervrészletet, amelyben a tervező a 'Bankszámlaszám' fogalmat a 'Partner' fogalomhoz kötötte. Ezzel komoly hibát követett el. Vajon miért?

Valamikor a régi időkben minden szervezetet köteleztek arra, hogy a számláit egy banknál vezesse. Úgy is nevezték az azonosítót, hogy 'Egyszámlaszám'. Így tehát akkoriban jogos volt az a feltételezés, hogy a számlaszám fogalom valóban a partnert jellemzi. Időközben azonban a helyzet megváltozott: ma bárkinek joga van annyi számlát és annyiféle banknál nyitni, amennyit csak akar. Ezért tehát a 'Bankszámlaszám' fogalom nem önmagában a partnert, hanem a partnernek és a pénzintézetnek a viszonyát fejezi ki: az X partner az Y banknál ezen a számon vezetteteti a számláit. Következtetés: A T 3.8 történetben szereplő tervező rosszul - nem a valóságnak megfelelően - fogta fel egy fogalom lényegét, vagyis terve nem felelt meg a D 3.2 meghatározásnak.

A fogalmak tiszta használata és célszerű elrendezése az IR adatrendszerének a kialakításához szükséges alapvető feltétel. Ellenpélda: A fejlesztő 'Egységár'-nak nevezi a 'Megrendelés-tétel'-ben megjelölt árat, amely ugyanarra a cikkre nézve rendeltételenként eltérő értéket is felvehet

(pl. a származás függvényében). Ez megtévesztés, mert az 'Egységár' fogalom éppen az egységes, a cikkekre minden vonatkozásban jellemző árat jelenti.

Az IR adatvetületén belül legelőször a fogalmakat kell tisztázni, azaz meg kell alkotni a fogalmi szintű (1. szint) adattervet. Az 'Egységár' fogalma egy lényeg, amely független attól, hogy az egységárakat a számítógépen hogyan helyezik el és milyen módon ábrázolják a fizikai szintű (3. szint) tervben leírt módon. A két lényeg - a tartalom és az eszközön való megvalósítás - között azonban nem csak az eszközkörnyezet *kvázi-objektív* korlátai miatti ellentétek feszülnek. Figyelni kell az alkalmazási környezetből fakadó *kvázi-szubjektív* megkötésekre is.

D 3.3 Logikai szintű - röviden: logikai - adattervnek az alkalmazási környezet korlátainak megfelelően átalakított kompromisszumos tartalmi (fogalmi) tervet nevezzük.

A kompromisszumok lehetnek tudatosak vagy tudat alattiak. Az utóbbi akkor fordul elő, amikor a fejlesztő nem eléggé figyel az IR felhasználói tényezőjének a teljes rendszerére. A mindennapos gyakorlat szerint a fejlesztőt egy alkalmazói részleghez rendelik (vö. szervezeti strukturálási elv), és ezért ő természetesen az ismereteket kvázi-szubjektíven, az adott részleg szemszögéből érzékeli. Példa: a kötelező biztosítással foglalkozó felhasználók elmondják a fejlesztőnek, hogy a biztosítási szerződésben csak egy szerződő fél szerepelhet. A fejlesztő a közlést általánosítja és azt feltételezi, hogy mindenféle biztosítási szerződésben csak egy szerződő fél szerepelhet. Mivel pedig ez nem igaz, a fejlesztő tudat alatt torzítja el a valóság képét - nem fogalmi, hanem csak logikai szintű tervet alkot akkor, amikor a szerződésekben általában (!) is egyetlen partner megadására ad módot.

A tudatos kompromisszum szemléltetésére álljon itt a következő történet:

T 3.9 Számos olyan vállalattal lehet találkozni, amelyben a dolgozó adatai között úgy tüntetik fel a nyelvtudás ismeretét, hogy csak háromszoros ismétlésre adnak lehetőséget.

Ez az ismereti elrendezés nem tükrözi hűen a valóságot. A valóság ugyanis az, hogy néhány dolgozó egy nyelvet sem beszél, viszont akadhat olyan is, aki több, mint három nyelv birtokában van. Az előbbiek esetében a három 'nyelvrubrika' felesleges, az utóbbiaknál viszont nem elegendő. Gondolja csak meg az olvasó, hogy hányszor, de hányszor találkozunk olyan kitöltendő papírokkal, amelyeken vagy feleslegesen sok, vagy nem elegendő 'rovat' szerepel!

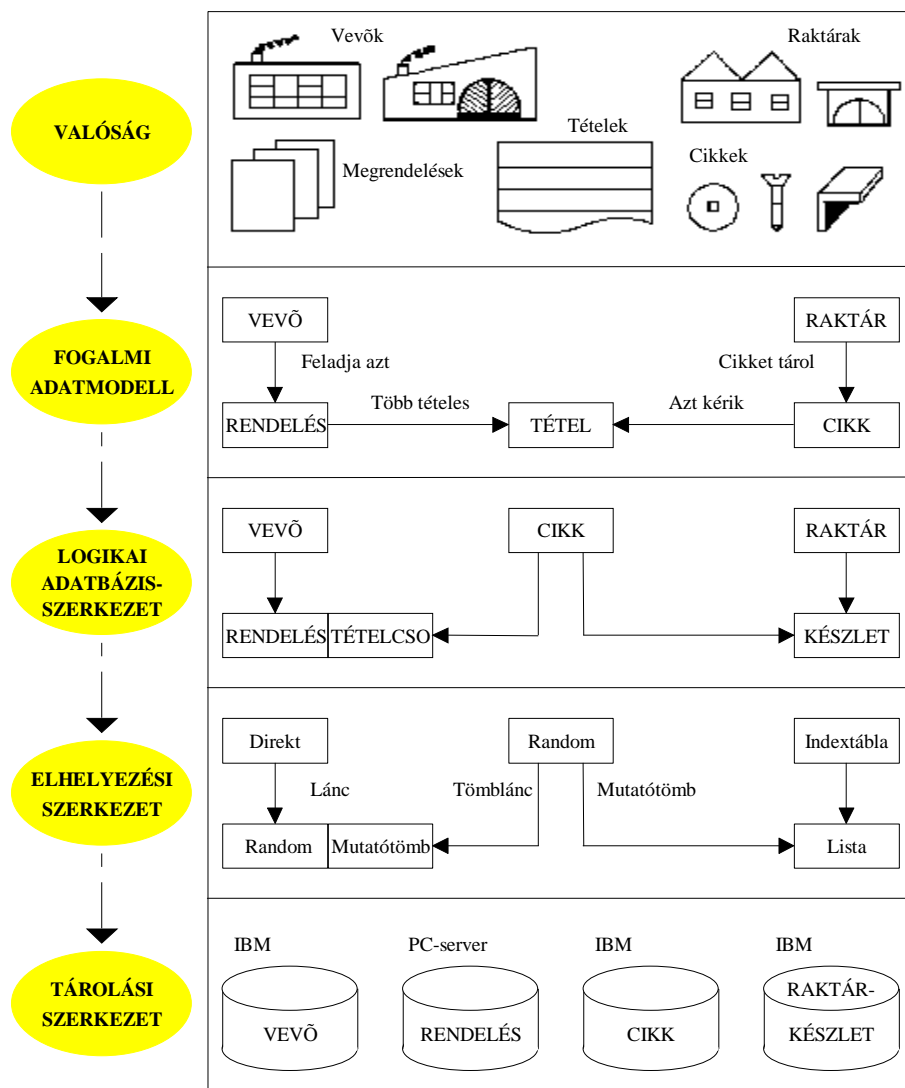
A kérdésünk az, hogy rossz-e a történetben szereplő adatterv? Válaszunk: ezt nem tudhatjuk. Ha pl. a vezetőség elhatározta azt, hogy csak három nyelv után fizet pótlékot; ha nem érdekli, hogy valaki szuahéliül is beszél-e negyedikként; ha nem zavarja, hogy mondjuk az angol nyelv hol első, hol harmadik adatként jelenik meg, akkor számára a megoldás elfogadható. És ez a lényeges: maga az érintett, a felhasználó elfogadja-e a megoldást?

Az a terv, amelyben háromra korlátozzák a nyelvismeretek számát, bizonyára nem fogalmi szintű, mert nem a valóság hű tükörképe. Ám még nem is fizikai szintű, mert még szót sem ejtettünk a gépekről. Logikai szintű (2. szint), vagyis az alkalmazási környezet által korlátozott, kompromisszumos megoldás.

A fejlesztőnek kötelessége keresni az ilyen áthidaló dolgokat. A mi bajunk az, hogy a fejlesztők ezt nem tudatosan teszik. A valós fogalmi képet nem rögzítik, hanem eleve a korlátozást írják le. Utána pedig azt is kijelentik, hogy a rendszer 'rugalmasan bővíthető'. Annak dacára, hogy akár a tudatos, akár a tudat alatti kompromisszumok pontosan a rugalmas bővíthetőség legalapvetőbb gátjai.

A tapasztalt informatikus igyekszik kikerülni az ilyen korlátokat. Ez azt jelenti, hogy a logikai szintű tervben csak minimális változtatásokat végez a fogalmi szintűhöz képest. A nyelv

példája jól érzékelteti a különbségeket. A fogalmi szinten nem két, hanem három jelenséggel kell törődni. Van személy, van nyelv és van egy viszony a kettő között: a nyelvtudás. Az okos fejlesztő ezért három ismeret-állományt fog alkalmazni. A nyelvtudás állományban ma a - korlátoknak megfelelően - egy személyhez csak max. három tétel tartozhat. Holnap viszont ez a korlát feloldható - anélkül, hogy az adatszerkezetet meg kellene változtatni. A tapasztalatlan fejlesztő csak két fogalomban gondolkodik: személy és nyelv. A nyelvtudást nem külön állományként, hanem a személyhez kötődő ismeretként tervezi meg, a mai korlátoknak megfelelően háromszoros adatként. Ezért akkor, ha holnap a korlát feloldódik, akkor - a negyedik nyelv felvételéhez - át kell alakítania a személy adatállományt négyszeres tétellel. Ha később a korlát öt lesz, akkor ismét a szerkezet átalakítására kerül sor. (NB.: Az alábbi ábrában az elhelyezési és tárolási szerkezet együttese a fizikai szint.)



3.5 ábra: Az adatvetület szintjei

Intermezzo

Közismert tény, hogy az informatikusok munkaidejének ma a 60-70 százaléka az eszköz- és a felhasználói koncepcióváltás miatti 'karbantartásokra' fordítódik. Az 'érdemi' fejlesztésre csak 30-40 százaléknyi idő marad - jó esetben.

Ennek az anomáliának több oka van. A legelső az, hogy a fejlesztők túl 'okos' túl 'szép', túl 'ravasz' egyéni megoldásokat alkalmaznak az adattervekben. Azaz a végtelenségig kihasználják az eszköz úgymond adottságait (3. szint) és egyéni módon csúrik-csavarják az adatállományok képét (2. szint). A második ok az, hogy még ezt is következtetlenségül teszik, mert egyetlen leírt tervben rögzítik az elképzeléseik végeredményét, nem választva szét a fizikai és logikai elemeket. Végül a fejlesztőt megtévesztik a korszerűnek mondott fejlesztési eszközök is, amelyekben például fogalmi szintűnek titulálják a valójában csak fizikai szintű - mivel a kezelőrendszer által eleve korlátozott - adattervet.

Most pár szó erejéig térjünk át az adatvetületről a *feldolgozásra*! Ha a szintek fogalma az előbbi illetően sem világos a fejlesztők számára, akkor nem csoda, hogy az utóbbi esetében még kevesebben látják át azok lényegét és fontosságát. Talán egy példa segít megvilágítani a mondani-valónkat.

Tegyük fel, hogy személyeknek és az általuk birtokolt gépkocsiknak az adatait kell kezel-nünk! Az bizonyára mindenki számára világos, hogy ezen feladatnak elvileg és az eszközök függvényében számtalan megoldása lehet. Ezzel már el is árultuk a lényegét: a feladat, az elvi megoldás és az eszköz hármasáról van szó.

Az információs rendszerben olyan valós események lépnek fel, amiket valós tevékenységeknek kell követniük. Valós esemény az, hogy megvásárolunk egy kocsit, amit át kell íratnunk a nevünkre (tevékenység). Ám a valósat mindig az információs kíséri. Ha eddig nem szerepeltünk kocsitulajdonosként az IR-ben, akkor ezt a hiányt pótolni kell. Ugyanez a megállapítás értelem-szerűen a járműre is vonatkozik. Információs esemény az, hogy belép egy új tulajdonos illetve egy új kocsi. Információs tevékenység az, hogy e tényeket rögzítjük. Az új kocsi és az új tulajdonos esemény ill. az új ismeretek bevitelére maga a megfogalmazott feladat, vagyis a *fogalmi szintű* feldolgozás lényege.

A fejlesztőnek ezek után számtalan választási lehetősége van. Itt csak hármat említünk. Az egyik az, hogy egy eljárásba szervezi a tulajdonos és a gépjármű adatainak a bevitelét, egy másik-ba azok karbantartását, egy harmadikba a törlés feladatát stb. A másik az, hogy egyetlen eljárásba tervezi a tulajdonos adatainak a bevitelét, karbantartását és törlését, külön eljárásba foglalva ezen műveleteket a kocsira vonatkozóan. Ad absurdum semmilyen összevonást nem alkalmaz, hanem külön-külön eljárásokat készít a személy és a kocsi bevitelére stb.

Itt a feladatok elvi megoldásáról van szó, és ez jelenti a *logikai szintű* tervet. E szinten a fejlesztő kvázi-objektív és kvázi-szubjektív mércék alapján dönt. Ha például új járművet kell felvenni, akkor azt tulajdonoshoz is kell kötni (objektív momentum). Részben a fejlesztő szokásai, részben a felhasználó igényei - ő úgy szeretné látni a menüket, hogy... - szabják meg azt (szubjektív momentumként), hogy a lehetséges megoldási változatok közül melyikre esik a döntés. Persze e szinten már az alkalmazott adatkezelő is 'közbeszól'. Mert pl. az is eldöntendő kérdés, hogy a személy ismereteinek a lekérdezésére írjunk-e külön programot, vagy erre célszerűbb az adatkezelő általánosított visszakereső rutinjait használni.

A *fizikai szintről* már alig kell beszélnünk. A képernyők elrendezése, a funkciógombok kijelölése, a program tagolása ezerféle módon - de mindig a felhasznált eszközökhöz kötötten - oldható meg. A fejlesztő számára nemcsak lehetőség, hanem kötelesség is a feladat (fogalmi szint) felhasználónak megfelelő elrendezése (logikai szint) és a korlátok közötti hatékony és szép

megoldások (fizikai szint) megkeresése. Nem az a baj, ha a fejlesztő annyit töpreng a lehetséges változatokon. Hanem az, ha a rendszertervben nem választják szét a három feldolgozási szintet.

T 3.10 A szerzőnek ‘szerencséje’ volt egy olyan dokumentációt olvasni, amely így kezdődött: „Üsse le a zöld F6-os gombot!” Úgy véljük, hogy ez a történet nem igényel külön magyarázatot.

A szintekre vonatkozó gondolatsort egy általános észrevétellel kell zárunk. Öröndetes dolog, hogy ma már a számítógép nem áll távol az emberektől és mindenki, aki egy kis figyelmet szán erre az eszközre, azt meglehetősen sikerrel képes használni. Majd egyszer csak az emberekben valami megpattan. Hirtelen és váratlanul bekövetkezik a számítógéppel való viszonyukban egy törés. Hogy miért nem úgy, meg miért nem azt... Vajon mi ennek a jelenségnek az oka?

Az emberek nem veszik észre, hogy amint az eszközt többféle célra kezdik használni, önkéntelenül is - információs rendszert fejlesztenek. Akkor is, ha erre nem gondolnak. A leges-legelső állomány létrehozásakor és a legeslegelső tétel kezelésekor már megvetették valaminek az alapjait. Többnyire rosszul, a hamis - mert nem átgondolt - információs kép birtokában. A sokadik ismeret kijelölése után döbbennek csak rá arra, hogy valamit - sokkal korábban - elrontottak. És ekkor következik be a ‘válság’ korszaka, amely ilyesfajta kifakadásokkal kísért: a számítógép nem is tudja, a számítógép rosszul működik, a számítógép...

Nem az eszközökkel van a baj (no, azért azokkal is akadnak gondok). Hanem az a probléma, hogy a laikusok és a számítástechnikusok nem gondolkodnak a feladat, a megoldási mód és a tényleges megvalósítás hármában. Sem az adat-, sem a feldolgozásvetületben nem ismerik a fogalmi, logikai, fizikai színhármas lényegét. Ezért ha legalul (eszköz) valami változik, nem képesek a középső szint (megoldás) újbóli megragadására. A megoldást pedig mindig összekötik a felső szinttel (feladat), elfeledkezve saját egyéb mindennapi és józan gyakorlataikról. Ha valaki Olaszországban akar nyaralni (‘feladat’), akkor eldönti, hogy miként fog oda eljutni (‘megoldás’) és kiválasztja a közlekedési módot (‘eszköz’). Ha ugyanez a valaki egy személy-nyilvántartást vezet, akkor e három szintre épülő józansága hirtelen elhagyja.

Az alábbi ábra sablonosan összegzi az információs rendszer vetületeit és szintjeit. A leképezés lényegét majd az ábra után fejtjük ki.

KÖRNYEZETI vetület	ADAT vetület	FELDOLGOZÁS vetület	LEKÉPEZÉS
	Fogalmi szint Logikai szint Fizikai szint	Fogalmi szint Logikai szint Fizikai szint	Fogalmi-Logikai Logikai-Fizikai

3.6 ábra: Az információs rendszer vetületei és szintjei

A profi informatikus a fenti ábrának megfelelően strukturálja magában az információs rendszert. Az előző alponthban a vetületekkel, ebben a szintekkel foglalkoztunk. De vajon mit jelent az ábra ‘leképezés’ oszlopa?

Az IR fejlesztője - és mint rámutattunk, fejlesztő az is, aki otthon bekapcsolja a pécéjét - tudatosan vagy fél-ösztönösen alakítja ki a feladat, a megoldási-mód és az eszköz viszonyát. A tudatos és a fél-ösztönös megközelítés között van egy olyan különbség, amely nem csupán a tényleges eredményre hat ki. Nevezetesen a **dokumentációról** van szó.

Szemben a tapasztalatlan számítástechnikussal, a valódi informatikus nemcsak egyetlen adattervet készít, hanem hármat - fogalmi, logikai és fizikai szintűt. Ez a megoldás látszólag többletmunkát jelent és látszólag nincs értelme. Azonban egyrészt többletmunkáról nincs szó,

mert a tudatos fejlesztő amúgy is papírra veti a feladatot, a megoldást és a mikéntet. Másrészt ennek előnye abban rejlik, hogy az eszköz vagy a megoldási mód megváltoztatásakor nem kell a magasabb szintű dokumentációrészletbe belenyúlni, azt viszont jól lehet használni éppen az alacsonyabb szintű dokumentációrészlet átdolgozásánál. Ez az igen kellemes lehetőség viszont csak akkor használható ki, ha...

A fogalmi szintű feladat az, hogy... A fejlesztő az alkalmazási környezet által támasztott speciális igények és korlátok miatt a 'létező legjobb' - a valóságnak a leginkább megfelelő - elképzelést 'kompromittálja', vagyis egy valamilyen fokon tudatosan 'elrontott' képet alakít ki. Ezek után beleütközik az eszköz korlátaiba - tehát ismételtelen a lehetőségek közül kell válogatnia. Nem a legszebb, a legjobb megoldás az, ha a dokumentációban mindhárom szintű tervet rögzíti. A valóban célszerű eljárás az, ha a dokumentációban leírja a döntéseinek az *okait* is.

Százsor tapasztalhatjuk azt, hogy a fejlesztők egy többszörös értekezleten igen hevesen elvitatkoznak arról, hogy mit és miért. Majd jön a következő forduló, és lám: Kovács elfeledkezett a korábbi megegyezés eredményéről, Szabó pedig azt sem tudja, hogy az előbbi értekezleten miért döntöttek úgy, hogy... A parttalan viták legfőbb oka az, hogy a dokumentációk nem tartalmazzák a *leképezés* okát, miértjét. Nincsenek bennük ilyen kitételek: 'Azért terveztünk Vevő-1 és Vevő-2 állományt, mert...'. Más: 'Azért ábrázoljuk karakteresen a Cikkszámot, mert...'. 'Azért vettünk fel csak három nyelvet, mert...'.
A fogalmi-logikai-fizikai színhármas nem azt jelenti, hogy a korábbi tervet eldobjuk, újrafogalmazzuk, kiegészítjük másféle módon. Hanem azt, hogy a célt (fogalmi) az adott megoldási módnak (logikai) megfelelően képezzük le, illetve a módot (logikai) az adott eszközzel (fizikai) szabjuk. E leképezések 'miértjének' a rögzítése több dologtól kímélhet meg bennünket. Az utólagos vitáktól; attól, hogy valaki félreértelmezze a szándékokat; az újonnan belépő igen szubjektív tájékoztatásától stb. (Vegye észre az olvasó, hogy amikor Szabó belép, Kovács nem 'a' rendszert magyarázza el neki, hanem azt, hogy ő személyesen mit tart és mit gondolt a régi viták során is a rendszerről. Ha tehát Szabó Kovács markaiba kerül, akkor teljesen más szubjektív képe alakul ki a dologról, mintha Lakatost bízták volna meg a tájékoztatásával.)

3.2 RENDSZERFEJLESZTÉS

3.2.1 A fejlesztésről általában

Az informatikus mindenkor a valóság, az ismeret és az eszköz (VR - IR - ER) egymással összefüggő hármában gondolkodik. Gondolatában az információs rendszer (IR) egy olyan köztes helyet foglal el, amely kapcsolódik a felsőbbhöz (VR) és alapoz az alsóbbra (ER), de egyiknek sem szolgál alárendeltje. Ez az elv - a gyakorlat pedig, mint tudjuk, teljesen más. Az informatikusnak nincs módja sem arra, hogy átalakítsa a valóságot, sem arra, hogy beleszóljon az eszközökbe, mint másvalakik által meghatározott másodlagos, technikai valóságokba.

E gyakorlati helyzet következtében és az információs rendszerek vetületeinek illetve szintjeinek az elméleti ismerete hiányában aligha csoda, hogy a fejlesztés, az elemzés, a tervezés, a szervezés, a dokumentáció, a specifikáció stb. szavakat meglehetősen zagyván használjuk. Ennek a pontnak az az alapvető célja, hogy e fogalmakat illetően a fejekben némi rendet tegyen. Ámde a szavak pontosítását megelőzően kötelességünk, hogy a felhívjuk a figyelmet a fejlesztés egy mindig elhanyagolt momentumára.

Az informatikus ismeretekkel dolgozik. Az ismeret a valóság tükörképe. Ezért egyáltalán nem csoda, ha az értelmes informatikus „tükör által, homályosan”, de mégis látja magát a

valóságot. Például tudja azt, hogy egy biztosítási szerződés, egy csekk vagy éppen egy törvény helytelen megfogalmazása, rossz képe, azaz célszerűtlen informatikai elrendezése valójában rossz gyakorlatot takar.

Éppen ezért megdöbbenő, amikor az informatikust nem engedik beleszólni a valóságba, ám az általa nem is a legcélszerűbbnek tartott eszközök használatára kényszerítik. Így végeredményben a természete szerint értelmes, kreatív emberi munkáját a mechanikus 'gépesítésre' [computerisation] redukálják. Talán nem árt elgondolkodni azon, hogy az informatikus képességeit mennyire aknázzuk ki.

A mi fejlesztési felfogásunk teljesen más. Az eredeti rendszerelemző [systems analyst] mint informatikus fogalma áll hozzánk a legközelebb. A valódi elemző nem pusztán számítógépen rögzíti és kezeli - mondjuk - a szállítási útvonalakat, hanem keresi azt az algoritmust is (e ponton mélyszéles elnézést kérünk az eddigi a lelkükben netán megbántódott matematikusoktól), amellyel maga a gyakorlati probléma a leghatékonyabban megoldható.

Nem igaz, nem tartható, nem értelmes az a begyöpösödött nézet, amely szerint az információs rendszer fejlesztése tisztán - és főként csak technikai szempontú - megvalósítást jelent. Nem, nem és nem! Csakis az informatikus láthatja, hogy az ismeret meg-nem-felelősége valójában a valós rendszer gyengeségeit leplezi ill. az eszközrendszer korlátainak tulajdonítható. Ezek után már érthető, hogy miért tartjuk annyira döbbenetesnek azt, hogy az informatikusnak mind a feladatokat, mind az eszközöket tekintve 'Hallgass!' a neve.

A fejlesztés lényegét más vonatkozásban is félreértik. Az információs rendszer ún. **életciklusának** [life cycle] három alapvető szakasza van: a legelső kialakítás, a működtetés és a változás fázisairól van szó. Sajnálatos, hogy ezek közül egyes helyeken csak az első lépést tekintik fejlesztésnek. A fejlesztő informatikusnak a működtetésbe nincs sok beleszólása; ez volna a kisebb probléma. A nagyobb az, hogy a változásokat sem tudjuk megfelelően kezelni. Ez abban nyilvánul meg, hogy a felhasználó és/vagy a működtető dönti el egy módosítási, kiegészítési ill. javítási igény felmerülésekor, hogy a fejlesztőkhöz fordul-e vagy sem. Ami baj, mert 'kisebb' - fejlesztésnek nem tekintett - változtatások ürügyén úgy nyúlnak bele a rendszerbe, hogy arra még a fejlesztője sem ismer rá, és - természetesen - a 'kisebb' változtatásokat nem vezetik át a dokumentációkon.

A komolyabb cégeknél a már működtetett rendszerrel kapcsolatos igényeket az ún. **változáskezelés** [change management] szigorú eljárásai ill. mércéi szerint mérlegelik. A változtatást magát is fejlesztésnek tekintik, mert hiszen a rendszer szervezett együttes, tehát bármilyen tényezőjének a módosítása másokra is hat. Ezt a hatást meg kell vizsgálni, az összefüggéseket újra kell dokumentálni stb. Az képtelenség, hogy a rendszert óriási erőfeszítéssel és gondosan megtervezik, majd teljesen ad-hoc módon átalakítgatják. Ezzel már el is jutottunk a következő témánkhoz: a tervezés lényegének az ismertetéséhez.

3.2.2 Háromszoros terv

Az információs rendszer egyes tényezői absztrakciók. Elgondolások, tervek eredményei. Sajnálatos tény, hogy a magyar nyelv fogalmai az IR fejlesztését illetően meglehetősen szűkösek. Például nekünk csak egyetlen 'terv' szavunk van - szemben az angollal. Talán ez a körülmény is közrejátszik abban, hogy helytelenül látunk bizonyos összefüggéseket.

Mondanivalónkat egy egyszerű példával világítjuk meg. Tegyük fel, hogy egy asztalost bízunk meg a konyhabútor elkészítésével. Ekkor ő három tervet fog alkotni. Mielőtt ezeket bemutatnánk, fel kell hívnunk a figyelmet arra, hogy minden példa sántít. Nem mondhatjuk azt, hogy az asztalos nem tudatosan készíti a terveit, de annyit állíthatunk, hogy nagyon sokszor rutinból cselekszik. Ő úgy szokta, hogy... Ez egyáltalán nem baj. Ezzel szemben az informa-

tikusnak nem tiltott ugyan a 'rutin', ámde neki számos emberi tényezőre is figyelnie kell. Éppen ezért minden tervezés mindig újszerű feladatot jelent a számára.

Az első terv [**plan**] az átfogó elrendezésre vonatkozó elképzelést rögzíti. Az asztalos persze globálisan tudja, hogy a konyhaberendezés az nem azonos a hálószooba-bútorral. Viszont van ekkora meg akkora konyha. Az egyikbe két asztal is elfér, a másikba pedig egy sem. Az egyik főzőhelyiség ilyen, a másik olyan alakú, fixen meghatározott falak és szerelvények szerint elrendezett. A 'plan' ugyan még a 'tervezgetés' idejében készül - azonban ez egyáltalán nem jelenti azt, hogy az asztalos elrugaszkodhat az elsődleges valóságtól, vagyis az alkalmazási környezet korlátaitól.

Az átfogó tervnek nemcsak ez a környezet, hanem az erőforrások is korlátot szabnak. Az asztalos - miközben a tervet készíti - meg fogja kérdezni tőlünk, hogy mennyit szánunk rá és mikorra kell elkészülnie a dolognak. Mahagóniból vagy pozdorjalemezből legyen-e a konyhaszekrény. Különbözik pedig addig nem is érdemes hozzáfognia, ameddig a vízcsap bekötési módjáról nem döntöttek. A funkciók sem közömbösek, mert adott használat esetében a nyitott, egyébként az ajtóval zárt megoldás a célszerű.

Mindennek roppant sok köze van az IR tervezéséhez. Nem ártana ugyanis, ha az IR-fejlesztő meg tudná különböztetni a különböző rendszereket egymástól, és nem hálószooba-bútort tervezne a konyhába. (Sajnos az ilyesmi bizony előfordul. A fejlesztő nem a felhasználó által kért rendszert alkotja meg. Vö. T 2.19.) Ezért elengedhetetlen, hogy az anyaggazdálkodási IR-részt tervező egyén jártas legyen - vagy mielőtt munkába fog jártasságra tegyen szert - az anyaggazdálkodásban. Különbözik miképpen is lehetne képes kialakítani egy átfogó elképzelést?

Az információs rendszerrész átfogó terve, a 'plan', a célt, a globális tartalmat, a makroszerkezetet (elrendezést), a helyet, az időt, az eszközöket stb. rögzíti. Ezek a tényezők a projektmenedzsment témakörébe tartoznak, és ezért nem kívánunk itt többet mondani róluk. Már csak egyetlen egy mozzanatra szeretnénk felhívni a figyelmet.

Az asztalos megbízója éppen úgy vágyakat kerget, mint az IR felhasználója. Az asztalos és az IR-fejlesztő mai reflexiói, mai viselkedésmódja ennek dacára teljesen eltérő. Amikor az asztalos felfedezi, hogy a megbízó vágyai irreálisak, akkor a becsületes mester figyelmezteti a lakástulajdonost: 'No de, uram...!' Nem szalad bele rossz megoldásokba, mert félti szakmai hírnevét. Ha a megbízó igen erősködik, akkor vagy talál átmeneti - kompromisszumos - tervet, vagy távozik. Jellemző módon **lebeszéli** a végső-felhasználót a szakmailag hibásról.

Ezzel szemben a - valamilyen értelemben - gyenge informatikus pontosan az ellenkező módon jár el. Isten ments, hogy nemet mondjon a t. felhasználóknak! Éppen ellenkezőleg, a realitásokat semmibe véve még 'egy lapáttal rá is tesz' az amúgy is irreális vágyakra. Egyáltalán nem elszigetelt jelenségről van szó, nem is túlzásról. A T 3.1 történet fényes bizonyítéka annak, hogy az IR fejlesztői ma még nem eléggé képesek a globális tervezésre [planning].

Mindennek az oka a türelmetlenség, a megfontolatlanság. Még pontosabban: a fejlesztési tervek összehangolatlanságának a hiánya, a helytelen sorrend.

Az asztalos nem kapkodja el a végső tervét. Céljaink, elképzeléseink, mi több: gondjaink meghallgatása után nem ad végleges választ, hanem csak azt közli, hogy a végleges ár, időpont, megoldás stb. majd feltehetőleg az lesz. Ez nem a rejtélyeskedés, nem a habozás, netán a becsapási szándék jele, hanem pontosan ellenkezőleg: a megfontoltság. Az asztalosnak ugyanis egyelőre az a feladata, hogy - képletesen szólva - rajzoljon. Felvázolja a konyha minden egyes leendő bútorának külön-külön is a fizikai képét, illetve megadja azok elrendezését.

A konyhakredenc és a többi tárgy ilyen képe már nem plánum, már nemcsak globális elképzelés, hanem a tényleges megvalósítás alapját képező nagyon is konkrét 'műszaki rajz'. Olyan és abban a mértékben pontos eligazítás, amelynek alapján már akár a segédek is meg tudják alkotni a konyhaberendezés tárgyait.

Ez a fizikai kép is terv. Ezt a második tervet angolul **design**-nak nevezik. Itt a képletes értelemben vett (műszaki) rajzról van szó. Arról, amely megmutatja a konyhaszekrény elemeit

éppen úgy, mint azok összefüggéseit. Mi több, azt is, hogy az ajtó - ha van - merre nyílik, hol van a felfüggesztés stb. Nem holt és nem önmagában él ez a 'design', hanem nagyon is kölcsönös a 'plan'-nal való együttműködése. Elvégre a konyhaszekrény ajtajának a nyílási módját és a felfüggesztés lehetőségét a konyha körülményei szabják meg.

Az asztalosok - és minden kézműves - példájából több dolgot tanulhatunk.

Először is azt, hogy van asztalos mester és vannak segédek. Maga a mester készíti az átfogó tervet [plan], majd felvázolja a részletek körvonalait [design], vagyis 'felskicceli' az egyes bútordarabok rajzát. Ezek után a mester magát a kivitelezést - a finom, a csakis általa értett részletektől eltekintve, mert hiszen ő az ezeket értő művész - a segédekre bízta. Érdekes módon az informatikában nem létezik és/vagy nem működik ez a munkamegosztás. Mindenki 'mester' akar lenni, a segéd szerénysége nélkül.

Másodszor tudnunk kell, hogy a régi szakmákban a tervnek [design] bevett konvenciói vannak. Ebben a pillanatban az asztalosénál jobb példa az építészé. Az épület műszaki rajzát ugyan a tervezőmérnök készíti el, de azt a kivitelező is pontosan ugyanúgy érti. Az ábrázolási, reprezentációs szabványok hiányában a kőműves nem tudná jól megalkotni az épületet. Ehhez képest kell mérlegelni azt a tényt, hogy az informatikában nincsenek ilyen megegyezések és/vagy azokat nem alkalmazzák. Ezért nem csoda, hogy az IR egyik fejlesztője nem érti meg a másikat. Az sem szokatlan, hogy az érthetatlenség jogcímére hivatkozva a segéd visszabeszél a mesternek. A jobbik esetben a programozó a szervezőre olvassa a terv úgymond hibáit. A rosszabbik esetben viszont hallgat azokról és a saját feje szerint - képletesen szólva - konyhaszéket farag abból a valamiből, amit pedig eredetileg konyhakredencnek szántak...

Ezer és egy oka van annak, hogy az informatikát egyes helyeken nem mesteri szinten gyakorolják. Az egyik - fentebb említett ok - az, hogy az informatikában a mester és a segéd funkciója még ma sem vált szét. Olyannyira, hogy akadnak szép számmal olyan cégek, amelyekben az IR fejlesztéséhez nem is alkalmaznak rendszerszervezőt, mindent a programozókra bíznak. Az sem ritka, hogy a farok csóválja a kutyát; a rendszertervezőre rálegyintenek, mert mindenki az úgymond 'szuperprogramozó' bűvöletében él.

T 3.11 A szerző egyik munkahelyén a fejlesztői társaság nagyobb része lelkes és lelkiismeretes munkával formálgatta a tervet. Majd jött 'a' programozó, aki arra fittyet hányva a saját ötletei szerint egy pillanat alatt egy másik megoldást vezetett be.

Reméljük, hogy az olvasó megérti: nem abból lesz az asztalos mester, aki a legjobban ért a körfűrészgéphez...

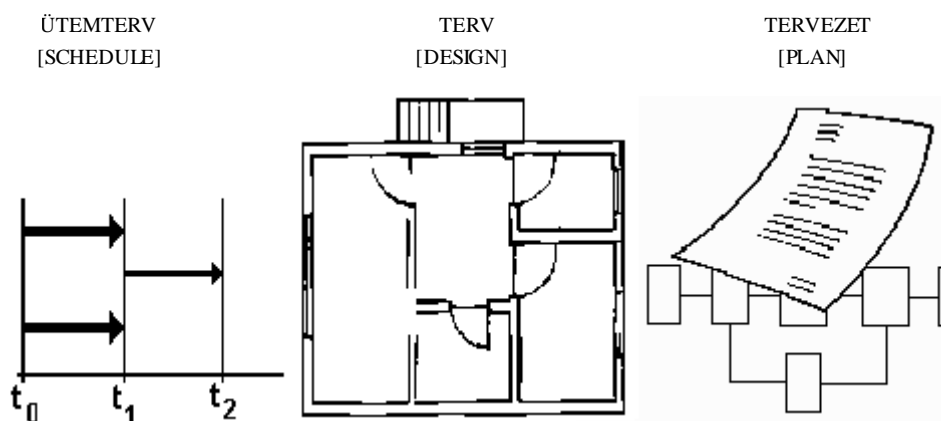
Az asztalos példája egy további dologra is figyelmeztet bennünket. Az átfogó terv, a 'tervezet' és a részletes elképzelés szükségképpen visszahat egymásra. Az asztalosmester a 'tervezgetés' fázisában be tudja mérni a nagyságrendeket: ha így, akkor ennyi, ha úgy akkor annyi. Az azonban tőle sem várható el, hogy már a legelső alkalommal pontos erőforrásbeli kalkulációt adjon. Azért nem, mert az anyagok beszerezhetősége, ára, minősége stb. folyton változik. Ezért a legvégső megvalósítás még számos - sokszor rajta kívül eső - tényező függvénye.

Pontosan ez a helyzet az informatikában. Mindaddig, ameddig nincs legalább valamilyen fokon részletezett egzakt elképzelésünk az adatok, az események és a tevékenységek elrendezéséről, a megvalósítás erőforrásait nem lehet pontosan megadni. Ehhez képest meglepő, hogy a vezetők a fejlesztőket 'madárjósításokra' kényszerítik, a fejlesztés legelején várva tőlük az idő- és költségadatokat. Pedig másutt már általánosan elterjedt az ún. '**csúszótervezés**' gondolata. Ez azt jelenti, hogy az IR erőforrásigényeit legalább két (valójában több) lépésben határozzák meg. Az első behatárolás becslés, amely az átfogó tervezeten alapul. A terv első vázlatának az elkészítése után kerül csak sor a fejlesztés végső költségének és időigényének a megadására úgy, hogy azt is

még több alkalommal tudatosan módosítják. Minderről azonban majd csak a projekt-menedzsmenttel foglalkozó kötetben fogunk bővebben szólni. Most a harmadikféle tervről kell szót ejteni.

Tervének [design] első felvázolása után asztalosunk többféle számítást végez. Számol az anyagokkal, a segédanyagokkal, az eszközökkel, a munkálatokkal és azzal, hogy lesznek-e segédei - és ha igen, akkor beosztja azok feladatait is. Ezt a harmadikféle elképzeléssort **kivitelezési terv**nek vagy ütemtervnek [schedule] szokták nevezni. Egy dolog a ház általános képe [plan], más annak részletes műszaki rajza [design] és ismét más a munkálatok ütemezése [schedule].

Természetesen ezt a tervet tekintve is megtalálható az informatikai párhuzam. Az információs rendszerek fejlesztésében is be kell osztani az erőforrásokat és ütemezni kell a munkát. Néha ezt a tevékenységet projekt-tervezésnek is hívjuk. A **projekt-terv** [project-plan] megnevezés talán egy picit furcsa, mivel magának a projekt szónak az első jelentése is terv, tervezet. Tehát a projekt-terv eredetileg és szó szerint terv-tervezetet jelent. Ma projektnek a viszonylag huzamos ideig tartó és erőforrások összehangolását igénylő fejlesztési feladatot nevezik. Ezért a projekt-terv nem más, mint a fejlesztési feladat terve.



3.7 ábra: A háromféle terv

A projekt-tervezéssel - mint már említettük - itt nem foglalkozunk. Most csak arra akarunk rámutatni, hogy egy igen összetett gazdasági, műszaki, személyzeti és jogi - és persze nem utolsósorban pedig informatikai - feladatról van szó. Az ütemezés [schedule] a projekt-tervnek csak az egyik, bár igen fontos része. Az átfogó tervezet [plan] ill. a terv [design] egyes részletei szintén a projekt-tervhez tartoznak. Ezért visszaszál az a gyakorlat, amikor projekt-terv címen csak néhány oldalnyi, a munkálatok ütemezését mutató ún. Gant-diagramot készítenek.

3.2.3 Tervezés és elemzés, dokumentáció és specifikáció

A magyarban nem teszünk különbséget terv [plan] és terv [design] között. Az angolnak viszont nincs külön szava a tervezésre és a tervre (adott viszonyokban mindkettőnek 'design' a neve). Ezért a nemzetközi szakirodalomban a két dolog megkülönböztetésére az alább bemutatott pontosító terminológiát vezették be:

D 3.4 Tervezési-folyamatnak [design-process], azaz tervezésnek hívjuk azt az időben lezajló tevékenység-sorozatot, amelynek során az információs rendszer adott vetületekbe és szintekbe sorolt elemeit meghatározzuk illetve azokat tudatosan összehangoljuk.

A tervezés nem tévesztendő össze a fejlesztéssel, amelyen mi feljebb az IR kialakítását, működtetését és állandó átformálását értettük. A tervezés része a fejlesztésnek, de az utóbbi másféle tevékenységeket is felölel.

D 3.5 Terv-terméknek [design-product], azaz tervnek a tervezési-folyamat végeredményét, produktumát nevezzük.

A terv nem a fejlesztés egyetlen terméke, mert hiszen a tervezés sem egyenlő a fejlesztéssel. A képernyőterv a tervtermék része, ám az azt megvalósító program is termék, ami viszont egyáltalán nem terv, hanem kész valóság. Legújabbban a fejlesztés teendőit és produktumait a 'tasks & deliverables' kifejezéssel illetik. A szópáros jól mutatja, hogy a fejlesztés fázisaiban vannak elvégzendő *feladatok* [tasks], amelyek végén 'leszállítandó' tételeknek [deliverables], *résztermékeknek* kell születniük. Ezek után az alapvető kérdés az, hogy miben áll e feladatok és e termékek lényege?

Régi vita tárgya, hogy miképpen kell tekinteni a **rendszerelemzés** [analysis] és a **rendszertervezés** [design] viszonyát. Az elméleti informatikusok egy része a kettő azonossága, más része a kettő eltérése mellett voksol. (Nem habozást, nem következtelenséget mutat, hanem a dilemma elvi nehézségét jelzi, hogy egyes szakemberek időről-időre átpártolnak az egyik táborból a másikba.) A probléma nem pusztán elméleti: a gyakorlat szempontjából egyáltalán nem közömbös az, hogy elvileg miképpen rendezzük el magunkban az elemzés és tervezés dolgait. Például amikor használni óhajtunk egy fejlesztési módszert és eszközt, akkor mi mit értünk elemzés és tervezés alatt.

A viták feloldása érdekében induljunk ki abból, hogy az elemzés fogalmát két értelemben lehet megközelíteni. Az elemzés lehet egy *tevékenységgör*, és lehet nem egy, hanem több tevékenységgörön belüli *rész munkálát*. Ebben a műben ez a kétféle szemlélet nem kizárja, hanem kiegészíti egymást. Nem pusztán az ellentét rugalmas elsimítására kell itt gondolni: az IR fejlesztése során valóban kétféle elemzést kell alkalmazni.

A fejlesztési munkálatokat fázisokra, szakaszokra szokták bontani. Maga a fejlesztési - mondjuk így - ötlet két dologból fakadhat, amely két dolog persze rokon annyiban, hogy mindig a tökéletesítés a cél. Az ötlet megfogalmazása a projekt legelső szakasza. Az új gondolatnak alapot adhat az, hogy valami *nem megy a rendjén* (negatív mozgatórúgó), de ne legyünk annyira pesszimisták, mint a régi informatikai szakkönyvek, amelyek a fejlesztést mindig valamilyen 'problémával' indokolták. A fejlesztésnek indoka lehet az is, hogy az IR egyik vagy másik elemét illetően *új megoldásra bukkanunk* (pozitív mozgatórúgó). Mindkét esetben közös, hogy az ötletet pontosan megfogalmazott tudatos céllá formáljuk (1. szakasz), majd megvizsgáljuk, hogy megalapozott-e a kitűzött cél (2. szakasz). Ezt a vizsgálati fázist nevezzük elemzésnek e szó első értelmében. Tehát az elemzés a fejlesztés egyik szakasza, annak egyik tevékenységi köre.

Az ötlet jellegétől függően más és más ennek a fázisnak a lényege. Ha valami problémát észlelünk, akkor helyzetelemzést kell végrehajtanunk. Ezen a *tények feltárását* [fact finding] kell érteni. Ezzel szemben az új gondolat felmerülésekor annak értelmességét kell vizsgálnunk, amit - nem szép ez a szó - *kiértékelésnek* [evaluation] hívunk. Persze ez a kétféle dolog sokszor egybeesik, és ezért a fejlesztés elemzési szakaszában a tényfeltárás és az értékelés összekapcsolódik.

Az elemzés első jelentésének a magyarázatát nem zárhatjuk le anélkül, hogy vissza ne térjünk a rendszerelemzésnek egy nálunk elhanyagolt, ám roppantul lényeges mozzanatára. Az

elemzés *tények* feltárásával jár. A *rendszerelemzőnek* [systems analyst] ezért nemcsak az informatikai értékelés a feladata, hanem az is, hogy magát a valós rendszert is vizsgálja. Ezért nem helyes az a gyakorlat, amikor az informatikainak nevezett fejlesztés azzal kezdődik, hogy 'itt van ez a kötvény, tessék erre számítógépes rendszert építeni'. Egyrészt a kötvény maga is az IR tényezője, tehát annak kialakítása is - informatikai fejlesztés. Másrészt a nevezett papír esetleg olyan valós tevékenységi gyakorlatot is tükrözhet, amely az informatikai szempontok szerint tökéletlennek, hibásnak minősül. Ezért a mögöttes valóság átalakítása is fejlesztést igényelne.

Most térjünk át az elemzés szó másik értelmére. Sokakkal szemben nekünk az a véleményünk, hogy a helyzetfeltárás/kiértékelés szakasza után már nincs újabb elemzési projekt-szakasz. Az elemzés és a tervezés ekkor már elválaszthatatlanul összekapcsolódik annyira, hogy még a fenti részmunkálat kifejezés sem tükrözi helyesen e szoros viszony lényegét.

A tervezés - a design értelmében - az elrendezett elgondolás megfogalmazását, képletesen szólva annak 'papírra vetését' jelenti (vö. műszaki rajz). Mármost az építész készít ugyan skicceket, de számára a tervezés nem azzal kezdődik, hogy azonnal felrajzolja a ház ilyen-olyan vetületét. Az építőmérnök először mérlegel: mi lenne, ha így és mi lenne, ha úgy... Mi az előnye ennek a megoldásnak, és mi a hátránya a másiknak? Változatok lebegnek a szeme előtt úgy, hogy mindegyik alapos, komoly és megfontolt összevetés tárgya. Ez az elemzés mozzanata, ami viszont elválaszthatatlan a tervezéstől. Meg kellene végre érteni az elemzésnek és tervezésnek ezt a teljes dialektikus egységét! Mert miképpen lehetne az egyik megoldást a másikkal összevetni (elemezni), ha nincs a fejünkben mindkettőnek a képe (terve)? És nevezhetjük-e tervnek az első ötlet megfogalmazását, ha azt nem mértük más megvalósítási változatokhoz?

Sajnos az informatikában nem nagyunk ügyelünk a szavakra. (Csak játékosan említjük, hogy ez óriási ellentmondás, mert a szó - ismeret, azaz az informatikus alapvető anyaga.) Amikor megvásárlunk egy olyan eszközt illetve módszert, amelynek a nevében az IR 'elemzési és tervezési' kifejezés szerepel, akkor némi csalódás fog érni bennünket. Hamarosan kiderül, hogy elemzésen az értékelést fel sem ölelő helyzetfelmérést kell érteni, viszont a tervezési részmunkálatban az elemzés momentuma nem kapcsolódik a kiötléséhez.

T 3.12 1993-ban e könyv írójának egy adatbázis-részlet tervét kellett elemeznie. Az eredeti elképzelést átvizsgálva több tucat változási javaslatot tett. Majd később döbbenet tapasztalta, hogy a régi rossz terv a maga változatlan formájában került rögzítésre a CASE eszköz alatt.

A számítógéppel támogatott rendszertervezési [CASE = Computer Aided Systems Engineering] eszközök és az ezek alapjául módszerek ma még nem biztosítanak megfelelő elemzési lehetőségeket. Nem az alapos *mérlegelés*, hanem az első gondolat gyors és kényelmes *leírása* támogatásának a jegyében születtek. Így előáll az a furcsa ellentmondás, hogy az információs rendszerben csak úgy-ahogy jártas számítástechnikus a legkorszerűbb eszközzel könnyen és gyorsan készíthet úgy a szerkezetében, mind a tartalmában teljesen korszerűtlen - a modern informatikai elveknek meg nem felelő - rendszert.

Intermezzo

E könyv szerzőjét gyakran megkérdik, hogy miért nem szentel több figyelmet a ma annyira népszerű **objektum-orientált** [object-oriented] gondolatnak. Most itt nincs lehetőség arra, hogy ezt a lényeget pontosan kifejtjük. Nagyon lazán és pontatlanul szólva az 'objektum' egy olyan számítógépes tényezőegyesítés, amely együtt szemlélhető, formálható és mozgatható a rendszer-részek között. Például a képernyő felépítése, tartalma, a megjelenítő és kezelő programok, a hiba- és más üzenetek, amelyek a káphez kötődnek, együtt egy objektumot jelenthetnek.

Az alapgondolat nemes, ha nem is újkeletű (mint sokan hiszik). De itt és most nem ez a lényeg. Hanem az, hogy az objektum-orientált eszközök és módszerek semmiféle támpontot nem adnak az objektum célszerű kijelölésére. Ergo: bárki fejleszthet ilyen eszközzel és módszerrel - nem objektum-orientált rendszert.

T 3.13 A minap este egy kis vitám támadt három programozóval egy rendszert illetően. Ők objektum-orientált eszközzel készítették az itt-ott hibás tervet. Megkérdeztem tőlük, hogy mi az az objektum. A sűrű csendet késsel lehetett volna vágni...

Objektum-orientált eszközzel fejleszték objektum-orientált módon, de nem is tudom, hogy mi is az az objektum. Csodás! Ezek után hiába várjuk azt, hogy a számítástechnikus az elemzés sokkal bonyolultabb lényegét megértse. Nincs is rá igénye: a gyors leíráson van a hangsúly.

Ezzel áttérünk a **dokumentáció** kérdésére. A házat nem lehet műszaki rajzok nélkül megépíteni, az IR-t nem lehet tervek nélkül fejleszteni. A terv 'írásosan' - szövegekben, listákban és rajzokban: azaz dokumentációban - jelenik meg. Az eddigi dörgedelmek jogosságából semmit sem von le az a tény, hogy a korszerű eszközök legalább a dokumentáció terén hatalmas előrelépést jelentettek.

T 3.14 1992-ben e könyv szerzőjét megbízták egy rendszer felügyeletével. Bekérte az X alrendszer dokumentációját. Egyetlen egy mocskos és gyűrött A4-es papíron három értelmetlen és magyartalan mondat volt 'a' programozó dokumentációja.

Az előző történet három programozójáról becsülettel el kell mondani, hogy a CASE-zel - és nem kézzel - készített tervnek azonnal fel tudták mutatni az akár szépnek is nevezhető dokumentációját. A korszerű eszközöknek van egy másik nagy előnye is. Nevezetesen az, hogy az IR egyes programrészeit automatikus módon lehet velük összeállítani.

Vajon miért ez a nagy ováció? A kézzel készített tervek sorsa közismert, nem csak az informatikában! A szerző lakásában akad jó pár dolog, amely még csak távolról sem emlékeztet a tervezőmérnök egykori elgondolására. Most nem az a lényeg, hogy a megoldás jobb-e vagy rosszabb, mint a terv. Hanem az, hogy a valóság és annak tükörképe - a dokumentáció - nem áll összhangban egymással. A korszerű eszközök automatikus programgenerálási képessége némi reményt ad arra, hogy az említett olló összezárljon. Persze az továbbra sem zárható ki, hogy a CASE-zel fejlesztett rendszerbe valaki 'gyalogos módszerrel' belenyúl.

Egy IR fejlesztése során számtalanféle dokumentáció születik. Rendszerleíró, felhasználói, működtetési stb. Mi itt csak kétféle dologról beszélünk. A tervezési folyamat során - remélhetőleg - több tervvariáns születik. Ezeket a **tervezésinek** nevezett dokumentáció rögzíti. Ezek legvégső változata a **tervdokumentáció**. Az előbbieket - a meg nem valósuló elgondolásokat - is komolyan veszi a tapasztalt informatikus. Ugyanis gyakran előfordul, hogy egy régebbi változat bizonyul a jobbnak valamilyen okok miatt. A tervezés kutatást, útkeresést, zsákutcába való tévedést stb. jelent. A régi változat komoly dokumentálása és a dokumentáció megőrzése sok többletmunkától mentheti meg az informatikust.

Már csak a tervdokumentáció céljait kell helyesen látnunk. A tapasztalatlanok azt hiszik, hogy az általuk elkészített dokumentáció csakis a saját magánügyük. Azt - ilyen-olyan okokból - 'szemérmesen' rejtegetik mások elől. Abban a téves hitben élnek, hogy a terv csak az általuk történő megvalósításnak az alapja. Ez a felfogás többszörösen helytelen. Egyrészt a komoly cégeknél bizony előfordul, hogy a terv készítője és a projekt kivitelezője nem ugyanaz a személy, még csak nem is ugyanaz az informatikai vállalat. Másrészt a dokumentáció a vezetőnek az eszköze is arra, hogy mérje a fejlesztők teljesítményét. (Közismert az a szinte népi játék, amely

szerint a fejlesztő azt állítja, hogy pl. 80 százalékgig elkészült a dolgokkal, pedig még 30 százaléknál sem tart...) Harmadrészt a dokumentáció a felhasználóval való kommunikációnak is az eszköze. Negyedrészt a fejlesztésbe újonnan belépő résztvevők abból ismerhetik meg a munka lényegét. Ötödrészt a terv a tervezet [plan] erőforrás-becsléseinek a kiigazítására is szolgál.

E sokféle használati mód miatt a tervdokumentációt igen komolyan kell venni. Már csak az a kérdés maradt hátra, hogy mit kell tartalmaznia annak? Ugyanis a felsorolt célok érvényesülésének az is gátja, ha a dokumentáció hiányos, no meg az is, ha túl részletes. Általában az utóbbi jelenséggel van a több gond.

A régi fejlesztési módszerekben tudatosan megkülönböztették a tervezés és a specifikálás szakaszát. Ennek megfelelően különbséget tettek a dokumentáció és a **specifikáció** között. Az előbbi az átfogó, az utóbbi a részlet-jellegű tényezőket ölelte fel. Ma erről a két dologról elfeledkezünk, és egyetlen rendszer-leírásban zsúfoljuk össze valamennyi tudásunkat. Ráadásul rosszul szerkesztett módon, mert nem a céllal, a körülményekkel kezdjük a tervünket, hanem másként.

T 3.15 A szerző kézhez kapott egy dokumentációt, amely azzal kezdődött, hogy tételesen (!) felsorolta a rendszerben alkalmazott kód-név párosokat.

A másik oldalon mára már a specifikálás és a program-specifikáció igen sokat veszített korábbi természetéből és jelentőségéből. Ennek a dolognak egy roppant fontos elvi oka van. Régebben a **lényeges** - és az ekként áttekinthető, a vezetőre illetve a felhasználóra is tartozó - összefüggéseket tették a tervdokumentációba, az úgymond **lényegtelen** - mert csakis a szervező és a programozó dialógusát érintő - dolgokat pedig a specifikációba rakták. Csak egy példát említünk: az ún. **validálási** rutinok - a kezelt adatok formai/tartalmi ellenőrzését végző eljárások - részletekként a specifikációba kerültek.

Mindennek az volt az oka, hogy a szervező és a programozó egy felhasználó ill. alkalmazói részleg ismeret-igényeiben gondolkodott. A közösen használt és korszerű adatbázisok ezt a régi szemléletet elsöpörték. Az is kiderült, hogy nem tartható a régi 'lényeges - lényegtelen' felfogásmód, mert éppen például a közös használatú ismereteknek a validálása előre megbeszélendő és súlyos lényeg. Ez logikus is, hiszen egyáltalán nemcsak a program és a programozó dolga az, hogy az alkalmazási-felhasználó miként töltheti ki például a 'Számlaszám' adatot.

A fentiek miatt ma már a korszerű módszerekben nem a lényeges - lényegtelen páros, hanem a feladat - megoldás - eszköz hármas a mérvado. Vagyis e ponton szépen visszakanyarodtunk a fogalmi-logikai-fizikai színhármashoz. Tehát az IR tervét nem 'dokumentálni és specifikálni' kell, hanem el kell készíteni annak a fogalmi, logikai és fizikai szintű tervét. A korszerű eszközök (CASE) ma már nagy segítséget nyújtanak ehhez. Egyelőre ugyan még egy picit összezavarják a három szint tényezőit, de legyünk türelemmel: a helyzet majdcsak javulni fog.

3.2.4 Tervezés, szervezés, modellezés

Elérkeztünk az egyik legnehezebben érthető gondolatsorhoz. Ismét csak arra kényszerülünk, hogy a szavakhoz és tartalmukhoz folyamodjunk, ha az alcím három fogalmát meg akarjuk világítani. Az angolszász nyelvterületeken az IR lényegének a megálmodóját hol tervezőnek [designer], hol elemzőnek [analyst] hívják. Nálunk a **rendszer-szervező** szó terjedt el, amelynek nincs is angol párja. (Egy ideig a folyamatszervező kifejezés is dívott. Ha valaki felsőfokú képzettség birtokában volt, akkor belőle rendszer-, amúgy csak folyamatszervező válhatott az informatikai vizsgák letétele után. Ez az ostobaság nálunk akkor uralkodott, amikor legnagyobb színészeink azok voltak, akik színiiskolát nem is végeztek...)

Vajon miért nem közömbös számunkra az, hogy miképpen hívják a rendszer lényegének az elképzelőjét? Meg fogjuk magyarázni.

A **tervezés** azt jelenti, hogy az információs rendszer - mint szervezett együttes - absztrakt természetű tényezőit (adat, esemény, tevékenység) a három tervezési szintnek megfelelően meghatározzuk és ezeknek illetve a konkrét tényezőknek a viszonyait is elgondoljuk. Nem kialakítjuk, hanem csak elgondoljuk. Az építész csak képletesen épít házat. Azt valójában nem ő teszi, ő 'csak' elgondolja a ház képét éppen úgy, mint a ház és a lakó (vö. felhasználó) viszonyát. A bizonylatot nem lehet 'megszervezni', csak 'megtervezni'. Viszont a bizonylatáramlást el kell gondolni (tervezés), ám ugyanakkor gondoskodni kell a gyakorlati megvalósítás módjáról is (szervezés).

A **szervezés** a feladatok és az erőforrások gyakorlati összehangolását jelenti. A ház felépítéséhez meg kell szervezni az anyagok szállítását, a munkások idejét, a gépek használatát stb. Az információs rendszerben meg kell szervezni az ismeret áramlását, a felhasználók együttműködését, a számítógépek működtetését stb. A rendszerszervező szakkifejezés elleni ódzkodásunk a tervezés és a szervezés rövid magyarázata után talán már nem tűnik alaptalannak.

Nálunk a rendszerszervező az az egyetlen személy, aki - tisztes címe dacára - bele sem szólhat a bizonylatok áramlásába, a felhasználók együttműködésébe stb. Valamilyen furcsán ellentmondásos módon nálunk a számítógépen kívüli dolgok nem a szervezőre tartoznak. **Ügyvitelszervező**, vállalatsszervező az, aki nálunk valóban felelős ezekért a lényegekért. Ez pedig tudathasadásos állapotot eredményez. Az 'ügy vitele' ugyanis informatikai dolog. Ehhez képest meglepő, hogy az 'ügyvitelek' az 'informatikus' - a rendszerszervező - megkérdezése és bevonása nélkül új bizonylatokat, kódokat, adatokat találnak ki - a már meglévő rendszerreszeknek tökéletesen ellentmondó módon.

Ezért lássa be az olvasó, hogy ismét csak nem a szavakkal, hanem a mögöttük lévő lényegekekkel van a gondunk. Kitaláltuk azt a rendszerszervezőt, aki mindent csinál, csak éppen rendszert nem 'szervez'. Ami nagyon nagy baj. Azért az, mert amit az egyik ember (tervező) elméletileg jól elgondol, azt egy másik (szervező) - a kölcsönös megértés hiánya miatt - tökéletesen elronthatja. És megfordítva: a majdani működés (szervezés) átlátásának a hiányában a kialakítás (tervezés) sem lehet tökéletes. Hányszor, de hányszor tapasztalhatjuk azt, hogy egy elméletileg jó rendszer gyakorlatilag csődöt mond!

Ám ideje áttérnünk a legutolsó - bár korántsem a leglényegtelenebb - témára. Az angol 'design' szó nemcsak tervet és tervezést jelent, hanem van egy olyan harmadik értelme is, ami számunkra nagyon is fontos. Iparművészetnek illetve **formatervezésnek** is fordíthatjuk (vö. 'industrial design'). Itt gondoljanak arra, hogy vannak olyan tárgyak, amelyek adott értelemben szépek, csak éppen nem használhatók a céljuknak, funkciójuknak megfelelően. Mások viszont csúnyák, de például kényelmesek. (Az a szék, amelyen ülve írom ezt a könyvet gyönyörű, de kényelmetlen - nem számítógép elé való. Viszont másikat nem akarok venni, mert kaphatók kényelmes, számítógép elé való székek, de mind randák.)

A formatervezés szóban is szerepel a 'tervezés', ám a 'forma' jelző csak részben árulja el a lényegét. A formatervezett tárgy - jó és szép. Azaz egyrészt megfelel a funkciójának, másrészt eleget tesz az esztétikai követelményeknek is. Ebből az következik, hogy nem minden terv 'design'; csak a jól működő, elegáns terv az. Ilyenek készítésére kellene megtanítani az informatikusokat is. Nem elegendő, ha a fejlesztő egy elfogadható, így-úgy elműködgető rendszert készít. Szépet és jót - igényeset - kellene alkotnia.

Az informatikus kollégák jó része eme követelmény hallatán - mert itt bizony arról van szó - fel fog horkanni. Honnan veszi a szerző a bátorságot... Ők arra törekednek - igényes szakemberekként -, hogy a létező legszebb és legjobb IR-t alkossák meg. Ámde 'formatervezett' információs rendszerek kialakítására nincs elegendő idő. A szerző ne őket, hanem a vezetőket és felhasználókat győzködjé!

Elismerve azt, hogy az idő mindig kevés, két dologra legyen szabad felhívni a becses figyelmet. Az egyik az, hogy a gyakorló informatikusok 90%-a még csak nem is hallott a következő fejezetben taglalt **optimum-kritériumokról**. Nem az időhiány, hanem e tájékozatlanság az oka

annak, ha az IR nem elegáns. Sajnos a fejlesztők szubjektív optimum-érzékét nem fogadhatjuk el objektív mércének. A 'jó' és a 'jobb' ma már egészen pontosan mérhető. A másik dolog az, hogy hiba azt hinni: a 'jobb' kialakítása több időt igényel, mint a 'jóé'. Például ha a tervezés legeslegelején nem fordítunk kellő energiát az egyértelműsége, akkor a későbbi fázisokban azt sokkal nagyobb erővel kell helyreállítanunk. Mivel már ezernyi helyen félreérthetően írtunk le egy dolgot, azt ezernyi helyen kell korrigálnunk.

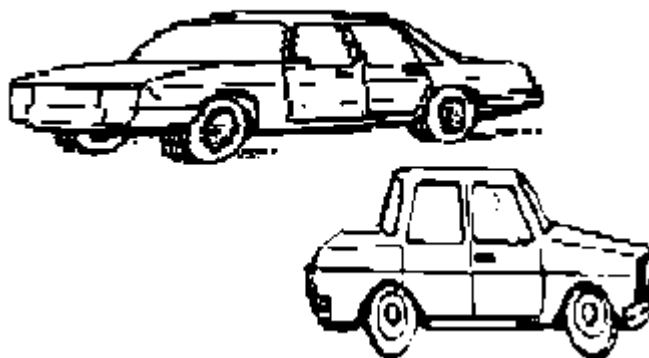
„Háromszor mérj, mielőtt vágasz!” - ez a bölcs asztalosmester tanácsa. A fiatal segéd prüsszöl, a felhasználó toporzékol. Majd az első téves levágást követően döbbennek rá arra, hogy nincs mit visszacsínálni. A hetvenes években a rendszer életciklusában 30 százaléknyi időt - erőforrást - szántak a tervezésre, 70 százalék jutott a megvalósításra. Talán érdemes elgondolkodni azon, hogy ma pontosan a fordított idő-megosztást sugallja minden fejlesztési módszer.

Már csak egy téma maradt hátra. Az utóbbi időkben az információs rendszerek tervét nem is így, hanem **modellnek** nevezik. Mivel ez az újabb varázsszó - mint minden felröppentett újdonság - félelmetes sebességgel terjed, nem kerülhetjük el, hogy pár gondolatot fűzzünk hozzá.

A modell szót igen sokféle értelemben használjuk. A festő a valóságot a maga módján tükrözi, amihez neki 'modellt ülnek'. A hajónak - az építése megkezdése előtt - elkészítik a kicsinyített mását, a modelljét. A szerszámok, az építőelemek legyártása előtt mintadarabban - modellel - folytatnak kísérleteket. De modellnek tekinthető a matematikai, fizikai, kémiai stb. képlet is. Amikor egy sablonnal pl. kört rajzolunk, akkor modellezünk, mert hiszen adott mintára készítünk valamit. Modell a tervrajz, a prototípus, a példa, sőt még a példakép is.

Az információs rendszerek terve magában hordozza a modell-fogalom szinte valamennyi jegyét. A terv a valóság tükröképe és az interjúk során a felhasználó - képletesen szólva - modellt ül a fejlesztőnek. A terv kicsinyített más; absztrakt és egyszeres formában foglalja össze a konkrétumok ezreit. A jó informatikus a terven gondolati kísérleteket folytat. Eltöpreng az IR tényezőinek a viszonyain - vagyis az információs rendszer képletén. A másutt egyszer már használt és jónak bizonyult megoldásokat újra előveszi, azaz sablonokat alkalmaz. Azt pedig nem is kell mondanunk, hogy az értelmes informatikus mindig prototípust készít.

A mi gondunk ismételt az, hogy a szavak elcsépelődnek és elveszítik eredeti jelentésüket. Pusztán attól, hogy máttól modellnek nevezzük a tervünket, az még nem lesz sem szebb, sem jobb. Márpedig manapság meglehetősen jellemző az, hogy a számítástechnikusok nem terveznek, hanem modelleznek - és ez a piciny szóhasználati eltérés a megbízókat roppantul megnyugtatja.



3.8 ábra: Design és terv

A modell nem a terv szinonimája, legalábbis az informatikában nem az. Egyes elméleti informatikai tudorok a modellezés fogalmát azért találták ki, mert látták a gyakorlati szakemberek képtelenségeit. Az új fogalom eredeti értelme szerint modellnek az optimum-kritériumoknak megfelelő tervet, modellezésnek pedig az azok betartására irányuló erőfeszítést nevezték. És ezzel visszakanyarodtunk az elemzéshez. Minden modell terv, de nem minden terv modell. Csak az, amely megfelel a roppantul tudatos elemzés során betartott kritériumoknak. A szavak nem játékok!

3.3 SZEMLÉLETI ÖSSZEFOGLALÓ

Az információs rendszer eltérő jellegű tényezőket felölelő, roppant bonyolult inhomogén lényeg. Ismeretek, események, tevékenységek, erőforrások stb. stb. alkotják. Ezért teljesen természetes és érthető, hogy az emberek a szerepüknek megfelelően más-más szemszögből közelítenek az IR-hez. A végső-felhasználót nem érdekli, hogy milyen számítógéppel és hogyan készül pl. a csekkje, csak az legyen korrekt. Az alkalmazói-felhasználó e tényezők iránt már nem közömbös, mert a 9 karakteres vagy a 24 karakteres azonosító - hmm! - közelebbről érinti. A hálózattervező madzagokat, modemeket, műholdakat lát. A szoftveres érthető módon a programcsomag-változatokba van belezúgva. Nincs mindebben semmi baj - az lenne az egészségtelen, ha nem így lenne.

Az informatikusnak viszont az a feladata, hogy ezt a több maréknyi bolhát egy kézben tartsa. Nem lehet elfogult. Mindenkinek a legjobbat kell adnia mégpedig úgy, hogy a tág értelemben vett felhasználó szinte ne is lássa - mert nem tartozik rá, őt csak terhelné - a rendszer elrendezésének a mikéntjét. Ugyan ki a fene fog törődni a kocsija fékrendszerének a lelkével addig, ameddig az nem csikorog? Csak a rosszul megtervezett információs rendszerről rí le, hogy valami baj van.

Az eszköz, a szervezet, a területi megoszlás, a tevékenységek megtevéseztetik a fejlesztőt. Mindezekre a dolgokra ügyelnie kell, ám helyesen csak akkor jár el, ha az IR-t az ismereti elv szerint tagoltan szemléli. Mindez nem elegendő. A cél, a tartalom, a statikus elrendezés és a dinamikus változás, a fontossági sorrend - hogy csak pár gondolatra térjünk vissza - mind-mind helyet kell, hogy kapjon az informatikus agyában. Ugye, hogy nem is olyan könnyű feladat mindez?

Holott még szinte el sem kezdtük a mondanivalót. Az informatikusnak azt is látnia kell, hogy minden IR a tartalmától, összetettségétől, a földrajzi helyétől, az alkalmazó embertől függetlenül vetületek és szintek szerint strukturált. Az IR az elgondolt (absztrakt) viszonylag stabil adat-, az ugyancsak kitalált viszonylag dinamikus feldolgozás- és a nagyon is konkrét környezetvetület együttese. Úgy, hogy az elképzelt adat- és feldolgozás-tényezőket a feladat, a megoldási mód és az eszköz fogalmi-logikai-fizikai színhármásában kell megfogalmazni.

Az információs rendszer maga is nagyon bonyolult. Ezt a tényt azért kell újra meg újra hangsúlyoznunk, mert a számítógépekkel kapcsolatba kerülő emberek abban a tévhitben ringatják magukat, hogy a számítógép-alkalmazás egyszerű. Nem az. Abban a pillanatban, hogy barátom az Excel-ben készítette el a legelső számláját (technikailag könnyedén), már el is vágta maga előtt az utat. Mert nem képes együtt (!) látni az X-valaki számára készült A és B számlát. Már csak azért sem, mert nincs dokumentációja. Nem volt eredeti elképzelése (plan) a számlák kezeléséről, és mindegyiket ad-hoc módon (design) alakította ki.

A tudatos informatikus ismeri a rendszerek életciklusát (születés, éltetés és állandó változás). Azt is tudja, hogy mindhárom szakaszban az átfogó képet, a részletes rajzot és az ütemezést összhangba kell hozni egymással. Nem létezik egyetlen terv: a megoldás átfogó módja (plan), a megvalósítás részlete (design) és a tényleges végrehajtás (schedule) elválaszthatatlan egymástól.

Az IR-ek sikerének a titka a jó dokumentáció, amely többszörös célt szolgál. Éppen ezért kell ügyelni annak célszerű szerkezetére és tartalmára. Nem a régi értelemben vett átfogó (dokumentáció) és sajátos részlet (specifikáció) szerint, hanem az IR vetületek és szintek szerinti általános szerkezetének megfelelően kell elkészíteni a rendszer rajzos, ábrás, szöveges leírását.

Az információs rendszert modellezni kell. Nem az első, roppantul hasznosnak tűnő megoldás a modell. Hanem a 'formatervezett', az optimum-kritériumoknak megfelelően átgondolt terv az. Az informatikust mindig csalódások fogják érni, amikor a létező legjobb, valóban mérlegelt gondolataival áll elő. Azért, mert ő látja az alkalmazási- és az eszközkörnyezet hiányosságait. A valós- és az eszköz-rendszerbe viszont nem szólhat bele. 'Szervező' titulusa dacára nem őt, hanem másokat fognak megbízni a szervesen az IR-be tartozó, az általa megfogalmazott dolgokkal összefüggő tényezők (pl. bizonylat, kód) kialakításával. Mindez azért van így, mert a következő fejezetben ismertetett módszer lényegét nem ismerik.

ELLENŐRZŐ KÉRDÉSEK - 3

Az olvasó három kérdésben vizsgálja meg a saját szemléletét. Vajon gondolt-e arra, hogy hányféle szempontból kellene vizsgálnia leendő rendszerét, mielőtt az alkalmazásba belefogna? Vajon eszébe jutott-e, hogy az adat, a feldolgozás és a környezet három vetület, amely közül kettőnek három absztrakciós szintje van? Végül: eligazodik-e a fejlesztéshez kapcsolódó terminológiai káoszban?

- 301 Állítson fel egy sorrendet! 1-5 közötti számokkal mérlegelje, hogy az alábbi szempontok közül melyik a fontosabb. Két tényező azonos értéket is kaphat. Vigyázat, a kérdés becsapós! Szóval mi a lényegesebb: a cél, a tartalom, a felépítés, a működés vagy az irányítás?
- 302 A T 1.1 történetben a rendszertagolás egyik helytelen módját mutattuk be. Miért nem célszerű az eszközök és a földrajzi hely szerinti szemlélet? A válaszhoz egy kis támaszt adunk. Tegyük fel, hogy Önnek az a feladata, hogy egy cikksorozatot készítsen a hazai kollégák által elkészített és Önhöz eltérő módokon eljuttatott anyagokból. Mondja el, hogy Ön mit tekintene ebben az esetben 'rendszernek'!
- 303 Az értékesítési részleget tavaly két csoportra bontották, idén pedig újra egyesítették ezeket az egységeket. Mondja el saját szavaival, hogy ezek a változások miként és milyen bajokat okozhatnak az IR-ben, illetve mikor nem jelent a szervezeti változtatás 'informatikai meg-rázkódtatást'!
- 304 Az információs rendszerek egyik legfontosabb kritériuma a stabilitás. Az Ön feladata kettős. Először 1-3 közötti számmal rangsorolja az adat-, a feldolgozás- és a környezet-vetületet a stabilitás szempontjából. Azután említse meg szóban azt a két elvet, amely a stabilitás kulcsa.
- 305 Ön szerint melyik szintre tartozik a 'duplapontos' numerikus adatoknak az ábrázolása? A fogalmira (1), a logikaira (2) vagy a fizikaira (3)?

- 306 Ön szerint melyik szintre tartozik az a részlet, hogy a 'Megrendelés-kele' adatot dátum-típusúnak határozzák meg? A fogalmira (1), a logikaira (2) vagy a fizikaira (3)? Több válasz is elfogadható!
- 307 A minősítő jelzők olykor megtévesztőek. Egy valakinek lehet több felső-fokú végzettsége és több korábbi munkahelye. Lát-e különbséget abban, hogy a személy adatsorában megjelenik a 'legmagasabb végzettség' és az 'előző munkahely' adat? Érveljen! Egyik megoldás sem elfogadható? Mindkettő az? Netán csak az egyik?
- 308 Milyen elvet sért az, ha a barátom Excel-ben kezeli a számláit, Word-ben a vevőit? Barátommal együtt írunk valamit. Ő angol Word 6-ot, én magyar Word 2-öt használlok. Lesznek-e gondjaink?
- 309 „Hárman egy csónakban...” eldöntöttük, hogy volt gimnazista társainkról nyilvántartást vezetünk. Név, lakcím, telefon stb. Mondja el, hogy milyen szinten (fogalmi, logikai, fizikai) kellene egyeztetnünk a gondolatainkat!
- 310 Az előző feladatban lát-e feldolgozási elvi szinteket is? Mit? Érveljen!
- 311 Ön újjá kívánja varázsolni a hálósobáját. Mondja el saját szavaival, hogy hányféle tervben gondolkodik!
- 312 Ön kedvelné-e, ha az előző példa munkálatai kapcsán külön-külön kellene egyeztetnie a parkettással, a festővel stb. Sajnos, az információs rendszer ma ezt feltételezi. Azért, mert nincs... Ki nincs?
- 313 Mi a különbség a terv és a 'design' között? A 311-es feladatra emlékezve: Ön mit várna el a mestertől? Ön dirigálna, vagy hagyná érvényesülni?
- 314 Ön családos ember. Előrelátó. Mit és miért fog megkövetelni a hálósobát újrarácsoló mesterektől? Segítségként: mire szolgál a dokumentáció?
- 315 Mit gondolna arról, ha a lakásfelújító mester helyett más szervezné meg a munkálatokat, az anyagszállítást, a - remélhető - takarítást, miegymást?
- 316 Végül arra kérjük, hogy foglalja össze: mi köze van a 311-315 kérdések mögötti tartalomnak az információs rendszerhez? Van? Nincs? Miért nem? Miért igen?

4. FEJLESZTÉSI MÓDSZER

Az élet minden területén az alkotás sikere a megfelelő **módszer** megtalálásától és annak tudatos alkalmazásától függ. Egyáltalán nem véletlen az, hogy az igazi nagy nevelők, a régi neves tanárok arra törekedtek, hogy elsősorban a módszert sajátíttassák el tanítványaikkal, és csak másodsorban oktattak megoldásokat ill. konkrét tényeket. A történész nem a dátumok, az események ismeretétől válik jó szakemberré (persze az is nélkülözhetetlen a számára), hanem attól, hogy a kellő módszerrel közelíti meg vizsgálata tárgyát. Volt középiskolás nevelőnk, Pogány tanárúr, el sem fogadta azt a matekdolgozatot, amelynek ugyan tökéletesen jó volt a végeredménye, de a példamegoldás felépítése nem felelt meg az általa oly precízen ismertetett és elvárt módszernek. Akkor talán egyesek nehezteltek ezért rá, viszont ma már mindannyian tudjuk, hogy helyesen járt el.

Az informatikai fejlesztések sikeres vagy sikertelen volta is mindenek előtt az alkalmazott - vagy éppen a nem használt - módszernek a függvénye. Senkit sem akarunk megbántani ezzel a kitételrel, de mi egyáltalán nem hiszünk az ösztönös - értsd: módszer nélkül dolgozó - úgymond 'számítástechnikai zsenikben'. Az IR rengeteg ember hatékony együttműködését feltételezi. Ebben és más értelemben véve is túl összetett dolog ahhoz, hogy módszer nélkül lehessen megközelíteni.

A mai magyar fejlesztésekre nem jellemző a végeredmény átütő sikere. Ennek a ténynek számos oka van. Először is sok szervezetben egyáltalán nem ismerik a másutt már széleskörűen elterjedt információs rendszer**fejlesztési módszerek** [information systems development method] egyikét sem. Másodsorban tipikus az a jelenség is, hogy a módszer fontosságát elismerik, alkalmazását kívánatosnak is tartják, de ilyen-olyan okokra hivatkozva használata mégis csorbát szenved. (Az úgymond időhiányt szokták érvként emlegetni.) Az utóbbi időkben a módszerek a figyelem középpontjába kerültek, ám azok szemlélete körül igen komoly bajok mutatkoznak. Ez ad alapot arra, hogy maguknak a módszereknek a lényegét egy kicsit körüljárjuk.

Ennek a fejezetnek az a célja, hogy tisztázza az információs rendszer-fejlesztési módszerek lényegét és bemutassa e módszerek mint sajátos rendszerek alkotó-elemeit.

A módszerek maguk is rendszerek, mert tényezők tudatosan kialakított, azaz szervezett együttesei. Az alábbiakban arra törekszünk, hogy az IR fejlesztési módszerét ebben a rendszer-szemléletben ismertessük.

4.1 SZEMLÉLETI ZAVAROK

A fejlesztési ún. módszertanokkal foglalkozó szakkönyvekben éppen úgy, mint az IR fejlesztését támogató eszközök kézikönyveiben számunkra zavarónak ható módon összekeveredik két dolog. Mielőtt erre a lényegre rátérnénk, tisztáznunk kell egy alapvető kérdést.

A **módszer** [method] és a **módszertan** [methodology] nem azonos dolgok. Ez a két lényeg a specifikus és a generikus viszonyában áll egymással. A módszer az adott tárgy (esetünkben: az IR) megvalósítását elősegítő technikák és megoldási módok szervezett, konkrét együttese. Ezzel szemben a módszertan 'a módszerek módszere'. Azzal foglalkozik az absztrakció magasabb

síkján, hogy miképpen épülnek fel a konkrét módszerek, mint olyan dolgok, amelyek saját maguk is rendszerként szemlélendők. A specifikusban a *módszer-alkalmazó* érintett, a generikusban a *módszer-alkotó* érdekelt. Az iskolákban módszert tanítanak, a szervezetekben módszert alkalmaznak - és nem módszertant. Ennek a könyvnek ez a fejezete viszont kimondottan a módszertannal foglalkozik, hiszen magát az IR-fejlesztési módszert vizsgálja rendszerként.

Most pedig térjünk vissza a szemléleti zavarokra. A fejlesztési módszereket leíró kézikönyvekben egymástól elválaszthatatlanul összekeveredik a 'mit' és a 'hogyan' kérdése. Még pontosabban szólva: kétféle módszer tényezői kerülnek összevegyítésre. Amikor például egy fejlesztési számítógépes segédeszköz útmutatója az adott fejlesztési fázis leszállítandó dolgait [deliverables] sorolja fel, akkor azok listájában éppen úgy megtalálható a szakasz *érdemi* produktuma (vö. tervtermék), mint a megalkotás tényét és körülményeit leíró *adminisztratív* jellegű papírok garmadája.

Félreértés ne essék: erre az adminisztrációra is szükség van. Ámde egyrészt az a baj, hogy így a tervtermék kap sokkal kevesebb figyelmet. Az érdemi vizsgálat helyett a 'kipipálásra' helyeződik a hangsúly: elkészült - könnyebbülünk meg - hiszen átadtuk/átvettük. Másrészt az a gond, hogy így maga a tervezési módszer szétszóródik a könyvekben. Belső logikája felismerhetetlenné válik, hiszen azt unos-untalan megtörik a (szükséges) papírtologatásra irányuló eligazítások. (Itt emlékeztetnünk kell az olvasót arra, hogy egyes mai széleskörűen alkalmazott fejlesztési módszerek olyan előbbiekből alakultak ki, amelyek egykor kizárólag a **projekt-menedzsment** feladatát támogatták, magát a tervezést nem!)

A fejlesztés minden egyes résztvevőjének tisztában kellene lennie azzal, hogy minden fejlesztésben két párhuzamos fonal húzódik végig. Tárgyától teljesen függetlenül minden alkotásban kétféle módszer kíséri egymást. Az egyik szál az *irányítási* ('hogyan'), a másik a tárgy szerinti *érdemi* ('mit'). Természetesen egy konkrét módszerben a két fonal elválaszthatatlan, ám hiba lenne, ha azokat az absztrakt módszertanban nem különítenénk el egymástól. (Minderre nézve lásd az előző fejezet végén lévő feladatokat és megoldásokat.)

Ezen a ponton az olvasó bizonyára a szerző szemére veti, hogy már megint túlságosan elméletieskedik. Ugyan mi értelme van az ilyesfajta boncolgatásnak: megvesszük az X módszert illetve eszközt - és kész. Nos, éppen ettől kell félni.

T 4.1 1993-ban egy neves pénzintézet 100.000 dollárért megvett egy CASE eszközt. A mai napig nem használta érdemi módon.

Vajon miért nem? Hanyagságból? Tudatlanságból? Ne feltételezzünk ilyesmit: egészen másutt kell a bajok gyökere után kutatni. Ugyanis tüzetesebb vizsgálat után kiderül, hogy a mai módszerek - voltaképpen nem is azok! A számítógépes eszközök bevetése, a technológiai korszerűsítés önmagában nem eredményezhet jobb rendszereket. Szemléleti váltásra lenne szükség, márpedig a kézikönyvek szemléletet nem tanítanak. Elmondják, hogy miképpen kell kezelni a gombokat a gépen; melyek a végrehajtható funkciók, programok; egy igen picit segítenek a végtelen rossz terv elkerülésében; néha javaslatot tesznek a következő lépésre; ennyi. Ennyi és nem több. Ez még nem módszer, mert ez még nem a rendszerbe foglalt szemlélet és megoldási mód szervezett együttese. Így hát pontosan ez az oka annak, hogy a drága pénzen megvett eszközt nem tudjuk igazán használni.

T 4.2 Nemrég került a kezembe egy kereskedelmi vállalat rendszerének az adatbázisisterve. DELPHI-vel - korszerűnek tartott eszközzel - készítették. Mondani sem kell, hogy meglehetősen rossz. Azért, mert a módszer nem állt az eszköz színvonalán.

Azért ne tessék elméletinek tartani az itteni gondolatokat, merthogy nagyon is a gyakorlatról van szó. A kézikönyvek semmilyen útmutatást sem adnak három roppantul fontos kérdésre. Az első az, hogy általában *miképpen épül fel* minden egyes információs rendszer, vagyis mi minddel kell a gyakorlatban törődni. A második az, hogy - furcsa szóhasználat - miképpen kell praktikus-*módszeresen* használni az alkalmazott módszert. A harmadik az, hogy nem tájékoztatnak a módszerek *általános lényegéről*. Ezért a gyakorló fejlesztő természetesen még azt sem tudja eldönteni, hogy a kérdéses módszer adott résztechnikája miképpen alkalmazandó/alkalmazható.

Térjünk vissza egy pillanatra a második momentumhoz. A korszerű fejlesztési módszerek meglehetősen gazdag megoldási-technikai rész módszer-készlettel rendelkeznek. Olyan módszertani eszköztárral, aminek a töredékét sem tudja érdemben kihasználni a fejlesztő. Ha a módszerben foglalt minden egyes tevékenységet végrehajtana és elkészítené az összes leszállítandó dolgot, akkor az felérne egy fejlesztési 'lassító-sztrájjal'. Ezért evidens, hogy a gyakorló fejlesztőnek mérlegelnie kell: ezt igen, azt nem. Azonban e gyakorlati szelekció nem lehet sikeres, ha a fejlesztő nincs tisztában a módszerelmélettel.

Ugyancsak a módszertan ismerete szükséges ahhoz, hogy a fejlesztés minden résztvevője át tudja látni: a konkrétan alkalmazandó módszer miben gyöngye és miben erős. Ne feledjük, hogy a módszer - és az azt támogató eszköz is - emberi termék, ezért nem feltétlenül tökéletes. Az X módszer/eszköz gyengesége miatti réseket nem szégyen, sőt, nagyon okos dolog az Y módszer/eszköz technikáival betömni. Persze ehhez tudni kell, hogy hol is vannak azok a bizonyos korlátok.

Szumma szummárum: egyetlen konkrét módszert sem lehet jól felhasználni a kellő absztrakt módszertani ismeretek hiányában. Ezért az alábbiakban kísérletet teszünk arra, hogy a módszert módszeresen, azaz rendszerként vizsgáljuk. Ebből a célból összeállítottunk egy módszertani kelléktárat. Felsoroljuk és részletesen is kifejti azt, hogy a módszer-alkotóknak és -alkalmazóknak mi mindenre kell ügyelniük. A főbb tételek, a további alpontok címszavai a következők:

- alapkoncepció
- struktúra és terminológia
- megfogalmazási mód
- kritériumrendszer
- tervezési algoritmus
- dokumentáció-kezelés
- technikai támogatás.

4.2 ALAPKONCEPCIÓ

Az információs rendszerek fejlesztésének, működtetésének és működésének az első és leg-alapvetőbb feltétele az **egységes alapkoncepció**. E kellék kapcsán két dolgot kell megmagyaráznunk. Az egyik az, hogy mit jelent maga a fogalom. A másik az, hogy mit kell érteni az 'egységes' jelzőn.

A hétköznapi nyelvben is igen sűrűn alkalmazzuk a koncepció fordulatot. Ez a szó eredetileg (érdemes figyelni a fokozatokra) a beszélgetés, a megbeszélés, a tervezés, a fejlesztés illetve a megalkotás aktuális tárgyára vonatkozó komolyan megfontolt elképzelést jelent. A koncepció nem pusztán vágy és nem csak lazán összehordott gondolatok kusza együttese. Ugyanakkor nem tévesztendő össze a pontosan kidolgozott tervvel sem. Éppen ezért helytelenül használjuk magát a

fogalmat, ha az „Az a koncepció, hogy ...” kezdetű kitétel után vagy tárgyunk lelkes, de összefüggéstelen taglalásába fogunk, vagy - éppen ellenkezőleg - egy részletesen összeállított tervet teregetünk ki.

Mi tehát a koncepció? Valójában nem kevesebb és nem több, mint komolyan megalapozott **szemléletmód**. Azon gondolatainknak a magva, amelyek szerint a szóbanforgó tárgyat - legyen szó bármiről is - megközelítjük. Ez a felfogásmód nem változik, nem változhat attól, hogy még csak a beszélgetés, a megbeszélés, vagy már a fejlesztés, a megalkotás fázisában vagyunk. Hiszen roppant nagy baj is lenne, ha más koncepcióban történne a megvalósítás, mint amilyen nézetben született maga az elgondolás. Ehhez képest igen döbbenetes, hogy információs rendszereink fejlesztése során hányszor történik ‘koncepcióváltás’.

Az előbbi kitétel nem jelenti azt, hogy a koncepció sohasem változhat. Az IR roppantul összetett dolog: egyszerre és előre senki sem képes azt átlátni. Ezért a rá vonatkozó elképzelések halmaza a fejlesztés előrehaladásával módosulhat. Ez azonban nem váltást, hanem csak pontosítást jelent. Az is bizonyos, hogy maga az informatikai tudomány és technika is fejlődik. Ezért az informatikusnak az új módszerek és eszközök ill. az ezek által generált új igények garmadájjával kell szinte nap mint nap szembenéznie. Ezek az újszerű dolgok bizonyára finomítani fogják az informatikai **részkoncepcióinkat**. Teljesen evidens, hogy a mikrogépes és a nagygépes, a helyi és az elosztott feldolgozású, a pusztán nyilvántartási és a gazdálkodási célú információs rendszert más-más szemléletmóddal kell, hogy megközelítsük.

Itt azonban nem a részletekről óhajtunk beszélni, hanem az **alapkoncepcióról**. Az információs rendszer igen összetett dolog. Ez az egyetlen lényeg száz és ezer megoldási módot kínál. Megvalósítása történhet így is, meg úgy is; ilyen, meg olyan alkonceptióban. Ugyan ki is tudná felsorolni, hogy hányféle variációban lehetne létrehozni - mondjuk - az anyaggazdálkodási információs rendszert? E hihetetlen változatosság dacára minden információs rendszerben van egy közös lényeg. Bár tautológiának hangzik, mégis kimondjuk: ez a közös lényeg az, hogy információs rendszerről - és nem autóról, házról stb. - van szó.

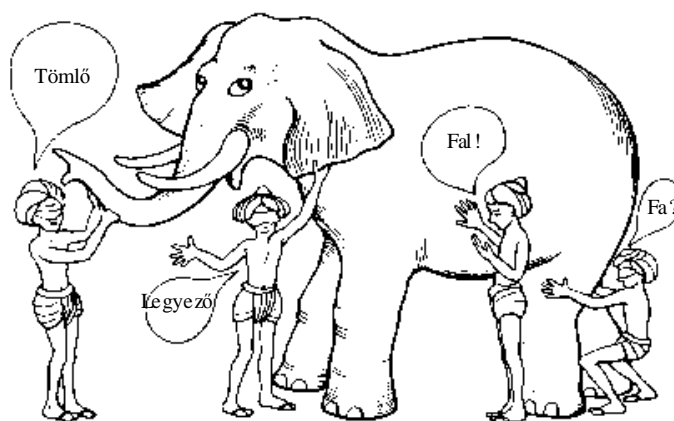
D 4.1 Az alapkoncepció azon szilárd és megalapozott elképzelések együttese, amelyeknek megfelelően szemléljük - konkrétumtól függetlenül - tevékenységünk alapvető tárgyát.

A definíció értelmében az IR esetében az alapkoncepció azon megalapozott és szilárd elképzelések együttese, amelyeknek megfelelően szemléljük általában az információs rendszereket. Teljesen függetlenül attól, hogy az anyaggazdálkodási vagy pedig a humánpolitikai rendszerről van-e szó.

Vajon miért ennyire fontos az alapkoncepció taglalása? Hiszen - látszólag - itt is csak egy elméleti gondolatmenetről van szó. Látszólag. Mert a gyakorlatban és valójában az IR fejlesztése kapcsán a legeslegelső megválaszolandó kérdés az, hogy: „**Minek tekintem én az információs rendszert?**”

Most képzelje el az olvasó, hogy az építész-mérnöknek építendő házanként változik az alapkoncepciója. Nem a részkonceptiói (ezt így, az úgy), hanem hol ezen, hol azon a módon fogalmazza meg pl. a lakóház lényegét. Nyaralót kérnek tőle, ő pedig társasházat tervez - vagy megfordítva. Netán garázst. Nevetségesen hangzik mindez? Inkább szomorúan - ha az informatikára gondolunk. Ha tízezer mindennapi embert - laikust - megkérdezünk, hogy minek tekinti ő személyesen az IR-t, tízezerféle választ fogunk kapni. Ez a kisebbik baj. A nagyobbik gond az, hogy az X fejlesztő - a profi - hétfőn más választ ad erre a kérdésre, mint pl. kedden. Vagyis - nincs alapkoncepciója. Ebből fakad a sok ‘koncepcióváltás’; a közismert eszközörületre visszavezethető állandó változtatás; a megbízhatóság, a stabilitás, a mérnöki nyugalom hiánya.

Az informatikustól rendszert (ház) kérnek, ő pedig csak valami részt (garázs) épít. Rádásul úgy, hogy az nem illeszkedik az egészbe. Ezzel a jelenséggel nap mint nap találkozhatunk. Mindez szemléleti hiányosságokra vezethető vissza. A fejlesztők az IR-t nem a D 2.1 definícióban adott tényezők szervezett együttese módján, hanem számítógépes programok egyvelegeként értelmezik. Vagyis az az alapkoncepciójuk, hogy az IR = számítógépes programhalmaz. Nem tartozik bele az ember, a szabvány, a bizonylat, a kézi eljárások tömkelege, a váratlan és kivételes - manuálisan intézendő - esemény stb. Ezek után nem kell csodálkozni azon, hogy a rossz alapkoncepcióból - rossz információs rendszer születik.



4.1 ábra: 'Alapkoncepciók'

Ezt a pontot az 'egységes' jelző magyarázatával zárjuk. Hiába tekintem én és kolléganőm - Marika - az IR-t annak, ami. Hiába helyes a mi alapkoncepciónk, ha az ugyanazon a rendszeren dolgozó többiek szemléletmódja más, korlátos. Gondolja meg az olvasó, hogy von Braun mennyire jutott volna a rakétáival, ha egyik-másik munkatársának más lett volna az alapkoncepciója a rakétákról, mint a sajátja. Ezért a fenti kiemelt kérdést módosítani kell. Úgy kell megfogalmazni, hogy: „**Minek tekintjük mi az információs rendszert?**” Mikrogépes programok együttesének? Windows-alkalmazásnak?

Az alapkoncepció nem más, mint az IR értelmezett meghatározása. Ez minden fejlesztési módszer alapja, szegletköve. Ezernyi gyakorlati tapasztalat mutatja, hogy ebben a kőben botlunk meg, erre esünk rá vagy ez esik ránk - és szétfűz bennünket.

És most tessék jól figyelni! E könyv szerzője már ezernyi cégnél megfordult, de egyetlen egy (!) helyen sem talált olyan szabályzatot, dokumentációt, netán egyéb eligazítást, amelyben szerepelt volna az IR meghatározása! Akár rosszul, akár jól. Ezután a „**Minek tekintjük mi ...?**” kérdés felvetése már felesleges. Mert nincs 'mi' és nem 'tekintjük'. Az IR ma ez, holnap az. Hol ház, hol garázs, hol háznak mondott garázs, hol fordítva.

Az alapkoncepció a módszer olyan sarokköve, amit a módszeralkotónak kell lefektetnie és a módszer-alkalmazónak mindenkor figyelembe kell vennie.

4.3 STRUKTÚRA ÉS TERMINOLÓGIA

Az információs rendszer inhomogén tényezőkből szervezett együttes. Felöleli a viszonylag stabil adatvetületet, a sokkal dinamikusabban változó feldolgozási vetületet és a környezeti vetületet, amiben az alkalmazási- és az eszközrendszer tényezői - a felhasználók és az erőforrások - szerepelnek. Mindehhez járul az a tény, hogy az ismeretek és az események/tevékenységek több - fogalmi, logikai és fizikai - szinten foghatók meg.

Mindezzel azonban még messze nem mutattunk rá az információs rendszerek belső szerkezetének az összetettségére. A vetületek - ráadásul mindig a megadott szinten - egymáshoz kapcsolódnak. Az adatok a feldolgozásokhoz, mindkettő a felhasználókhoz kötődik. A vetületeken belül is vannak összefüggések. Az adat másik ismerethez, a feldolgozás az őt megelőző/követő másik tevékenységhez rendelt. Az pedig közzismert, hogy a felhasználók éppen úgy nem függetlenek egymástól, mint ahogyan az eszközökben is 'rendszert' kell találni.

Az információs rendszer tényezői egy roppant bonyolult hálót alkotnak. A fejlesztőnek két dolga van. Képletesen szólva egyrészt 'hálót kell szőnie' és másrészt 'be kell hálóznia'. Az adat és a feldolgozás másodlagos valóság. E dolgokat és összefüggéseiket ki kell találni (hálósövés). Majd mindezeket és az IR többi tényezőit össze kell rendezni (behálózás). De vajon miképpen tehetné meg ezt a fejlesztő egy-egy konkrét feladatnál, ha nem volna általános képe a hálóról magáról?

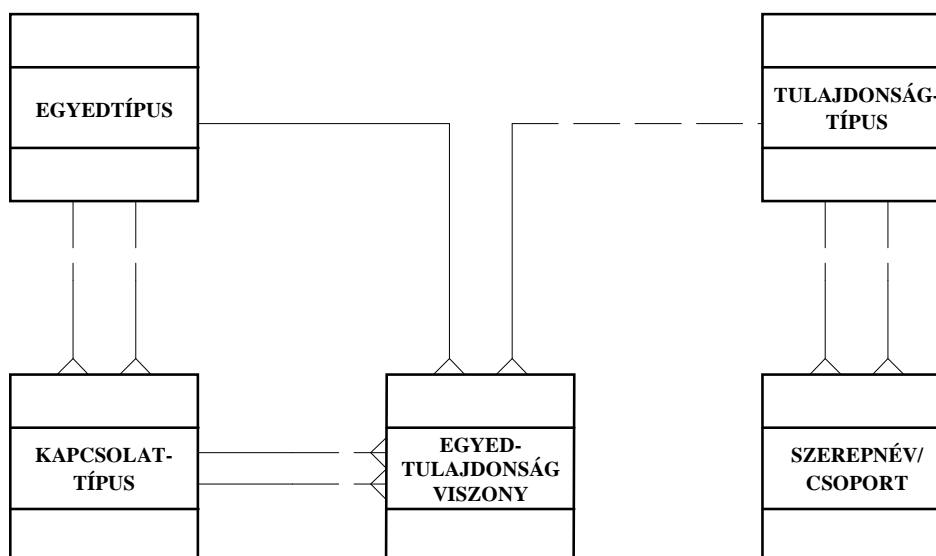
D 4.2 A rendszerstruktúra azon szilárd és megalapozott elképzelések együttese, amelyeknek megfelelően szemléljük - konkrétumtól függetlenül - tárgyunk tényezőit és azok összefüggéseit.

A módszert tudatosan alkalmazó fejlesztőben és a fejlesztés résztvevőiben élnie kellene egy közös strukturális képnek arról, hogy az IR tényezői általában - a konkrét rendszertől függetlenül - miképpen kapcsolódnak egymáshoz. Mindez nem ördögösség. Nem hisszük, hogy akadna olyan olvasónk, aki ne tudná azt elmondani, hogy miképpen épül fel a lakása illetve a civilizált lakás általában. Biztosan remekül felvázolja a helyiségek kapcsolatait, a szobák és funkcióik viszonyát, a használati módot, a bennük lévő eszközöket (vetületek). Mindezt gond nélkül megteszi fogalmi, logikai és fizikai szinten is. A fürdőszoba arra való, hogy ... (fogalmi). Mi nem terveztünk bele bidét, mert ... (logikai). A kád jobboldalt áll, mert ... (fizikai).

Mi tehát a gond? Az, hogy az IR fejlesztésének a résztvevői messze nem látják annyira világosan az IR általános szerkezetét, mint például a lakásukét.

Minden konkrét információs rendszernek van egy konkrét terve, vagyis egy konkrét *modellje*, mint ahogyan ez igaz minden konkrét lakásra nézve is. Ám a lakásoknak van egy absztrakt, általános modellje is - a lakás úgy szokott kinézni és úgy szokott 'funkcionálni', hogy ... Ennek párhuzamjára elmondhatjuk azt is, hogy az információs rendszer úgy szokott kinézni és úgy szokott funkcionálni... A 'szokott' kitétel a specifikusban az általánosat, a kivételben a törvényszerűt jelzi. Itt a konkrét képnek az absztrakt nézetéről van szó. Ezt az informatikában a modellek modelljének, vagyis **metamodell**nek nevezik. A konkrét lakás terve specifikus minta. A lakások mindenkor célszerű elrendezése generikus minta - azaz metamodell.

Az általános minta a hely és az idő függvényében - e két faktor is összefügg - fokozatosan változik (evolúció). Nem úgy néz ki egy magyar lakás, mint pl. egy indiai és a mai magyar lakás sem azonos a századelejével. A fejlődést ezért nem hagyhatjuk figyelmen kívül: metamodellünk - az IR szerkezetéről alkotott mai elképzelésünk - időről-időre módosulni fog. Az informatika - minden ellenkező híresztelés dacára - ma már érett tudomány. Ezért azzal nem kell számolni, hogy forradalmi változások következnek be az IR-struktúrát illetően (revolúció).



4.2 ábra: Részlet az IR metamodelljéből

A 4.2 ábra az információs rendszer adatvetületének az általános elvi modelljét szemlélteti. A rajta lévő fogalmakat majd a következő fejezetben ismertetjük. Itt nem a részletek megértése a lényeg. Arra kívánunk utalni, hogy a 4.2 ábrához hasonló képnek kellene élnie a fejlesztési résztvevők fejében, ám a módszerek kézikönyveiben hasonló ábrákkal a legritkább esetben lehet csak találkozni. Ami azért is figyelemre méltó, mert az ábra szerinti kép örök érvényű. Noha bővíülhet újabb elemekkel, az általa tükrözött összefüggések nem fognak megváltozni.

Érdekes módon éppen a számítástechnikai profik azok, akik nehezen képesek a speciálisban az általánosot megragadni. Ők mindig a legújabb technika szerint gondolkodnak. Ma relációsan, holnap objektum-orientáltan. Bizonyára néhány olvasó számára ismeretlen ez a két fogalom. Ez most ne zavarja. Az ismeretek elrendezésének és kezelésének két, részben eltérő technikai módjáról van szó. Itt a hangsúly a *‘technikai’* jelzőn van. Mert árulja el az olvasó, hogy szerinte más lesz-e a számlázási ismeretek lényege attól, hogy a számítógépen azokat milyen technika szerint rendezik el? Ugye, hogy nem? Akkor pedig hol marad az adatok elrendezésének és kezelésének a technikától független *tartalmi* módja?

Nos, éppen erre való a metamodell. Természetesen az informatikust is érdekli a napi aktualitászú technika. A második megközelítésben. Viszont az elsőben ő a valódi tartalommal törődik, amit a metamodell szerint rendez el. A megrendelés, a számla, a vevő, a cikk akkor is az és akkor is úgy függ össze egymással, ha az ismereteket rekordokban, relációkban vagy objektumokban tárolják és kezelik. Az informatikus az általános struktúrához próbálja igazítani a speciálisat.

A fenti gondolatok nem az újabb termékek és elképzelések ellen szólnak. A házak építésében is mindig újabb és újabb technológiákat alkalmaznak. Csacsi és elmaradott az, aki nem figyel az új áramlatokra. Mi csak két dologra akarjuk felhívni a figyelmet. Az egyik az, hogy a ház lényege nem a technológián múlik. A ház az ház. Azt kell először érteni, utána a technológiát. A másik az, hogy az informatikában a változások gyorsabbak, mint az építészetben. Sokszor nem is tartósak. Jön egy új fuvallat, amiről azt hiszik egyesek, hogy orkán lesz, közben pedig csak gyenge szellő. Ezért a józan informatikus nem dől be ingó nádként minden új áramlatnak. Az ismeretek kezelésének a módja időközben rengeteget fejlődött, ám a szerző bátran állíthatja, hogy az IR általános elvi struktúrája az elmúlt 25 év alatt alig változott.

Ennek ellenére az informatikát az értetlenség és a kommunikáció-képtelenség uralja. Ennek a jelenségnek az oka kettős. A divathullámokat az jellemzi, hogy új jelszavakat röppentenek fel és

új terminológiát vezetnek be. A tapasztalatlan informatikus sokszor *nem tudja eldönteni*, hogy valóban új dologról van-e szó, vagy csak egy régi árut reklámoznak az új jelszóval. Például amióta divatosá váltak az adatbázisok, azóta a régi, más célú programok tucatjaira ragasztották rá az előkelő ‘adatbáziskezelő’ címkét, holott semmi közük sincs az adatbázishoz. Ha tehát a hozzáértő a képzetlennel beszél és az ‘adatbázis’ szót használja, akkor a kettejük kommunikációja a ‘jó napot, sógor’-ra hasonlít. A másik jelenség az, hogy az új szakszavak mögötti tartalmat egyesek nem pontosan ismerik. Ebből következően *a terminológiát rosszul használják*. Ami persze szintén kölcsönös értetlenségre vezet.

T 4.3 1994-ben saját vállalatunk adatbázisának a tervét vizsgáltam. Sehogyan sem értettem, mert a tervező az ‘alaptábla’ és a ‘reláció-tábla’ fogalmakat használta. Hmm, mik lehetnek ezek?

Rövid magyarázat következik. Azt nem fogjuk elmondani, hogy mi is az a relációs adatbázis, mert arra nincs terünk. A relációról annyit kell csak tudni, hogy az az egy jelenségcsoportra (pl. gépkocsik, személyek stb.) vonatkozó ismeretek tárolásának és a kezelésének az egysége. Nem más, mint speciális korlátoknak megfelelő adatállomány, amit kétdimenziós elrendezése miatt táblának is neveznek. (A tábla sorai pl. egy-egy gépkocsi ismereteit írják le úgy, hogy az oszlopai a jellemzők. Rendszám, tulajdonos, évjárat stb.) Vagyis a tábla nem más, mint a reláció [relation] szinonimája. A dolgok közötti viszonyokat - pl. a személy a gépkocsi tulajdonosa - kapcsolattal [relationship] fejezzük ki. Történetünkben a fejlesztők nem tudták, hogy mi a ‘relation’ és a ‘relationship’. A ‘relációtábla’ ugyanis azt jelenti, hogy ‘táblatábla’, ami csacsóság. A tervező a viszonyra kívánt utalni, így a ‘kapcsolati tábla’ fogalmat kellett volna használnia.

Itt érkeztünk el az alcím második felének a magyarázatához. Az információs rendszer tényezőit és azok viszonyait az ember nevekkel illeti, hogy beszélni tudjon róluk. Most nem az a lényeg, hogy elmondjuk a reláció és a kapcsolat definícióját, vagy akárcsak elkezdjük magyarázni e dolgokat. A fontos az, hogy rávilágítsunk: az informatikának mint szakmának is van saját *szakszargonja*. Ám ugyanakkor arról sem szabad elfeledkezni, hogy az IR-nek szerves része az ember is, mint felhasználó, ezért ezzel a műnyelvvel óvatosan kellene bánni.

„Kreálom a relációt és megindexelem natnullra definiálva az attribútumot.” - ilyen és hasonló úgymond rendszerleírásokat lehet manapság olvasni. Az csak részletkérdés, hogy itt szó sincs valódi szakmai szargonról: az ilyesmi nem más, mint magyartalan ocsmányság. A „Létrehozom a táblát és indexelem úgy, hogy a tulajdonság üres értékét is megengedem.” kitétel tényleges szakszargon, amit viszont nem köteles mindenki érteni. Ezért baj, ha a rendszerleírásokban nincs egy olyan fogalomjegyzék, amely megmagyarázza, hogy a mi a tábla, az index, a tulajdonság és az (üres) érték.

Ennél még nagyobb probléma, ha nem egységes a **terminológia** használata. Az információs rendszer tipikus tényezőit és azok tipikus viszonyait szakmai fogalmakkal jelöljük meg. E fogalmak együttese a terminológia. Mivel az IR-t több szinten (fogalmi, logikai, fizikai) lehet szemlélni, a terminológia maga is strukturált. A fogalmi szinten még viszonylag szűk a szakkifejezések tára, ám a szinteken lefelé haladva ez a készlet egyre bővül. Nemcsak amiatt, mert a terv egyre részletezettebbé válik, hanem azért is, mert eltérő koncepciókban készült eszközöket lehet alkalmazni a megvalósítás szintjén. Amit az egyik rendszerben ‘rekordtípus’-nak neveznek, az a másikban a ‘reláció’ nevet viseli.

Lám-lám: a terminológiai zűrzavar máris arra vezetett, hogy elkövettünk egy hibát. Az egy jelenségre (egyedre) vonatkozó adatok sora - például Kovács Pali személyi adatai - a logikai szinten vagy a személy rekordtípus egy rekordjaként, vagy a személy reláció (tábla) egy soraként jelenik meg. Azonban e két valami (rekord és táblasor) nem teljesen azonos. A szakembernek tudnia kell azt, hogy minden táblasor rekordnak is tekinthető, de ez fordítva nem igaz. Ugyanis a reláció nem más, mint bizonyos szigorú korlátoknak eleget tevő rekordtípus.

Megismételjük: e pontnak nem az a feladata, hogy fogalmakat tisztázzon. Így a fentiek csak a mondanivalónk szemléltetésére szolgáltak. Ha valaki 'rekordot' mond 'reláció' helyett vagy fordítva, akkor pontatlanul - másokat megtévesztő módon - használja a terminológiát. Ez pedig, sajnos, számtalanszor előfordul. A fejlesztők az 'egyed' kifejezéssel (fogalmi szint) illetik a 'relációt' (logikai szint). Egyes - olykor nagyon neves - szoftverforgalmazók pedig 'relációs' rendszernek titulálják az adatkezelőiket, amelyek pedig a régi rekord-szemléletben készültek.

Ismételten felhívjuk a figyelmet arra, hogy mindez nem teoretikus fejtegetés.

T 4.4 Alig pár napja egy meglehetősen komoly adatbázis tervével kellett foglalkoznom. Az adatbázis fejlesztése az X, majdani működtetése pedig az Y rendszerrel történik. Mindkettő 'relációs'. Csak éppen az X rendszer 'reláció' fogalma szűkebb az Y rendszerénél ...

A terminológia következetlen használata gyakorlati bajokra vezet. Korlátoz, megtéveszt és gátolja a valóban helyes megoldás megtalálását.

4.4 MEGFOGALMAZÁSI MÓD

Kevesen figyelnek fel arra a tényre, hogy az információs rendszer terve maga is információs rendszer, mert a D 2.1 meghatározásban leírt tényezők szervezett együttese. A terv is adatokban testesül meg; vannak tervezési események illetve tevékenységek; a tervhez is erőforrások kötődnek stb. Az információs rendszer általános képét fentebb *metamodellnek* neveztük. Ezért meglehetősen logikus, hogy az annak tervével kapcsolatos tényezők szervezett együttesét pedig a **meta információs rendszer** kifejezéssel illessük.

Sokan elfeledkeznek arról, hogy ezt a rendszert is fejleszteni kell. Azonban itt nem a fejlesztési szisztéma megalkotásáról és állandó tökéletesítéséről kívánunk beszélni. Csak egy momentumot akarunk kiemelni: maga a terv is ismeretekben testesül meg. Minket most nem ennek az ismeretnek a tartalma, hanem egyelőre csak annak közlési formája érdekel.

D 4.3 Az információs rendszer tervének a közlési formáját nevezzük megfogalmazási módnak.

A pont lényegének a megértéséhez fontos azt tudni, hogy az IR fejlesztési módszerei igen különböző gyökerekből fakadnak. A kezdetek-kezdetén még nem is készültek valódi tervek: a leírt programok garmadáját csak utólagosan becézték tervnek. (NB.: Mindez a hetvenes évek elején történt. Ám kísérteties módon ma is találkozunk olyan vállalatokkal, amelyekben hasonló szinten fogják fel a terv lényegét ...) Később a terv egyes részeit papírra írták úgy, hogy diagramokkal, folyamatábrákkal, tablóképekkel, lyukkártya vagy -szalag képpel, fizikai lemezelhelyezési sablonnal stb. szemléltették a tervet.

A hetvenes évek közepétől kezdtek próbálkozni a számítógéppel támogatott fejlesztéssel. Akkorra már kiderült, hogy az IR terve is egy 'adatbázis', aminek legtermészetesebb kezelési eszköze maga a számítógép. A kísérletek - a fentebb említett gyökereknek megfelelően - eltérő irányokban indultak el.

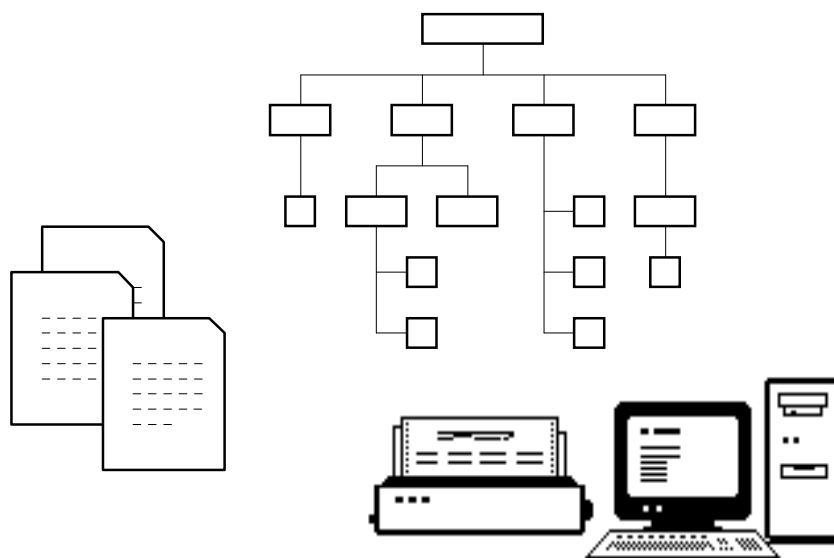
Egy érdekes, ámde viszonylag gyorsan elvetélt próbálkozás a programnyelvek mintájára nyelvi alapokon akarta megoldani a fejlesztést. Ezért kialakították a **tervezési nyelv** formai (szintaktikai), tartalmi (szemantikai) és egyéb szabályait. Vagyis megfogalmazási módként metanyelvet alkalmaztak. A próbálkozás igen látványos kudarcának az oka nem magában az alapgondolatban rejlik. E hőskor kezdetén még papír-alapú bemenetekkel dolgoztak. Mármost az ellyukasztások miatt a tervet igen nehézkes volt számítógépre vinni. Mégis ennek a kísérletnek köszönhető, hogy a struktúra és a terminológia fontossága (vö. előző pont) egyre világosabbá vált éppen úgy, mint a tervezési szintaktikáé és szemantikáé.

Nem kell ezektől a dolgoktól annyira megijedni. Ha valaki programot ír vagy adatot visz be a számítógépre, akkor eleget kell tennie előre lerögzített formai és tartalmi szabályoknak. A programok esetében mindmáig a(z elő) fordítórendszer kényszeríti ki ezek betartását. (Most nem akarjuk részletezni, hogy ugyanez áll a nem fordító, hanem interpretáló módú programrendszerekre is.) Az ismereteknél pedig 'adatrerögzítési utasítás' és a programokba épített ellenőrzés (vö. validálás) igyekezett biztosítani a formai és tartalmi helyességet. Ha tehát valaki tervezési adatot visz be a számítógépre, akkor annak is eleget kell tennie előre lerögzített formai és tartalmi reguláknak. Csak éppen nem a felhasználói adatokra, hanem a fejlesztési ismeretekre vonatkoznak a megkötések.

A tervezési nyelv koncepciója megbukott, mert ötven megtervezett adatféléseggel szemben ötvenszer kellett leírni, lelyukasztani, ellenőrizni pl. az 'Attribute-name' (tulajdonságnév) szöveget. Az is világos, hogy a finn, a portugál vagy a magyar tervezőnek az ilyesfajta megoldás nem 'smakkolt', mert nemcsak a terv tárolása, hanem annak dokumentációja is az angol nyelv szabályaira épült.

A hetvenes évek közepén egy olyan addig szokatlan adatbáziskezelő rendszer koncepcióját alapozták meg, amely ugyan szintén megbukott, de nyomai mind a mai napig minden egyes rendszerünkben fellelhetők. A QBE nevezetű [Query By Example - lekérdezés példa alapján] kezelőről van szó. Az egész ötlet azon alapult, hogy az ember a hétköznapi életben bizonylatokban gondolkodik, tehát hozzászólt ahhoz, hogy formanyomtatványokat tölt ki. Miért ne lehetne akkor számítógépes formanyomtatványokat - sablonos képernyőket - alkalmazni akár a tervezésben is? Hiszen - erről eddig nem szóltunk - erre már manuális kísérlet is történt. A fejlesztőnek éppen úgy **tervezési nyomtatványokat** kellett kitöltenie, mint ahogyan a felhasználónak alapelőzeteket.

A QBE - mint neve is mutatja - a képernyőn megjelenített egy alapelőzetet úgy, hogy annak 'rovataiban' mintákkal, példákkal - tehát kitöltési javaslattal is élt. A 'like' (olyan, mint ...) adatkezelés minden mai ismeretkezelő rendszerben visszaköszön. Ha beírom a 'Személynév:' rovat melletti dobozba: 'KOV', akkor a rendszer fellistázza a számomra az összes - többek közt - Kovács nevű személyt. Maga a QBE időközben csendesen elhalálozott elsősorban a korabeli technikai háttér hiányosságai, másodsorban a kizárólagos angol nyelv, harmadsorban a felkínált, de nem-kívánt minta állandó törlési kényszere miatt. Ám alapötlete az alapja a mai legtöbb fejlesztési eszköznek.



4.3 ábra: Megfogalmazási módok

A **formaorientált tervezés** előnye a nyelvorientálttal szemben nyilvánvaló. Nem a fejlesztőnek kell mindig leírnia azt, hogy 'Attribute-name', mert ez a szöveg automatikusan megjelenik a képernyőn és csak mellé kell írni az adatféle nevét. Mi több, a korábbi hibákon okulva - részleges sikerrel - a mai szoftverek már a nemzeti nyelv használatát is támogatják. Tehát a tervezési bizonylaton, a fejlesztési formanyomtatványon nálunk a 'Tulajdonságnév' szöveg jelenik meg.

Természetesen a tervezési formák mögött is mindig egy adott tervezési nyelv, egy meghatározott terminológia búvik meg. Amennyiben a fejlesztő nem ismeri jól az alapfogalmakat, a tervezési képernyő 'rovatainak' a pontos jelentését, úgy meglehetősen - hmm, mondjuk így - 'érdekes' tervekkel képes összekovácsolni.

Egy ábra sokkal szebben beszél ... A vizuális beállítottságú emberek egy-két leskiccelt vázlatból sokkal többet értenek meg, mint több oldalnyi szövegesen megfogalmazott leírásból. Ezért nem csoda, hogy a számítógéppel támogatott (mérnöki) tervezés [CAD - Computer Aided Design] nyomdokaiba lépve maga az IR-fejlesztés is rátért az **ábraorientált** kurzusra. Az IR tényezőit és ezeknek viszonyait - az ismereteket éppen úgy, mint a feldolgozásokat és felhasználókat - szimbólumokkal jelölték meg. Például a 'dobozok' az egyedtípusok (gépkocsi ill. személy), a közöttük húzott vonal a kapcsolattípus (a gépkocsinak a személy a tulajdonosa) ábrázolásbeli megjelenítési eszköze, reprezentánsa. Manapság az ábrákon alkalmazott jelek egyre inkább egységessé válnak (vö. konvenciók). Azonban egyelőre az „Ahány ház, annyi szokás.” elve érvényesül. Magyarul: a jel és a jelentés viszonya fejlesztési módszerenként eltérő. Példa: az egyedtípust mindenütt 'dobozzal' ábrázolják, de kicsit másként és picit mást is jelent ebben meg abban a rendszerben az 'egyedtípus' lényege.

Az informatikai módszer-alkotók társadalma ezek szerint még nem jutott el arra a felismerésre, miszerint az ábra - mint a kommunikáció eszköze - maga is nyelv. A reprezentálási módtól függetlenül egységes terminológiát és nyelvi - szintaktikai/szemantikai - szabályokat feltételező lényeg. Egyáltalán nem kizárt, hogy kényelmesebb 'megrajzolni' ábraorientált módon egy információs rendszer tervét, mint azt tervezési bizonylaton formaorientáltan vagy még bonyolultabb módon egy mesterséges tervezési nyelvorientációban megfogalmazni. Azonban sohasem lenne

szabad megfeledezni arról, hogy az alapszabályok függetlenek a megfogalmazási módtól. Nem az utóbbi diktálja az előbbi, hanem megfordítva.

Figyelni kell egy további tényezőre is. Az ábra látványos, de kifejezőereje - ezt be kell látni - nem elégséges. Most teljesen mindegy, hogy melyik tényezőt emeljük ki. Ha az IR X tényezője változatainak az alváltozatainak a sajátos részleteit is tükrözni akarnánk (ami szöveges megfogalmazásban semmi gondot nem okoz), akkor az ábraorientált fejlesztésben rengeteg szimbólumot kellene bevezetnünk, ami ellehetetlenítené magát az áttekintést.

Mi tehát az eddig levonható tanulság? Az, hogy az információs rendszerek fejlesztésében az egyetlen és közös strukturális/terminológiai lényegre alapozva (ld. előző pont) a különböző orientációjú (nyelv, forma, ábra) megközelítéseket együtt kell alkalmazni. Azért, mert ahol gyenge az egyik, épp ott erős a másik.

Azonban az orientációknak e jószándékú, célszerű és értelmes vegyítése ma még gondokat okoz. Azért, mert nem állnak rendelkezésre a kellő eszközök.

T 4.5 1992-ben az X fejlesztőeszköz ábraorientált részével fogalmaztam meg az adatbázistervnek egyik részletét. Nagy megdöbbenésemre ugyanazon eszköznek a formaorientált része egyszerűen nem vette be, amit az ábrákon megrajoltam. Fordított esetre is volt példa.

A fenti történethez nem fűzünk kommentárt. Áttekintésünket viszont nem zárhatjuk le anélkül, hogy a ma divatos számítógépes eszközök két, az adatbázis tervezését (de csakis azt) illetően nagyon veszélyes kategóriájára ki ne térnénk.

Az első ilyen segédlet a **szövegszerkesztő**. Roppantul kényelmes, sokoldalú, időnként - a bonyolultság miatt - kiakadó, de azért (ezt valljuk be) célszerűen használható szoftverről van szó. Mármost abban az esetben, ha a cél és az eszköz egymáshoz illik. Történtek kísérletek arra, hogy az IR tervét is ilyen termékkel fogalmazzák meg. „Egy pillanat alatt lecserélem a neveket másokra.” - szólott egy kolléga. Meg is tette. Csak éppen a betű szerint analóg neveket is megváltoztatta - viszont a betű szerint nem hasonlóak továbbra is a régié maradtak.

A második ilyen eszköz szinte nem is kategorizálható. A **programfejlesztés**re nagyon is alkalmazható segédletek szinte végtelen soráról van szó. A **negyedik generációs nyelvtől** [4GL - Fourth Generation Language; ma már ötödikről is beszélnek] kezdve az 'objektum-orientált' dolgokig. Ezen a ponton egyáltalán nem lényeges, hogy az olvasó ismerje-e furcsa kifejezések szakmai lényegét. A fentebbi nyelvi, formai, ábrai, szövegi megfogalmazásokat itt-ott egészen kiváló módon ötvöző eszközökkel/módszerekkel csak egyetlen baj van. Arra roppantul alkalmasak, hogy a 'szobát' fizikailag megtervezzék velük (programépítés), arra viszont egyáltalán nem használhatóak, hogy magát a teljes 'épületet', a rendszert megfogalmazzák a segítségükkel.

E pontot nem zárhatjuk le anélkül, hogy ki ne térjünk az **ikonmánia** roppantul káros következményeire. „Látod, most rákattintok az egérrel erre a képecskére, és máris definiálok egy adatbázist.” - mondja az eszközkezelésben jártas, ám az adatbázishoz mit sem értő fél-szakember. Azt ugyan nem tudja, hogy mit jelent az egyed, a tulajdonság, a felhasználó, a szabvány stb., de iszonyatosan gyorsan egerészik a képernyőn. Nem azon töri a fejét, hogy miképpen lehetne szebb és jobb adatbázist alkotni, hanem azon, hogy miért olyan színű, miért ott van és miért nincs más ikon. A szoftverforgalmazók, akik nem saját maguk ellenségei, ezt pontosan tudják. Több fejlesztési képességgel nem, de egyre több 'zanzával' látják el eszközeiket.

Legyen csak bőven ikon minden fejlesztő rendszerben. Ám úgy, hogy a képek elfogadható alapkoncepciót, teljes és egyértelmű struktúrát illetve terminológiát támogassanak. Ha a fejlesztő ez utóbbiakra illetve a következő pontban leírandó tényezőkre koncentrálna, akkor egyáltalán nem zavaró az ikonmánia. Ha viszont az átfogó kép helyett képecskékben gondolkodik...

4.5 KRITÉRIUMRENDSZER

Az IR terve egy részben *absztrakt kép*. Ez a módszer alapkoncepciójára épül; megfelel az általánosan elfogadott struktúrának és terminológiának; végül pedig formai szempontból az alkalmazott megfogalmazási mód konvencióit követi. Ez a három kellék azonban még messze nem elegendő ahhoz, hogy egy módszert a célnak megfelelőnek nevezhessünk. A három eddig ismertetett tényező pusztán keretként szolgál - a célszerű tervezéshez egyéb kellékek is szükségesek.

Vannak olyan tervezési módszerek, amelyek csak arra nézve adnak eligazítást, hogy miképpen kell összeállítani az IR tervét ('hogyan') és annak mi kell, hogy legyen a tartalma ('mit'). Ez igen kevés. Ugyanis az 'IR terve részben absztrakt kép' kitétel eleve azt sugallja, hogy azt gondolati úton állítjuk össze. Márpedig ez azzal jár, hogy húsz fejlesztőnek húszféle elgondolása támadhat ugyanazt az IR-t illetően. (Mi több, még ugyanaz a szakember sem tervezné meg kétszer a rendszerét teljesen azonos módon.) Az eddigi negatív példákkal szemben legyen szabad itt egy pozitív történetet elmesélnünk.

T 4.6 Az elmúlt hónapokban egy eléggé összetett adatbázis-tervet volt alkalmazni. A lelkes fejlesztők állandóan kérdeztek, és újabb meg újabb ötletekkel álltak elő - így még jobb lesz.

Természetesen a gondos tervezés nem azonos az aggodalmassal, a bíbelődéssel - mert előbb-utóbb valaminek születnie is kell. De a változatok - van mások, van jobb is - átgondolása sohasem nélkülözhető a fejlesztés során. A fejlesztőnek nem pusztán egy működőképes, hanem egy *optimális* elképzelést kell - kellene - az asztalra tennie. Erre pedig csak akkor van módja, ha tudja: az IR terve mitől jó illetve rossz. Tehát ha kritériumok szerint képes értékelni a saját megoldásait.

D 4.4 Tervezési kritériumoknak azon objektív mércék rendszerét nevezzük, amelyekhez viszonyítva mérlegeli a fejlesztés minden résztvevője a terv illetve a tervváltozat megfelelőségét.

A definícióban három momentumra kell figyelni. Először: a fejlesztés minden egyes résztvevője szerepet játszik a mérlegelésben. Ezért megengedhetetlen az is, hogy a mércéket valamilyenük ne ismerje, no meg az is, hogy egyesek más mércével mérjenek, mint mások. Másodszor: a kritériumok objektívak. Ezen azt kell érteni, hogy akár számszerűen is mérhetőek illetve bevett szabályokat kell, hogy kövessenek. (A mai fejlesztésekben túlzottan gyakori a „De én pedig úgy gondoltam, mert ...” kezdetű szubjektívizmus. A tervező nem a 'de én', hanem a 'de az általános elvek' módjára kellene, hogy gondolkodjon.) Harmadszor: az optimum-kritériumok rendszert alkotnak.

Az utóbbi kitétel két dolgot takar. Az egyik az, hogy a kritériumok szervezett együttest képeznek, vagyis tudatosan elrendezettek, egymástól nem függetlenek. A másik az, hogy egyenként is meglehetősen összetettek. Itt egy példával kell élnünk, megelőlegezve a továbbiakat. A redundancia - kétszeres ill. többszörös tényező-meghatározás a tervben - hiba. Tehát a redundancia-mentesség - erény, amelynek követését a minimalitás kritériuma fogalmazza meg. Ez maga is egy összetett mérce, mert a terv sok helyen - több összefüggésben - lehet redundáns.

Ennek a pontnak nem lehet feladata, hogy a több száz (!) konkrét kritériumot ismertesse. (Zárójelben jegyezzük meg, hogy a mai fejlesztő eszközök a legjobb esetben is csak maximum pár

tucat konkrét mércét állítanak a tervező elé.) Ezért csak arra vállalkozhatunk, hogy az absztrakt kritériumokat, vagyis az optimalitás általános vezérlőelveit mutassuk be.

Minden jó megoldást ('erényt') két szélsőséges kilengés ('bűn') kísér. Ezért a kritériumok magyarázatánál a jó út mellett a kétféle rosszat is be kell mutatnunk.

4.5.1 Valóságosság

A terv akkor valóságos, ha a valós rendszerben (VR) lévő konkrét tényezőket és azok összefüggéseit az információs rendszer (IR) elképzelése torzítás nélkül tükrözi. Ez így egy picit butácskán, mi több tautológiának hangzik. Legjobb, ha egy példával világítjuk meg a mondani-valót.

Akad jónéhány olyan ember (a szerző nem tartozik közéjük), aki két vagy még több diplomával rendelkezik. Mármint az általunk ismert legtöbb szervezetnél a 'legmagasabb végzettség' adatcsoportot (mikor, hol, mit végzett) kizárólag csak egyszeres értékekkel lehet kitölteni. A többdiplomás ezért arra kényszerül, hogy vagy ezt, vagy azt a végzettségét diktálja be 'legmagasabbnak', jöllehet az egyik nem magasabb, mint a másik. Vagyis egyszerűen szólva hazugságra, lényegének a megtagadására kényszeríti az információs rendszer.

Ezért helyesen csak az a fejlesztő jár el, aki követi a valóságosság kritériumát, aki lehetővé teszi, hogy egy személyhez több legmagasabb iskolai végzettség ismeretcsoporthoz kapcsolódjon. Durva hibát követ el az a tervező, aki a tervében semmibe veszi a jelenségek (esetünkben: személy és végzettség) viszonyainak a valóságát. Ám a másik szélsőséges kilengés sem jelent kisebb gondot.

A cégek - számos ok miatt - nyilvántartást vezetnek arról, hogy az újonnan felvett munkatársnak mik voltak az előző munkahelyei. Ha csak egyetlen ilyen adatcsoportot biztosít az IR, akkor tervezője az előbbi hibába esik, hiszen az új kollégának több előző munkahelye lehetett. (Itt már a következő kritériumhoz is kapcsolódunk. Nem egyértelmű a terv akkor, ha abban nem tisztázzák, hogy az 'előző' alatt a közvetlen megelőzőt, vagy az összes korábbi kell érteni.) Most azonban inkább a túlzott precizitási vágyról kell szólnunk.

A valóság az, hogy húsz évvel ezelőtt Kovács Pali az ilyen-olyan nevű cégben és ebben-abban a munkakörben dolgozott. Ezért egyes tervezők kényszerítve érzik magukat arra, hogy a személyek és a cégek között meghatározzanak egy olyan többszörös viszonyt, amelyből kitapintható az előző cégek minden adata. Elméletileg ugyan helyesen járnak el, de gyakorlatilag ki az ördög kíváncsi arra - mi több, kinek van pénze és energiája annak tárolására/kezelésére -, hogy húsz éve (az azóta talán már meg is szűnt) X cégnek mi volt a postai címe?

A valóságosság ellen kétféle módon lehet védekezni: pontatlansággal és túlzó pontossággal. Ez az az egyetlen kritérium, amely semmilyen automatával sem ellenőrizhető. Azért nem, mert nincsen olyan *szemantika* ill. *matematika* alapú eszköz, amely szembeszállhatna a fejlesztővel: nem egy, hanem akár több (számszerűség, matematika) 'legmagasabb' végzettség is létezhet, így tehát már maga a fogalom is csacsiság (értelmezés, szemantika).

NB.: A fejlesztési segédletek vagy tartalmi (szemantikai), vagy számszerűségi (matematikai) alapon vizsgálják, hogy a terv eleget tesz-e a kritériumoknak. Itt nem 'kizáró vagy'-ról van szó. Sokszor előfordul, hogy a szemantikai elemzés világít rá a számszerűségi problémákra - vagy éppen megfordítva.

4.5.2 Érthetőség

Ez a kritérium éppen úgy két szintű, mint az előző. A módszer absztrakt és a terv konkrét szintjén is megfogalmazható. Úgy tűnik, hogy itt egy pillanatra vissza kell, hogy kanyarodjunk az előző alponthoz.

Ha a konkrét tervben egyetlen 'legmagasabb végzettség' dolgot tételezünk fel, akkor mi tévedünk. Ámde maga a módszer is lehet hibás. A leghétköznapiabb információs egységünk például az **adatszoport**. A dátumot, a lakcímet stb. a normális ember egyszerre egységként is és részekként is (év, hó, nap) szemléli. Számos adatkezelő rendszer viszont kizárja a csoport alkalmazását. Ezért a tervező vagy csak egyben, vagy csak részekben határozza meg a dátumot. Mindkét megoldás már az absztrakt szinten torzít, mert a valóság az, hogy...

Az érthetőség az absztrakt szinten az alapvető tervezési fogalmak megfelelő használatát jelenti (ld. 4.3 pont). Ha a tervező kapcsolat helyett relációt mond és ír, akkor terve zavaros, nem érthető. A konkrét szinten éppen úgy a helytelenül használt **elnevezések** jelentik a gondot, mint az általánosabb absztrakt szinten. Ismét csak arra kényszerülünk, hogy példával szemléltessük mondanivalónkat.

T 4.7 Két évvel ezelőtt egy egyébként neves vidéki kórház adatbázis-tervét kellett (volna) vizsgálnom. Terv helyett kaptam egy olyan COBOL-ban készült állománylistát, amiben a fájlok Fx, azokon belül az adatok az Ay 'nevet viselték'. Természetesen egy kukkot sem értettem az egészből.

A hetvenes években talán megengedhető volt, hogy a fejlesztők a rendszerek tényezőit mondjuk (és ez bizony gyakran megtörtént) éppen aktuális szerelmeik nevével illessék. Talán nem meglepő, hogy ma már ez nem elfogadható mód.

Ki jár tehát az érthetőség erényének az útján? A IR tervében a tényezők, mint fogalmak névvel jelöltek. Az okos fejlesztő ezért **beszélő neveket** használ. Két módon vétekezhet. A túlzottan tömör, nem-beszélő megnevezésekkel (pl. típus - mivel ezernyi típus fogalmunk van) vagy a túlzottan csacsogókkal. (A szerzőnek volt alkalma találkozni a 'családi pótlékot igénylő hátrányos helyzetű édesanya személyi száma' nevű adattal is, aminek a helyében a 'személyi szám' egyszerű megnevezés is bőven megtette volna.)

Persze azok, akik nem gondolkodnak fejlesztési szintekben (fogalmi, logikai és fizikai) nehezen értik meg, hogy itt miről is van szó. Az általuk alkalmazott fejlesztési segédeszköz hosszban, karakterkészletben, mintában - nem lehet pl. üres a névben - már a logikai szinten is korlátozza a megnevezéseket, a fizikai szint behatárolásai pedig még szigorúbbak (és kezelőről-kezelőre változnak). Az IR tervé még nem megvalósítás, hanem a kommunikáció eszköze. Tehát arra hivatkozni az elkészítése kapcsán, hogy majd a megvalósításban úgyis másféle neveket kell alkalmazni - csacsiság vagy szándékos önáltatás, mi több: olykor durva megtévesztés. Egyes fejlesztőknek 'szent' célja az, hogy rajtuk kívül senki se értse a rendszer lényegét. Tőlük azért nem kell annyira félnünk, mert előbb-utóbb a saját dugájukba dőlnek: pár hónap múlva már saját maguk sem tudják, hogy régebben mit is neveztek Ax-nek...

4.5.3 Egyértelműség

Tegyünk csak gyorsan két próbát. Először azt mondjuk, hogy 'Gabi'. Majd azt, hogy 'nem' és 'szex'. Az utóbbi esetben az olvasó azonnal tudja, hogy egyazon lénységnek a kétféle nevről van szó. A 'nem' és a 'szex' - **szinonima**. Viszont az előbbi esetben senki sem képes kideríteni a

nemet, a szexet: azt, hogy a 'Gábor' vagy a 'Gabriella' becenevét használtuk-e. Azért, mert két lényegnek az egyféle nevére van szó. A 'Gabi' - *homonima*.

Mind az általános tervezési fogalmakban, mind az általunk kialakított konkrét tervekben kerülni kell a szinonimákat és a homonimákat. Akinek ez sikerül, az eleget tesz az egyértelműség kritériumának. Nem is mellesleg ez a mérce szinte elválaszthatatlan az előző kettőtől. Ha valaki nem törekszik a valósághűsége és az érthetősége, akkor tervében hemzsegni fognak a szinonimák és homonimák. Például az 'F1' és az 'F2' nevű állományban (vö. T 4.7) egyaránt lesz 'A1' módon megnevezett, de eltérő tartalmú adattétel.

Az értelmesen beszélő és egyértelmű nevek alkalmazásának az erénye ellen is kétféle módon lehet vétkezni. A történet kapcsán emlegetett szükséztűséggel illetve a felesleges bőbeszédűséggel. A legtipikusabb hiba mindkét esetben az, hogy nem- vagy visszaélnek a *minősítés* lehetőségével. Mivel pl. a rendszerben lehet gépkocsi- és cégtípus is, a pusztán 'típus' név - csúnya szó - alulminősítés. A személyi számról viszont mindenki tudja, hogy mit jelent. Ezért nincs szükség arra, hogy azt a személy ismereténél a 'személy személyi száma', ezzel szemben a gépkocsi ismeretek során a 'tulajdonos személyi száma' jelöléssel illessük. Az előbbi példában homonimát, az utóbbiban szinonimát generálunk.

Az egyértelműség kritériuma matematikai és szemantikai módszerekkel igen jól ellenőrizhető. Mivel ez a mérce közel áll az érthetőséghez is, nincs kizárva az sem, hogy akár az automata is felhívja a figyelmet az előző mérce megsértésére. Arra azonban figyelni kell, hogy a homonima és a szinonima eltérő természetű. A homonima olykor redundanciaként mutatkozik (ld. a 4.5.4 alpontot), viszont a szinonima gyakran csak közvetett módon, például a hiányok elemzésével (ld. a 4.5.5 alpontot) küszöbölhető ki.

Az olvasónak egyelőre meg kell elégednie ezzel az átfogó áttekintéssel. Ám azt is tudnia kell, hogy az egyértelműségi bajok kiküszöbölésére ma már igen jó elméleti arzenál áll a rendelkezésre. Kár, hogy a mai fejlesztési eszközök ennek a fegyvertárnak csak egy csöppnyi hányadát vetik be a tervek értékelésekor.

4.5.4 Minimalitás

Ha maga a terv nem minimális, akkor persze az IR sem lesz az. A minimalitás kritériuma nem számszerű kicsiséget jelent. Ha a valós rendszer (VR) összetett, akkor senki se várja, hogy egyszerű legyen az információs rendszer (IR) terve.

T 4.8 Egyik tervező-kollégám hirtelen elkeseredett: lehetséges, hogy ez a rendszer ennyiféle tényezőt tartalmaz? Megnyugtattam: azokon kívül, amiket most látsz, még rengeteget.

És valóban: annak a bizonyos tervnek a mérete idővel a háromszorosára nőtt. A minimalitás ugyanis nem elhagyást, csak okos takarékoságot jelent.

A minimalitás mércéjének azt tesz eleget, aki kerüli a tervben a felesleges - erre a jelzőre fogunk alább visszatérni - átfedéseket, vagyis redundanciákat. Ha ugyanis a tervben valami feleslegesen (!) redundáns, akkor magában a leendő információs rendszerben is átfedések fognak fellépni. Ezek pedig több ok miatt kerülendők.

Tegyük fel, hogy a tervben a személy neve jellemző nemcsak a személyhez, hanem az általa birtokolt gépkocsihoz is kapcsolódik! Ez a duplikáció nem csak azzal jár, hogy a nevet iksz-plusz-egyszeresen kell tárolni (egyszer magánál a személynél és x-szer a birtokában lévő kocsi-nál). Nem csak azt eredményezi, hogy - ahol az adat kétszeres, ott a tevékenység is az! - többször kell megírunk a személy nevének a kezelését végző rutint. Ezeknél ugyanis még nagyobb gond a konkrét adatok egyértelműségének a lehetséges hiánya. Szabó Aranka férjhez megy és felveszi a

Kovács Pálné nevet. Vajon mi is garantálná, hogy az itt-ott - több helyen - tárolt nevét mindenütt egyértelműen, egységesen és egyidejűleg fogják megváltoztatni?

Az egyértelműség kritériumának (ld. előző alpont) a megsértési veszélye miatt egyes fejlesztők minden redundanciától úgy félnek, mint a tűztől. Azonban nem csak jobbra, hanem balra is ki lehet lengeni e mércét illetően is.

Az IR tervének van fogalmi, logikai és fizikai szintje. Az egyértelműségnek a minimalitással szembeni elsődleges volta miatt jól teszi minden fejlesztő, ha az első - a fogalmi - szinten kerüli a redundanciát. Ámde azt is tudomásul kellene venni, hogy az ismereteket adott felhasználók meghatározott kezelőrendszerrel (környezet - logikai szint) és korlátos számítógépeken (eszköz - fizikai szint) fogják kezelni.

Ha tehát a fogalmi szinten a redundancia kerülendő, ez nem jelenti azt, hogy a logikai és a fizikai szinten nincs szükség a tudatos és szándékos bevezetésére. Kevesen akarják megérteni, hogy a minimalitás kritériuma - az egyértelműség kedvéért - a fogalmi szinten mindig betartandó, viszont az alsóbb szinteken éppen 'szükséges rossz' is lehet. „Az ismétlés a tudás anyja.” Bár ez a kitétel látszólag nem illik ide, gondolja meg az olvasó, hogy ebben a könyvben is hányszor kényszerülünk ismétlésekre. Azért, mert az ismerethalmaz nehezen átlátható, fizikailag szétszórta található csak meg stb.

Nem a fizikai adattárolás és -kezelés redundanciája jelenti a legfőbb veszélyt az információk rendszer optimalitása szempontjából. Mi több, az igazándiból tudatosan tervezett többszörösség a *finombehangolás* [tuning] egyik legfőbb célja és eszköze. Csak az a baj, ha még maga a hangszer sincs kész, és azt a tervező máris be akarja hangolni a legelső lépésben. A fogalmi szinten nincs apelláta: törekedni kell az abszolút minimalitásra. Azért, mert különben az egyértelműség elve szenvedhet csorbát. Ha több helyen adjuk meg a 'típus' tulajdonságot, akkor vajon homonimáról vagy csak redundanciáról van szó?

4.5.5 Teljesség

Feltételezve azt, hogy az átfedéseként mutatkozó homonimákat kiküszöböltük a tervből (nincs pl. kétféle eltérő értelmű 'típus' tulajdonságunk), a redundancia mértéke matematikai eszközökkel nagyon pontosan megállapítható. Tehát nem nehéz kimutatni azt, hogy a terv mennyire sérti a minimalitás elvét.

A teljesség mércéje egy fokkal bonyolultabb. Nincsen olyan automata, amely rávilágítana arra, hogy a személyeknél elfeledkeztek a 'lakcím' adatról és ezért a terv hiányos. (Itt ismételtetnünk kell a kritériumok összefüggésére. Ha a terv ebben az értelemben nem teljes, akkor nyilván nem is valóságghű.) Viszont a valóban jó fejlesztési segédeszköz az általános struktúra alapján (vö. 4.3 pont) dolgozik. Ezért az absztrakció magasabb szintjére alapozva azonnal képes annak kimutatására, hogy *általában* mi hiányzik a fejlesztő tervéből. Ez ugyan nagyon is fontos dolog, de bennünket most inkább az érdekel, hogy a tervező egy adott és *konkrét* rendszer esetében miképpen kell, hogy törekedjen a teljességre és mi segítheti ebben a rész céljában?

A teljesség mércéjének a betartását egészen komoly szemantikai/matematikai eszköztárral lehet figyeltetni. Tegyük fel például, hogy a tervező a gépkocsinál elfelejt utalni a tulajdonosra. Ezzel az adatkapcsolati háló durván megszakad, a terv nem teljes, egy valamire való automata viszont ezt a hiányt fel tudja tárni.

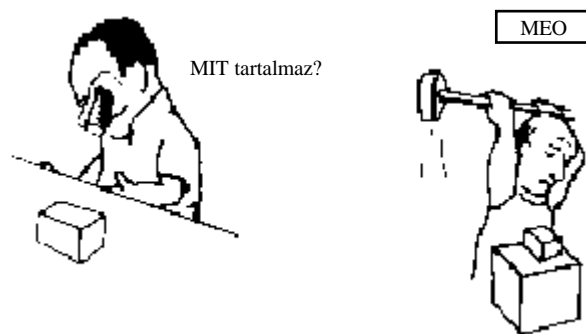
Erény az, ha valaki betartja a teljesség követelményét. Hiba az, ha valaki nem tesz eleget annak, vagyis hiányos tervet kreál. Viszont az olvasó megkérdesheti, hogy a teljesség erényét miképpen lehet másféle kilengéssel is megsérteni? Ha valamit nem tartalmaz az IR terve, az baj (hiány). De miképpen lehet teljesnél is teljesebb a terv? Miként tartalmazhat többletet?

Erre a kérdésre már feljebb választ adtunk. A teljességre úgymond gondosan ügyelő tervező mindent kétszer mond, mindent kétszer ír le. Ezzel a minimalitás - és esetlegesen az egyértelműség - elvét hagyja figyelmen kívül.

4.5.6 Harmónia

Az IR terve csak akkor optimális, ha az a fentebb ismertetett ötféle kritérium mindegyikének megfelel. A mércék összefoglalásaként ismételten felhívjuk a figyelmet arra, hogy a terv értékelési szempontjai szorosan összefüggnek és vizsgálati sorrendjük sem közömbös. Két példával szemléltetjük a bonyolult kritérium-viszonyokat.

A szinonima nem más, mint rejtett redundancia. Ha ugyanis a 'járműfajta' és a 'kocsitípus' ugyanazt jelenti, de egynél több helyen jelenik meg a tervben, akkor ugyanaz az ismeret többszörös tárolást és kezelést igényel. Ez a többszörösség két problémát rejt magában: egyet a működtetés, egyet a tervezés szintjén.



4.4 ábra: Rendszervizsgálat

A működtetés szintjén a redundancia az *inkonzisztencia* veszélyét hordozza magában. A rendszerben a két (vagy több) azonos lényegű, de eltérő nevű dolog (szinonima) azonosságára nem figyelve ugyanahhoz az autohoz az egyik névnél másféle értéket adnak be, mint a másik megnevezésnél. Az ilyen jellegű tartalmi ellentmondást nevezzük inkonzisztenciának. (A konzisztens ugyanis önmagával azonosat, megegyezőt, harmonikusat jelent. Ha a BMZ 873 rendszámú gépkocsi 'kocsitípus' adatának értéke itt 'Lada', másutt pedig a 'járműfajta' adatának értéke 'Seat', akkor a rendszer inkonzisztens, ellentmondó.)

A tervezési szinten az a baj, hogy a fejlesztő elkönnyveli a terve minimalitását, miközben pedig az álcázott redundanciát tartalmaz. A szinonima ugyanis nem más, mint rejtett redundancia. Ezért a terv minimalitási kritériuma addig nem is vizsgálható, ameddig a terv nem egyértelmű, mert szinonimákkal terhelt. Tehát először ezeket kell kiküszöbölni - és csak azután lehet a minimalitást elemezni.

Ellenkező természetű problémát okoz a homonima. A tervező - valamilyen ok miatt - a 'típus' névvel illeti mind a jármű fajtáját, mind a biztosítás jellegét. Így a terv a fejlesztési szinten nem tesz eleget a minimalitás követelményének, mert úgy tűnik, hogy kétszer szerepel benne ugyanaz a dolog ('típus'). A homonima - egyazon névvel illetünk eltérő tartalmú dolgokat - mindig a látszat-redundancia jelenségét okozza. Ha valamelyik 'típus' adatot eltávolítjuk a tervből, akkor az nem lesz teljes. Ha ezt nem tesszük meg, akkor formailag nem lesz minimális. Itt sincs más megoldás mint az, hogy először az egyértelműség mércéje szerint - világos nevek

(‘járműtípus’ és ‘biztosítástípus’) alkalmazásával - kiemeljük a tervből a homonimákat. Csak ezután van lehetőségünk a minimalitás, majd a teljesség elemzésére.

Az optimum-kritériumok nem pusztán arra szolgálnak, hogy tetszetős tervet készítsünk és eleget tegyünk bizonyos *elméleti* követelményeknek. Éppen úgy, mint az eddigiekben, most is éppen hogy nagyon is *gyakorlati* megfontolásokról van szó. Az IR legvégső felhasználója a közönséges átlagpolgár, aki nap mint nap arra kényszerül, hogy ilyen-olyan papírokat töltsön ki. Ennek során akarata ellenére is ‘hazudnia’ kell, mert például öt nyelvet beszél, de csak háromra akad rovat a papíron (valósághűség). A számlákon, bizonylatokon olyan ‘rovatnevek’ találhatók, amelyeknek a felével sincs tisztában (érthetőség). A mindenféle ívek, kimutatások hemzsegek a szinonimáktól és homonimáktól (egyértelműség). Az pedig köztudott, hogy leveleinket elcímzük, nevünket hol így, hol úgy írják - és ennek az adatbázis inkonzisztenciája az oka, ami a nem-egyszeres adattárolásból és -kezelésből fakad (minimalitás). Ugyanennek köszönhető, hogy ha hússzor fordulunk egyazon a céghez, társasághoz, intézményhez, akkor ugyanennyiszor kell újra meg újra leírogatnunk ugyanazokat az adatainkat. Végül az is gyakori, hogy a terv tényezői közötti viszonyok kialakításának az elmaradása miatt rossz, hiányos ismeretek jutnak csak el a végső-felhasználóhoz (teljesség).

A kritériumrendszerrel az olvasónak nagyon csínján kell bánnia.

T 4.9 1989-ben az IFIP által készített és a szintén általa hibátlannak nevezett adatbázis tervét vizsgáltam. A 100-nál nem sokkal több tényezőt tartalmazó tervben 72 hibát fedeztem fel - közöttük egészen durvákat is.

A kritériumok tekintetében a módszer-alkotók messze nem állnak a helyzet magaslatán. Ugyanez vonatkozik a fejlesztési segédeszközökre is. A ma széles körben alkalmazott módszerek illetve eszközök átlagosan nem több, mint másfél tucat mércét alkalmaznak, és azokat is egymástól elszigetelten. Az információs rendszerek generikus szerkezete (ld. 4.3 pont) annyira összetett, hogy legalább tízszer ennyi mércét kellene optimum-kritériumként tekintetbe venni úgy, hogy ráadásul azok egymással bonyolult módokon fűggenek össze.

A kritérium-hiánynál is nagyobb baj, hogy egyesek módszerük ill. rendszerük *erényeként* állítják be és kritériumként tekintik azt a mozzanatot, ami valójában a módszer *korlátja*. Például egy adatkezelőből hiányzik az adatcsoport szerkezeti tényezőjének a meghatározási képessége. Készítője nem, hogy elrejtje ezt a hibát az alkalmazó elől, hanem abból még érvet is kovácsol: így jó, mert..., holott a csoportok hiányában az adatbázis egyértelműsége és teljessége nem biztosítható.

4.6 TERVEZÉSI ALGORITMUS

Ebben a pontban nem a rendszerfejlesztés szakaszairól, fázisairól, nem a teljes rendszer-kialakítás lépéseiről, eljárásáról lesz szó. A *fejlesztési eljárás* ugyanis a projekt-menedzsment témakörébe tartozik és nem pusztán tágabb, hanem más fogalom, mint a **tervezési algoritmus**.

D 4.5 Tervezési algoritmusnak azt a tevékenység-sorozatot nevezzük, amelynek megfelelően a fejlesztő a rendszer modelljének az egyes tényezőit - az egyiket a másik után - kialakítja, majd a különálló részleteket egymással összehangolja.

Egy triviális hasonlaltal próbáljuk megvilágítani a dolog lényegét. A tervezési algoritmus leginkább a konyhai receptre hasonlít, amennyiben valahogyan ilyen módon fogható fel: „Végy X darab M-tényezőt, Y darab N-tényezőt, majd ...” Itt tehát nem absztrakt elvről, hanem egy adott szintig konkrét ajánlásról van szó.

Ötödik kellékünk kapcsán több gondolatot kell körüljárnunk. Amikor az IR már elkészül, annak működtetése során az alkalmazók előírással felhasználási eljárásokkal kezelik a felhasználói adatokat. Ezek az ‘alkalmazási algoritmusok’ olykor merevek (például előbb kell a számítógépre vinni az X-féle adatot, mint az Y-félét), máskor rugalmasabbak (a két bevétel sorrendje közömbös, de mind a kettőnek meg kell történnie). A tervezési algoritmus felhasználója a fejlesztő, akinek - az előbbi gondolatsor analógiájára - előírással tervezési eljárásokkal kell kezelnie a tervezési információkat. A ‘tervezési algoritmus’ is lehet merev illetve rugalmasabb.

Akadnak olyan fejlesztők, akik a kimondottan *merev* recepteket kedvelik. A módszer-alkotó igenis írja elő a számokra, hogy melyik a legelső lépés, hogyan követi azt a második, majd végül az utolsó. Ne kelljen gondolkodni azon sem, hogy a tervezési részlépésben miképpen kell megragadni az aktuális tényezőt. A hetvenes években a módszer-alkotók az ilyen szemléletű módszer-alkalmazók kiszolgálására törekedtek. A BISAD [Business Information Systems Analysis and Design] módszer ‘forsrifos’ algoritmusai vagy az ARDOS becenevű eljárás forma-orientált (vö. 4.4 pont) módszer ugyancsak kötött receptje után még ma is sóvárog a régivágású fejlesztők egy része. Mert mennyire egyszerű, ha kézhez kapunk egy, az előírással kitöltési sorrendnek megfelelően összeállított tervezési bizonylat-készletet, hiszen akkor csak a kitöltésre kell ügyelnünk.

Ez a gondolatmenet téves, mert éppen a fejlesztés legfontosabb mozzanatát hagyja figyelmen kívül. Nevezetesen azt, hogy a tervezés mindig *kreatív munka* volt és lesz. Ezért a merev tervezési eljárás csak annak kedvez, akinek a papír kitöltése a végcélja, nem pedig az, hogy optimális rendszert kreáljon. Annak, aki a felszínen szigorúnak látszó - de minőségi korlátokat a mélyben nem támasztó - algoritmusra való hivatkozás az egyetlen mentsvára, képtelenségének leplezője.

A mai számítógéppel támogatott módszerek egyikének-másikának a tervezési algoritmusára éppen a fentiek ellentettje a jellemző, mert ezek annyira *rugalmas* tervezési eljárást kínálnak, hogy az már szinte nem is tekinthető algoritmusnak. A fejlesztő a rendszer tényezőit teljesen tetszőleges sorrendben jelölheti ki, ám éppen ezért időnként segéd-algoritmusokat kell(ene) alkalmaznia a tényezők összefüggéseinek az elemzésére. És éppen itt kezdődnek a gondok, mert pont ezen összehangoló tevékenységek szoktak törvényszerűen elmaradni.

Ám a legtöbb módszer mereven-rugalmas vagy rugalmasan-merev. Például a CASE-ek (ld. 4.8 pont) nem adnak egy Ariadne-fonalat a fejlesztő kezébe, hogy az eligazodhasson a számtalan tervezési lehetőség labirintusában. Viszont egyes tényezők tervezési sorrendjét megkötik. Tipikus példa, hogy például a ‘személy lakcíme’ tulajdonságot, adatféleséget csak valamilyen egyedhez kötötten lehet kijelölni, önállóan nem. Ez egyrészt kellemetlen, másrészt e szigor nem vezet sehová. Kellemetlen a dolog azért, mert a fejlesztő ismerhet már minden fontos vonást egy adott tényezőt (itt: a lakcímet) illetően, de ezt az ismeretét nem tudja számítógépre vinni a fejlesztési adatbázisba az elgondolás pillanatát követően. Célszerűtlen azért, mert ugyanakkor az eszközök nem zárják ki, hogy több ill. nem a megfelelő egyedhez kössenek pl. egy tulajdonságot. Mondjuk a lakcím a személy egyeden kívül megjelenik a gépkocsit leíró adatsorban is.

A fejlesztési módszer kellékei összefüggenek. Ha tehát maga a struktúra (ld. 4.3 pont) tökéletlen, akkor az arra épülő tervezési algoritmus sem lehet jó. Az ismeretek - pl. személy-azonosító, lakcím - bármennyire is nehéz elképzelni, az egyedektől részben függetlenek. Továbbá részkritériumok rögzítik (ld. 4.5 pont) pl. azt is, hogy a személy-azonosító miért, a lakcím miért nem kapcsolható több egyedhez is. Ezért természetes, hogy amelyik módszerben sem a struktúra, sem a kritériumok rendszere nem kellőképpen átgondolt, abban a tervezési eljárás maga is meglepő vonásokat fog mutatni.

A személyi számítógépek megjelenését követően terjedt el egy meglehetősen közkedvelt fejlesztési részmodszert, amelyre a tervezési algoritmus teljes hiánya a jellemző. A **próbálkozás és tévedés** [trial and error] megközelítésről van szó. A fejlesztő - az on-line eszközök birtokában ma már erre módja van - teljesen lazán, az első ötletei szerint állítja össze a rendszertervét. Gyors programozást követően mindjárt futtatják is a programokat (próbálkozás), majd kiértékelik az eredményeket. Ha azok nem kielégítőek (tévedés), akkor újabb ötletek szerint újra próbálkoznak. Ez a megközelítés a mi szemünkben nem módszer. Csakis azért említettük, mert a komoly módszerhátterű eszközöket ma is sok fejlesztő ilyen „vak tyúk is talál szemet” alapon használja.

T 4.10 A hetvenes években sok számítástechnikai cégnél székelytáblára írták ki annak a programozónak a nevét, aki harmadszorra sem fordította le sikerrel a programját. Ma ...

Szerintünk a próba/hiba megközelítés ma sem elfogadhatatlan, ha annak a tudatos alkalmazásáról van szó. Ha célja nem a végletesen rossz kikerülése, hanem az, hogy a jónál is jobbat alkossunk. Különösen a fizikai tervezés - a számítógépen történő tényleges megvalósítás - szakaszában alkalmazhatjuk ezt a részmodszert. Ekkor azonban ezt már **finombehangolásnak** [tuning] nevezzük, ahol már maga a fogalom is a meglévő jónak a tökéletesítését sugallja.

Az ún. **előprojekt** [pilot-project] kapcsolatos mondanivalónk meglehetősen hasonló a fenti gondolatmenetre. Nem véletlenül, hiszen olyan próbálkozás és tévedés részmodszerről van szó, amely nem egy-egy feldolgozásra, hanem egy célszerűen kiválasztott teljes rendszerreszre vonatkozik. A fejlesztők ugyanis két hibát követhetnek el. Az egyik az öncélú kísérletezgetés, a másik a kísérletek teljes elhagyása. Az előbbiről volt szó a fentiekben, az utóbbiról itt beszélünk.

Manapság a fejlesztők egyre többet hallanak a tervezés fontosságáról és arról, hogy a „minden mindennel összefügg” elve miatt a rendszert nem igazán lehet részenként megtervezni. Elfeledkeznek arról, hogy az IR tervének három szintje van illetve ha erre emlékeznek is, nem tudnak mit kezdeni e szintekkel. Pedig a képlet világos. A fogalmi szint szolgál az IR terve egészének az összefogására, a kritériumok érvényesítésére. A logikai és a fizikai szinten a teljes rendszer egy-egy rendszerresze igenis lehet - átmenetileg - eltérő pontossággal kidolgozott. Az okos kísérletezésnek éppen az a lényege, hogy kiragadunk egy tipikusnak mondható - az IR legfontosabb adatait és eljárásait felölelő - rendszerreszt és azt (de csakis azt) a kvázi-megvalósítás szintjéig határozzuk meg, mielőtt a többi rész logikai/fizikai tervezésébe fognánk.

A kiválasztott rész arra szolgál, hogy képet, mintát nyerjünk a leendő rendszer mélyebb részleteinek egyes vonásairól, mielőtt a teljes munkába fognánk. Ez a megoldás a **prototípus-alkotás** [prototyping], amivel csak egy bajunk van. Az, hogy - siker esetén - a fejlesztők elfeledkeznek arról, hogy mi is volt az eredeti cél. A sikeresnek ítélt rendszerreszt - már az időhiány miatt is - úgy könyvelik el, mint véglegesen megvalósítottat, és a többi rendszerresznél nem alkalmazzák a tapasztalatokat. Magyarul: a prototípus elveszti ezt a célját, jellegét.

A 3.2.3 alpontban elmondtuk, hogy a tervezés és az elemzés egymást szorosan kiegészítő momentumok. Ezért legfontosabb mondanivalónk az, hogy csakis az a tervezési algoritmus az elfogadható, amelyben e két mozzanat összhangban áll. Sajnálatos módon a legtöbb mai tervezési algoritmusban az elemzés vagy nem is kap helyet, vagy nem megfelelő a súlya. Nincsenek olyan részlépések, amelyek az optimum-kritériumok betartásának az ellenőrzésére kényszerítenék a tervezőt. Optimális terv csak többszörös **iteráció** eredményeképpen születhet mégpedig úgy, hogy a fejlesztő **változatokat** mérlegel. Nem lehet kielégítő az a tervezési algoritmus, amely nem épít az iteráció és a változat párosára.

Utolsó előtti gondolatként a „hol kezdjem” felvetésre kell kitérnünk. Nagyon sok tervező számára ez a 'kályha' jelenti a legfőbb problémát. A merev tervezési algoritmusok mindig kínálnak egy fix kiindulópontot, viszont a rugalmasak ránk bízják a választást. Vajon melyik a jobb megoldás?

A hetvenes évek elején a *kimeneten* [output] alapuló lépéssor volt divatban. A fejlesztő szépen megrajzolta a 'tablótervet', majd azt egyeztetette a felhasználóval. A tervezési algoritmus így abban merült ki, hogy a rendszer összes többi elemét a kimenet produkálásának a szolgálatába állították. Ez a 'kályha' nagyon gyorsan alkalmatlannak bizonyult: ha az ismeretigények változtak - ezt pedig megtették többször, még a fejlesztés kellős közepén is -, akkor újra kellett rajzolgatni a kimeneteket, újradefiniálni az állományokat és a feldolgozásokat stb. Nem járt nagyobb sikerrel a *bemenetre* [input] alapozó szemlélet sem. Mert hamarosan kiderült például az, hogy nem a 'számlafeldolgozás' a valódi feladat, hanem a számlával mint bemenettel kapcsolatos valamennyi ismeret egységes és integrált kezelése az.

A hetvenes évek közepétől a *feldolgozás* ún. strukturált tervezése jelentette a kiindulást. Az események és tevékenységek hierarchikus ábráját kellett először kitalálni. Ekkor volt nagy divat az ún. **fekete doboz** [black box] koncepciója. A fejlesztés kezdetén csak a főbb tevékenység-csoportok lényege volt ismeretes, azok belső tartalma nem. E fekete dobozok tartalma fokozatosan vált szürkévé, ahogyan a feldolgozási hierarchiát egyre pontosabban specifikálták, hogy végül egészen kifehéredjen. Azonban a folyton változó feldolgozási igények miatt ez a 'kályha' sem bizonyult tartósan fejlesztőt-melengető kiindulópontnak.

Mivel az információs rendszerben az adatvetület a legstabilabb, a nyolcvanas években az **adatbázis-tervezés** [database design] kezdett a tervezési algoritmus első és meghatározó elemévé válni. Mindmáig ez számít korszerű megközelítési módnak. Azonban ezt a szemléletet sem szabad túlzásba vinni. (Mondja ezt az a szerző, aki egyébként az adatmodellezés iránti különleges vonzalmáról ismert.)

Tudomásul kell venni, hogy az információs rendszerrészek nagyon különböző természetűek lehetnek. Lehet, hogy az egyik ilyen, a másik amolyan szemléletet követel meg. Ezért nem gondolkodik egészségesen az, aki csak egyféle tervezési algoritmust óhajt látni. Nem az egyszerűség a legfőbb követelmény. Két dolog fontos: a tervezési eljárás helyes struktúrára épüljön és támogassa a kritériumok elemzését. Ha e két feltétel teljesül, akkor tökéletesen mindegy, hogy kimenetre, bemenetre, feldolgozásra stb. alapozza-e a fejlesztő a tervét. Az elemzési eljárás során úgyszólván kibuknak az egymással nem harmonizáló tényezők.

4.7 DOKUMENTÁCIÓ-KEZELÉS

Az IR fejlesztése során számos különböző célú dokumentáció születik. Ennek a pontnak nem az a célja, hogy ezek mindegyikét bemutassa. Itt kizárólagosan a rendszerterv dokumentálásáról lesz szó. Bár a tervezési eljárás végeredménye a tervtermék, a fejlesztés során több tervváltozat (ld. előző pont) is születhet - és természetesen mindezeket dokumentálni kell. Éppen ebből következik, hogy az IR terve nem csak egyetlen célt szolgál.

A fejlesztés során annak résztvevői az átmeneti dokumentációkon keresztül *kommunikálnak* egymással. Az IR terve részben absztrakt kép. Így nem csoda, ha a rendszertervező, a programozó, a felhasználó szerepű résztvevők ill. éppen két - a rendszer eltérő, de kapcsolódó részeiben érdekelt - szereplő személy más-más elképzelést alakít ki magában a rendszer valamelyik tényezőjére vagy több rendszerelem összefüggésére nézve. A baj másutt rejtőzik.

T 4.11 Egy igen összetett rendszer fejlesztése során a résztvevők hosszú viták után megállapodtak abban, hogy mit fognak közösen érteni az 'objektum-viszony' nevű egyeden. Később többen elfeledkeztek a lényegről, mások pedig - akik újonnan kerültek kapcsolatba a rendszerrel - szintén saját értelmezéseket kezdtek kialakítani.

Most tökéletesen közömbös, hogy mi is az az 'objektum-viszony'. Az is csak részben fontos, hogy egy eléggé bonyolult dologról van szó, és annak lényege többször pontosítódott. A mindenkori aktuális változat mindig dokumentálásra került. Ám ennek ellenére a régi résztvevők némelyike nem követte az aktuális verziót, az újak pedig meg sem nézték a dokumentációt.

Pedig a dokumentáció második célja éppen az **új felhasználók** tájékoztatása, eligazítása lenne. A fejlesztések során végtelen sok idő megy el azzal, hogy a friss erőknél a régiéket kell, hogy elmagyarázzák: mit is jelent az X valami. Mi több, történetünk szerint ez nem is vezethet sikerre akkor, ha Kovács (a korábbi résztvevő) a saját nézetét fejti ki Szabónak (új tag) ahelyett, hogy kezébe tenné a dokumentációt.

Azt, aminek éppen a magánértelmezések tág lehetősége miatt a vitathatatlan és közös **etalon** szerepet kellene játszania. A tervezetek kapcsán nem ritkán hallani ilyen kitételt: „Az X úgy képzelte, hogy, de nem kell azt komolyan venni...”. Ez a magatartás gyakran odáig fajul, hogy a logikai tervet készítő valaki semmisnek tekinti a fogalmi szintű tervet, aminek az 'áldásos' következményeit még akkor sem ismeri el, amikor a fizikai szintű tervet fejlesztő ugyanezt teszi az övével. A mélyebb szintű tervekben ugyanis sohasem megengedett az ártértelezés. Csak arra van lehetőség, hogy a sokféle megoldási mód közül a célszerűt kiválasszuk.

Végül - de nem utolsó sorban - a dokumentáció nemcsak a megvalósításnak, hanem az **elemzésnek** is az alapja. Értelmes közegben a 'hosszas viták' abból is fakadnak, hogy érvényesíteni akarják a kritériumrendszert. A mércék összetett módon függenek össze egymással és olykor egymás ellen hatnak. Az egyiket ma fontosabbnak tekintjük a másikinál, de holnapra már ez az elsőbbségi sorrend (a prioritás) is változhat. Az viszont biztos, hogy az egyetlen aktuális és érvényes dokumentáció hiányában a változó természetű elemzés is csődöt mond.

A jó dokumentáció nem csak azt írja le, hogy mi az aktuális elképzelés szerint az IR terve ('mit'). Az emberek feledékenyek: még korábbi döntéseik indokaira sem emlékeznek. Mérlegelnek jobbra, mérlegelnek balra, majd döntést hoznak. Nem telik el két hét, és máris elfelejtik a korábbi érveket. Így minden kezdődik előlről: miért is ilyen, miért nem olyan módon csináljuk? Éppen ezért az okos dokumentáció a közös megfontolások okait - a 'miértet' - is rögzíti.

Az alapvető célok megvilágítása után térhetünk át a voltaképpeni tárgyunkra: a dokumentáció kezelésére.

A dokumentációkban felsorakoztatott, a rendszer tervére vonatkozó ismeretek remélhetőleg maguk is szervezett együttest alkotnak. Számos felhasználója van ennek az ismerethalmaznak. Magában a dokumentálásban is események szerint hajtunk végre tevékenységeket - meghatározott szabványok szerint. Eszközöket veszünk igénybe a dokumentálásra. Ergo: az IR rendszer dokumentációja maga is rendszer. Itt nem kívánunk külön meghatározást adni, de azt feltétlenül tudni kell, hogy a **dokumentációkezelés** e másodlagos rendszernek az elrendezését, szervezett együttesként való felfogását jelenti. A dokumentáció nem pusztán - képletes értelemben vett - papír. Ismeretekről szóló ismeret, azaz metaadatbázis.

Az pedig világos, hogy a dokumentációnak megfelelően kell tükröznie magát az alapkoncepciót; struktúrájában és fogalmaiban az IR célszerű szerkezetét kell, hogy kövesse; a dokumentáció egy adott megfogalmazási módban készül; nem csupán az első 'ötletroham', hanem az elemzett kritériumok alapján; végül egyes fejezetei a tervezési algoritmus megfelelő lépésében születnek meg. Tehát az IR fejlesztési módszerének ez a kelléke (is) összhangban kell, hogy álljon az összes többivel.

A fentiek dacára struccpolitikára vallana, ha azt hinnénk, hogy nincs is önálló - a terv tartalmától független - élete a dokumentációnak. Milliószer előfordul az, hogy a már működtetett információs rendszeren úgymond 'apró' változtatásokat hajtanak végre anélkül, hogy ezeket átvezetnék a dokumentáción. Megfordítva: a fejlesztőnek támad egy 'jobb' részötlete, és azt ráadásul le is írja. Csak éppen nem létezik semmilyen olyan mechanizmus, ami garantálná, hogy ez az újabb elgondolás a működtetett információs rendszerben meg is valósul és úgy, hogy...

A dokumentáció-kezelésnek az lenne a feladata, hogy biztosítsa a tervezés alatt álló, a megtervezett, a bevezetett, a működtetett illetve a megváltoztatott IR tényeinek és az azokról alkotott képeknek az egységes, harmonikus voltát. A teljes harmónia persze elérhetetlen váagnak, megvalósíthatatlan álomnak tűnik. Senki sem látott még kis hazánkban egy olyan épületi műszaki rajzot, amely a kivitelezés során egy az egyben megvalósult volna. Megfordítva: amikor e mű szerzője a lakását sajátkezűleg átalakította, eszébe sem jutott a 'blue-print' - a tervrajz, mint dokumentáció - átpofozása.

Egyelőre nincs értelme annak, hogy a dokumentáció és a valóság legteljesebb harmóniájára törekedjünk. Erre még nincsenek is meg a kellő eszközeink illetve módszereink. Azt viszont látni kell, hogy az IR és a lakás között van némi - nem is csekély - különbség. Az utóbbi egy-, az előbbi sokfelhasználós. Ha a szerző a lakását netán eladja valakinek, akkor a dokumentáció és a valóság közötti eltérés nem igazán lesz zavaró. Ezzel szemben az információs rendszernek olykor ezer, olykor akár millió felhasználója is van. A dokumentáció és a valóság harmóniája ekkor nélkülözhetetlen. Tessék csak a mikrogépes programok olyan garmadájára gondolni, amelyekben a dokumentáció vagy hiányos, vagy megtévesztő!

4.8 TECHNIKAI TÁMOGATÁS

A huszadik század végén a világ legtermészetesebb dolga, hogy mindazokat az ismereteket, amelyek bonyolultnak fűggenek össze egymással, számítógépen tároljuk, kezeljük és elemezzük. Az információs rendszer terve maga is összetett ismerethalmaz, ezért logikus, hogy azt is számítógéppel menedzseljük. Ma már nem az a kérdés, hogy ez lehetséges-e, hanem az, hogy milyen a jó megoldás?

Már a hatvanas évek közepétől fogva arra kényszerítettük a számítógépeket, hogy az IR bizonyos tényezőit illetve azok összefüggéseit automatikus módon dokumentálják (vö. előző pont). Már akkor is alkalmaztunk részmodszereket, de módszertanilag megfontolt és kielélt megoldásaink még nem léteztek. Az előző pontokban felsorolt módszertani kellékeket csak nagyon korlátozottan fogtuk fel. Nem az IR teljes fejlesztése, hanem a számítógép alkalmazása jelentette akkor az alapkoncepciót. Nem ismertük az IR struktúráját és - finoman szólva - némi változatossággal alkalmaztuk a terminológiát. A megfogalmazási mód például a lyukkártyás rögzítésre szorítkozott. Csakis technikai, nem pedig informatikai mérce volt mérlegelésünk alapja. Tervezési algoritmust vagy nem ismertünk, vagy az első - mereven biztonságos - koncepciót fogadtuk el. A dokumentáció pusztán papírmunkát jelentett.

Szépen, fokozatosan a módszertani kellékek fegyvertára lassan kibővült. Ma már nem a tehetetlenség jelenti a legfőbb gondot. Éppen ellenkezőleg: a rajzos, ikonos, zenés, gyorsan működő eszközök hada áll rendelkezésünkre. Ezért a volt technikai képtelenség helyett (vegyük számba, hogy mindez egy pár rövidke év alatt zajlott le) manapság éppen az a fő probléma, hogy a technikai lehetőségek garmadája ringatja el a fejlesztés résztvevőit. Az 'elringatás' lényege pedig az, hogy az eszközökben, nem pedig a módszerekben keressük a megoldás kulcsát. Magyarul: eszközmániában élünk, miközben a módszerre nem figyelünk eléggé. Többek között azért nem, mert az eszközök mögötti módszerek messze-messze az előbbieket mögött kullognak. Képletes hasonlattal élve: vettünk egy igazi, egy csodás televízió-készüléket, de még ma sem tudjuk, hogy mit árt és mit hasznos abban nézni, mert ebben a tekintetben nincs kire hallgatnunk.

A CASE [Computer Aided Systems Engineering - számítógéppel támogatott rendszertervezés] névvel illetnek legújában minden olyan programcsomagot, amely ilyen-olyan módon alkalmas arra, hogy az IR tervének valamely részletét számítógépen határozzák meg. A nem a fennkölt elvekre, hanem a mindennapi materiális gyakorlatra ügyelő valaki is felfigyelhet arra,

hogyan az egyik úgymond CASE-t pár tízezer forintért, a másikat több százezer dollárért árulják. (Persze az ár önmagában nem mutatja a valódi képességeket, és olykor csak piaci helyzetet tükröz, amit szintén nem árt tudni.)

A zűrzavaros helyzetet jól jellemzi, hogy egyes program-mániákus valakik a CASE betűszóban az 'S' betűt rendszer (system) helyett ma is programcsomag (software) módjára fordítják és értelmezik. Az információs rendszert szoftverre korlátozni óhajtó eme 'ösbölnyek' - szegények máshoz nem értenek - szemlélete az alapkoncepció (vö. 4.2) helytelen felfogásáról árulkodik. Az is igen tipikus, hogy utólag húzzák rá a roppantul korlátos valahányadik generációs eszközre [például 4GL, fourth generation language, **negyedik generációs nyelv**] a CASE sokkal tetszetősebb - divatosabb - maszkját.

Ne tévedjünk meg az álcák láttán! Vizsgáljuk meg az alapkoncepció mögötti tartalmat is! Megengedi-e a CASE az összetett - fogalmi, logikai, fizikai szintű - struktúrák meghatározását és jól használja-e a terminológiát? A megfogalmazási mód nem attól kényelmes-e, hogy a struktúra-készlet szegényes? Erősen tipikus, hogy a CASE a grafikai megoldásaival villog, csak éppen a képernyőn ide-oda cipelhető, megforgatható, kivágható stb. rajz szimbólumai képtelenek a valóság hű tükrözésére. Kérdés az is, hogy milyen kritériumokat, tervezési algoritmust, dokumentálási módot támogat a technikai eszköz? E ponton két történetet is el kell mesélnünk.

T 4.12 1990-ben egy svédországi CASE-konferencián a szerző a folyosón felsorakoztatott eszközök mindegyikét 'kiakasztotta'. Egészen egyszerűen olyan mintapéldákat diktált be a CASE forgalmazói számára, amelyekről előre tudta, hogy azok csalafinta hibáira az eszközök kritériumrendszerében nincs megoldás.

Csúnya dolog volt ez? De még mennyire! És illő dolog az, hogy egy százezer dollárért árult CASE még az egyszerű adatciklusoknak a tervben való feltárására sem képes???

T 4.13 1992-ben a szerző egy relációs rendszer CASE-eszközét próbálta használni. Egy egészen aprócska - mindössze harminc tényezőt felölelő - rendszerrész dokumentációját a CASE írd és mondd 180 oldalas 'dokumentációban' adta ki!

Mi történt volna, ha a több ezer tételes valódi tervét dokumentálta volna e CASE segítségével? Valószínűleg nem lett volna szükség a nagymarosi gátra.

A számítógépes nem program-, hanem információs rendszerfejlesztés ma még csecsemőkorban van. Azért, mert az ikonokra, a kezelésre - és nem a tartalomra - helyezik a hangsúlyt. Ez a helyzet változni fog. Itt és most csak annyit tudunk leszögezni, hogy jól teszi az, aki figyel a CASE csacsiságaira. Hibázik viszont az, aki nem ügyel a számítógéppel támogatott fejlesztés lehetőségeire.

4.9 METASZABVÁNYOK

A 4.1 pont végén található felsorolásban az itteni alcímben jelzett tényező nem szerepelt. Azért nem, mert nem helyezendő egy sorba a többivel, hanem mintegy azok fölött áll. A 2.1 meghatározásban az információs rendszer különálló, fontos tényezőjeként jelöltük meg a szabványt. Ott az ún. **felhasználói** szabványokról volt szó. Azon megegyezésekről, amelyek a felhasználói adatokra, eseményekre, tevékenységekre stb. vonatkoznak. Azonban a fejlesztést magát is rend-

szerként kell szemlélni. Vannak fejlesztési adatok, események, tevékenységek stb. Ezeket az ún. **fejlesztési** szabványok szabályozzák, amiket metasabványoknak hívunk.

A fejlesztési konvenciók köre igen tág. Az alapkoncepciótól kezdve az esetleg alkalmazott technikai eszközökig felöleli a fejlesztési módszer mind a hét kellékét. Így a tervezési algoritmust is. Ezért nemcsak a felhasználói, hanem a fejlesztési szabályokat is szabványok és szabványos eljárások együtteseként kell tekinteni.

A megegyezések a részletekre és az átfogóbb dolgokra egyaránt ki kell, hogy terjedjenek. Számunkra meglepő, hogy egy-egy vállalatban minden fejlesztő a saját szokásai szerint ad neveket az adatoknak; az egyik 1-2-3 kódokkal, a másik a B-M-T betűkkel jelöli a bevitel, módosítás és törlés műveleteit; a képernyőn hol az F10, hol az Esc, hol az Abort jelenti a megőrzés nélküli kilépést stb. De folytathatnánk a sort azzal is, hogy az egyik szervező így, a másik úgy készíti el a dokumentációt. Magyarul és összefoglalóan: a fejlesztésben iszonyatos káosz uralkodik a metasabványok hiánya miatt. Ez a megállapítás éppen úgy érthető egy nagyvállalatra, mint arra az emberkére, aki számítógépet használ, de hol így, hol meg úgy rendezgeti el a dolgait.

A 'rendszer' szót ma annyian, de annyian használják. Anélkül, hogy e kifejezés által sugallt lényegre - **REND** - valamit is figyelnének...

Ezt az igen rövid pontot mindjárt le is zárjuk. A módszer-alkotónak a feladata az, hogy felhívja a figyelmet a szabványosítandó/-ható tényezőkre. A módszer-alkalmazóra hárul az a kötelesség, hogy a konkrét szabványokat ki is dolgozza. Mivel a metasabványok sora szinte végtelen, itt és most nincs alkalmunk arra, hogy velük behatóbban foglalkozunk. Csak a szükségességükre hívhattuk fel a figyelmet.

T 4.14 Egy rendszerterv magyarázata során kollégám így szólt: „Tudod, nekem az a konvencióm, hogy...”

Ez a történet az IR lényegének a teljes meg-nem-értéséről tanúskodik. Ugyanis a konvenció - közös megegyezés. Kollégám 'megegyezett önmagával'. Mi pedig úgy véljük, hogy a (rossz) egyéni szokások nem konvenciók, nem szabványok.

Végül legyen szabad elmesélnünk egy mosolyra késztető esetet:

T 4.15 Valahol be kellett diktálnom a foglalkozásomat. Első ötletként azt találtam mondani, hogy rendszerszervező vagyok. A hölgy ezt meg is értette. Azóta a papíron ez áll: „rendszerző”.

Nem javítottam ki. Azért nem, mert az informatikusnak ez a munkája. Nem a rendszertervezés, nem a rendszerszervezés, nem a rendszerszerzés - hanem a rendszerezés.

4.10 MÓDSZERTANI ÖSSZEFOGLALÓ

Az előző pontok nyilván meggyőzték az olvasót arról, hogy az IR fejlesztése és annak módszere meglehetősen összetett dolog. Ez már akkor is így igaz, ha a végső-felhasználó az IR egyedüli alkalmazója, netán saját maga a fejlesztő is, és kizárólag csak személyes használatra óhajt fejlesztette(ni) egy kis rendszert. Ámde akkor, ha az IR több-felhasználós, sőt: több-fejlesztős, akkor nyilván a gondok is megsokszorozódnak.

A fejlesztés minden résztvevőjének - akkor is, ha a 'minden' csak egyet jelent - helyes **alapkoncepcióval** kell rendelkeznie. Az IR helyes szemléletmódja nem korlátozódhat a szoftverre

ill. - kicsit tágabban - a számítógépre. Az információs rendszerbe tartoznak a gépen kívüli rendszertényezők is. Ezért a gépi és a kézi adatokat, eseményeket, tevékenységeket stb. csakis egységben szabad szemlélni.

A fejlesztések sikertelenségének az egyik alapvető oka az, hogy a résztvevők nincsenek az IR világos és egyértelmű **strukturális** képének a birtokában. Így az általuk látott szerkezet többnyire korlátozott. A felülről-és-kívülről való nézetmód helyett a rendszereket alulról-és-belülről, sokszor az alkalmazott eszközök által behatárolt módon közelítik meg. A gyakori váltások és az eszközök sokfélesége miatt az informatikára ma még a **terminológiai** káosz jellemző. Vagy nincsenek is pontosan definiált fogalmaink, vagy az egzaktnak meghatározottakat nem azok eredeti értelmében használjuk.

Az IR-ek fejlesztése mindig valamilyen segédletekkel történt. Öröndetes az a jelenség, hogy egyre tudatosabban alkalmazzuk magát a számítógépet is ebben a folyamatban. A tervek **megfogalmazási módja** így kedvező irányba változik. Az ábrákon, számítógépes tervezési képernyőkön alapuló megfogalmazás egyrészt gyorsabb és kényelmesebb, mint a hagyományos papíralapú tervezés. Másrészt a tervek belső és egymás közötti konzisztenciája, összhangja magasabb fokú lehet. Arra viszont ügyelni kell, hogy ne a segédeszköz vegye át a végcél helyét, mint az ma még több helyen megtörténik.

A végcél nem pusztán egy megvalósítható, hanem a lehető legelegánsabb terv kidolgozása. A terv eleganciája optimalitásában rejlik. A jó **kritériumrendszer** a valósághű, érthető, egyértelmű, minimális és teljes IR megalkotásának a kulcsa. Rendszerről kell beszélnünk, mivel a mércék bonyolult módon függenek össze egymással. A tervezés nemcsak magának az elgondolásnak a rögzítését jelenti; annak nélkülözhetetlen mozzanata a terv optimalitását biztosító elemzés is.

A fejlesztési projekt tervezési szakaszának részlépései az IR egyes tényezőinek a kidolgozására irányulnak. Külön feladat az adatbázisnak, a bemenetnek illetve a kimenetnek, a feldolgozásnak a megtervezése és e tényezők összehangolása. A **tervezési algoritmus** az a követendő eljárás, amely eligazít e résztvevő tevékenységek lényegét illetve azok sorrendjét illetően. Nagyon sokféle tervezési eljárás létezik a tervezési orientáció (adatbázison, bemeneten, feldolgozáson alapuló tervezés), a kísérletezés szerepe és a lépések meghatározottságának szigora (rugalmas vagy merev algoritmus) szerint. Elméletileg közömbös, hogy melyiket alkalmazzák és az is gyakori, hogy a többféle megoldási módot egy projekten belül is vegyítik. Sokan a megoldási módra koncentrálnak, holott nem ez a résztechnika, hanem a végső tervtermék minősége kellene, hogy a figyelmünk központjában álljon.

„A szó elröptül, az írás megmarad.” Nem sokat ér az a terv, amely csak egyes fejekben létezik. A rendszer tervét minden fejlesztési résztvevő számára elérhető módon kell dokumentálni. A dokumentáció ugyanis nem pusztán etalon, vagyis nemcsak a megvalósításhoz szükséges tervezet, minta, hanem az ismerkedésnek, a résztvevők közötti kommunikációnak az eszköze is. A **dokumentáció-kezelést** ezért úgy kell kialakítani, hogy a terv ne maradjon véka alá rejtett gyertya.

A fejlesztést **segédeszközök** támogatják, amelyek korszerűbb megfogalmazási módokat biztosítanak, mint a hagyományos manuális tervezés. Az eszközöknek azonban nem csak az a feladata, hogy megkönnyítsék a fejlesztő munkáját. Sőt nem is csak az, hogy szerepet játszanak a dokumentáció-kezelésben. Amikor a segédeszközt megvásároljuk és alkalmazzuk, mérlegelnünk kell azt is, hogy ez a technika mennyiben támogatja az optimum-kritériumok érvényesítését. Sajnos ma még az érdeklődés sokkal inkább a tetszetős és kényelmes felületre, semmint a szigorúan rugalmas dokumentáció-kezelésre illetve az optimalizálásra irányul.

Az információs rendszerek fejlesztésében két módszer-fonal váltja egymást: az **irányítási** és a **tervezési** módszer. A gyakorlatban ezek szorosan összefonódnak. Egymást kíséri a tervrészlet mint tervtermék és a hozzá kötődő adminisztrációs papírok garmadája. Ez így helyes. A probléma az, hogy a fejlesztési résztvevők nem ismerik e két fonal eltérő természetét, lényegét, ezért a hangsúly olykor pl. a kritériumok vizsgálata helyett az 'elintéztük' jellegű papírmunkára tevődik át.

E bajok gyökere a *módszertan* hiányában található. A *módszer-alkotók* maguk sincsenek tisztában az IR-fejlesztési módszerek lényegével. Ez abban nyilvánul meg, hogy az ebben a fejezetben felsorolt hét módszertani kellék némelyikével egyáltalán nem törődnek, másokat pedig helytelenül ill. korláatosan szemlélnek. Ezért a gyakorlati *módszerek* maguk is féloldalasak. A fejlesztés résztvevőinek, vagyis a *módszer-alkalmazóknak* csak konkrét módszert tanítanak, módszertani ismereteket nem. Ezért azok észre sem veszik az előbbieket korlátait, és meg sem kísérlik, hogy az elsajátított módszer hézagait valamilyen módon betömjék.

Ennek a fejezetnek a fentiek miatt éppen az volt a célja, hogy kitágítsa az IR fejlesztésében szereplő személyek szemléleti horizontját függetlenül attól, hogy fejlesztőről vagy felhasználóról van-e szó. Ugyanis a módszertani kelléktár jó ismerete a sikeres fejlesztési közreműködés egyik alapvető feltétele.

ELLENŐRZŐ KÉRDÉSEK - 4

A lényeg az, hogy az olvasó megértse a hét-plusz-egy módszertani kellék lényegét és viszonyát. A kellékek mindegyike szükséges. Arra törekedtünk, hogy őket fontossági sorrendben mutassuk be - a többit átfogó metaszabvány kivételével. Az olvasó gondolkodjon el a sorrenden.

- 401 Ön saját használatú számítógépet vásárol. Mi lenne a legelső teendője? Az alábbiak közül egyetlen tétel kiválasztása a feladat.
- 1 - beszerezné a Norton Commandert
 - 2 - elgondolkodna az alkalmazásai lényegén
 - 3 - felkérne egy szakembert néhány program megírására
 - 4 - telepítené a Microsoft termékeit.
- 402 Az Excel-ben is táblázatnak nevezik az ismeretek együttesét, a relációs adatkezelőknél is használják ezt a fogalmat. Próbálja saját szavaival elmondani, hogy milyen problémát lát. A 'reláció' lényegét a fentiekben elmagyaráztuk.
- 403 Próbálja meg igazolni a szerzőt! Mondja el, hogy a megfogalmazási mód miért került a listánkban ilyen előkelő helyre, pl. a kritériumok elé.
- 404 Milyen kritériumokat sérthet az, ha Ön több eltérő ismeretegyüttesben is a név, cím, darab, mértékegység, azonosító stb. adatneveket alkalmazza?
- 405 Ön egy anyagnyilvántartást óhajt vezetni. Az események függvényében lesznek bemenetei, kimenetei, feldolgozásai, tárolt adatai, azokat kezelő eszközei stb. Ön mit tekintene e fejlesztés 'kályhájának'?
- 406 A dokumentáció több célt szolgál. A szerzőt a minap meghívták egy olyan értekezletre, amelynek az adatbázisterv 'kivesézése' volt a célja. „El tudnál jönni?” - ennyit tartalmazott a meghívó. Csak egy levelet: és mást nem. Miről feledkeztek el a meghívók?

- 407 Aransárga ikonokkal választhatom ki az 'objektumokat' a tervezési segédeszköz képernyőjén. Megteszem. Azt a szívességet is, hogy leírjam - mások helyett és utólag - az X adatbázis tervét. Ma az adatbázis még csak távolról sem hasonlít az általam leírthoz. Viszont véletlenül megláttam egy másik tervét, amit ugyanazzal az eszközzel készítettek. Mi hiányzik? Érveljen a saját szavaival!
- 408 A nyegle számítástechnikus szerint magánügye, hogy az adatot az adatbázisban Személyi_száma, A1, Sz_személy, Szemszáma vagy más néven nevezi-e. Az informatikus szerint ez nem magánügy. Ön miként gondolkodik? Érveljen!

5. AZ IR ADATRENDSZERE

Az információs rendszer legfontosabb tényezője az ember. Azonban az IR-ben az ember nem ebben az *általános* minőségében szerepel. Testi-lelki emberként a valós rendszernek a része. Az IR-ben a helye *különleges*, vagyis nem abszolút. Az ismeret tágértelmű felhasználójaként, az ismerettel való viszonyában lép fel. Így „Az ismeret maga az ember.” alapon kijelenthetjük, hogy az IR legfontosabb célja az ember, viszont ennek nem ellentmondóan a központi eleme az ismeret.

Az információs rendszeren belül az ismeretek saját rendszert alkotnak. Ez az utóbbi lényeg - az **adatrendszer** - maga is rendkívül összetett. Lehetetlen dolog arra vállalkozni, hogy a vele kapcsolatos könyvtárnyi tudást egyetlen fejezetbe sűrítsük. Az ismeret számos aspektus szerint osztályozható és vizsgálható. Ezek közül a számítástechnikusok és a számítógép felhasználók a gépi megjelenítés - tárolás és kezelés - formai és technikai szempontjait emelik ki. Sokkal kevesebb figyelmet szentelnek az ismeretek és kezelésük *természetének*. Így pl. annak, hogy az ismeret a valóság tükörképe; továbbá a számítógépen tárolt és kezelt adat elválaszthatatlan a mindennapos kommunikációtól. Márpedig ha valaki nincs tisztában a kommunikáció és a tükrözés lényegével, akkor vajon miképpen tud jó információs rendszert készíteni?

Az informatikust a megvalósításnál (is) jobban érdekli a feladat. Nem a gépet 'kell' használni, hanem az eszköz segítségével egyértelműbb, jobb és - igenis - szebb ismeretekkel kell ellátni az embereket. Ez az igazi kihívás! Nem technikai varázslatokkal - bár azok is lehetnek hasznosak - kell elbűvölni a felhasználót, hanem használhatóbb ismeretekkel. Ehhez pedig tisztában kell lenni egyrészt az adatrendszer felépítésével, másrészt az ismeretkezelés módjaival.

Ennek a fejezetnek a célja kettős. Egyrészt az, hogy feltárja az ismeretek természetét és erre alapozva bemutassa az adatbázisok mint adatrendszerek elvi felépítését. Másrészt az, hogy a megoldás helyes kiválasztása érdekében kifejtse az ismeretkezelés lehetséges alapvető módjait.

Az információs rendszer tényezői összefüggenek egymással. Az adat nemcsak a felhasználóhoz, hanem az IR másféle elemeihez is kapcsolódik. Most az adat-felhasználó viszonyra koncentrálunk. A következő fejezet tárgya az események és a tevékenységek rendszere. Mivel ezek a tényezők is az ismerethez kötődnek, az a fejezet további (kötelező) alkalmat nyújt a számunkra, hogy az adatrendszer egyéb aspektusait is áttekintsük.

5.1 AZ ISMERET DIMENZIÓI

Korábbi meghatározásunkban (D 2.2) leszögeztük, hogy az adat értelmezhető, de nem értelmezett ismeret. Az érzékelés, az észlelés és a felfogás momentumait jelöltük meg az értelmezhetőség követelményeiként. Azonban az előbbieken túl az ismeretek befogadásának vannak egyéb feltételei is. Gondoljunk csak pl. arra, hogy *az ismeretek nem függetlenek egymástól*. A mindennapos beszélgetéseink során a mondanivalóinkat egybefűzzük; az egyik mondatunkra épül a másik; az adott/kapott ismeret végül is a teljes gondolati láncnak a gyümölcse.

Az információs rendszerben az ismereteket valamilyen módon formalizáljuk. Kitöltendő bizonylatok rovataiként, számítógépen tárolt állományok mezőiként, képernyőn megjelenő ‘dobozokként’, nyomtatott kimenetek adataiként látjuk azokat. Az ismeret így *mesterséges* képet ölt, és elveszíti *természetes* képét. Mi több, hajlamosak vagyunk arra, hogy az ‘információt’ előbb-utóbb annak az előbbi - mesterséges - formájával azonosítsuk. „Az ismeret természetes színét a számítógép halványra betegíti.” - mondhatnánk, kiforgatva a nagy informatikus előd szavait.

Pedig a mesterséges nyelv gyökerei a természetesben rejtőznek. Roppant nagy kár, hogy az IR fejlesztői erre a tényre nem figyelnek eléggé. Reméljük, hogy e pont végére a mostani rébusz megvilágosodik.

Mondanivalónk megvilágosításához induljunk ki két egyszerű közlésből:

„A BMZ 873 rendszámú gépkocsi színe piros.”

„Az FGS 802 rendszámú gépkocsi típusa FORD Escort.”

Minden tényt közlő - vagyis kijelentő - mondatnak van *alanya* és *állítmánya*. Az előbbi a közlés tárgyra, az utóbbi a róla szóló mondanivalóra utal. Erre az alapvető nyelvtani tételre még mindenki emlékszik. Senkinek sem okoz gondot, hogy két példamondatunkat alanyra és állítmányra bontsa a következő módon:

5.1 tábla

ALANY	ÁLLÍTMÁNY
A BMZ 873 rendszámú gépkocsi	színe piros.
Az FGS 802 rendszámú gépkocsi	típusa FORD Escort.

Arra már valószínűleg igen kevesen emlékeznek, hogy az alany megadásához, vagyis mondanivalónk tárgyának a kijelöléséhez két dolog szükséges. A logika szabályai szerint hivatkoznunk kell a ‘*legközelebbi nem*’-re [genus proximum] és meg kell adnunk a ‘*megkülönböztető jegy*’-et [differentia specifica].

Persze a mindennapi közlések során bele is örülnénk, ha ezekre a dolgokra így kellene figyelni. Ettől függetlenül igaz, hogy az ismeret általános tárgyát fel kell fogunk a közlés értelmezéséhez. Példamondataink nem könyvekről és nem személyekről, hanem gépkocsikról (legközelebbi nem = gépkocsi) szólnak. Nem egyről, hanem kettőről, mégpedig a rendszám tartalma által meghatározottakról (megkülönböztető jegy = rendszám).

Ez a kettősség nemcsak az alany, hanem az állítmány esetében is látható. Az első példában a színre (legközelebbi nem) utaltunk, majd megadtuk a konkrét színt (megkülönböztető jegy). A második példában hasonlóan jártunk el, csak éppen a típust határoztuk be pontosabban. Ezért a fenti kis ‘táblázatot’ akár át is alakíthatjuk a következő módon:

5.2 tábla

ALANY		ÁLLÍTMÁNY	
Megkülönböztetés	Nem	Nem	Megkülönböztetés
A BMZ 873 rendszámú	gépkocsi	színe	piros.
Az FGS 802 rendszámú	gépkocsi	típusa	FORD Escort.

Erről a kis táblázatról rengeteg dolgot kell elmondanunk. Legelőször is azt, hogy *a sikeres közlés feltétele kétszer-kettő*. Meg kell adni mindkét mondatrész (alany és állítmány) mindkét tényezőjét (nem és megkülönböztetés). Másodszor azt, hogy az első tételünk akkor is igaz, ha a

mindennapok során másként látjuk a dolgokat. Minden további nélkül mondhatjuk, hogy ‘Ez a kocsis FORD Escort.’ és ráadásul azt is, hogy ‘Ez piros.’, mi több, csak ennyit: ‘Piros.’. Vegyük észre, hogy az ember mindennapi tudatában a dolgok eleve összekapcsoltak. Tudjuk, hogy a ‘piros’ az nem a kocsis típusa, viszont a ‘FORD Escort’ nem a színe. (NB.: Ezt tudja a jártas felnőtt, mert vannak összekapcsolt ismeretei. És a kisbaba...?) Azt se feledjük el, hogy az ‘Ez a kocsis...’, sőt az ‘Ez...’ szintén behatárolás akkor is, ha nem szóban, hanem mozdulatban (rábökünk), képen stb. utalunk a közlés tárgyára. Végül az emberi kommunikáció-sorozatban nem ismételtetjük a tárgy hivatkozását. ‘Lajos kocsija FORD Escort. Színe kék.’ A második mondat már nem tartalmazza a ‘Lajos kocsija’ behatárolást, mert az már adott.

Harmadik mondanivalónk az, hogy a ‘kétszer-kettő’ valamelyik elemének az elmaradása egyértelműségi hiányokat okoz. Lássuk csak a következő példát!

„A piac sarkán a rózsa ára X forint.”

Ezt a mondatot nem látni, hanem csak hallani kellene. Mivel hiányzik belőle a legközelebbi nem - és szóban a nagybetűk sem emelődnek ki - a közlés számos félreértésre adhat okot. A nyájias olvasó nyilván virágra gondol. A csintalanabb a magyar nyelvtudásomat vonja kétségbe (nem teszünk ‘a’-t a személynevek elé, így Rózsa elé sem). Pedig engem csak az asszony küldött le - krumpliért.

Lehet más próbákat is tenni, ámde be kell látnunk, hogy kétszer-kettő - négy. Az alábbi három ‘süket’ példamondat igazolja, hogy nemcsak az alany nemének, hanem megkülönböztetésének ill. az állítmány valamelyik részének a hiánya is kommunikációs zavart okoz.

„A gépkocsi színe piros.” - Melyiké?

„A BMZ 873 rendszámú gépkocsi 1300.” - A súlya? A motorja?

„A BMZ 873 rendszámú gépkocsi típusa.” - No és mi az?

Negyedik mondanivalónk egyszerű. Az ember gondolatfűzéssel, rámutatással és ezernyi más eszközzel tudja pótolni a szavak hiányát. Az ismeretek kétszer-kettő tényezője közül bármelyiket. Azért képes erre, mert agyában eleve el van rendezve a négy dolog. A számítógép ‘agyát’ viszont fel kell világosítani. Még akkor is, ha egérrel mutogatunk az ismeret valamelyik kellékére a négy közül. A számítógépen tárolt ismeretek elrendezése az alábbi logikai kép szerint történik:

5.3 tábla

ALANY	ÁLLÍTMÁNY	
Rendszám	Szín	Típus
BMZ 873	piros	?
FGS 802	?	FORD Escort

Első példamondataink alapján nem ismerhetjük a piros kocsi típusát illetve a FORD színét. Ezért hiányos ez a táblázat. Hiányzik belőle továbbá a generikus alany megnevezése (gépkocsi). Ez nem baj, ha tudjuk: miről szól a táblázat? A teljes és valós kép persze a következő lesz:

5.4 tábla

GÉPKOCSI		
Rendszám	Szín	Típus
BMZ 873	piros	LADA 2104
FGS 802	bordó	FORD Escort

Ezek után már visszatérhetünk a pont címére, és elárulhatjuk, hogy mi az adat négy dimenziója. Az első a **generikus alany** (GÉPKOCSI), ami nem más, mint a táblázat neve. Annyi táblázatunk lesz majd a számítógépen, ahányféle dolog (gépkocsi, személy, könyv, számla, város stb.) érdekel bennünket. A második a **specifikus alany** (pl. BMZ 873). A fenti kettővel szemben annyi sora lesz majd az igazi GÉPKOCSI táblázatnak, ahány ilyenféle járműről vezetünk ismereteket. A harmadik dimenzió a **generikus állítmány** (Szín). Annyi ilyen oszlopunk lesz a táblázatban, ahányféle gépkocsi-ismeretre vagyunk kíváncsiak. A negyedik dimenzió a **specifikus állítmány** (piros), ami az adott táblázat kijelölt sorában a vonatkozó oszlopnál található bejegyzés. A BMZ 873 Rendszámú GÉPKOCSI Színe - piros.

Mielőtt az olvasó ezer - jogos - kérdést zúdíthatna ránk, két dolgot kell sietve megjegyeznünk. Az egyik az, hogy a legutolsó tábla által mutatott kép **logikai** elrendezés. Egyáltalán nem biztos, hogy a számítógépen az adatok pontosan e képnek megfelelően kerülnek **fizikai** elhelyezésre. A másik az, hogy az ismeret dimenziói nem mindig jelennek meg a számunkra a fenti explicit módon. Ezért a későbbiekben ki kell majd térnünk az ismeretkezelés eltérő módjaira (5.5 pont).

5.2 AZ EGYED ÉS A TULAJDONSÁG FOGALMA

Az információs rendszerben a közlés dimenzióit nem az alany és az állítmány fogalmakhoz kapcsoljuk, bár az informatikusok ezzel is megpróbálkoztak.

Intermezzo

A hetvenes években kísérlek történtek az úgynevezett **szemantikai adatbázis** bevezetésére. Ennek lényege az, hogy az adatbázisban nem adatokat, hanem az angol (!) természetes nyelvhez közelálló (!) nyelvben leírt mondatokat tároltak. Pl.: „Peter lives in London, has two children, is engineer.”, „Paul lives in Paris, has one children, is doctor.”. Ezt az ismeretkezelést azért hívták szemantikainak, mert az az alany és az állítmány(ok) szerint tagolt módon történt. A próbálkozás számos ok miatt megbukott. Egyrészt csak a tört angol nyelvre épült. Másrészt túl sokat kellett írni. Harmadrészt a mondatok kapcsolása nem volt megoldott. A „Peter lives...” és a „Peter has a car” mondatokat vagy összetolták, ekkor nagyon hosszú mondatokat kaptak, vagy szétvágták - akkor lépett fel a kapcsolat hiánya.

A mai ismeretkezelésben a töltőszavakat ('is', 'has' stb.) nem használják. Nevet adnak a jellemzőknek (lakóhely, gyerekek száma, foglalkozás) és ezekhez kell kijelölni a tartalmat (Paris, 1, doctor). Így az adat korábbi definíciója mellett egy új, **szűkebb adatfogalom** született. Szűk értelemben az adat egy jelenség-csoport (személy) egy konkrét jelenségének (Paul) a vonatkozó jellemzőjéhez (lakóhely) tartozó érték (Paris). Így kerekedik ki a négy dimenzió, amelyekre - azért, hogy beszélni tudjunk róluk - sajátos fogalmakat alkalmazunk. Ezek bemutatására fog sor kerülni a következőkben.

D 5.1 Az ismeretekkel leírni kívánt jelenségek fogalmi tükörképét egyedeknek nevezzük.

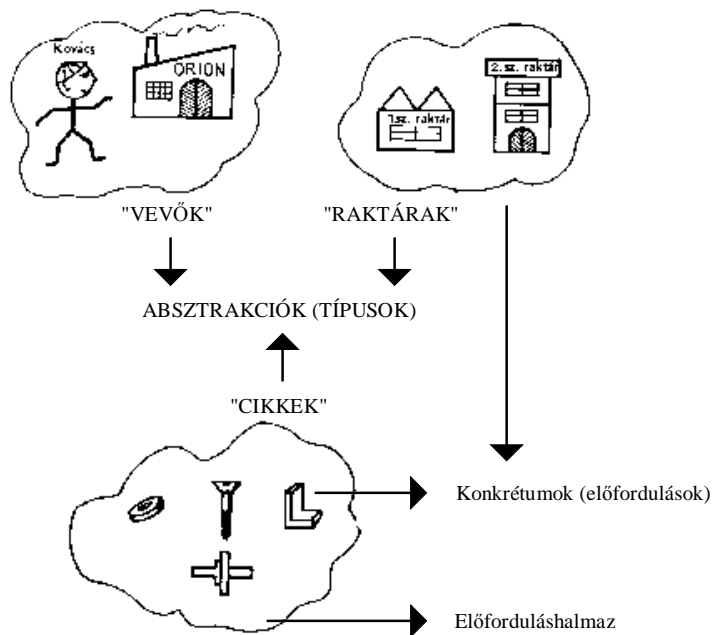
A fenti kijelentésekben Peter és Paul - egyedek, mert őket ismeretekkel akarja valaki leírni. Ez világos, csak a 'fogalmi tükörkép' kitétel okozhat némi zavart. Az olvasó gondoljon az alkatrészekre, például az M6-os csavarra! Minden egyes darab M6-os csavar a szó valódi értelmében

egyed, mert eredetileg ez a fogalom az 'önálló léttel bíró jelenség'-et takarta. Azonban az IR-ben természetesen nem óhajtjuk mind az X (száz)ezer M6-os csavart egyenként leírni, hanem az M6-os csavarok absztrakt lényegét - fogalmi tükörképét - kívánjuk csak megragadni.

D 5.2 Az ismeretekkel leírni kívánt jelenségek osztályainak fogalmi tükörképét egyedtípusoknak nevezzük.

Az ember a valós jelenségeket osztályokba sorolja. Ezt azok közös jellemzői szerint teszi, ám erre a lényegre majd alább térünk ki. Itt azt kell elmondanunk, hogy ez a besorolás **mesterséges absztrakció**, amennyiben mindig a célunktól függ. Ha Peter és Paul egyedek, akkor világos, hogy létezik egy 'személy' nevű egyedtípus mint absztrakció. Az M6-os és M8-as csavarokban közös, hogy ők 'alkatrészek', tehát létezik egy ilyen nevű egyedtípus is. A 'mesterséges' jelző nem 'mesterkélten' jelent. Bár elképzelhető, de mesterkélten a 'dolgok' egyedtípus, ha oda soroljuk Pétert és a természetében tőle különböző M6-os csavart is. Az absztrakció azért mesterséges, mert Péter és Pál érdekelhet bennünket úgy is, mint saját 'dolgozó', úgy is, mint külső 'ügyfél', a kórházban 'beteg' vagy éppen 'orvos' (Pál mindkettő lehet!) stb. Az absztrakció miatt mondjuk azt, hogy az egyedtípus a jelenségek osztályainak a fogalmi tükörképe - és nem maga a jelenség. Hiszen a 'dolgozó', az 'ügyfél', a 'beteg' - egy-egy fogalmi tükörkép.

Attól fogva, hogy megalkotjuk az **absztrakt** egyedtípust (pl. személy), az abba sorolható **konkrét** egyedeket (pl. Peter) **egyed-előfordulásoknak** nevezzük. A vonatkozó alkalmazási környezetben minden ottani egyedtípus az előfordulások (időben változó) **előfordulás-halmazával** rendelkezik. Példánk esetében Peter és Paul alkotja ezt a halmazt.



5.1 ábra: Egyed típus, -előfordulás és előforduláshalmaz

Vegyük észre, hogy az egyed típus a **generikus**, az -előfordulás a **specifikus** alanyak felel meg. Ezért nem is annyira nehéz az egyedek megtalálása. A „Az FGS 802 rendszámú gépkocsi

típusa FORD Escort.” mondatunkban az FGS 802 rendszámú valami a ‘gépkocsi’ nevű egyed-típus egyik előfordulása, ismeretekkel leírandó jelenséggént.

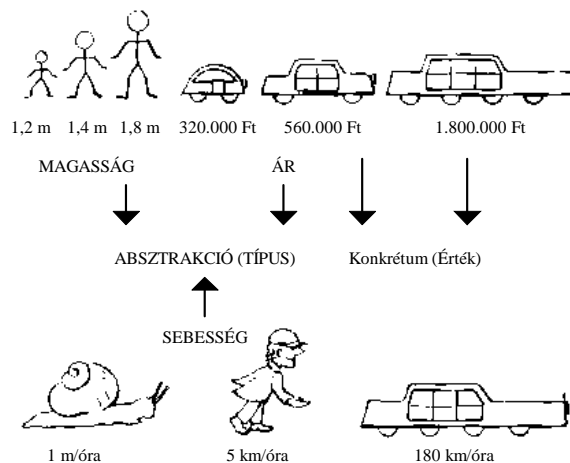
A gépkocsiknak van rendszáma, az embereknek van foglalkozása, de fordítva ez nem igaz. Az FGS 802 rendszámú előfordulás éppen ezért tartozik a gépkocsi egyed-típusba, míg a Paul nevű előfordulás a személy egyed-típusba. A kocsiknak és az embereknek külön-külön vannak olyan sajátos jellemzői, amelyek a másik osztályhoz (egyed-típushoz) nem kapcsolódnak.

D 5.3 Az egyed-típusok jellemzőinek a fogalmi tükörképét tulajdonság-típusoknak nevezzük.

A rendszám, a szín, a típus - a gépkocsi egyed-típus egy-egy tulajdonságtípusa. A személy egyed-típust pedig a lakhely, a gyerekszám, a foglalkozás jellemző írja le. A tulajdonságtípus is absztrakció, mert az ember ad nevet a fogalmaknak. Ezt önkényesen, a céljainak megfelelő módon teszi. A gépjármű egyed-típust a rendszám tulajdonsággal jellemezheti annak ellenére is, hogy a kismotor illetve a kombáj is gépjármű, de az ilyen járműveknek nincs rendszáma.

D 5.4 A tulajdonságtípus (lehetőség) tartalmának a fogalmi tükörképét (tulajdonság)értéknek nevezzük.

A XII., a 12. és a december ugyanazt jelenti, csak másképp ábrázoljuk a hónap tulajdonságtípus tartalmát. Az érték éppen ezért fogalmi tükörkép. A ‘lehetőség’ kitétel magyarázata pedig a következő. A tulajdonságértékek a tulajdonságtípus *előfordulásainak* tekinthetők. A tulajdonságtípusnak is van előforduláshalmaza, amit *érték-halmaznak* hívunk. Idővel ez is változik. Ma még nincs kék gépkocsi az egyedeink között, tehát ma még a ‘kék’ nem a ‘szín’ halmaz része, de holnap már az lehet. Az egyedhez hasonlóan itt is a típus *absztrakt*, míg az előfordulás - vagyis az érték - *konkrét*. Az egyed absztrakt és konkrét jellemzőjéről van szó. A szín absztrakció, mert a pirosból, a bordóból stb. elvont fogalom.



5.2 ábra: Tulajdonságtípus és -érték

Vegyük észre, hogy a tulajdonságtípus a *generikus*, az érték pedig a *specifikus* állítmánynak felel meg. A piros és a bordó specifikus jellemző, viszont maga a szín tulajdonság generikusan jellemzi a gépkocsikat.

A fenti definíciók mindegyike fogalmi tükörképet emlegetett. Ennek az egyik okát már elmondtuk: mesterséges elgondolásokról van szó. Van azonban egy másik megfontolás is, amit alapelveként is rögzíthetünk:

E 5.1 Az egyed és a tulajdonság relatív fogalmak.

Vegyük csak alapul a szín jelenséget! Nagyon sokan - elhamarkodottan - úgy vélnék, hogy a szín az tulajdonság, amely pl. a kocsit jellemzi. Ez ugyan igaz, de éppen az autógyárban a szín számos ok miatt maga is ismeretekkel leírandó jelenség - tehát egyed is. Az 'is' a kulcsszó, ami majd további gondolatoknak ad alapot.

Intermezzo

Még ma is létezik olyan informatikai koncepció, amely szerint viszonylagos voltuk miatt nem szabad egyedben és tulajdonságban gondolkodni. Egyetlen lényeg van csak: az **objektum**. (Ez nem tévesztendő össze a fejezet későbbi részében említett objektum-orientált adatbázis azonos nevű fogalmával!) E koncepció eredeti változatában (vö. bináris relációk modellje) az ismeretek mindig két dolog párosaként jelentkeznek úgy, hogy a viszonyban álló két dolog egymáshoz képest adott szerepet játszik. A gépkocsinak van színe, a szín a gépkocsi jellemzője. A gépkocsinak van típusa, a típus a gépkocsi jellemzője. A koncepció gyakorlatilag megbukott. Azért, mert nem természetes. A természetes nyelvben is használunk összetett mondatokat, és az FGS 802 kocsira vonatkozó közlésünket - 'színe piros és típusa FORD Escort' - nem tagoljuk két mondatba. Továbbá nincs olyan örült, aki Rendszám - Szín, Rendszám - Típus párokként fogja tárolni és kezelni az ismereteit a Rendszám - Szín - Típus hármas helyett.

A viszonylagosság egy másik módon is megnyilvánul. Vegyük észre, hogy az egyedekről képtelenek vagyunk úgy beszélni, hogy meg ne említenénk valamely jellemzőjüket. Ezzel a felvetéssel pedig eljutottunk az azonosítás tárgyköréhez.

A logika szabályai szerint az azonosításnak kétféle módja van. **Nominatív** az azonosítás akkor, ha a szóbanforgó dolognak az egyetlen és jól behatároló nevét adjuk meg. A 'jól behatároló' itt azt jelenti, hogy a név a jelenségcsoport minden egyes tagjára más és más. A neven pedig itt ne tessék tulajdonnevet érteni! Név például a rendelésszám is, mert hiszen minden rendelésre nézve más az értéke, tehát általa a rendelést 'meg lehet nevezni'.

A másik azonosítási mód a **deskriptív**, amikor a jelenség konkrét jellemzőit addig soroljuk fel, ameddig a dolgot egyértelműen be nem határoltuk. Egy példa jól mutatja a lényegét. Ha a személy-azonosítóval hivatkozunk valakire, akkor az előbbi - nominatív - azonosítással élünk. Ha pedig felsoroljuk a nevét, születési dátumát és helyét, édesanyja nevét és - ha még szükséges - a lakcímét is, akkor a leíró azonosítási módot használjuk.

A mindennapi életben az utóbbi megoldást gyakran alkalmazzuk. Figyeljük ki két hölgy beszélgetését a közértben: „Nem jut az eszembe a neve; tudod annak a szőkének gyereke született.” „Melyik szőkének, a Pista feleségének?” „Nem, na segíts már, akik itt a szomszédban laktak.” „Ja, azok a felvágósak a nyaralóval?” „Igen-igen, és tudod...”

Természetesen az informatikában az effajta azonosítás nem célszerű, ha nem is kizárt egyes ad-hoc kereséseknél, például egy nyomozás során. Újabban gyakran alkalmazzák a deskriptív azonosítást az on-line lekérdezésnél is, vagyis a képen megjelenő jellemzők némelyikéhez pár karaktert beütve (Név = 'Pet', Lakcím = 'Lon' stb.) tallózva azonosítják a londoni Petert. A mai adatkezelési technikáknál általában viszont elkerülhetetlen a kvázi-nominatív azonosítás. A 'kvázi' jelzőt rögvest megmagyarázzuk.

D 5.5 Azonosítónak nevezzük azt a tulajdonságot vagy tulajdonság-együttest, amely az adott egyedtípus minden előfordulására eltérő értéket vesz fel és ezért azonosítóként jelölik ki.

E bonyolult meghatározás magyarázata a következő: Ritka esetben találunk a jelenségekre jellemző olyan egyetlen természetes tulajdonságot, amelynek az értékei minden egyedelőfordulásra eltérőek. Ekkor két megoldásunk van. Néha mesterséges 'nevet' vezetünk be, mint amilyen a fent említett Rendelésszám. Ha ez nem lehetséges vagy nem célszerű, akkor viszont összekapcsolunk két vagy több tulajdonságot úgy, hogy azok értékeinek az együttese legyen eltérő az egyedtípus minden előfordulására. Ez a 'kvázi'-nominatív azonosítás, aminek a legszebb példája a mesterséges személyi-szám.

Néha viszont az is előfordul, hogy egynél több azonosításra alkalmas adatunk van. Vállalatunk dolgozóira hivatkozhatunk a személyi-számmal is (sajnos ma már nem) és a mesterségesen bevezetett törzsszámmal is. A megrendelés egy tételét azonosíthatja a rendelésszám+cikkszám páros is, meg a tételsorszámmal kiegészített rendelésszám is. Számos ok miatt (ld. alább a kapcsolat fogalmát) a több lehetséges azonosító közül az egyiket ekként kell kijelölnünk - és erre utalt a meghatározás utolsó félmondata.

Az alternatív azonosításnak nincs akadálya. Az ismeretkezelés során az A és a B tulajdonság a felhasználó által egyaránt alkalmazható keresésre. Az adatbázis belső szerkezetében van szükség az egyértelmű azonosítóra. Éppen ezért a fenti közértbeli 'traccs' roppantul természetesnek ható azonosítási megoldása szóba sem jöhet az információs rendszerekben.

5.3 A KAPCSOLATTÍPUS FOGALMA

A mindennapi életben kiejtett mondataink gyakran egynél több állítmányt is tartalmaznak. Figyeljük csak meg a következő két példamondatot:

„Az FGS 802 rendszámú gépkocsi típusa FORD Escort és ötszemélyes.”

„Az XYI 999 rendszámú gépkocsi típusa FORD Escort és ötszemélyes.”

Az olvasónak nyilván feltűnt a két mondat 'zakatolása'. Lehet, hogy ezt nem így fogalmazza meg magában, de a két állítás - **redundáns**. Ugyanis minden FORD Escort gépkocsi ötszemélyes. Számos - itt nem részletezendő - ok miatt a redundanciát általában kedvezőtlen dolognak tartjuk az informatikában. Például azért, mert valaki közölhetné az alábbi - téves - ismeretet:

„Az XYI 999 rendszámú gépkocsi típusa FORD Escort és kétszemélyes.”

Ha valahol - egy helyen - elkönyvelnénk a következő dolgot, akkor a tévedés nem fordulhatna elő:

„A FORD Escort típusú gépkocsi ötszemélyes.”

A fenti két mondatból hármat 'faragva' egyértelműbb ismereti képet kapunk, mert a férőhelyet illetően nem tévedhetünk:

„Az FGS 802 rendszámú gépkocsi típusa FORD Escort.”

„Az XYI 999 rendszámú gépkocsi típusa FORD Escort.”

„A FORD Escort típusú gépkocsi ötszemélyes.”

A mindennapi életben gyakran élünk a 'szillogizmus' lehetőségével. Nemcsak önmagukban, hanem összefüggéseikben is nézzük a jelenségeket. Logikusan kötődik az egyik mondatunk állítmányához a másik mondat alanya. A fenti három mondatból még a gyerek is levezeti azt az ismeretet, hogy mindkét kocsi ötszemélyes.

Most egy nagyon lényeges mondanivaló következik. Az informatikusok igen sokat foglalkoznak az ismeretek *extenzionális* vetületével, már jóval kevesebbet az *intenzionálissal*, a *transzverzálissal* pedig alig törődnek. Nem kell megijedni - egészen egyszerű dolgokról van szó.

Extenziónak a szóban forgó dolgok fogalmi kiterjedését nevezzük. Ez a fenti egyedhalmaz-tényezőhöz kapcsolódó lényeg. Az alkalmazási környezetben a dolgozó egyedtípus azokat a személyeket öleli fel, akik cégünkkel belső munka-viszonyban állnak. Van X dolgozónk. Pesten X1, Győrött X2, Pécsen X3. Az extenzió a minőséghez (dolgozó) kötött mennyiség ($X = X1 + X2 + X3$) úgy, hogy a minőség a meghatározó, de a mennyiség sem elhanyagolható. Ma jellemző az, hogy a számítástechnikus inkább az utóbbival törődik. Például azzal, hogy majd létre kell hoznia egy ilyen meg olyan méretű pesti, győri stb. adatállományt.

Az intenzió a jelenségre vonatkozó bennfoglalt fogalmak köre. A gépkocsikat a rendszámmal, színnel, típussal akarjuk jellemezni. Az Y1, Y2, ..., Yx jellemző foglaltatik az egyedtípus jellemzésébe. A mi kifejezéseinkkel élve az egyedtípus tulajdonságtípusainak az együttese az intenzió. Az 5.4 táblára hivatkozva akár azzal az egyszerűsítéssel is élhetünk, hogy a tábla *sorai* jelentik az extenzionális, *oszlopai* pedig az intenzionális vetületet. Ez pedig összefügg a harmadikkal, és így meg kell magyaráznunk azt a kitétele, amely szerint az informatikusok nem eleget törődnek az intenzionális dolgokkal.

T 5.1 Nemrégiben láttam egy számla 'adatbázist'. Egy-egy tételében szerepelt a számlaszám; a vevőkód, a vevő neve, címe, banki számlaszáma; a rendelésszám, dátum; a cikk száma, neve, ára, stb. Magyarul: Ahány rendelést adott fel egy vevő, annyszor jelent meg a cikk neve, ára stb. a tételek sorokban.

A vevő, a rendelés, a cikk, az arra vonatkozó rendeltetés, a számla és annak tétele mind-mind különböző jelenségek. Hiszen a vevőt és a cikket nem azonos típusú ismeretek jellemzik - és ez vonatkozik a többi felsorolt dologra is. Ez az 'adatbázis' végtelenül silány. Az olvasó el tudja képzelni, hogy ha pl. egy cikk X vevő Y megrendelésének a Z darab tételében szerepel, akkor mennyire fog a tár a cikknevet ismétlésével ($X*Y*Z$) és mi az esély arra, hogy azt azonosan írják.

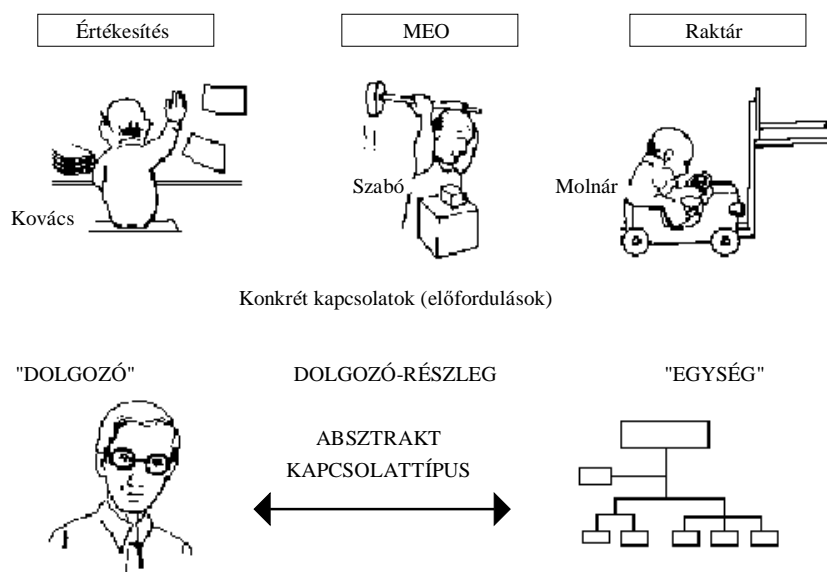
A példában az ismeretek intenzionális elrendezésén nem is gondolkodtak. Ha azt megtették volna, akkor - többek között - külön vevő és rendelés egyedtípust alkottak volna. Ezzel az igen célszerű intenzionális megoldással viszont fellép az ismeretek kapcsolásának a transzverzális problémája. Ha külön állományban van a megrendelés és a vevő, akkor miként kapcsoljuk a kettőt egymáshoz?

Transzverziónak a jelenségcsoportok közötti ismereti váltást nevezzük. Három - a kocsikra vonatkozó - példamondatunk jól szemlélteti ezt a logikai lépést. Ha az FGS 802-es rendszámú gépkocsi típusa FORD Escort (most még csak az első jelenségnél tartunk), akkor (itt jön a váltás) az ötszemélyes (itt már átléptünk a másikba). Ha most az olvasó a transzverziót a logikai dedukcióval (levezetéssel) tekinti azonosnak, akkor (részben) téved. Ugyanis nemcsak az a kérdés tehető fel, hogy egy adott gépkocsi a vonatkozó típus esetében hány férőhelyes, hanem az is, hogy melyek az ötszemélyes gépkocsik.

Az előző pontban utaltunk arra, hogy az egyed és a tulajdonság relatív dolgok. A gépkocsi-típus nemcsak a gépkocsi tulajdonsága, hanem egyben olyan egyed is, amely saját jellemzőkkel (pl. férőhely) írható le. Ez a kettősség egy nehezen emészthető gondolatmenetnek ad alapot, amelynek kifejtése előtt látnunk kell a könnyebb dolgokat. Meghatározás következik:

D 5.6 Az egyedek közötti viszonyok fogalmi tükörképét kapcsolatoknak nevezzük.

Péter az FGS 802-es rendszámú gépkocsi tulajdonosa. Pál a BMZ 873-asé. Itt két **konkrét** kapcsolatról van szó, amelyeknek van minősége is - 'tulajdonosa'. Lehetne más minősége is. Például Péter fizeti a BMZ 873-as kocsit Cascoját. A kapcsolat velejét ki kell fejtenünk - ezért van szó fogalmi tükörképről. Ezért a viszonyok **absztrakciója** is elkerülhetetlen. Ismét generikusban és specifikusban kell gondolkodnunk. Általában a gépkocsiknak vannak tulajdonosai és vannak, akik a Cascot fizetik (nem feltétlenül a tulajdonos!). Speciálisan az X gépkocsi tulajdonosa Y, Cascoját pedig Z fizeti, aki többnyire azonos Y-nal.



5.3 ábra: Kapcsolattípus és -előfordulás

Péter az FGS 802-es rendszámú gépkocsi tulajdonosa. Péter - személy. Ezért általában is fennáll a 'személy - gépkocsi' kapcsolattípus, amelynek a jelentése ebben az esetben az, hogy tulajdonos. A 'személy a kocsit tulajdonosa' illetve a 'személy fizeti a kocsit Cascoját' absztrakt viszonyok - kapcsolattípusok. Azt pedig már szinte felesleges megismételni, hogy a kapcsolattípus előfordulásai is halmazt képeznek. Ezek a tulajdonos-gépkocsi illetve a Casco-fizető-gépkocsi egyedpárosok.

D 5.7 Az egyedtípusok közötti generikus viszonyok fogalmi tükörképét kapcsolattípusoknak nevezzük.

A kapcsolatok az ismeretek világában nagyon sokfélék lehetnek. Itt nincs mód arra, hogy a rész-egész (a kocsinak része a motor); a birtoklás (enyém a kocsit); a hasonlóság; az altípus stb. kapcsolat-jellegekre kitérjünk. Elég szomorú, hogy a mai információkezelésben a szemantikában ismeretes huszonvalahány jelleg helyett mindössze csak kettőt-hármat ismernek és ismernek el. Nem feladatunk a kizárás/bennfoglalás (ha a kocsit az enyém, akkor nem a tied; ha az enyém, akkor nekem kell gondoskodnom a kötelező biztosításáról) és az opcionálitás (minden kocsinak van tulajdonosa - már akkor, ha nem rendőrségi ügyről, talált kocsiról van szó, amelynek a

tulajdonosa ismeretlen -, de nem minden személynek van kocsija.) A kapcsolattípus számtalan jellemzője közül csak egy dologra kell kitérnünk: ez pedig a **kapcsolat foka**.

A jelenségek 1:1, 1:N vagy M:N fokú viszonyokban állhatnak egymással. A kocsinak csak egy kötelező biztosítása lehet (1:1). Egy kocsinak mindig csakis egy bejegyzett tulajdonosa lehet, de ez fordítva nem igaz, mert egyeseknek van több kocsijuk is (de nem a szerzőnek - 1:N). Egy balesetben több gépkocsi vehet részt és megfordítva, a peches kocsi több baleset szenvedő alanya lehet (M:N).

Intermezzo

Egyes informatikai koncepciók szerint a jelenségek közötti kapcsolatot magát is mindig ismerettel kell leírni, a kapcsolat fokától függetlenül. Másokban pedig a kapcsolat nem egy lényeg-et jelent, hanem mindig legalább kettőt. Mást-mást a kapcsolt dolgok 'oldaláról nézve'.

Ezekkel az informatikai megközelítésekkel nem értünk egyet. Talán kezdjük az ellenvélemény kifejtését a második tétellel. Kétségtelen, hogy a megrendelés és a vevő viszonyában az előbbi a 'feladott', az utóbbi a 'feladó' szerepét tölti be. Mindez azonban nem jogosít fel bennünket arra, hogy kétféle viszonyt lássunk. Egyrészt ez a látásmód redundáns, mert maga a viszony - egy. Másrészt bornírt is, hiszen a lényeg az, hogy rendelés történt, ha pedig azt feladták, akkor van feladó is, ha van feladó, akkor van feladott is. Azonban az olvasó ne vegye csak szóhasználati fordulatoknak az előző gondolatmenetet! Pl. a házasság - egyetlen viszony. Pontosan akkor romlik meg, amikor kettősnek tekintik, és a feleség ill. a férj oldaláról kezdik ezt a kapcsolatot értelmezgetni.

E 5.2 Az ismeret szemlélete = az élet szemlélete.

Az ismeretek - absztrakciók. Menthetetlenül tükrözik az elvonatkoztatónak a valóságos dolgokkal kapcsolatos szemléletét. „Mutasd meg az adatmodelledet, és én megmondom ki vagy.” (Kiemelés: a szerzőtől.) Az adatmodell lényegére majd alább térünk ki. Most a megszakított gondolatot a kapcsolatfokra történő visszatéréssel folytatjuk.

Egy megrendelésben több cikket lehet kérni és egy cikket több megrendelésen lehet igényelni (M:N fok). Az Intermezzo első mondata szerint gondolkodók itt kapcsolatot látnak. Az első ránézésre jogosan, mivel a rendelés és a cikk nem talál egymásra, ha nem kötjük össze őket valamilyen módon. Nevetséges dolog lenne rendelést feladni cikk-megjelölés nélkül. Gazdaságilag indokolatlan lenne olyan cikket foglalkozni, amire nem adnak fel rendelést. Tehát kapcsolatban áll a rendelés és a cikk?

Nem! A D 5.1 definícióban világosan elmondtuk, hogy egyednek nevezzük az ismeretekkel leírni kívánt dolgokat. Ismeretekkel óhajtjuk-e leírni a rendelés és a cikk viszonyát (rendeléstétel)? Persze, például a megrendelt mennyiséggel. Így a rendeléstétel - bár kétségtelenül viszony - egyedként tükrözendő dolog. Maga a viszonyt leíró rendeléstétel egyed áll azután kapcsolatban a rendeléssel éppúgy, mint a cikkel. A rendelés és cikk **közvetett** M:N fokú viszonyát két 1:N fokúra bontjuk le: a rendelés és rendeléstétel illetve a cikk és rendeléstétel kapcsolatára. Mert hiszen egy rendelésben több tétel van (1:N) és egy cikkre több tétel (1:N) vonatkozhat.

Figyelmeztetnünk kell mindenkit, hogy az M:N fokú viszonyra való alapozás egészen egyszerűen téves következtetésekre is vezet(het)! Tegyük fel, hogy egy szervezeti egységben - nevezzük költséghelynek - a személyek mellett gépek is találhatók. Ezért a személy és a gép egyedtípusokat egyaránt jellemeznünk kell a költséghely-kód ismerettel azért, hogy tudjuk e 'dolgok' hová tartoznak. Viszont nagy baj lenne, ha e közös jellemző alapján valaki kapcsolatot keresne a gépek és a személyek között. Egy pillanat alatt kimutatná, hogy az X személy az Y gép kezelője, holott az egyszerű igazság csak az, hogy az X és az Y 'dolog' egyazon Z költséghelyhez

tartozik. Ott a személyek és gépek távoli viszonya M:N fokú, és a közös adatból (költőhely-kód) nem vonható le semmilyen következtetés. A ma annyira gyakori hamis információk e téves kapcsolások következményei.

Itt kell visszatérnünk az azonosítóhoz. Két egyed akkor és csak akkor köthető egymáshoz, ha az egyik azonosítója a másikban is megtalálható. Tehát akkor, ha - képletesen szólva - az egyik mondat állítmánya a másik alanyában folytatódik. Lássunk egy példát, majd egy ellenpéldát!

„Az FGS 802 rendszámú gépkocsi típusa FORD Escort.”

„A FORD Escort típusú gépkocsi ötszemélyes.”

Helyes következtetés, hogy az FGS 802 rendszámú gépkocsi ötszemélyes. Az első mondat a másodikban ‘folytatódik’.

„Kovács az értékesítési osztályon dolgozik.”

„A 116-os számú számítógép az értékesítési osztályon működik.”

Helytelen következtetés, hogy Kovácsnak bármi köze is lenne a nevezett 116-os számú számítógéphez. Vagy van, vagy nincs. Ha van, akkor ezt külön kitétel kell, hogy kifejezze.

A kapcsolat a legnehezebben érthető ismereti fogalom. Egyetlen egy adattáblát (egyedtípust) könnyű elrendezni. Lesz X sora (egyedelőfordulása) és Y oszlopa (tulajdonságtípusa) úgy, hogy az oszlopok és sorok találkozásánál jelenik meg a tulajdonságérték (ld. 5.4 tábla). Sokkal nehezebb az ismereteket átköthetően, az átjárhatóságra ügyelve megszervezni. Az alábbi táblakettős egy jól megtervezett adatbázis-részletet mutat:

5.5 tábla

GÉPKOCSI		
Rendszám	Szín	Típus
BMZ 873	piros	LADA 2104
FGS 802	bordó	FORD Escort

GÉPKOCSI-TÍPUS	
Típus	Férőhely
LADA 2104	ötszemélyes
FORD Escort	ötszemélyes
Polski Fiat	négyszemélyes

Érdekes, eddig a Polskiról nem is volt szó! Lám-lám, az ismeretek célszerű tagolása ismét egy újabb gondolattal gazdagít bennünket.

Az 5.5 tábla két része egyértelmű módon kapcsolható egymáshoz. A felső táblából kiindulva megtudhatjuk, hogy a konkrét gépkocsik hány személyesek. Az alsóból is visszajuthatunk a felsőhöz a közös ‘típus’ jellemző által. Látjuk, hogy két darab ötszemélyes és nulla darab négyszemélyes kocsink van. Nincs Polskink.

A logikailag célszerűen tagolt adatbázis - már másodszor említjük ezt a szót, aminek a lényegét majd alább magyarázzuk meg - több ismeret felölelésére képes, mint a rosszul meghatározott. Az 5.5 táblában el tudtuk mondani, hogy a Polski négyszemélyes - ami ismeret akkor is, ha ilyen kocsink nincs. Viszont az 5.6 tábla erre nem ad módot mindaddig, ameddig nem lesz egy adott rendszámú Polski járművünk:

5.6 tábla

GÉPKOCSI			
Rendszám	Szín	Típus	Férőhely
BMZ 873	piros	LADA 2104	ötszemélyes
FGS 802	bordó	FORD Escort	ötszemélyes

A kapcsolat az egymáshoz kötött ismeretek célszerű elrendezésének a kulcsa. A kapcsolat a végső válasz arra a kérdésre, hogy végül is van-e adatbázisunk, vagy sem. Mivel ezt az ‘adatbázis’ fogalmat már többször említettük, úgy illik, hogy végre megmagyarázzuk a lényegét.

5.4 ADATBÁZIS, ADATMODELL, ALMODELL

A mindennapos ismeretközléseinkben használt mondatok alanyt és egy vagy több állítmányt tartalmaznak. (A kocsi színe piros és típusa FORD.) Mondataink nem lezártak, hanem az állítmányo(ko)n keresztül másféle állításokhoz vezetnek át bennünket. (A FORD ötszemélyes.) Ezért ki kell jelentenünk, hogy az ismeret nemcsak az egyféle dologról (egyedek) megszerezhető tudást (tulajdonságok) jelenti, hanem a dolgok jellemzőinek az összefüggéseire (kapcsolatok) alapuló információt is. Ezt a szigorúan hangzó kitélt a mindennapok során állandóan érvényesítjük, amikor beszélgetéseinkben az egyik témáról a másikra váltunk. Viszont az információs rendszerek kialakításában elfeledkezünk róla, és nem szentelünk kellő figyelmet a kapcsolat tényezőjének.

Mindebben ‘segítenek’ bennünket a rémrossz eszközök (szoftverek), amelyek nem követelik meg az egyik oldalon a tisztánlátást (vö. T 5.1), a másik oldalon viszont ott is ‘adatbázist’ kiáltanak, ahol pedig ez a szelíd-farkas nem is látható.

Az információs rendszernek az a feladata, hogy az ismeretekkel leírni kívánt valós jelenségek (egyedek) jellemzőit (tulajdonság) illetve azok összefüggéseit (kapcsolat) hűen tükrözze. A **konkrétumokról** van szó. Konkrétan arról, hogy az FGS 802 rendszámú gépkocsi (egyed) színe (tulajdonság) bordó, típusa (másik tulajdonság) FORD Escort, ami a másik egyed (gépkocsi-típus) felé kimutató, a közös ‘típus’ adaton alapuló viszony (kapcsolat) szerint azt jelenti, hogy a kocsi ötszemélyes (újabb tulajdonság, de immár a második egyedé).

D 5.8 Az adatbázis az egyed-, tulajdonság- és kapcsolat-előfordulások adatmodell szerint szervezett együttese.

Az olvasó vegye figyelembe, hogy a fizikai adatszervezés módjáról illetve a tételek ábrázolásának a mikéntjéről nem szól a meghatározás. Csak azt jelenti ki, hogy vannak jelenségek, amelyeket ismeretek írnak le úgy, hogy ezek közül egyesek más jelenségekre mutatnak. Az ‘adatmodell szerint’ kitélt pedig majd alább magyarázzuk meg.

Intermezzo

Hál’ Istennek a mai ‘multimédiás’ korszakban lehetőség van arra is, hogy akár videofelvételkel, fényképekkel, szövegekkel, hangokkal gazdagítsuk ismeret-tárunkat. Azonban ez az ábrázolás nem végcél, hanem eszköz. Ezért az olvasó ne feledje, hogy például egy kép a **szerve-**

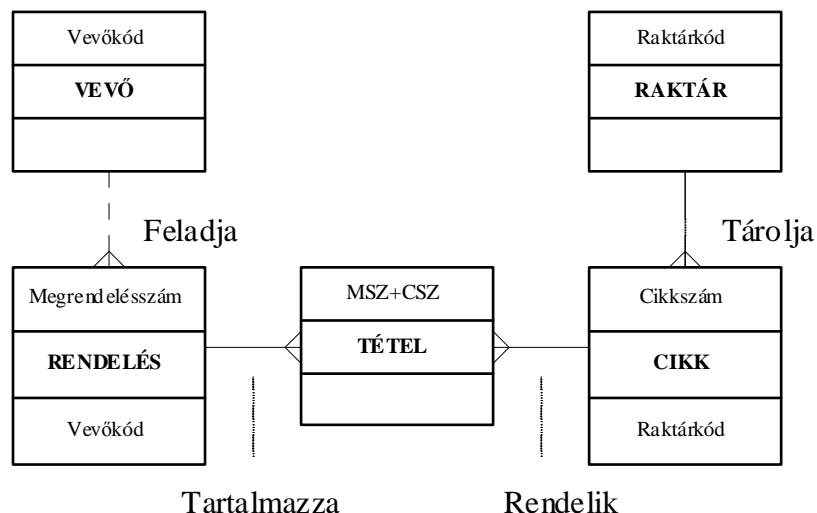
zett adatbázisban nem akárhol és nem véletlen módon jelenhet meg. A kép is egy jellemző, és ezért a jellemzett dolog tulajdonságaként kell feltüntetni a képre való utalást is.

A közértbeli hölgyek az ismereteket tetszőleges módon rendezhetik el. Ez így helyes, ez így emberi. Az információs rendszernek adott esetben ezer és millió embert kell kiszolgálnia. Nem épülhet az egyedi elképzelésekre. Az kizárt, hogy hol ilyen, hol olyan egyedekben; hol ezekben, hol azokban a tulajdonságokban; most ebben, máskor abban a kapcsolati rendszerben gondolkodjanak. Előre meg kell formálni az adatbázis absztrakt képét akkor is, ha tudjuk: az idővel változni fog. Ugyanis a konkrétumokat csak az absztrakt kép birtokában lehet általánosan elrendezni.

D 5.9 Az adatmodell az egyed-, tulajdonság- és kapcsolat-típusok ill. az ezekre vonatkozó korlátok szervezett együttese.

Minden konkrétum az absztrakt képnek kell, hogy megfeleljen. Például nem létezhet olyan egyedelőfordulás, amely ne tartozna valamelyik egyed típusba. A 'szervezett' jelző ebben az esetben azt jelenti, hogy komolyan megvizsgáljuk az egyed típusok intenzióit, vagyis tulajdonságtípusainak együttesét és célszerűen elrendezzük a transzverziókat, vagyis az egyed típusok kapcsolattípusait. Ez a kétféle elrendezés - az egyed típus belső és külső szerkezete - számos megkötést implicál. Például az egyed típusnak kell, hogy legyen azonosítója, és legfeljebb csak egy ilyen jellemzője lehet. Két egyed típus csak akkor áll kapcsolattípusban, ha van egy közös tulajdonságuk, amely legalább az egyik azonosítója stb. Ezen *strukturálisokon* kívül léteznek **leíró korlátok** is. Például a vevőkód úgy épül fel, hogy..., a rendelésszám értéke X és Y között lehet stb.

A meghatározásban a korlát kifejezést az utóbbi értelemben kell venni, hiszen a szerkezeti korlátok magában a struktúrában találhatók. A definíció tehát arra utal, hogy az adatmodell nem pusztán szerkezeti, hanem ahhoz fűzött szöveges leírásokat is kell, hogy tartalmazzon. Ezek lehetnek meghatározások, feltételek, megkötések, korlátozások, értelmezések stb.



5.4 ábra: Adatmodell-szerkezet

Az előbbi ábra az adatbázis tervének az ún. **Bachman-diagramja**. Az effajta ábra igen alkalmas az alapvető ismeretek összefüggéseinek a bemutatására. Az ábrázolás alapvető konvenciói a következők:

- Az egyed típusokat téglalapok jelképezik úgy, hogy az egyed típus neve szerepel a téglalap közepén (vastag szedettel).
- A kapcsolattípusokat a téglalapokat összekötő vonalak mutatják; a név a vonal mellett található. A sima vonal vég egyes, a 'szarkalás' N-es fokot jelez. A folyamatos vonal kötelező, a szaggatott opcionális viszonyt mutat.
- Az egyed neve fölött található az azonosító tulajdonságtípus illetve az összetett azonosító részei. Az egyed neve alatt tüntetjük fel a kapcsoló- és az egyéb tulajdonságtípusokat. Általában - a zsúfoltság elkerülése miatt - csak az előbbieket.

A diagramon számos egyéb jelzés is feltüntethető, de egyelőre elegendőnek tartjuk a leggyakoribbak ismertetését. Most fontosabb rámutatni arra, hogy az ilyen adatmodell-szerkezeti diagram nemcsak az adatbázis tervezési eszköze, hanem az adatbázis lényegének a felhasználóval való megértetésére is szolgál. Az értelmezés viszonylag könnyű. Az 5.4 ábra a következőket mondja:

- A vevők rendeléseket adnak fel, akár többet is (szarkalás vég), de minden rendelés csak egy vevőhöz tartozik (egyenest vég). Az utóbbi viszony kötelező (folyamatos vonal), tehát rendelés nem létezhet vevő nélkül, de ez megfordítva - időlegesen - elképzelhető (szaggatott vonal).

- A rendelések többtételűek is lehetnek, de a rendeléstétel mindig csak egyetlen rendelésnek a része. A Tartalmazza kapcsolat kétirányban kötelező. Evidens az is, hogy tétel nem létezhet rendelés nélkül és egyúttal minden rendelésnek kell, hogy legyen legalább egy tétele.

- Minden egyes tétel kötelezően egy cikkre mutat. Nem minden cikkre létezik adott időpontban rendeléstétel, viszont egyesekre akár több is vonatkozhat.

- A cikkeket raktárakban tárolják. A kapcsolat kötelező: nincs raktár cikk nélkül és fordítva. Viszont...

Az olvasó - a valós életben a tervező és a felhasználó - elgondolkodhat azon, hogy valóban igaz-e az az ábra által sejtetett tény, hogy minden cikkféle csakis egyetlen raktárban lelhető fel. Tekintettel a 'kéziraktárakra', ez kétséges. Melyik egyedbe kerül például a cikk-készlete tulajdonság?

Az adatmodell-diagram az IR dokumentációjának a szerves része. Ha egyszer megtanuljuk azt 'olvasni', akkor rengeteg dolgot elárul nekünk az ismereteinkről. Illetve pontosabban szólva arról, hogy mi azokat hogyan látjuk.

Mivel az információs rendszerben 'minden-mindennel összefügg', elvileg az adatmodell az IR-ben fellépő valamennyi ismeretet fel kellene, hogy ölelje. Ez azt jelenti, hogy az összes bennünket érdeklő jelenség absztrakcióit - vagyis az összes létező egyedtípust - tartalmaznia kellene. Csak ezen az áron kerülhető el a redundancia (az ismeretek többszörös tárolása és kezelése) illetve a fordított baj, a kapcsolati hiány (nem tudjuk az ismereteket összefüggésbe hozni).

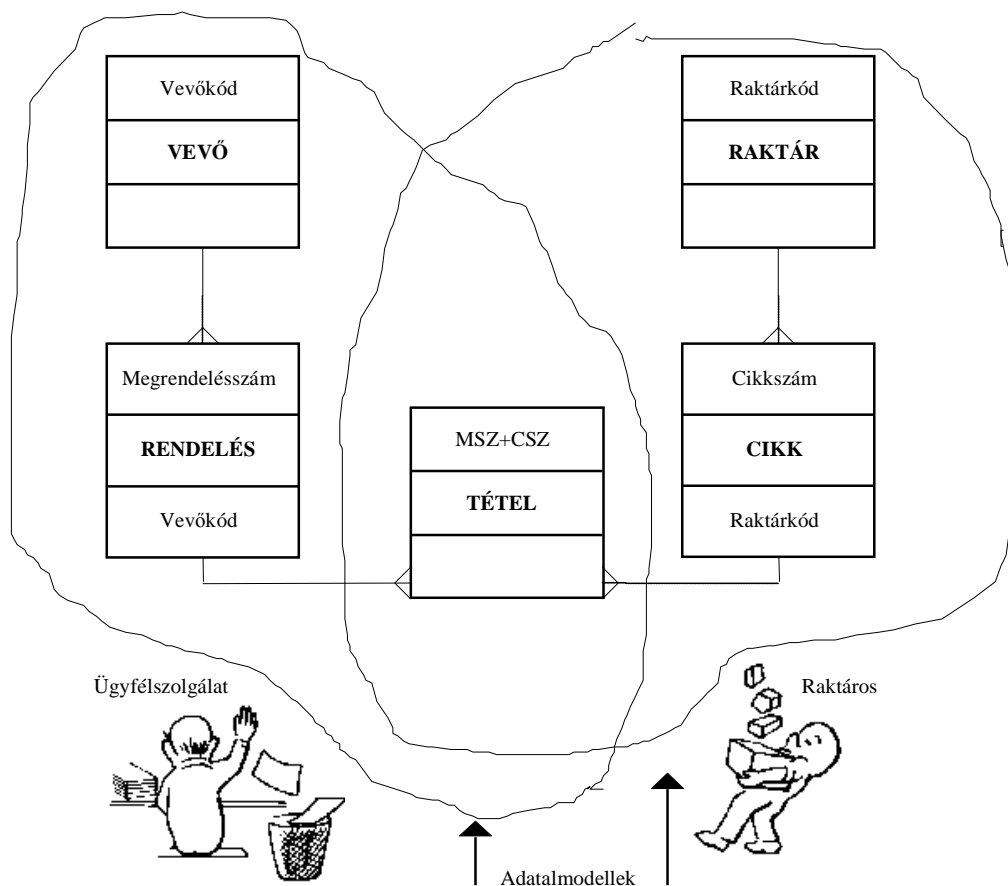
Természetesen azt senki sem várja el az alkalmazói-**felhasználótól**, hogy ilyen átlátással rendelkezzen. Ez egy különleges szerepű **fejlesztőnek**, nevezetesen az adatbázis-tervezőnek a dolga. Neki kell megoldania két nem is annyira egyszerű dolgot. Az egyik az, hogy a különböző felhasználók ismeretigényeit harmóniába hozza, mert hiszen nem lehet mindenkinek saját adatbázisa egy cégen belül. A másik az, hogy az igényekből egy olyan tervet gyúrjon akár kompromisszumok árán is, amely amellett, hogy mindenkit kiszolgál, ráadásul még optimális is.

Nem ennek a könyvnek a feladata az adatbázistervezés titkainak a feltárása. Itt csak arra akarunk rámutatni, hogy a közös adathasználatról nem kell megijedni. Manapság mindenki 'saját adatbázist' akar, ami azért ellentmondás, mert az itt nem ismertetett kiegészítő meghatározás szerint az adatbázis osztott használatú. Az *osztott* [shared] lényegét sokan félreértik. Egyesek az *elosztott* [distributed] - azaz földrajzilag különböző térségeket felölelő - rendszerekre gondolnak. (Már csak azért is, mert a szaksajtó is rendszeresen 'osztott'-nak fordítja a 'distributed' - teljesen mást jelentő - angol szót.) Mások a közös on-line használatot látják e szó mögött. Holott az osztott valójában azt jelenti, hogy több felhasználó - akár látszólag egyidejűleg is, akár a közös adatbázisnak a teljesen eltérő részeit is, akár teljesen eltérő célokból és módokon is - kezelheti. Nem is adatbázis az az ismerethalmaz, amely ilyen módon nem kezelhető.

Az 'in pluribus unum' kérdése, a közös adatbázis egyedi szemléletű kezelése az ún. adatmodellek által lehetséges.

D 5.10 Az adatmodell az egyed-, tulajdonság- és kapcsolattípusoknak viszonylag önálló, önmagában is szervezett csoportja.

A rendszernek a része is rendszer, az adatmodellnek az almodellje is az: szervezett egyed-, tulajdonság-, kapcsolattípus együttes. Az olvasó nem téved, ha az egyébként meglehetősen - mondjuk - visszafogott meghatározás lényegét a 'viszonylag önálló' szópárosban keresi.



5.5 ábra: Adatmodellek

A szervezetekben tevékenykedő emberek meghatározott feladatokat látnak el. Ezért elsősorban az azokhoz kapcsolódó ismeretekben érdekeltek. Így tehát nem csoda, hogy a jelenségeket, azok jellemzőit és viszonyait egymástól igen eltérő módokon látják. Az informatika nyelvén szólva: más-más egyed-, tulajdonság- és kapcsolattípusok érdeklik őket. Szemléletük akkor is különbözik, ha egyazon jelenségről van szó. Tessék csak pl. megfigyelni, hogy egészen másként néz az ügyfélre a biztosítási ügynök, mint a kárfeltevő! A látásmódbeli különbségek miatt végeredményben minden egyes - hm! - 'terület' a legszívesebben az egyéni ízlésének megfelelő, csakis a saját igényeit kiszolgáló, önálló adatbázist szeretne elképzelni magának.

Az önálló adatbázisokra azonban sem szükség, sem valódi lehetőség nincs. (A rossz mai gyakorlatot ne tekintse az olvasó lehetőségnek!) Az ügyfél ismeretei azért nem 'köszönnek vissza' mondjuk a kárfelvételnél, mert ott más 'adatbázis' él, mint a szerződés kötésben. Ezért kell a polgároknak annyi papírt feleslegesen kitölteniük - az egyszer már megadott ismeretekkel. Azért bátorkodtunk a 'nincs lehetőség' fordulattal élni, mert bár az ügyfelet egyesek képesek több dolognak tekinteni, az ügyfél - egyetlen lényeg. A 'nincs szükség' kitétel is ilyen egyszerű magyarázatú. Képzeld el az olvasó, hogy a biztosítási ügynök és a kárfeltevő egy és ugyanaz a személy! (Ennek számos elvi akadálya van, de csak gondolati játékról van szó.) Fogadjunk, hogy a kettős szerepű Kovács lesz a legelső, aki tiltakozni fog a kétféle 'ügyfél adatbázis' ellen! Mint ahogyan a kisvállalatokban, ahol egy személyre több szerep jut, nap mint nap éppen a felhasználó nem érti, hogy miért nem tudja 'összehozni' a mondjuk az anyagokra vonatkozó X- és Y-féle ismereteket?

Az adatbázis-tervező feladata nehéz, mert az 'enyém-tiéd' szemlélet az IR-ben is uralkodik. Meggyőzni a résztvevőket a 'miénk' fontosságáról és kialakítani azt az adatbázist, amely mindenki részéről egy kicsi engedékenységet igényel, ámde a teljes közösséget jól kiszolgálja, nem kis kihívás. Az 5.5 ábra viszont mutatja, hogy a probléma nem megoldhatatlan. Az ügyfélszolgálat és a raktáros teljesen más szemmel nézi a rendeléstételeket. Az előbbi arra kíváncsi, hogy miképpen áll a vevő rendelése. Az utóbbi viszont készletekben gondolkodik. Lesz-e elég a düzniből, nemcsak az éppen toporgó X vevő számára, hanem úgy általában is?

Az eltérő szemléletek dacára a két felhasználó ismeretigénye jól kiszolgálható egyetlen adatbázisból. Ennek egy feltétele van: a viszonylagos önállóság. A két felhasználó ismeret-igénye eltérő is lehet (önállóság), de csak addig a mértékig, ameddig nézeteik megfelelnek a közös képnek (viszonylagos). Ez magyarul azt jelenti, hogy előbb együtt kell látni a *teljes* adatmodellt ahhoz, hogy utóbb azon és annak megfelelő *részeket*, almodelleket képezhessenek.

A 'képezés', az almodell-kialakítás ismét csak nem a felhasználó feladata. Nem is lehet az, hiszen ő nem akarja látni a teljes modellt. A lényeg megvilágításához egy egyszerű példát fogunk alkalmazni. Tekintsünk az 5.5 ábrára és tegyük fel, hogy valaki csakis arra kíváncsi, hogy melyik vevők (vevőnév) rendeltek az X (cikkszámú) cikkből! Más ismeretre nincs is szüksége. Mivel a cikkszám értéke végig X, ez a valaki voltaképpen egy szelektív vevőnév-listát akar.

Az ilyen és hasonló kérdést feltevő felhasználó az adatbázist nagyon sajátosan szemléli. Erről a szemléletről sok mindent kell elmondanunk. A szemlélet, alias **nézet** [view] az adatmodellnek olyan absztrakt vetülete, amelynek segítségével a teljes adatbázisból kijelöljük a bennünket érdeklő konkrét ismereteket. Maga ez a kijelölés intenzionális, transzverzionális és extenzionális vonzatú. Csak egyes egyedtípusok bizonyos tulajdonságai (intenzió) illetve kapcsolatai (transzverzión) érdekelnek bennünket, ráadásul csak sajátos egyedelőfordulásokra vonatkozóan (extenzió). A fenti példa esetében a nézet csak a vevő és a cikk egyed egy-egy tulajdonságát érinti (vevőnév, cikkszám - intenzió); kizárólag az X számú cikk vonatkozásában (extenzió); a két egyed között pedig nincs is kapcsolat, mert az transzparens (transzverzión). Ez utóbbi momentumra mindjárt visszatérünk.

Az adatmodellen valós és virtuális nézetek képezhetők. **Valós** a nézet akkor, ha a felhasználó ugyanúgy látja az adatmodell tényezőit, mint ahogyan azokat a tervező megfogalmazta. Például

lekérdezi az előző napi rendeléseket úgy, hogy a feladó vevők néhány adatára is kíváncsi. A rendelés és a vevő egyedei, azok tulajdonságai, a két egyed közötti kapcsolat valóságosan létező tényezők. Ezzel szemben a felhasználó *virtuális* képet is kialakíthat. Létrehozhat például olyan egyedtypust, amely a modellben nem is szerepel. Ámde az ilyen virtuális egyed csak olyan tulajdonságokkal rendelkezhet, amelyeket a modell tartalmaz vagy a modell által tartalmazott tulajdonságokból - például számítással - levezethető. A fentebbi esetet kibővíthetjük így is: „Jelenjen meg egy vevő-cikk egyedtypus az X számú cikke nézve úgy, hogy lássam a vevő nevét és az általa az X cikkből megrendelt összes mennyiséget.” A ‘vevő-cikk’ egyedtypus nem létezik: azt úgy kell kreálni a nézet összeállítása során. (NB.: Az adatkezelő rendszer képességei szerint ez akár automatikusan is történhet.) A megrendelt összes mennyiség is virtuális tényező: azt ki kell számítani a meglévő, valós tulajdonságokból.

A virtuális egyedek *rejtett* kapcsolatokat feltételeznek. Ábránkon a vevő és a cikk egyedtypus nem áll valós kapcsolatban és a vevő-cikk feladathoz virtuálisra sincs szükség. Ám ahhoz, hogy a kérdést megoldjuk, a cikk egyedből a tételen és a rendelésen át el kell jutnunk a vevő egyedig úgy, hogy a felhasználó által nem látott - rejtett - módon használjuk az egyedeket összekötő kapcsolatokat.

Ezen a ponton már megmagyarázható a nézet és az adataimodell különbsége. A nézet a felhasználó által *látni kívánt* ismeretek részmodellje, amelyben lehet akár virtuális tényező is. Az almodell a teljes modellnek az a *valós része*, amely a nézet előállításához szükséges. A példában a felhasználó nem akarja látni sem a tételt, sem a rendelés egyedtypust, sem azok tulajdonságait, sem a kettő közötti kapcsolatot. Viszont a kérdésre csak úgy tudunk válaszolni, ha - bár rejtetten - kezeljük a rendelések és a tételek adatait (pl. Megrendelésszám) is. Azért, mert a vevő valójában ezeken keresztül áll kapcsolatban a cikkel.

A korszerű adatbáziskezelő rendszerekben a nézeteket és azok virtuális elemeit dinamikusan lehet kijelölni. Ráadásul ezekre is és az alkalmazói programok által látott almodellekre is igen széleskörű védelmet lehet kiépíteni. Ezért nincs ok az osztott adatbázisokkal kapcsolatos aggodalmakra. Éppen ellenkezőleg: csakis a modell/nézet/almodell koncepciók adhatnak alapot a célszerű ismeretkezelésre, ha ún. faktografikus ismeretekre van szükségünk.

Az ismeretek kezelése ma nem az itt leírt céltudatos módon történik. Sokakat megrendít, hogy a számítógépre vitt ismeret-kincsüket miért nem úgy és akkor tudják elővenni... Ezért itt kitérőt kell tennünk.

5.5 ISMERETKEZELÉSI MÓDOK

Most egy pillanatra feledkezzenek el a multimédiáról! A grafikák, hangok stb. tárolásától és kezelésétől eltekintve - ám azoktól egyáltalán nem függetlenül - az ismeretkezelésnek alapvetően két módja van: az *adatszerű* és a *szövegszerű*. Az előbbit tényszerűnek (faktografikusnak) is hívják, aminek az okai majd mindjárt megvilágosodnak. Előbb azonban következzenek egy kis történeti áttekintés.

A lyukkártyás szöveggepek, a mechanikus számoló-apparátusok, a szobányi - valóban számításokra szolgáló - ős-számítógépek hosszú korszaka után ezeket az eszközöket olyan berendezésekké ötvözték, amelyek elsődleges célja az volt, hogy kiváltsák a könyvelő-masínákat. Lehet, hogy ez a kitétel egy picit túlzás, de annyi bizonyos, hogy a mai számítógépek elődeit elsősorban könyvelésre és nyilvántartásra alkalmazták.

Már a görögök is feljegyezték a rekordokat. Ez a kitétel kicsit tautologikus. A rekord ugyanis nem más, mint - feljegyzésre méltó tény. Sok helyen még ma is *rekordnak* hívják az embereket érdeklő lényeges - vagyis feljegyzésre méltó - ismeretek együttesét. Nemcsak a százméteres sík-

futás világcúcsára vonatkozó adatok sora (ki, mikor, hol, mennyi) rekord, hanem a számlaadatok sora is az.

A hasonló - valamilyen értelemben együvé tartozó - dolgok nyilvántartására a precíz könyvelők irattartókat nyitottak. Ezek becsületes magyar neve 'dosszié', angolul file (fájl). A **fájl** a valamilyen alapon egybetartozó dolgok ismereteinek a tárolója úgy, hogy annak egy tétele a rekord. Van egy számla fájlunk, aminek egyik tétele az X, másik rekordja az Y számú számla. Egy lényeges megjegyzés: Az állománynak hívott fájl időközben elveszítette az eredeti jelentését. Ma már fájlnek nevezik az olyan ismeretegyüttest is, amelynek nincsenek is tételei...

Az eredeti fájl- és rekord-fogalom jól megfeleltethető az ismeret egyed típus és -előfordulás dimenziójának. A számla egyed típus fájlban testesül meg úgy, hogy egy-egy konkrét számla-előfordulást egy-egy rekord tükröz. (Nem azonosságról van szó! A fájl és az egyed típus nem azonos dolgok. Az egyed típus számítógépi tükrözése egy vagy több fájlban is történhet - és ez megfordítva is igaz.)

Az egy dossziében tárolt papírok néha távolról sem hasonlítanak egymásra. Pl. a Scotland Yard egy-egy fájl - aktát - nyit az egyes nyomozásokkal kapcsolatos 'feljegyzésre méltó' ismereteknek. Ebben a legkülönbözőbb feljegyzések (azaz: rekordok találhatók). A könyvelő aktáiban lévő számlák formailag ugyan csak részben hasonlítanak egymásra, ámde tartalmuk már előírásos: roppant szigorú bizonylati szabályok szerint kell kitölteni a kötelező rovatokat. Ezek a rubrikák a **mező** [field] nevet viselték és viselik még ma is egyes helyeken, így például a vasúti menetrendekben. Ám mezőnek tekinthető a képernyőnek az a kiemelten jelzett része is, amit előírásos tartalommal kell kitölteni.

A mező és annak tartalma az ismeret másik két dimenziójának feleltethető meg. Itt sem azonosságról van szó! A tulajdonságtípus (pl. Rendszám) értéke (FGS 802) a mező tartalma. Ám adott esetben a tulajdonságtípus megbontható több mezőre, vagy több tulajdonságtípus vonható össze egybe. Az ismeretek négy dimenziója ugyanis fogalmi szintű lényeg; a fájl, a rekord, a mező pedig a logikai és/vagy fizikai szintű megvalósítás eszköze.

A rubrikákba kényszerített ismeretek ilyen módú kezelésének az elterjedésével az adat fogalom a mi **tág** értelmezésünkkel szemben egy **szűk** értelmet is kapott. Adatnak a mezőnek és tartalmának a párosát kezdték nevezni (szín = piros), és - időlegesen - még a másik két dimenzióról is elfeledkeztek. Ennek okaira rögtön visszatérünk. A lényeg az, hogy faktografikus vagy **adatszerű** kezelésnek hívták a rovatok szerint történő ismerettárolást és -kezelést, egy picit elfeledkezve arról is, hogy minden adat valóság (az elképzelés is az), de nem minden adat - tény.

Az adatszerű ismeretkezelésnek két módja van. A hagyományos mód, aminek az ideje lejárt, az **állománykezelés** volt. A fejlesztő nem egymással kapcsolt egyed típusokban, hanem elszigetelt jelenségekben gondolkodott. Amikor pl. az anyag állománnyal bíbelődött, akkor eszébe sem volt pl. a vevő állomány. Egy ideig ráadásul az állomány fizikai elrendezése lényegesebbnek tűnt a tartalomnál is. Nem a vonatkozó anyagtétel, hanem az ikszedik fizikai rekord állt a figyelem középpontjában. Ez a magyarázata az adat helytelen értelmezésének. Ha mindig csak az anyaggal és azzal is fizikai szinten foglalkozunk, akkor az egyed típusról elfeledkezhetünk, és az egyedelőfordulás is csak korlátozott értelmet kap.

A volt fejlesztőket persze menti a technika korabeli állapota. Ám a maiaknak nincs mentségük, ha a korszerű **adatbáziskezelés** helyett a régi megoldásokhoz ragaszkodnak. A logikai és fizikai adatbázis az állományok szervezett együttese. Ezért az állomány- és az adatbáziskezelés közötti különbség sokrétű. Az utóbbi kulcsfogalma a(z állományok közötti) kapcsolat és az arra épülő transzverzió. A hagyományos adatkezelés elsősorban extenzionális, a korszerű pedig elsősorban transzverzális. A magyarázathoz lapozzunk vissza a T 5.1 történetre!

A fejlesztő egy táblába zsúfolta a számla, a vevő, a cikk stb. adatait. A kezelés nála a tábla sorai mentén való függőleges (tehát: extenzionális) mozgásra, tehát az egyik számláról a másikra való átlépésre korlátozódik. Ezzel szemben az 5.5 ábra teljesen más logikát sugall. Az ismeretek, jelenségcsoportok szerint, egynél több állományban fognak megjelenni. A „kik rendelték az X

cikket” kérdésnek a megválaszolásához az előbbi fejlesztőnek végig kell böklátsznia az egyetlen egy tábláján. Az 5.5 ábra esetében viszont a táblák közötti transzverzió, a vízszintes irányú mozgás lesz a jellemző. Elsősorban, mint azt fentebb jeleztük, ugyanis az adatbáziskezelés felöleli az állománykezelést is. A függőleges mozgás sem kizárt, mert kereshetjük például az adott keltezésű megrendeléseket, ami csak a rendelés táblán belüli mozgást igényli. (NB.: A tábla az állomány szinonimája).

A négy dimenzióba kényszerített adatszerű ismeretkezelés egyeseket irritál, mivel túl feszesnek, korlátoznak és egy picit embertelennek tűnik. A továbbiak előtt legyen szabad felhívunk a figyelmet arra, hogy a korszerű számítógépek atyja - Neumann János - a programokat is adatoknak tekintette... Továbbá a mai korszerű szövegszerkesztők a kezdetleges programírási segédletekből nőttek ki.

T 5.2 Barátom Word-ben írja a dolgait és Excel-ben tárolja az adatait. A minap kifakadt: „De akkor miért nem tudom kikeresni azt, hogy... Miért nem tudom levezetni azt, hogy... Miért vannak redundáns, kapcsolhatatlan, ellentmondó információim?”

A rovatokba kényszerített - adatszerű - ismeretkezeléssel szemben csak sokkal később terjedt el a természetes nyelven alapuló ún. *szövegszerű* ismeretkezelés. Ennek is kétféle módja van. Az emberek többsége a dokumentumíráshoz alkalmas *szövegszerkesztéssel* téveszti össze a szöveges információkezelést. Ez a tévedés sok későbbi csalódás forrása. A szövegszerkesztő ugyanis arra való, amit a neve is mutat: szövegek megírására, formátumozására, kijejeztetésére stb. Egyáltalán nem való adatkezelésre és adatelőállításra. Miért nem?

A .doc és hasonló szöveg‘állományok’ esetében elvész az ismeretnek mind a négy dimenziója. Az ilyen állomány - egyetlen tétel. A fájlnak nincsenek sorai, képletesen szólva rekordjai (a szövegsorok nem azok!). Explicit módon nem is foghatók össze - mondjuk - a szerződések egyetlen állománnyá, hiszen minden egyes szerződés egy állomány. Persze az egyedtípus ill. - előfordulás fogalmának és megvalósításának a hiányában a kapcsolatokról egyáltalán nem beszélhetünk. A szövegszerkesztővel elkészített számla sohasem lesz a szerződéshez köthető.

Már csak azért sem, mert a kapcsolat mindig közös tulajdonságon alapul - és a szövegszerkesztő a tulajdonságtípus és -érték dimenziókat sem ismeri. E ponton az olvasó hajlamos tévhitre ringatni magát azon az alapon, hogy de bizony ő képes rákeresni ugyanarra az ismeretre. Megnyugtadjuk, hogy nem képes. Most ne tessék meghökkenni: mi ismereten nem egy karaktersort értünk. Három gond is fellép ezzel a bizonyos visszakereséssel kapcsolatosan. Az első a *homonima*. Ha mi a rózsza virágot keressük, akkor bizony találatként jelentkezik a burgonya és a Rózsza nevű személy is. Jó, az utóbbi akkor nem, ha..., de pl. a ‘vár’ szó igen meglepő találatokat fog eredményezni. A második a *szinonima*, amely lehet ún. technikai is. Az utóbbi az eltérő írásmódból fakad (pl. 12. vagy XII. hó). Valós szinonima például az anyag és a cikk, ha a két dolog ugyanazt jelenti. (Apropó, mindkettő melleleg homonima is, mert az anyag lehet szövet is, a cikk az újság része is.) És vajon amikor Olaszországot keressük, akkor rákérdezzünk-e az Itália - sőt Italia - jelsorra is? A harmadik probléma a *valóságghűség*. Semmi akadálya annak, hogy egy szövegben nem-létező, nem-ismert dolgokra utaljunk.

Ha az egyes szavak nem egyértelműek, akkor az ismeretek összekapcsolásával hamis információk birtokába jutunk:

T 5.3 A hetvenes években egy amerikai riporter szenzációs felfedezésre tett szert. Marilyn Monroe-ról közölte, hogy egyetlenemet végzett - miközben mindenki tudta, hogy még az általános sem ment neki.

A számítógépeken tallózó újságíró az 'alma' szót fedezte fel két szövegben. A csodaszép színész nő kedvenc bútora almafából készült, az egyik kollégiumnak ugyancsak 'Almafa' volt a beceneve. (Tehát itt nem az 'alma mater'-ről van szó.) A derék zurnaliszta tévesen kötött össze két ismeretdarabot - mert nem ismerte az adatbázisokat.

A média szakembereinek az ismeretei szövegekben testesülnek meg. Maga a szövegszerkesztő ezeknek a tényszerű kezelésére nem alkalmas. Ezért hozták létre már a hetvenes években a **szövegkezelő rendszereket**. Az 'alapanyag' ezek esetében is a szöveg, de azt egy picit előfeldolgozzák. Egyrészt kiegészítik őket a címet, szerzőt, helyet, keletkezési időpontot, osztályozást, formát stb. rögzítő valódi adatokkal. Másrészt ún. deskriptorokon alapuló szótárral (itt: tezaurusz) egészítik ki őket. A kezelő kijelöli, hogy melyik kulcsszónak a helyét jegyezze meg a rendszer. A kulcsszó ellenőrzésre is szolgálhat és még számos más igen kellemes lehetőséget nyújtanak a szövegkezelők. Számunkra most az a lényeg, hogy a tezaurusz és deskriptor páros adatszerűen működik, megadható pl. az, hogy a fehér a szín értéke, és ezzel máris kezünkben van az adat két dimenziója - a tulajdonságtípus és az érték.

A szövegkezelő képességei nem itt zárulnak le. A dokumentumok - szemben a szövegszerkesztővel - explicit módon csoportokba sorolhatóak, valódi állományt lehet belőlük képezni. Ezeket a kezelő együtt látja. Ezzel eljutottunk az adat két további dimenziójához, mert például a 'könyvek' állomány egyedtypusként, míg az egyedi könyvre vonatkozó dokumentum annak előfordulásaként tekinthető.

Számos elméleti és elvi kezeléstechnikai oka van annak, hogy a szövegkezelő rendszereket mégsem tekinthetjük adatbáziskezelőknek akkor sem, ha az előbbi egyesek a mindennapi életben az adatbáziskezelő névvel illetik. Engedje meg az olvasó, hogy csak egy indokot említsünk. Az adatbázis adatmodellre épül - lásd e két fogalom meghatározását - úgy, hogy a kapcsolat tényezője kiemelt szerepet játszik (vö. az állomány- és az adatbáziskezelés különbségével). A szövegkezelő szövegszerű állományok kezelésére alkalmas, az állományközi - előre kijelölt - kapcsolatokon alapuló kezelésre nem. Egy példa: a hírek állomány nem köthető a személyek szövegállományhoz, holott a hírek személyekre is vonatkozhatnak. Az „X elnököt megölték Y-ban” ismeretet külön-külön kell bevinni a két fájlba és együttes visszakeresésükre sincs mód.

A fentiek miatt téves az a köznap szöhasználat, amely szerint valaki 'belépett a nemzetközi adatbázisba'. Az adatmodell szerint strukturált, adatszerű kezelésű adatbázisok a legritkább esetben 'nemzetközi' és az azokba való belépést nem szokták csak úgy megengedni. A számítógépen célszerűen elrendezett szöveges állományok együttesét **adatbank**nak nevezik az informatikusok. Az ebben lévő állományok közös kezelésére csak technikai mód van: az ismeretek tartalmilag nem kapcsolhatók össze. Például fellapozható külön-külön a gyógyszerek és a gyógyszergyárak közös adatbankban lévő két állománya, de azt nem tudhatjuk meg egyszerre (!), hogy melyik gyár melyik gyógyszert gyártja. Nekünk kell tallóznunk a két ismerethalmazban, hacsak az egyikben vagy mindkettőben nem tüntetik fel a másik teljes (!) ismeretsorát, például a gyógyszerneveknél a gyártók minden adatát - vagy megfordítva. Ez a megoldás viszont hallatlan redundanciát és inkonzisztenciát okoz, aminek a megítélését az olvasóra bízunk.

Még egy dologra kell utalnunk. A **táblázatkezelők**re néha az adatbázis címkét ragasztják, mert az utóbbi fogalom 'nemesebben cseng'. Ez gyatra megtévesztés. A táblázatkezelő látszólag adatszerűen dolgozik, mivel rovatokat kell kitölteni tartalommal. Ám az első csalafintaság ott van, hogy itt nem tulajdonságtípusok értékeiről van szó. A táblázat oszlopaiba bármi és bármilyen ellenőrzés nélkül bevihető - gyakorlatilag szövegszerű módon, hogy az ábráról ne is beszéljünk. A második elringatás a táblák összekapcsolásának az úgymond lehetősége. Azt most ne firtassuk, hogy egy valódi adatbázisban e kapcsolásnak nincs számszerű és összetettségi korlátja, mint a táblázatkezelőkben. Az adatbázisban tetszőleges számú és tetszőleges jelenségeket tükröző egyedtypusok köthetők egymáshoz. A táblázatkezelőkben még az egyedtypus fogalma sem ismeretes, az előfordulásról pedig majd alább ejtünk szót. A lényeg az, hogy az adatbázisban az adatmodell - a kapcsolattípus tényezőn keresztül - előre és tudatosan rögzíti az összefüggéseket

rendszerét. A táblázatkezelőben nincs is ilyen kép - azt mindig teljesen ad-hoc módon alakítjuk ki. Természetesen egymásnak ellentmondó és átfedő módon.

A táblázatkezelőben nincs egyedelfordulás sem, hiszen az állomány egyetlen dolgot tartalmaz: magát a táblázatot. Végeredményben a táblázatkezelő az ami: a szövegeket nem lineárisan, hanem többdimenziósan megfogalmazó szerkesztő, amely igen korlátos adatelőállításra is alkalmas.

Ezt a pontot egy megjegyzéssel és két kiegészítéssel kell zárunk. Nem arról volt szó a fentiekben, hogy az ismeretkezelés egyik módja jobb, mint a másik, hanem arról, hogy vannak eltérő ismeretkezelési megoldások. Kettős **célban** kell gondolkodni. Van egy felhasználási cél és egy másik cél, mégpedig az, amire az eszköz épült. Ha valóban táblázatot akarunk kezelni, akkor alkalmazzuk az erre való eszközt, tudva azt, hogy a tavaly az X-nek megírt számlánk az ideivel nem fog egy 'adatbázist' alkotni. Ha viszont nekünk fontos, hogy együtt, elrendezve lássuk a számlákat, akkor csakis az adatbáziskezelő felel meg e célunknak.

Eddig nyitva hagytuk azt a kérdést, hogy a nem-adatszerűen kezelt ismeretek vajon miképpen illeszkednek a 'valódi' adatbázisba. Nem túl szűk-e az általunk kifejtett adatbázis-fogalom a multimédia korszakában? Van-e mód arra, hogy a szövegeket, grafikákat, ábrákat, képeket, táblázatokat stb. a szűkebb értelemben vett adatokkal együtt - vagyis egy szervezett adatbázisban - tároljuk és kezeljük?

Erre a felvetésre kettős választ kell adnunk: egy tartalmit és egy technikait.

Tegyük fel, hogy feladatunk a személy képének vagy a lakás alaprajzának a megjelenítése akkor, ha a személy nevének ill. a lakás azonosítójának leütjük az Fx gombot!

Tartalmilag az a tény, hogy a személy neve adat-, míg a képe más formát ölt, semmit sem változtat az adatmodell és az adatbázis lényegén. A személy képét és a lakás alaprajzát olyan a személyhez ill. a lakáshoz kapcsolt egyedítípusként kell elképzelnünk, amelynek egy-egy előfordulása a konkrét kép/alaprajz. (Ezt már azért is így kell tennünk, mert pl. egy személynek lehet több képe is.) Azt is tudnunk kell, hogy a kép a személyhez, az alaprajz a lakáshoz kapcsolódik - és nem fordítva. Tehát ki kell jelölnünk a kapcsolattípusokat. Végül valami alapján ki kell tudnunk választani a sok kép közül a konkrétat, a bennünket érdeklőt. Ez a kép azonosítója alapján történik, amely kapcsolóadatként szolgál. Ezzel máris előttünk áll az adatmodell harmadik tényezője, a tulajdonságtípus.

Az adatmodell és az adatbázis fenti meghatározása (D 5.9 és D 5.8) csak azt mondja ki, hogy az adatbázis az adatmodellnek megfelelően, egyed, tulajdonság és kapcsolat tényezők szerint szervezett. Arról nem szól, hogy mi ezen tényezők ábrázolási formája. Ezért a két definíció akkor is fennáll, ha az adatbázisban az adatok mellett képeket, szövegeket stb. tárolunk és kezelünk.

A **technikai** téren már nem ennyire tiszta a helyzet. Természetesen a feladatot meg lehet oldani, csak a megvalósítás mikéntje nem közömbös. Ma általában úgynevezett **testreszabott** [tailored] - másként szólva: célraorientált - módon közelítik meg ezt a kérdést. Függetlenül attól, hogy saját programot írunk-e ill. megvásárlunk-e egy pl. térgeometriai programcsomagot, a testreszabott módot egy pozitív mellett több negatív vonás jellemzi. A testreszabás = 'bedrótozás'. Az ilyen program hatékony, de roppantul merev, vagyis nem bővíthető. Azért nem az, mert az adatszerkezet eleve és fizikai szinten rögzített, kezelésének a részletei pedig beleépülnek a programokba. Ezzel az adatfüggetlenség mindkét elvét (E 3.1 és E 3.2) megsértik. Természetesen adatmodellről szó sincs, ezért tehát valódi - az adatmodellnek megfelelő - adatbázisról sem beszélhetünk.

A korszerű adatbáziskezelők ún. **általánosított** [generalized] rendszerek. Ez azt jelenti, hogy olyan előre elkészített rutinokkal dolgoznak, amelyek a kezelt tartalomtól függetlenül, viszont az előre kiválasztott (néhány) fizikai szerkezeti tárolási megoldáshoz kötötten működnek. Óriási előnyük a tartalom rugalmas bővíthetősége. A fejlesztő akkor is új egyed-, tulajdonság- és kapcsolattípusokat illeszthet a meglévőkhöz, amikor pedig már él és működik az adatbázis. Amint az lenni szokott, az előny hátránnyal is jár. A mai adatbáziskezelő az operációs rendszernél is

bonyolultabb rendszer, amelynek számos funkciója a kiválasztott fizikai szerkezeti megoldás(ok)on alapul. Mármint az világos, hogy pl. a grafika tárolási módja nem ugyanaz, mint a szövegé vagy az adaté. Az újféle szerkezet befogadásához az adatbáziskezelőt át kellene írni. Ez roppant nehéz dolog. Így a szerző nem csodálja, hogy ma még a piacon nemigen található igazi multimédia szinten dolgozó általánosított (!) és valódi adatbáziskezelő.

Jelenleg az utolsó megbeszélendő kérdés az ún. **objektum-orientált** adatbázis. Sajnálatos módon e dolog lényegét mindmáig senki sem definiálta - legalábbis nem a szerző által elfogadható módon. Ezért - továbbá az alább elmondottakból is következően - mi sem fogunk formális meghatározással élni. Maga a briliáns gondolat eredetileg nem az ismerethez, hanem annak kezeléséhez kapcsolódik. Azt is el kell mondani, hogy - az új megnevezés dacára - egyáltalán nem valami új elképzelésről van szó. A két kulcsszó az *újrahasználatosság* [reusability] és az *átvihetőség* [transferability]. E két régen ismert, gyakorolni óhajtott, de eddig nem túl nagy sikerrel alkalmazott elv - a spanyolviasz újrafeltalálása - a felesleges munkaismétlés elkerülésére született. Egy triviális példa: ha egyszer valamikor és valahol már megfogalmaztunk egy hibakezelési módot, akkor azt ne találjuk fel újra. Használjuk további feladatainkban is és más körülmények között is.

Nem kell elmagyaráznunk, hogy a közös kép, benyomás mennyire fontos. Így pl. az egységes hibakezelés azt jelenti, hogy nem felülről kényszerítetten, hanem alulról kiindulva óhatatlanul megteremtünk egy egészséges belső szabványt. Az objektum-orientált koncepció annyiban jelentett előrelépést, hogy résztényezők helyett az összefüggő elemek részhalmazára terjesztette ki a két alapelvet. Egy példa: egyetlen objektumként lehet meghatározni az adatbeviteli képet, az azon megjelenő eligazításokat, a téves bevitel hibaüzeneteit stb.

Még szinte friss örömről ne csorbítsa két apró tény. Sem az, hogy néhány mai objektum-orientáltnak átkeresztelt eszközben - szemben a leírt gondolattal - az elemi és nem-transzferábilis tényezőket nevezik objektumoknak. Sőt még az se, hogy a tényezők kombinálása egy objektumba rugalmatlansággal jár. Mert ha az egyik felhasználó más képernyő-elrendezést akar látni, akkor...

Az adatbázisok terén az öröklés [inheritance] és az (egyed)altípus [subtype] a kulcsszó. Mindez az úgymond homogén hierarchiákra vonatkozik. Példa: az adatbázisban van partner, személy és szakértő egyed. A szakértő is személy, az utóbbi altípusa. A személy is partner, annak altípusa. Ámde nem minden partner személy és nem minden személy szakértő. Az egyedaltípusnak nyilván csakis akkor van értelme, ha a szakértőt a nem minden személyt, a személyt a nem az összes partnert jellemző külön ismeretekkel akarjuk leírni. A partner-személy-szakértő egyedtypusok így egy háromszintes homogén hierarchiát alkotnak.

Magát ezt a struktúrát régóta ismerjük és az adatmodellben régen meg tudtuk határozni. Az öröklés ezért nem adatszerkezeti, hanem -kezelési momentum. Az adatbevitel, -lekérdezés, -karbantartás stb. sajátos módja. Ha lekérdezem például a szakértőt, akkor megjelenik annak minden olyan adata is, amely nem magához a szakértő, hanem a személy és a partner egyedhez kapcsolódik. Ha módosítom a személy nevét, akkor a szakértő neve is ugyanarra változik.

Nos, ezen a ponton kell megállnunk, hogy a huncutságot leleplezzük. Egy jól szerkesztett adatbázisnak a szakértő egyedében természetesen nem szerepelhet a név - mert az már jellemzi a személyt, tehát redundancia lenne. Azt pedig, hogy egyes képernyőkön a szakértőnek csak az ilyen minőségben értelmezett adatai jelenjenek meg, vagy a fölérendelt személytől öröklöttek is, nekem kell előre megmondanom. Pontosan úgy, mint az inhomogén hierarchiáknál, amikor is el kell döntenem, hogy egy képen akarom-e látni a személyt és az őt foglalkoztató szervezeti egységet.

Az objektum-orientált gondolat legnagyobb 'felfedezése' egyben a legnagyobb képtelensége is. Az adatbázis úgymond hordozhatóságról van szó. Azért kell az adatbázist objektum-orientáltan megfogalmazni, hogy az illetve annak részei az 'alkalmazások' között átvihetőek, újrafelhasználhatóak legyenek. Mindez pedig nem más, mint egy olyan elméleti számárság, amely a gyakorlati csacsкасágot óhajtja leplezni.

Ha teljesen azonos tartalmi és technikai feltételek között egy adatbázis(rész)t az A környezetből átviszünk a B környezetbe, akkor ez másolás, amit felesleges a 'hordozás' szóval jelölni és semmi köze sincs az objektumokhoz: egy ezer éve ismert technikai műveletről van szó. Adott esetekben csak az absztrakt struktúrát másoljuk, a konkrét ismereteket nem, mert a szegedi és a kecskeméti részlegünk más-más aktuális adatokkal dolgozik. Ha a technikai feltételek eltérőek - vagyis más a tárolási mód, a fizikai szerkezet -, akkor adatkonverzióhoz folyamodunk, mert az adatbázis direkt átvitele lehetetlen. A konverzió [conversion] lényege, hogy a fogalmilag teljesen azonos lényegű (!) ismereteket az egyik ábrázolási és elhelyezési formáról a másikra hozzuk. Eközben a tartalom nem változhat.

Ha pedig az A és a B környezetben az adatbázisnak valamelyik - akárcsak egy picurka - részlete is fogalmi szinten eltér, akkor a hordozhatóság eleve kizárt.

Intermezzo

A hetvenes években kedvelt kifejezés volt a tulajdonság-vándorlás [attribute migration]. Ezen azt értették, hogy valamelyik jellemzőt a korábbi jellemzettől 'átpakolták' egy másikhoz. Magyarul: Az egyed típus adott tulajdonságtípusát nemes egyszerűséggel abból kivették és egy másik egyed típushoz kötötték. Pl. az egységár tulajdonságot a cikk egyed típusból áttették a megrendelés-tételbe. Mindez az ismeretek természetét illető teljes tudatlanságban történt. Hiszen az egységár - minőség. Éppen attól az, hogy minden rendelésben azonos értelmű. A rendelés-tételenként eltérő értékű ár már teljesen más fogalom, más minőség.

Az adatbázis nem hordozható. Azért nem az, mert az adatmodellnek megfelelő módon szervezett lényeg. Ha az adatmodell több helyen teljesen azonos, akkor másolásról van szó. Ha nem az, akkor a másolás kizárt. Ezek után feltesszük a kérdést: akkor mire jó az objektum-orientált adatbázis? A válasz egyszerű: egy újabb - és ismét nem átgondolt - technikai manőverről van szó, amely kizárólag azt hivatott leplezni, hogy eszközeink gyengék, mert nem tiszta elképzeléseken alapulnak.

5.6 ADATRENDSZER ÖSSZEFOGLALÓ

Az információs rendszer tényezői közül az emberi igényeket megtestesítő adat kiemelt fontosságú. Már azért is az, mert az ismeretek az IR-en belül egy olyan külön világot - adatrendszert - alkotnak, amely jóval összetettebb az események, tevékenységek stb. együtteseinél.

Az ismeretek számítógépes ábrázolásának, megjelenítésének, kezelésének ma már számtalan - olykor nagyon is vonzó - módja van. Ma sok embert - érthetően - ez a technikai bűvölet ragad magával. A józan informatikus feladata az, hogy a fecsegő felszín alatt meglássa a mélységeket is. Számára nem az az elsődleges kérdés, hogy miként fogja elrendezni és kezelni a számítógépen az ismereteket. Hanem az, hogy meglássa a természetes és a mesterséges ismeretközlésben az összhangot. Ugyanis nem attól válik mindennapivá, emberivé, elfogadhatóvá a számítógéphasználat, hogy több színt, ikont, ablakot, mozgást stb. varázsolunk a képernyőre. Hanem attól, hogy a számítógépes ismeretek tartalmi elrendezése és a fejünkben lévő kép harmonizál egymással.

Az emberek a mindennapok során mondatokkal érintkeznek egymással. Egy-egy mondatban kijelentéseket tesznek, állítanak valamit valamiről. A mondat két alapvető egysége az alany és az állítmány. (NB.: Az egyéb mondatrészek is állítások. Az „X rendszámú gépkocsi színe piros és típusa Y.” mondat ugyanazt a tartalmat közli, mint az „X rendszámú piros gépkocsi típusa Y.”.)

A lényeg az, hogy egy bennünket érdeklő jelenségnek valamilyen jellemzőjére utalunk. Nem feltétlenül egyre, hanem néha többre is. Gondolatainkat kitágítjuk, egybefűzzük. „Ha a gépkocsi típusa Y, akkor ólommentes benzinnel megy”. „Az ólommentes benzin ára ma annyi...”

A minket érdeklő jelenségeket egyedeknek, azok jellemzőit tulajdonságoknak, viszonyait kapcsolatoknak hívjuk. Mindhárom tényező esetében kétféle szinten gondolkozunk: konkrétan és absztraktban. Eleinte az előbbi a fontos; később a kép megfordul. A pirosban és a kékben van valami közös - tehát kitaláljuk a szín fogalmát. Később viszont úgy kérdezzük, hogy mi a színe? Ha a korábbi dolgok nem elégítenek ki bennünket, akkor kitaláljuk az 'okkeres téglasárga' fogalmat. Ekként a jelenségeket, azok jellemzőit és viszonyait osztályokba soroljuk, tehát egyed-, tulajdonság- és kapcsolattípusokat képzelünk el. Korábbi fogalmaink e típusok előfordulásaivá válnak. A típusnak (pl. szín) a mai előfordulás-halmaza holnap már újabb tételekkel bővíthet.

A mindennapi életben közléseink tárgyait - az egyértelműség feltételezésével - tetszőlegesen azonosítjuk. Név szerint éppen úgy, mint leírás szerint. Hasonló a helyzet a számítógépes ismeretkezelésben is. Szerencsére - ma már - a bankban az ügyintéző hölgy nemcsak a bediktált számlaszámunk (nominatív azonosító), hanem a felsorolt jellemzőink (deskriptív azonosító) alapján is elő tudja keresni a ránk vonatkozó ismereteket - legfeljebb az utóbbi esetben kissé lassabban. Az ismeretek összekapcsolásának viszont mindig feltétele a kapcsolótulajdonság: az a jellemző, amely két egyedtypusban is azonos értéket vesz fel. A mindennapos bajok többsége éppen abból származik, hogy például a gépkocsi egyedben nem vezetik át a tulajdonosra utaló kapcsolótulajdonság értékét. Ezért kapja még ma is a régi tulaj a már régen nem őrá vonatkozó csekket.

Ez az egyetlen példa is jól mutatja, hogy az ismeret tartalmi elrendezése nem tréfadolog. Vajon boldogabb lesz-e bárki is attól a rossz csekk fogadtán, hogy az ismereteit szép színekkel, egérrel, funkciógombokkal kezelték? Az ismereteket a természetes emberi igényeknek megfelelően kell elrendezni. Erre pedig csakis az egyed-, tulajdonság- és kapcsolattípusokat rögzítő adatmodellnek megfelelő, az ezen absztrakt ismereti osztályok által elrendezett adatbázis alkalmas.

Az egy szervezetben, de különböző funkciókban dolgozó emberek másként látják a lényegileg azonos dolgokat. Az egyiknek a személy bérallományát, a másik számára biztosítottat, a harmadiknak ... azt jelenti, hogy... Egyedül az almodell és az azt megalapozó nézet koncepció alkalmas arra, hogy a többszörösen is egyet lássunk. A szemléleteket össze kell egyeztetni és egy adatmodellt illetve adatbázist kell alkotni. Az nem megoldás, hogy szemléletenként mindenkinek egy-egy külön 'adatbázist' készítsünk. Ezzel ugyanis éppen a lényeg - a közös, de egyéni módon használható ismerethalmaz - veszik el. Következik a félreértés, a mellébeszélés, a „jó napot, sógor” effektus. Én azt gondoltam, te azt gondoltad, én azt vittem a számítógépre, te azt kezeled azon... Brrr!

A számítógépes ismeretkezelést manapság az üzleti érdekekből fakadó, félig-meddig szándékos félrevezetés uralja. Nem kizárt, hogy vannak olyan tisztességes szoftverforgalmazók is, akik roppant jóakaratúak, csak éppen nem eléggé értik az ismeretek lelki világát. („A pokolhoz vezető út...”.) Jellemzőbb azonban az, hogy az éppen aktuális szlogennek megfelelően a termékről leveszik az 'elavult' - például relációs - címkét, majd anélkül, hogy bármit is tettek volna, ugyanarra a dologra ráragasztják a 'korszerű' objektum-orientált nyalókát. Ebben az egész piaci fogásban az a 'legmegkapóbb', hogy sem a termék forgalmazójának, sem az alkalmazónak halvány dunsztja sincs a 'relációs' és az 'objektum' dolgokról. És ez így fog folytatódni, még igen hosszú ideig. Addig, ameddig...

Ameddig a felhasználó fel nem ébred. Az adatszerű, a szövegszerű, az egyéb ismeretkezelési módok közül ma még a felhasználó csak a reklámok alapján tud választani. Nem érti, hogy a dBASE és a Clipper állomány- (és nem adatbázis-) kezelő. Nem tudja, hogy mi a szövegszerkesztés és -kezelés különbsége. Nem látja a tallózás-szerűen használható adatbank és a strukturáltan megfogalmazott adatbázis közötti eltérést. Örül a képnek, a hangnak, a grafikának - de azt nem képes felfogni, hogy ezek együttes mai kezelésének mi a holnapi ára.

A becsületes informatikus feladata a felvilágosítás, ami egyáltalán nemcsak a felhasználó, hanem a fejlesztő korrekt tájékoztatását is jelenti. Sokszor éppen az utóbbi ragaszkodik adott megoldásokhoz, miközben az előbbi nyitottabb lenne. E fejezetnek éppen az volt a mondani-valója, hogy nem eszközökben, nem félig értett jelszavakban, hanem az adatrendszer természetében kellene gondolkodni.

ELLENŐRZŐ KÉRDÉSEK - 5

Ebben a fejezetben az a lényeges, hogy az olvasó pontosan értse az egyed, a tulajdonság és a kapcsolat fogalmát. Főleg azt kell látnia, hogy fogalmi szintű dolgokról - és nem megvalósításokról - van szó. Ez a feltétele annak, hogy lássa a különböző ismeretkezelési módok közötti eltéréseket és mérlegelni tudja azok alkalmazhatóságát adott feladatok esetében.

- 501 Egyes adatmodellekben rekordnak, másokban szegmensnek, relációnak vagy éppen objektumnak hívják az egy jelenségre vonatkozó ismeretek együttesét. Ön szerint milyen szintű tényezőkről van szó: fogalmi (1), logikai (2) vagy fizikai (3)? Érvényét veszíti-e az egyed, tulajdonság és kapcsolat hármasa pl. az objektum-orientált megközelítésben?
- 502 Válassza ki a helyes megoldást és érveljen! A mondatok így kezdődnek: „Az autók és az alkatrészek...”. A folytatások:
- 1 - mindig egy egyedtypusban tükrözendők
 - 2 - mindig két egyedtypusban tükrözendők
 - 3 - tükrözése történhet az előző két mód mindegyikén
 - 4 - tükrözése sokszor három egyedtypust igényel.
- 503 Ön milyen módon tükrözné a gépkocsi tulajdonosát?
- E - egyedként
 - T - tulajdonságként
 - M - mindkét módon.
- 504 Tulajdonsága-e (1) a cikk egyedtypusnak a kezelésekor rátett technikai zár, amely a műveletek összeakadását gátolja meg, vagy sem (2)?
- 505 Hogyan nevezik a jelenségek egyedi behatárolására szolgáló jellemzőt, ? Alkalmas-e a Rendszám a gépkocsik egyedi elkülönítésére?
- 506 A gépkocsihoz számos egyéb ismeretféle (pl. baleset, kár, biztosítás) tartozik. Ezeket a gépkocsihoz kell kapcsolni. Ön milyen tulajdonságot választana erre a célra elvileg és gyakorlatilag?
- 507 A kapcsolat fogalmát sokan félreértik. Két jelenség között akkor is látnak ilyen összefüggést, ha azok kölcsönösen többszörös viszonyban vannak egymással. Ön szerint van-e kapcsolat a rendelés és a cikk között?

- 508 Az Ön cégénél a saját vállalata tekinthető-e (1) egyedtípusnak, vagy sem (2)? Más szervezetben lehet-e egyedtípus az Ön cége?
- 509 Ön tanárokról vezet nyilvántartást. Lehet-e (1) vagy sem (2) a tanár egyednek tulajdonsága a foglalkozás jellemző?
- 510 Mondja el saját szavakkal, hogy miért nem kezel adatbázist a Word!
- 511 Az objektum-orientált adatbázis egyik erényének mondják, hogy egyben lehet látni és kezelni pl. a személyre vonatkozó összes ismeretet. Miért féloldalas ez a vélemény? Gondoljon a személy ismereteinek halmazára!

6. AZ IR FELDOLGOZÁSRENDSZERE

Az információs rendszer adat- és feldolgozásvetületének természete közötti eltérések egyikeként a relatív függetlenséget illetve függőséget említettük. Ezen azt kell érteni, hogy bár az adat és a feldolgozás egyaránt absztrakció, az előbbi a maga teljes egészében önállóan is elképzelhető, az utóbbi viszont csak az előbbi függvényében fogható meg.

Ez érthető: az adat a feldolgozás tárgya, ezért egyben annak meghatározója is. Egy ház rajzát el lehet képzelni anélkül, hogy az felépülne (valaha maga a szerző is kergetett hiú ábrándokat) és annak pontos működését előrelátnánk. Fordítva ez nem igaz. Az olvasó pedig nyilván bolondnak vélné azt a szakácsot, aki először eldönti, hogy ő milyen módon (feldolgozás) készíti el az ételt, majd azután kérdezi meg, hogy miből és mit (adat és információ)...

A **feldolgozásrendszer** ezernyi láthatatlan szálon függ az adatrendszertől. Ez egy fontos következménnyel jár. A feldolgozások maguk is rendszert alkotnak az információs rendszeren belül. Ezt a rendszert a különböző absztrakciós szintű feldolgozási tényezők alkotják a rendszerrészekről kezdve az elemi műveletekig. Minden ilyen szinten a feldolgozási tényező kétféle módon is összefügg az adat elemmel. Egyrészt a tevékenység adatbázis-adatra vonatkozik (pl. módosítjuk a partner címét), másrészt a feldolgozásnak része az esemény is, amely szintén mindig adatok formájában tükröződik (a bemeneti eseményt írja le a partner új címe). Tehát a feldolgozási rendszert nem lehet csak önmagában szemlélni. Azt az absztrakció minden (!) szintjén az adatrendszerhez kötötten is látni kell.

Ennek a résznek a célja az, hogy feltárja az IR feldolgozásrendszerének az összefüggéseit, mégpedig oly módon, hogy minden absztrakciós szinten kimutassa az adat- és a feldolgozás-tényezők összefüggéseit.

Tapasztalataink szerint a mindennapi informatikában két gyakori probléma lép fel. Az egyik az, hogy az emberek nem képesek az IR két absztrakt tényezőjét egymástól viszonylag függetlenül szemlélni. A másik az, hogy nem látják meg a közöttük lévő összhangot. Általában jellemző, hogy az adatot és a feldolgozást csak az absztrakció középső szintjén - az adatfolyam síkján, amiről majd később ejtünk szót - képesek együtt szemlélni. Az annál magasabb rendszer(rész) illetve funkciócsoport szinten éppen úgy nem, mint az annál alacsonyabb elemi művelet szinten. Pedig a tudatos - tehát elemzési módszerekkel támogatott - feldolgozás-tervezésben ez az áttekintés nélkülözhetetlen.

6.1 AZ ISMERET ASPEKTUSAI

Először térjünk vissza az IR **szintjei**hez. Az adatmodell (D 5.9) és az adatbázis (D 5.8) definíciói szándékosan **fogalmi** szintűek. Azért, mert szerintünk először a célt kell meghatározni és csak azután ill. ahhoz lehet kiválasztani az eszközt. Arra nincs is módunk ebben a könyvben, amely végül is nem az adatbázisokról szól, hogy a számtalan, nem-fogalmi szintű adatbázis-koncepciót ismertessük. Annyit viszont meg kell tennünk, hogy legalább néhány általános szót ejtsünk a másik két szintről is. Főleg azért, mert az adatbázist sokáig csakis számítógépes lényeknek képelték el és ez a téves nézet mindmáig hat.

A hetvenes években a *mágneslemezek* elterjedését követően szinte eufórikus hangulat alakult ki. Ekkor vált lehetővé, hogy valódi adatbázisokat készítsünk és ekkor kezdett elterjedni maga az adatbázis szó (tudniillik azt megelőzően arra is az adatbank kifejezést használták). A mágneslemezzel vált lehetővé az ismeretek címezése, amire az egyéb adathordozókon, így a *mágnesszalagokon* egyáltalán nem volt lehetőség. Korábban csak ún. kötegelt feldolgozásra volt mód (aminek a lényegét alább majd elmagyarázzuk), ami kizárólag a tiszta állománykezelésen (ld. 5.5 pont) alapult. A szalagos feldolgozásban egészen egyszerűen nem volt fizikai lehetőség az egyik állományból a másik adott tételére való kimutatásra, a transzverzióra. Tehát a kapcsolat fogalom nem is létezhetett, ergo nem is voltak a mai értelemben vett, a kapcsolatokat feltételező adatbázisok.

A címzési lehetőség egy szinte örült versenyt indított el. Kiemelkedően fontos témakörre vált az *állományszervezés* [file-organisation], amely az első komoly adatbáziskezelők megjelenésével *adatbázis-szervezés* [database-organisation] nőtte ki magát. A kérdés az volt, hogy ki tudja a legrafináltabb módon - láncok, mutatók, indexek stb. (mindezek a címzést feltételezik) segítségével - kialakítani az adatbázis tárolási/hozzáférési szerkezetét. (NB.: A tárolás, vagyis elhelyezés mindig egyféle, a hozzáférés viszont lehet többféle is. Ld. például az indexelt-szekvenciális szervezést.) Ebben a korszakban a *fizikai* szinten volt a hangsúly és adatbázisnak a 'komplex (fizikai) szerkezetű' állományegységeket nevezték.

Hamarosan kiderült azonban, hogy a 'komplex szerkezet' kialakításához az is szükséges, hogy felvázolják: milyen állományok milyen logikai kapcsolatokban állnak egymással. Az adatbázisnak ezt a logikai képét *sémának* nevezték. Akkor a séma nemcsak a logikai tényezőket írta le, hanem nagyon keményen a fizikai megvalósítás módjait is. Ezzel alaposan megsértették a logikai adatfüggetlenség (E 3.1) elvét. A logikai tartalom 'bedrótozódott' a fizikaiba. Ez egyrészt eljárási (procedurális) adatkezelésre vezetett (ld. később), másrészt igen nehézkessé vált a tartalmi szerkezet változtatása, mivel az az állományok fizikai átszervezésével [reorganisation] is járhatott és ekkor a programokat át kellett írni.

A nyolcvanas évek elején piaci szinten is elfogadhatóvá vált az ún. *relációs adatbázis* [relational database] koncepciója. Ez éppen úgy forradalmasította az informatikát, mint egy évtizeddel korábban a mágneslemez. Nem magyarázzuk el e koncepció lényegét, csak a legfontosabb vonásait mutatjuk be. A relációs gondolat a logikai adatfüggetlenséget helyezte az előtérbe. A kezelőrendszer - és nem a programozó - feladata, hogy a fizikai megvalósítással bíbelődjön, vagyis elrendezze, kezelje a láncokat, mutatókat stb. Az indexet kivéve az összes többi technikai megoldási módot a mai korszerű adatkezelők el is rejtik a programozó szeme elől. (Csak a rendszerprogramozók tudnak azokhoz férni, ők is csak...) A fejlesztő így komoly terhektől megszabadulva jobban koncentrálhatott magára a tartalomra. Már csak azért is, mert a relációs adatbázisnak egy igen fontos dolgot köszönhetünk: az adatbázis tartalmi optimalizálásának az alapvető módszereit. (Ezt *normalizálásnak* hívják. A normalizálás az adatbázis egyértelműségének, teljességének, de elsősorban redundancia-mentességének a biztosítását szolgáló eljárás. Vö. optimumkritériumok - 4.5 pont).

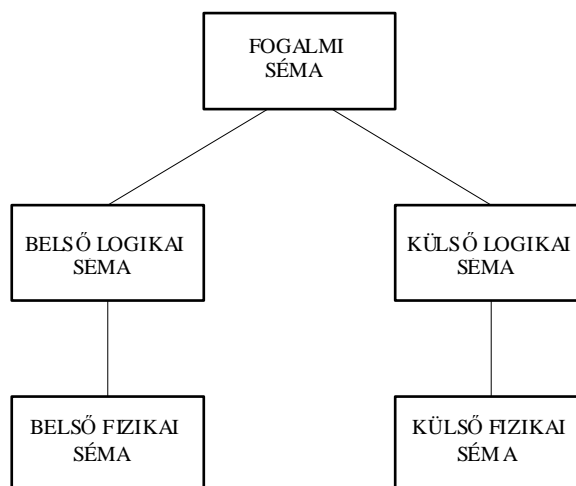
A változást követően a hangsúly a fizikairól áttolódott a *logikai* szintre. Nem a fizikailag 'komplex szerkezetű' állományok együttesét tekintették adatbázisnak - mindenki tudta, hogy igen, valahol van egy összetett fizikai szerkezet is, no de kérem az kit érdekeli? A tartalmilag összetett, vagyis több logikai állományt és azok viszonyait felölelő ismeretegyütteseket hívták adatbázisoknak függetlenül attól, hogy a logikai állományt relációnak, annak szinonimájaként táblának vagy rekordtípusnak hívták-e. A hangsúly a fizikai szervezésről a logikai *adatbázis-tervezésre* [database-design] helyeződött át. Nagyjából és egészéből ez a helyzet jellemzi a mai gyakorlatot, miközben az elmélet már régóta messze előrébb jár.

Bár maga a lényeg már a hetvenes évek óta ismert, a *fogalmi szintű modellek* nálunk csak mostanában kezdenek gyökeret verni. Ugyanis csak lassan látjuk be azt, hogy a logikai szinten megfogalmazott adatbázis szükségszerűen magával hordozza az alkalmazási- és az eszközkörnye-

zetből adódó korlátokat (ld. E 3.2). A relációs koncepció csak lehetőséget ad a normalizálásra, de nem kényszeríti ki azt. Ezért a fejlesztőt semmi sem gátolja meg abban, hogy egyes felhasználók alkalmazási igényeit túldimenzionálva, a többiére nem figyelve szuboptimális - magyarul: rossz - logikai adatbázis-tervet fogalmazzon meg. Maga a reláció mint az adatok logikai egysége adott esetben szuboptimális és végeredményben nem a fogalmi szintű cél, hanem a logikai szintű megvalósítás egyik eszköze. Azért az, mert pl. a megrendeléseket (fogalmi szint) nemcsak relációs, hanem például ún. hálós vagy hierarchikus (logikai szint) adatkezelővel is feldolgozhatjuk.

Persze most az olvasó megkérdezheti, hogy mindez szép, de mi köze van az IR feldolgozási rendszeréhez? Nagyon is sok. Ha valaki nem látja át a fogalmi szintű **adatmodellezés** [data-modelling] fontosságát és lényegét, akkor az nem lesz képes a feldolgozásmodellezés megértésére.

Fel kell figyelni arra, hogy adatmodellezést - és nem adatbázis-modellezést - emlegettünk. Néhányan abban a tévhitben élnek, hogy az adatmodell kizárólag az adatbázisnak a logikai terve. Ez a tévedés kettős. Egyrészt az adatmodell nem logikai, hanem fogalmi szintű lényeg. Másrészt mi csak azt mondtuk, hogy az adatbázis az adatmodellnek megfelelően szervezett. Ez nem jelenti azt, hogy az adatmodellben ne lehetne olyan tényező is, ami nem kerül az adatbázisba. Megfordítva: nem minden az adatbáziskezelővel feldolgozott állomány képezi az adatbázis részét. Minderre majd a következő pontban még visszatérünk. Előbb az a feladatunk, hogy a tiszta képet megalapozzuk. A következő ábra segít ebben bennünket:



6.1 ábra: Az ismeretek aspektusai

Az alábbiakban többször fogjuk használni a **séma** kifejezést. Ezért úgy illik, hogy megmagyarázzuk annak lényegét:

D 6.1 **Sémának a modellnek az adott szerkezeti, tartalmi és formai szabályoknak megfelelő leírását nevezzük.**

A séma tehát nem azonos a modellel, hanem annak reprezentánsa. Különböző szabályokat lehet elképzelni, ezért a modellnek - a szabályok függvényében - egynél több sémája is lehet. Amint a modellnek a része az almodell, ugyanúgy a sémának a kiválasztott része alsémában tükröződik. Most pedig lássuk az ábra magyarázatát!

Az ismereteket illetően a 'kint és bent' illetve a 'fönt és lent' párosokban kell gondolkodni. Az ismeretek két helyen léteznek. Az emberi tudatban illetve az azt 'materializáló' papírokon ('kint'), valamint a számítógépen ('bent'). Nemcsak a számítógépen tárolt, hanem a bizonylatokon lévő ismeret is adat. Ezek után több dolog világos. Az egyik az, hogy a kétféle dolognak harmonizálnia kell egymással. A másik az, hogy a papíron és a számítógépen lévő adat 'tárolása' és kezelése eltér egymástól. Vegyünk csak alapul egy megrendelési bizonylatot! Azon a vevő, a megrendelésfej és a -tétel adatai együtt 'tárolódnak', miközben a számítógépen belül azok jó tervezés esetén több állományban szóródnak szét, rossz tervezés esetén egy, de a papírtól eltérő elrendezésű állományban vannak.

A számítógépen tárolt ismeretek elrendezését a **belső séma** [internal schema] írja le. A számítógépen kívülieket a **külső séma** [external schema] tükrözi. A kettőnek illetve a kettő tényezőinek összhangban kell állniuk egymással. Ezért szükség van egy harmadik, az előbbi kettőt összerendező tényezőre. Ez pedig nem más, mint a **fogalmi séma** [conceptual schema]. Ez egyféle 'fordítókörön' szerepet tölt be a külső igény és a belső megvalósítás között. Egy példa erre az átfordításra: a rendelési bizonylaton szereplő 'megrendelt mennyiség' ismeret a számítógép rendeltétellel állományának a 'Rendmenny' nevű adatában tárolódik. Ez az összekapcsolás ma csak az emberi fejekben, jobb esetben a dokumentáció valamelyik eldugott részében létezik. A korszerű adatkezelésben ezt a kapcsolást a szintén a számítógépen kezelt fogalmi séma testesítené meg. (A feltételes mód arra utal, hogy az elv a gyakorlatban még nem terjedt el.)

Amint látjuk, a fogalmi séma nem pusztán az adatbázis tervének az alapja. A fogalmi modellben olyan ismeretek is szerepelhetnek, amelyek nem kerülnek az adatbázisba, mert azokat ilyen vagy olyan okok miatt nem akarják számítógépen kezelni, de magán a papíron rajta vannak. Megfordítva: akadnak olyan adatok is a számítógépen, amelyek viszont papíron nem kerülnek rögzítésre, viszont azok is a fogalmi sémában találhatók. Hogy ne beszéljünk általánosságban, két példát említünk. A számított adatok egy részét nem tároljuk a számítógépeken, de papíron megjelenítjük őket. Az egyedtípusok belső azonosítói az adatbázis igen fontos kellékei, de papírokon nem jelennek meg. Végül utalnunk kell arra is, hogy mind a papírokon, mind a számítógépeken vannak olyan ellenőrzési, menedzselési stb. - összefoglaló nevükön: technikai - adatok, amelyek elvileg nem képezik részét a fogalmi sémának. Azért nem, mert nem lép fel velük kapcsolatosan 'átfordítási' feladat.

A fogalmi séma ezek szerint nem pusztán a számítógépen kívüli és belüli ismereteknek az összege, hanem azok adott módon elrendezett ötvöze.

D 6.2 Fogalmi sémának a fogalmi adatmodellnek az adott szerkezeti, tartalmi és formai szabályoknak megfelelő leírását nevezzük.

A fentiek szerint a fogalmi szint egyben megegyezik a fogalmi **aspektussal** is. Ez azt jelenti, hogy az ismereteket lehet - sőt: kell - számítógéptől és papíroktól függetlenül is szemlélni. Kell, mert egyébként maga az adathordozó rányomja a bélyegét a tartalomra, ami helytelen, hiszen a számla akkor is az, ha az egyik ill. a másik cég formátumában érkezik hozzánk.

Itt jutottunk el ahhoz a gondolathoz, hogy a külső és belső modellnek egyaránt két szintje van. A tartalom a **logikai**, az elrendezés a **fizikai** szintre tartozik. Ezt a kettőt azért kell szétválasztani, hogy megteremtjük az eszközfüggetlenséget (azaz fizikai függetlenséget - ld. E 3.2). Gondolja meg az olvasó, hogy vajon a számla ilyen-olyan formátumú, de azonos tartalmú kijelzése vajon egy dolog-e vagy több? Ne hamarkodja el a választ, mert az így hangzik: 'is'. Ha erről netán elfeledkezünk, akkor elrendezés-függően írjuk meg a programjainkat és akkor, amikor az egyetlen tartalmat módosítjuk, a fizikai függés miatt több programban kellene a módosításokat végrehajtanunk. Kellene, mert vajon ki fog emlékezni arra, hogy a tartalom mely alkalmazások

melyik kimeneteiben azonos? Hiszen ez a tény sehol sem kerül leírásra a külső modell megbontásának a hiányában.

D 6.3 Külső sémának a külső modell logikai és fizikai szintek szerint megbontott, a fogalmi sémához illeszkedő leírását nevezzük.

A belső modellt és sémát tekintve analóg a helyzet. Az azonos tartalmú, de különböző tárolású és az eltérő helyeken tárolt fizikai szintű adatbázisok összhangja a belső séma kétszintű megbontása nélkül igen nehezen tartható fenn.

D 6.4 Belső sémának a belső modell logikai és fizikai szintek szerint megbontott, a fogalmi sémához illeszkedő leírását nevezzük.

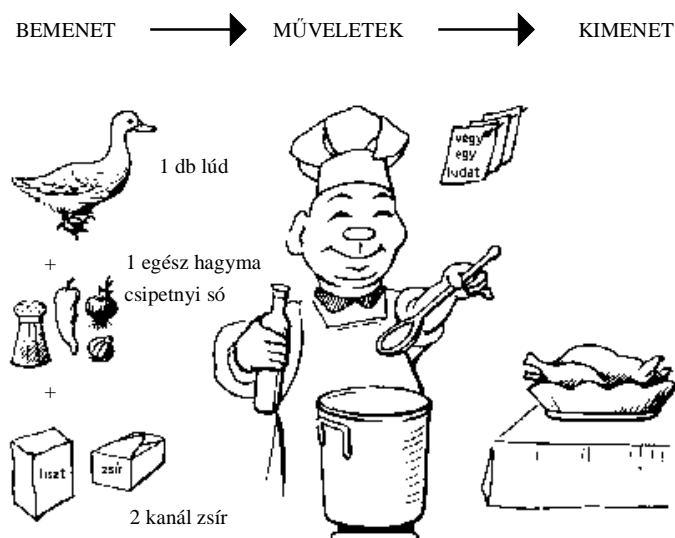
Természetesen a belső modell két aspektusa a logikai (D 3.3) és fizikai (D 3.1) adatbázis-tervnek felel meg, amelyekről már korábban szót ejtettünk. Az a kérdés viszont nyitva maradt, hogy mit kell érteni külső modellen illetve milyen módon kapcsolódik egymáshoz az ismeretek számítógépen belüli szemléletmódja, azaz számítógépes aspektusa a számítógépen kívülihez.

6.2 BEMENET, KIMENET, ÁTMENET

A számítógépen tárolt ismeretek kezeléséhez az szükséges (a folyamatirányító automatákról itt feledkezzünk meg), hogy az ember *kommunikáljon* a géppel.

Régen ez a kommunikáció úgy történt, hogy az adatokat az erre kiképzett emberek ún. másodlagos adathordozókra rögzítették az elsődleges - a bizonylat - alapján. A lyukkártyás, lyukszalagos eszközök korszakában az alkalmazói-felhasználó nem volt közvetlen 'vonalban' a számítógéppel, ezért a *vonalon kívüli* [off-line] megjelöléssel illették ezt a kommunikációs módot. (Ne feledjük, hogy erre még ma is akad bőven példa. No nem a bemeneti oldalon, hanem a kimenetin akkor, amikor például nem közvetlenül történik a nyomtatóra való kiírás.) Amikor a billentyűzeten kalimpálva, a képernyőre meresztett szemmel kezeljük az ismereteket, akkor viszont közvetlen '*vonalban*' [on-line] vagyunk a számítógéppel.

A számítógép őskorában még arra sem volt mód, hogy az ismereteket magán a számítógépen tároljuk: a masinák akkor csak beolvasást, műveletvégzést illetve kiírást végeztek. Például a következő ábra analógiájára:



6.2 ábra: Az ismeretelőállítás elemi tényezői

Természetesen mindenki tudja azt, hogy mi a bemenet és a kimenet. Ámbár még ez sem olyan biztos. Biztos az, ha adunk itt egy meghatározást:

D 6.5 Az adatfeldolgozás tárgyát bemenetnek, eredményét kimenetnek nevezzük. Mindkettő adatokat ölel fel. Ezért beszélhetünk be- és kimeneti adatokról.

Majd az alábbiakban meglátjuk, hogy mégsem felesleges ez az egyszerűnek látszó meghatározás. Itt csak ismételten megerősítjük, hogy az eredmény nem információ, hanem adat illetve azt, hogy a számítógépen mindig adatfeldolgozás folyik. (Most is, amikor a szerző ezt a könyvet írja.)

Az őskorban nem volt lehetőség a mai értelemben vett adatkezelésre, hiszen az ismereteket nem lehetett 'on-line' (a számítógépen belül elérhető módon) tárolni, ezért a feldolgozás csak a vezérlési és adatelőállító műveletek kombinációjából állt. Kis idő múlva megjelentek a mágnesszalagok, amelyek - ha éppen fölrakták őket - már 'on-line' tárolást biztosítottak. Akkoriban a feldolgozás úgy nézett ki, hogy a másodlagos adathordozóról (pl. lyukkártya) mágnesszalagra olvasták az ismereteket és a szalagokat rendezték, másolták, összeválogatták stb.

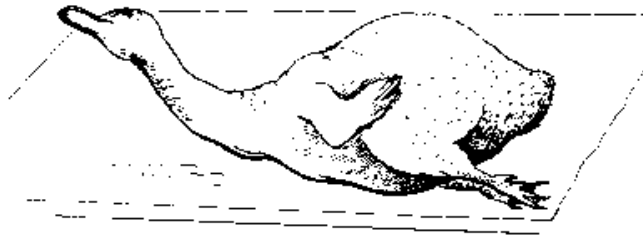
T 6.1 1974-ben az egyik kollégám készített egy rendszert, amelyben a szegény operátornak 21 (!) tekercset kellett ilyen-olyan sorrend szerint felrakogatnia a gépre és lekapkodnia onnan.

A mágnesszalagok arra is szolgáltak, hogy előkészítsék a kimeneteket, illetve azokon tárolják az ún. **átmeneteket**.

D 6.6 Az átmenet az adatfeldolgozás adott tevékenységének olyan belső eredménye, amely egy másik tevékenység tárgyaként szolgál.

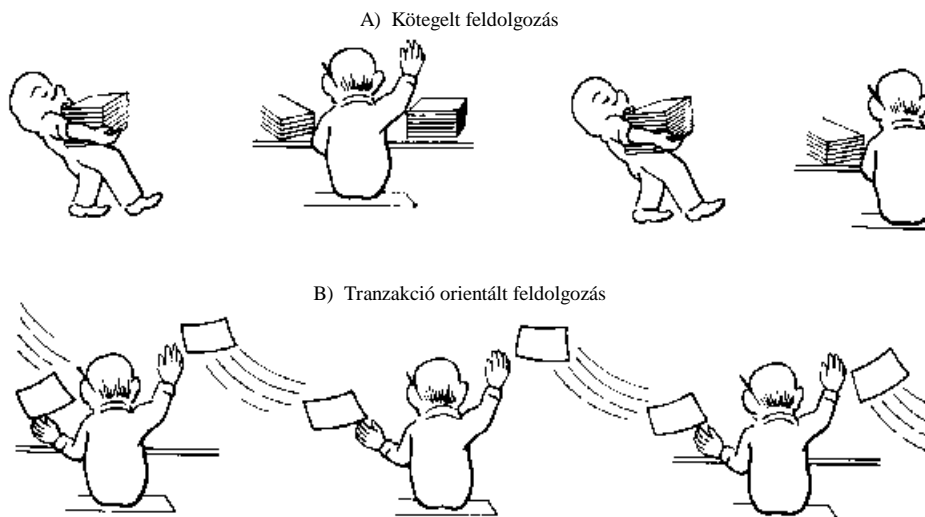
Az átmenetnek - szemben a **bemenettel** [input] és a **kimenettel** [output] nincs angol neve. (Legalább is a szerző csak az átmeneti állományok alább említett két típusának a nevével

találkozott.) Ennek talán az az oka, hogy az átmenet kétféle funkciót tölt be, és ezért az adott feldolgozási tevékenységben azt vagy ki-, vagy bemenetnek tekintik. Ez jogos is annyiban, hogy tágabb értelemben a bemenet és a kimenet bármilyen műveleti absztrakcióra vonatkozhat. Azaz pl. a $2 * 2 = 4$ képletben van két bemenet és egy kimenet.



6.3 ábra: Átmenet

Az átmenet a számunkra igen fontos fogalom, amely segít megvilágítani a két eltérő **feldolgozási mód** lényegét. A bemenet - feldolgozás - kimenet láncnak a bemenet - feldolgozás-1 - átmenet - feldolgozás-2 - ... - kimenet vonalra történő változása adott alapot az ún. **kötegelt** [batch] feldolgozáshoz. Ezen itt és most nem a kötegelt programozást kell érteni, ami azt jelentette, hogy a feldolgozási munkákat [job] sorbaállították. A köteg az együttesen feldolgozandó adatsorok halmaza, például egy 'köteg' számla. A kötegelt feldolgozás azt jelenti, hogy az adatsorok teljes halmazán (!) végrehajtunk egy műveletsort, az eredmény pedig átmeneti állományba kerül, amely kötegen megint végrehajtunk egy műveletsort stb. A következő ábra szemlélteti ezt a megoldást:



6.4 ábra: Kétféle feldolgozási mód

Az ábra a másikkéle feldolgozást is mutatja. Annak magyarázatához viszont az szükséges, hogy előbb megvilágítsuk a **tranzakció** [transaction] lényegét. Ezt a fogalmat az üzleti életből vette át az informatika. A tranzakció jelentése 'ügylet', ami eredetileg éppen hogy sokféle ismeretet felölelő komoly megegyezést takart. Ezzel szemben a számítástechnikában az 'ügylet' a

számítógéphez való fordulást jelenti, a tranzakció pedig az az elemi ismeretsor, amelyen az adott feldolgozást végre kell hajtani. Például ilyen ügylet az, hogy fel kell dolgozni a számlákat; az egyedi számlák pedig tranzakciók. A **kötegelt feldolgozási** módban a számlákat összegyűjtik (kötegelik) és a kötegre egyszerre 'eresztik rá' a feldolgozást. Ezzel szemben a **tranzakció orientált feldolgozás** [transaction oriented processing] esetében a műveleteket minden egyes tételre egyenként hajtják végre.

Itt kell megjegyeznünk, hogy az off-line kommunikáció korszakában nyilván nem egyesével olvasták be pl. a kártyákat, tehát a feldolgozás egyetlen módja a kötegelt volt. A mai on-line kommunikációs időkorszakban a tételeket általában a képernyő előtt ülve egyenként szoktuk kezelni, és ez az 'általában' sokakat arra a tévhitre vezetett, hogy a kötegelt ellentété az on-line feldolgozás. Ez két okból is tévedés, mert nem a kommunikáció módja, hanem az 'egyenként vagy kötegtben' megoldás közüli választás a lényeges. A két ok a következő:

A képernyő előtt ülve (on-line) is lehet indítani kötegelt feldolgozást. Ugyanis bizonyos műveletsorok a *természetük szerint* kötegeltek. Gondoljunk például az elemzésekre, a statisztikákra, az összegzésekre, a leválogatásokra stb. Bár ezeket egyetlen tranzakció indítja, a számítógépen belül átmeneti állományok (kötegek) jönnek létre. Ezzel teljesen ellentétes gyakorlattal is találkozunk. Például valós vagy mondvacsinált korlátokra hivatkozva a keletkező ismereteket nem viszik be a számítógépbe a születésük pillanatában, hanem a változásokat összegyűjtve kötegeket képeznek. Így előfordul, hogy a természete szerint tranzakció-orientált feldolgozást kötegtelen hajtják végre.

A számítástechnikában nem ritka jelenség a feldolgozások megalapozott vagy jogos késleltetése. Minősítés nélkül mondunk egy példát. A szerződéseket nem úgy dolgozzák fel, hogy megszületésükkor azonnal beviszik azokat a gépbe és a feldolgozás végén 'kipottyann' a számítógépből mondjuk a kötvény. A műveletek a bevitel után megszakadnak. A szerződésadatok - ellenőrzés után - egy átmeneti állományba kerülnek. Ezt nevezik **tranzakciós állománynak** [transaction file], mivel ez őrzi a tranzakciókat. A késleltetett tényleges és kötegelt feldolgozásnak - a 'startjelen' kívül - nincs is külső bemenete, mert ez a fájl a belső bemenete.

A feldolgozás másik végén sem ritka a késleltetés. Semmi értelme sem lenne például annak, hogy a kötvényeket egyenként nyomtassuk ki, mert akkor az ehhez szükséges böhöm nagy és sokat tudó nyomtatók ára sohasem fizetődne ki. A kimeneti adatokat a fizikai kimeneti kép (!) szerint elrendezetten - vagyis pl. több kötvény egymás mellett jelenik majd meg - egy olyan átmeneti állományba pakolják, amit **nyomtatási-kép állománynak** [print-image file] neveznek. Erre majd 'ráeresztik' a nyomtatóprogramot, amikor az úgy szükséges. Ez a megoldás akkor is beválí, ha a végső kimenet kiírására több földrajzi helyen van szükség és az adathordozó 'utaztatása' lényegesen olcsóbb, mint a kiírt papírtömegé.

Ezt a pontot egy kellemetlenkedő megjegyzéssel kell zárunk. Hosszú ideig az informatikusok ügyet sem vetettek az adatmodellre ill. a modellezésre. Amikor végre meglátták benne a fantáziát, akkor azonnal túllépték a józan mértékeket és a fogalmat félrehasználták - amint az lenni szokott. Az adatmodell a valóságnak a megalkuvás-mentesnek szánt, fogalmi szintű és egyszeres tükörképe. Erre föl egyesek az adatmodell fogalmat arra a vegyesen logikai-fizikai szintű, sokszor nem is célszerű kompromisszumokkal átítatott valaminek a leírására használják, amely mindig többszörösséget implikál. Ez hamisítás és (ön)félrevezetés.

Az adatmodell nem való arra, hogy abban a be-, át- és kimeneteket tükrözzük! Azért nem, mert ezek a tényezők szükségszerűen az adatbázissal is és egymással is átfednek (többszörösség). Igen sokszor nem-természetes (nem fogalmi szintű) alkotások, amelyek mögött rengeteg technikai alku búvik meg. Az adatmodellbe nem való tényezőknek az abban való feltüntetése zavarja a lényeg áttekintését. A modellezés ilyen félresiklásának pedig az az oka, hogy az adatkezelő rendszerek gyártói tudatosan fogalmi szintűnek állítják be a saját logikai szintű és technikai korlátokkal megterhelt ismeretegységeit. Például azt mondják a relációra, hogy az egyedtípus, holott az utóbbi egy valós jelenség, az előbbi pedig egy korlátos gépi tárolási megoldás.

Van még egy okunk arra, hogy a ..menetet nem tartjuk az adatmodellbe való dolognak. A ..menet általában nem-normalizált, vagyis nem egyetlen jelenség, hanem több összefüggő egyed ismereteit tartalmazza. Például a be- és átmeneti szerződésállományban még nem válnak szét az ügyfél és a tulajdonképpeni szerződés adatai. (Ha ezt megtennék, akkor az átmeneti állományok adatbázist alkotnának az adatbázison belül. De ezt már azért szerencsére nem teszik.)

6.3 ADATBÁZIS ÉS FELDOLGOZÁS

Az adatbázis a fogalmi adatmodell szerint szervezett ismeretek együttese. (NB.: A 'fogalmi' jelző lassan feleslegessé válik, mert az adatmodell mindig ilyen szintű.) Ehhez hozzá kell tennünk, hogy mindig csak a számítógépen tárolt adatokról van szó. Mivel nem osztott és egyidejű használatú, például a könyvtári kartonrendszert nem tekintjük adatbázisnak annak dacára sem, hogy 'szervezett ismeretek együttese'. Másrésztől tautológiának tekintjük a sokak által használt 'on-line adatbázis' kifejezést, mivel a fentiek miatt minden adatbázis - on-line.

On-line, azaz éjjel-nappal elérhető a feldolgozások számára. Csak az a kérdés, hogy az utóbbiak miképpen használják az adatbázist? És akkor, ha a bemenet, a kimenet és az átmenet nem a tulajdonképpeni adatbázis része, akkor vajon mi a voltaképpeni adatbázis tartalma? Válaszoljunk előbb ez utóbbi felvetésre.

A mai informatikusok már nem is használják a régi és jól bevált törzsadat ill. eseményadat (másképpen: tranzakciós adat) kifejezést. Valamiképpen a 'gépen van, vagy nincs gépen' párhuzamban gondolkodnak a változatlan és a változó párossal szemben. Ez kár, sőt olykor kimondottan bajok forrása. Ezért bizony mi szépen visszavezetjük a köztudatba ezt a két fontos fogalmat.

D 6.7 Törzsadatnak nevezzük a valós rendszer viszonylag tartós életű objektumaira vonatkozó viszonylag változatlan ismeretet.

A fenti definíció kapcsán célszerű több szintben gondolkodni. A meghatározás a viszonylagos változatlanságot emeli ki. Az emberek neve viszonylag állandó: nem jellemző, hogy nap mint nap új nevet veszünk fel. A fizetésünk is - sajnos - viszonylag állandó. A testsúlyunk - legalábbis a szerző - viszonylag változó (egy irányban), no de azért akár viszonylag állandónak is tekinthetjük. Ámde:

T 6.2 Volt alkalmam egy olyan adatbázistervet bírálni, amelyben a személy törzsadataként szerepelt az életkor. Mellette ott volt a születési dátuma, a személyi száma és a papír kitöltésének a kelte.

„No comment!” Az értékek változásának foka csak az egyik mérce. Maga a teljes állomány is lehet 'rövidéletű', mint amilyen például a tranzakciós-fájl.

A törzsadat fogalma a legeredetibb állomány-fogalomhoz kapcsolódik, mivel nem tévedünk akkor, ha a törzsadat és a nyilvántartás közé egyenlőségjelet teszünk. Mivel vállalatunkban nyilvántartást kell vezetni az anyagokról, vannak anyag törzsadataink. Ugyanez vonatkozik az eszközökre, a dolgozókra stb.

A viszonylagos állandóság magyarul azt jelenti, hogy a dolgok megváltoznak. A valós jelenségek keletkeznek, átfórmálódnak, letűnnek. Az azokat tükröző információs egységek - az egyedek - ennek megfelelően (némi ütemkésséssel) születnek, módosulnak és meghalnak. Ezeknek a fázisoknak együttesen a szép **egyed-életciklus** [entity life-cycle] nevet adták az érző informati-

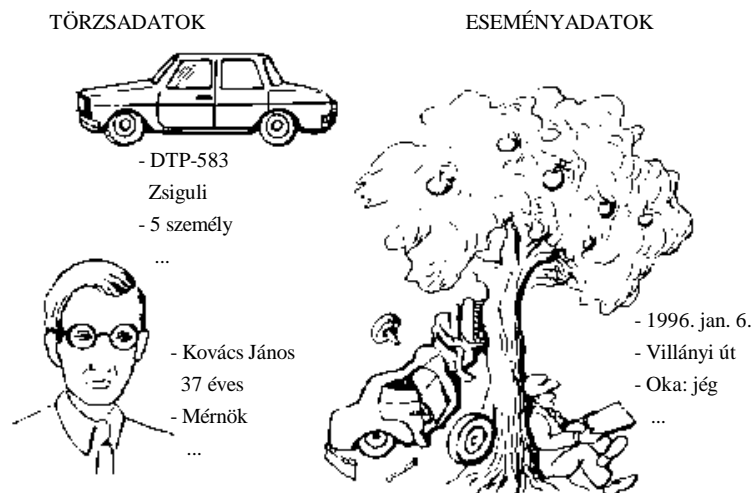
kusok. (NB.: Az ütemkésésnek az első két fázisban csak technikai oka van. Viszont a 'temetést' sokszor jogi okok - vö. megőrzési idő - miatt elnapolják.)

A valós rendszerben a változás fázisai a ténylegesen megtörtént dolgokhoz, a valós eseményekhez kapcsolódnak. Az információs rendszerben a változást az ezeket leíró eseményadatok tükrözik.

D 6.8 Eseményadatnak nevezzük a valós rendszer viszonylag tartós életű objektumainak a változását tükröző ismeretet.

Az esemény- és a törzsadat furcsa kettős viszonyban áll egymással. Egyrészt - szemben az alábbi ábrán mutatottal - a kétféle adat természetesen átfedésben áll. Ha valakinek a neve változik, akkor a jelenséget rögzítő személy és a változást leíró személy-módosítás ismeretsorokban egyaránt szerepelnie kell a 'név' tulajdonságtípusnak. Másrészt vannak olyan események, amelyeknek adatait az adatbázisban óhajtjuk tárolni. Tehát a rájuk vonatkozó ismeretek nemcsak arra szolgálnak, hogy más jelenség ismereteit megváltoztassuk, hanem egyben arra is, hogy magukat az eseményadatokat a számítógépen kezeljük.

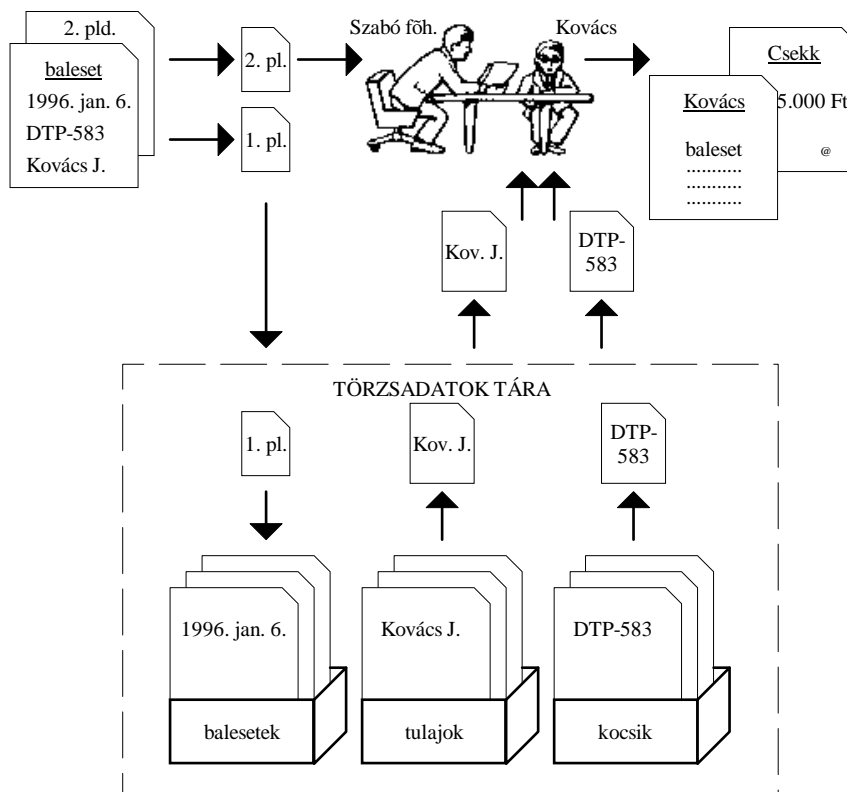
„Ami történt, megtörtént.” - mondjuk gyakran. Amikor az eseményadat törzsadattá is válik, akkor olyan egyedtípus keletkezik (pl. baleset), amelynek életciklusa nem teljes: hiányzik belőle a módosítás szakasza. Éppen elég baj, hogy az informatikusok az érdemi változtatást nem különítik el a *hibajavítástól*. Ugyan technikailag a két feladatot azonos módon hajthatjuk végre, de mennyire más a két feldolgozás elvi tartalma! Elütöttem-e a névben egy betűt és javítom, vagy valóban más lett valakinek a neve - ez csak egyes számítógépesek számára jelenti egyazon informatikai feladatot.



6.5 ábra: Esemény- és törzsadatok

Az eseményadatok általában bizonylatokon (elsődleges adathordozókon) jelennek meg, például egy baleseti és/vagy kárfelvételi jegyzőkönyvön. A törzsadatok - leszámítva azt, hogy a bizalmatlanság jegyében ma még kézzel is vezetik őket - általában a számítógépen tárolódnak. A kérdés az, hogy e kettő miképpen kapcsolódik egymáshoz és mi a törzsadatok szerepe az új ismeretek előállításában?

A korábbi 'bemenet - feldolgozás - kimenet' képszerűen lineáris vonal ma már nem jellemző. A bemeneti adatok csak igen ritka esetekben kerülnek közvetlen módon a kimenetre. Általában 'leparkolnak' a törzsadatok tárában, hogy onnan vegyünk elő őket a további feldolgozások céljaira. Ezzel természetesen egy picit módosult a bemenet és a kimenet korábbi lényege is.

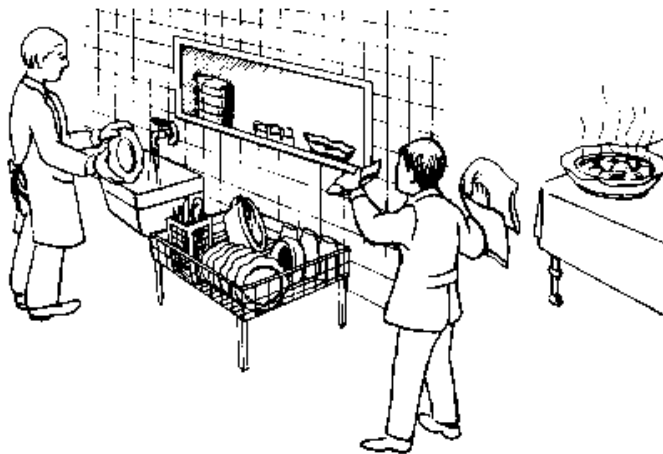


6.6 ábra: A törzsadatok kettős használata

A törzsadatokat őrző adatbázis megjelenésével a feldolgozások két szakasza sok esetben elválhat egymástól. Így a bemenet - feldolgozás - kimenet képletet a bemenet - feldolgozás-1 - adatbázis illetve az adatbázis - feldolgozás-2 - kimenet képlet-páros váltotta fel.

Az első láncot (ld. az ábra baloldalát) **karbantartásnak** [maintenance] hívjuk. A hozzáadás, aktualizálás [update] és törlés műveletsoroknak gyakran nincs is azonnali érdemi kimenete. Hibátlan bemenet esetén csak a tény nyugtázása ill. a képernyőn annak visszatükrözése jelenik meg. Hibásnál a kimenet a bemenet tükörképe, a hiba mineműségére utaló ismeretekkel kiegészítve. Természetesen mindig akadnak kivételek. Például pénzfelvételnél elvárjuk, hogy a tranzakciót bizonylatoltan lássuk vissza. Azonban ideje lenne tudomásul venni, hogy lejárt az a 'régi szép idő', amikor a helyesen bevitt adatokat is leporellókra írták ki - pénz- és papírpocsékoló módon - úgymond 'összeolvasás' végett.

Az első és egyszeres láncsal szemben a második (lásd az ábra jobboldalát) mindig többszörös. Lehet, hogy nem pontosan idézzük a kitételt, amely szerint „a jó gazda kincseiből régít és újat vesz elő”. A második lánc az egyszerűbb esetekben (szelektív) **visszakeresést**, a régi kincs elővételét jelenti. (Bár azt is meg kell jegyezni, hogy az okosan szelektáló keresés többletismeretet adhat. Mert például a 'kik nem fizették...' lekérdezésben nemcsak az alaphoz is hívott törzsadatok jelennek meg a nem-fizetőkre nézve, hanem azt is megtudjuk, hogy hányan nem fizettek.)



6.7 ábra: A feldolgozás fázisai

Itt meg kell állnunk egy pillanatra. A számítástechnikus most az informatikus szemére vetheti, hogy de hiszen a karbantartáshoz is szükséges a visszakeresés. Ezt az informatikus is tudja. Meg ennél még többet is. Például azt is, hogy nem a technikai műveletek mélysíkján, nem a programutasítások szintjén, hanem a felhasználási szándékok nívóján kell megközelíteni az információs rendszernek minden tényezőjét - így a feldolgozást is.

Az új kincs elővétele **ismeret-előállítással** történik. Ennek kétféle módja van. Az egyik a **rutinszerű** származtatási műveletek alkalmazása. A felhasználó azt kéri, hogy keressük ki az X vevő által feladott megrendelések összértékét. Ezt mondja, bár a 'keressük ki' megfogalmazás nem korrekt, hiszen ez az ismeret nem törzsadat, nem kerül tárolásra, ergo nem is kereshető vissza. Viszont - a valóban tárolt és visszakereshető mennyiségi és ár ismeretekből - előállítható rutinszerű módon. A szerző az olvasó helyében nem becsülné le az adatbázis **tallózásszerű** feldolgozását sem. Az adatkincsek között egészen váratlanokra is bukkanhatunk, ha ismerjük az adataink természetét. Kovácsnak két kocsija is van, de nem fizeti a gázszámláját. Szabó beteget jelentett szombaton, közben pedig... Az ikszedik kerületben sok a piros kocsi, a balesetek száma is magas. Még sorolhatnánk a példákat. A helyzet azonban az, hogy a felhasználók még a rutinszerű műveletek lényegét is nehezen tudják megfogalmazni.

Most pedig térjünk át az adatbázis feldolgozásának néhány kérdésére. Mivel a számítógépen tárolt adatbázis állományokban testesül meg, nincsen semmilyen gyakorlati akadálya annak, hogy a fejlesztők állománykezeléssel (ld. 4.5 pont) - és ne adatbáziskezeléssel dolgozzák fel azt. Gyakran előfordul, hogy valós vagy vélt indokok alapján a természete szerint tranzakció-orientált feldolgozás helyett kötegeltet alkalmaznak. Egy példa: Szerződések adatait visszük be folyamatosan a számítógépre. Nagyméretű és bonyolult állomány esetében, főként akkor, ha sok indexet alkalmazunk, a tételbevitel - korszerű eszköz ide, gyors gép oda - félelmetesen lelassulhat. (A kezelő sokszor szervezi újra a nagyméretű indexet.) Ezért csak az azonosító-indexet nyitjuk meg, a többieket nem. A karbantartást elvégezzük, majd 'ráindexelünk' a teljes állományra - ami kötegelt művelet, és amelynek a befejezése előtt a szerződés további feldolgozása lehetetlen. Tehát a fogalmi szintű tevékenységet logikai szinten kompromisszumosan végezzük el.

Az ilyen megalkuvások néha szükségesek és értelmesek, az alapvető gond az, hogy akkor sem adatbázis-szerűen dolgozzunk, amikor az lehetséges lenne.

T 6.3 A korszerű adatkezelők lehetővé teszik a változások automatikus átvezetését, amit propagálásnak hívunk. Kolléganőm kerek-perec kijelentette: „Hát én ezeket sohasem használok!”

Természetesen a kolléganőnek jogában áll nem élnie a valódi adatbázis-szerű feldolgozás egyes lehetőségeivel, bár néha talán kár, hogy azokat nem használja ki. A történet azonban alkalmat ad nekünk egy-két dolog megvilágítására. Nem fogunk elfeledkezni a ‘propagálás’ ismertetéséről sem, azonban messzebbre kell elindulnunk.

Az adatkezelés és -előállítás természet szerint procedurális és deklaratív lehet. **Procedurális** a művelet akkor, ha az alkalmazói programba maga a programozó illeszti be a végrehajtási utasítások sorát. **Deklaratív** akkor, ha ezeket már előre megírták az adatkezelő rendszer készítői, a programozónak nem kell utasításokat összeállítania, hanem az ismeret valamilyen technikai jellemzőjét kell megadnia. Egy példa szemlélteti az eltérést. Régen, ha volt egy dátum jellegű ismeretünk, akkor a programozónak kellett a dátumszerű formát kialakítania és neki kellett megírnia az ún. **validálási** rutinrészét, ami ellenőrizte, hogy a dátum év, hó, nap tétele megfelelő értékeket tartalmaz-e, pl. a hónap helyén nem áll-e 14. Ma már ez nem így történik. A dátum-érvényesítés automatikus és az adatkezelő része. A programozónak csak annyi a dolga - pontosabban ez nem is az övé, hanem az adatbázis-tervezőé -, hogy a vonatkozó adatot dátum-típusúnak deklarálja. Ezt az adatkezelő ‘észreveszi’, és valahányszor a kelezést beviszik vagy módosítják, automatikusan végrehajtja a tartalmi vizsgálatot. A példát folytatva elvileg arra is lehetőség nyílik, hogy a hibás tartalom esetén lezajló hibakijelzési műveletet deklarátívan kezeljék. Csak egyszer kell megírni ezt az eljárást és azt a kezelő részévé tenni, például egy általános hibaszöveg-állományra alapozva.

A feldolgozás procedurális és deklaratív párosa analóg a testreszabott és az általánosított adatbázisával (ld. 4.5 pont). Hasonlít, amennyiben a deklaratív műveletelemek a feldolgozás általánosíthatóságát fokozzák. Ennek előnyeiről pedig nem kell sokat gyözködni az olvasót. Régebben több száz programozó több tízezer kelezés-validálási rutint írt, természetesen eltérő módokon. Az is előfordult, hogy a validálás ‘véletlenül’ elmaradt. De gondoljunk más példára is, amelyben a rutin lényege megváltozhat. Hány meg hány helyen kellett (volna) - és ráadásul azonos módon - átírni mondjuk a cikkszámellenőrzést, ha azt nem deklarátívan, hanem procedurálisan készítették?

Vegyük észre, hogy a logikai adatfüggetlenség (E 3.1) elve rejtőzik e dolgok mélyén. Tegyük fel, hogy valaki valamilyen okból a korábbi dátum-szerű adatot később nem így akarja kezelni - vagy megfordítva. Ez az adatban bekövetkező változás procedurális kezelésnél programkarbantartáshoz vezet. A programokból vagy ki kell venni, vagy azokba be kell tenni a dátum-validálást. Ám deklaratív kezelésnél csak annyi a dolgunk, hogy átdefiniáljuk az adat típusát.

D 6.9 Az ismeretekre vonatkozó ismereteket metaadatoknak nevezzük.

A ‘rendelés kelte adat típusa dátum’ - ismeretre vonatkozó ismeret, metaadat. Nemcsak a felhasználói adat tartalma (a rendelés konkrét dátuma) módosulhat, hanem a fejlesztői adaté, a metaadaté (az adat általános típusa) is. Éppen ezért az informatikusok megfogalmazták a következő elvet:

E 6.1 Minden az ismeretekre vonatkozó ismeretet az adatbázis fogalmi sémájában kell leírni.

Ez az **ún. 100%-os elv** [100 percent principle], ami valóban csak elv, hiszen elméletileg a ‘minden’ nem teljesíthető. Összehasonlításképpen: a régi COBOL vagy más - procedurális - programokban 5%, a korszerű adatkezelőkben 20-25% a sémában leírt metaismeretek aránya - a többi

továbbra is a programban marad. Pedig gondoljuk meg, hogy az alkalmazási-programozás nélküli világ felé haladunk, ami feltételezi a procedurális részek elhagyását úgy, hogy nincs is szükség alkalmazói-programra. (A szerző készített egy olyan mini-adatkezelőt, amely minden kezelési, vezérlési, előállítási műveletet tud - és egyetlen egy sor programot sem kell a használatához leírni!)

Az adatelőállítási művelet maga is deklarálható és általánosítható. Már 70-ben létezett a SYSTEM 2000 nevű rendszer, amelyben volt 'function' típusú adat. Az ilyen adathoz - adatszerűen, állományban, tehát nem programban - meg kellett adni a műveletsort. Például az érték függvény típusú adathoz az 'ár*mennyiség' függvényt, amely automatikusan végrehajtódott, valahányszor az érték adatra mutattak rá. A szerző kolléganője 1986-ban készített egy olyan általánosított statisztikai rendszert (ASTRA), amely képes volt a meglévő exportadatokból bármilyen statisztikai tábla előállítására - programozás nélkül. A felhasználónak lehetősége volt arra, hogy a függvény-állományt karbantartsa. (Az értő olvasó ne tévedjen el ezen a ponton. Itt nem 'paraméterezésről' van szó, mint például egyes lekérdező [query] vagy jelentéskészítő [report writer] rendszerekben, hanem adat formájában bevitt, ekként módosítható és törölhető makrókról. Tehát olyan nyílt rutinokról, amelyek nincsenek a programba fordítva és értelmezési [interpreter] módon hajtódnak végre a futás közben.)

Mindezt részben igazolásként kellett elmondanunk. Ennek a fejezetnek a címe a feldolgozásokról szól, mi pedig unos-untalan az adathoz kanyarodunk vissza. Nem tehetünk másként. Ugyanis a régi hiedelmekkel, a programozási nézetekkel szemben (nem programozót mondtunk, hiszen vannak korszerűen gondolkodó programozók is) fel kell ismerni a következő tételt:

E 6.2 Az adat vezérli a feldolgozást - és nem megfordítva.

Most pedig kanyarodjunk vissza az *automatikus átvezetéshez* [propagation]. Az adatbázisban lévő ismeretek összefüggenek egymással. Előfordul, hogy az egyik változása fogalmilag maga után vonja a másikat, esetleg többet is. Azaz a feldolgozás fogalmi szintjén egy műveletből másik következik úgy, hogy ha azt nem hajtánánk végre, akkor az adatbázis inkonzisztens állapotba kerülne. Példa: Tároljuk a számla összértékét. (Az más kérdés, hogy ezt helyesen tesszük-e?) A számla egyik tételét sztorzírozzák. Ha a változást nem vezetjük át az összérték adatán is, akkor adatbázisunk inkonzisztenssé válik. Más példa: Megváltoztatjuk a cikk egyed típusban a cikkszámot, amely több másik egyed típusban is szerepel. (A kulcsváltoztatás az informatikai 'halálos bűnök' egyike, de azért elkövetjük.) Ha a többi egyedben ezt nem tennénk meg, akkor adatbázisunk széthullana.

Ha tehát a változások átvezetése nem automatikus, akkor az a programozóra marad, aki ember, tehát felelős, tehát az inkonzisztencia veszélye nő. Ezért a jó adatkezelők sok automatikus propagálási segédletet nyújtanak, amelyek közül most csak kettőt említünk. Legyen adott a vevő és a rendelés egyed típus úgy, hogy közöttük a vevő-azonosító teremti meg a kapcsolatot. Ha töröljük például az X azonosítójú vevőt, akkor nincs létjogosultsága azon rendelés egyedeknek, amelyekben a vevő azonosítója szintén X értékű: azokat is törölni kell. (Azért, mert a rendelés kapcsolata a vevő felé kötelező. Más esetben, amikor opcionális a viszony, nem törölni kell az alárendeltet, hanem üríteni a kapcsoló értékét.) A rendeléssel együtt annak tételei is törölendők, vagyis a törlést végig kell vezetni az adatbázis megfelelő részén. Hasonló a helyzet a - halálos bűnként elkövetett - azonosító-módosítás esetében. Ha egy főlérendelt egyedben változik a kulcs, akkor azt át kell írni az adatbázisban olyan mélységig, ameddig az alárendeltek hivatkoznak rá.

A *törlés* és a *módosítás* automatikusan propagálható, ami azáltal valósítható meg, hogy a kezelőrendszer függvényében - a kapcsolatot ill. a kapcsoló adatot ekként deklaráljuk és megadjuk kötelező/opcionális jellegét is. Nincs tehát arra szükség, hogy magunk programozzuk eljárási módon az átvezetést, csak be kell kapcsolnunk a deklaratív kezelési módot. A T 6.3 történetben kolléganőm ezzel a lehetőséggel nem élt.

6.4 AZ SQL ÉS A NAVIGÁLÁS KONCEPCIÓJA

Az általánosított adatkezelők éppen azért nyerték el ezt a címet, hogy rengeteg 'automatizmust' biztosítsanak a fejlesztő számára. A kezelő 'jósági' foka az általánosítás szintjével arányos. Nem közömbös ugyanis, hogy rendszerünk mennyire függ az emberi hibáknak kitett procedurális programozástól.

Azt most ne firtassuk, hogy a kezelő is lehet hibás - a gyártóknak roppantul fontos e hibák elkerülése és mindent meg is tesznek ennek érdekében. Jelen pontban a deklaratív adatkezelés alapjait szeretnénk megvilágítani, mert úgy tűnik, hogy véglegesen ez jelenti a jövőt.

A 70-es években az ún. *hálós adatbázisok* [CODASYL] kapcsán egy fontos fogalom látott napvilágot: ez a **navigáció**. Már felhívtuk a figyelmet arra, hogy az adatbáziskezelésre (ld. 4.5 pont) a transzverzális - kapcsolatok mentén való - mozgás a jellemző. Az egyik egyedtípusból kimutatunk a másikba vagy onnan áttérünk egy másikra esetleg úgy, hogy vissza is kell az eredeti ponthoz találni. Először vegyünk egy egyszerű példát! Ki akarjuk keresni a vevők rendeléseit. Az első vevő megtalálása után a vevőkód alapján áttérünk az ő első rendelésére, majd a továbbiakra (persze akkor, ha vannak). Az adott vevőhöz tartozó utolsó rendelés kikeresése után vissza kell találnunk az első vevőhöz, hogy rátérjünk a másodikra. Ehhez tudnunk kell azt, hogy hol - melyik előfordulásnál - tartottunk a vevő egyedtípuson belül. Ezért, mielőtt a vevő egyedtípust elhagynánk, meg kell jegyeznünk a helyet, vagyis 'le kell szűrnünk egy bóját'.

Ez a feldolgozás régen azon az alapon történt, hogy a kezelőrendszer belső ún. *adatbázis-kulcsot* adott ki minden egyedtípuson belül az előfordulásokra (vagyis a rekordokra). Ezek a kulcsok alapultak a láncok, a mutatók, az indexek is. Az adatbáziskulcs [database key] volt a bója. A programozónak a programban kellett megőriznie egyedtípusonként az utoljára kezelt tétel kulcsát, hogy módja legyen a visszatalálásra. E bóják mentén navigált végig a program az adatbázis vad és zavaros vizein, az eltévedés veszélye nélkül.

Az egyetlen 'szépséghibát' a fizikai függőség jelentette. A kulcs, a mutató stb. fizikai eszköz, amely az adatbázis *átszerkesztésekor* [restructuring], vagyis a logikai szerkezet változásakor maga is módosult. Ha a korábbi mutatót másik váltotta fel vagy az megszűnt, akkor át kellett írni az összes programot, amely ehhez a bójához kötötte a navigálást. (NB.: Az adatbázis időszakos fizikai ún. *átszervezése* [reorganisation] nem járt ezzel a hatással, hiszen csak a mutatók tartalma - adatértéke - változott, azok lényege - az adattétel - nem változott.)

A logikai szintű adatkezelésnek, amit a relációs gondolatnak köszönhetünk, az az alapja, hogy az adatbáziskulcsot a programozó egyáltalán nem látja, nem kell látnia. Az adatkezelő feladata a navigálási bóják megjegyzése, és ezzel roppant nagy teher került le a programozó válláról. Ami még fontosabb, az átszerkesztés ma már - éppen a fentiek miatt - nem jár programmódosítással. Kezdetben azok, akik a relációs koncepció hívei voltak, a navigációt elvetették. Később - ők is - rájöttek arra, hogy a navigáció nem feltétlenül fizikai szintű koncepció, azt meg lehet - sőt meg is kell - logikai szinten fogalmazni. Nem csak a feldobott labda kedvéért jelentjük ki elvként:

E 6.3 „Navigare necesse est.” - Navigálni, azaz hajózni pedig szükséges.

A fenti elvre még visszatérünk. Most arra kell rámutatnunk, hogy a relációs adatkezelő nyelv nagyfokúan deklaratív. Ez nem is csoda, ha ismerjük az ún. **SQL** nyelv [Structured Query Language - strukturált lekérdező nyelv] alapjait.

Az 'eszkűel' eredeti kiejtés szerinti neve 'sequel' [folytatás, következmény], ami több dolgot is elárul. A hetvenes években számos általánosított **lekérdező** [query] és azokkal rokon **jelentés-készítő** [report writer] programcsomag volt a piacon. Mindkettő a feldolgozás procedurális módját váltotta ki a deklaratívvá, bár a metaismeretek megadását nem deklarációnak, hanem 'paraméterezésnek' hívták akkoriban. Az általánosított lekérdező használatához egy formulát kellett kitölteni (paraméterezni), vagyis deklarálni kellett azt, hogy milyen ismeretet, honnan, milyen feltételekkel, milyen egyszerűbb függvényeket alkalmazva stb. kell megjeleníteni a képernyőn és/vagy a nyomtatásban. A jelentés-készítő két dologban tért el az előzőtől, bár a visszakeresés alapjai azonosak. Egyrészt az összegfokozatok és a tablóyszerű elrendezés, másrészt a formai csinosítás volt a paraméterezés lényege. A dolog sokszor úgy működött, hogy a lekérdezővel összeállítottak egy átmeneti, kvázi-nyomtatási-kép állományt, amit azután a jelentés-készítővel - a szó jó értelmében - 'kicsicsáztak' és kiírtattak akár több formában is.

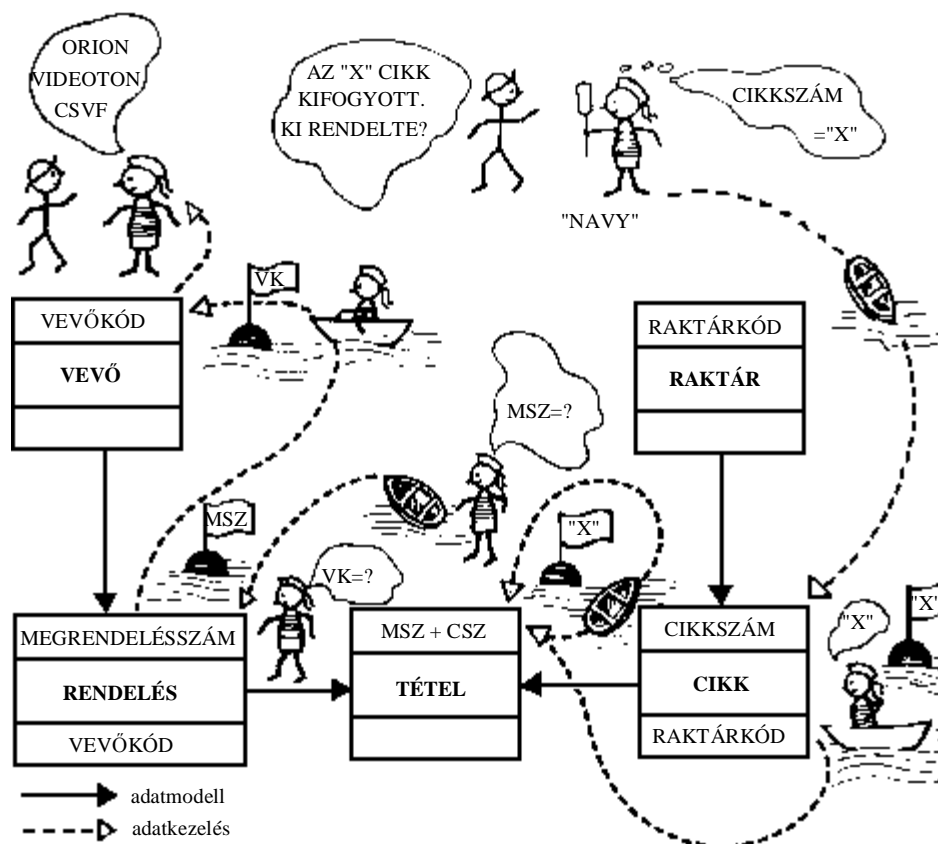
Minden korszerű adatkezelőnek ma is része ez a két funkció, csak tudni kell, hogy mire valók. A report-writer olyan lassú, mint a WORD-nyomtatás - éppen a formátumozás miatt: nem alkalmas gyorsjelentések készítésére. Mindenesetre a szép elemzések, statisztikák összeállításának a terhére leveszi a programozók válláról. Mint ahogyan a lekérdező is, amit viszont sok számítást igénylő vagy nagyon ravasz kiválasztási kombinációkat feltételező visszakereséseknél jobb elkerülni. (A kezelő ui. optimalizálni kezdi a lekérdezést, amitől Isten mentsen.)

Az SQL eredetileg lekérdezőnek készült, és így részben az előző tapasztalatok folyománya (sequel). A visszakeresés közben az egyik tétel megtalálása a másik előzetes kezelésén múlik, az egyik művelet a másik következménye (sequel).

Intermezzo

Sok ismerősöm panaszkodik, hogy a relációs adatkezelővel nehézkes az adat-feldolgozás. A bonyolult feltételeket, az összetett számításokat stb. C nyelvben kell 'hozzáprogramozni' az SQL-ben megírt programrészekhez. Miért van ez? Ilyenkor dörögök. Ha valaki megvesz valamit, aminek ráadásul még a neve is becsületesen elárulja, hogy mit tud, akkor miért kéri számon azt, amire nem jó? Az 'adatkezelő' - elsősorban adatkezelésre és nem adatelőállításra való. Az SQL 'lekérdező' nyelv, és bár ma már ennél sokkal, de sokkal többre képes, alapvető gyökerét, funkcióját, szemléletét stb. a név jól tükrözi.

A relációs nyelvek - csodálatos - matematikai hátterét nem ismertetjük. Nem való arra sem ez a könyv, hogy programutasításokat magyarázzon. Azonban a legalapvetőbb fogalmi szintű műveletek lényegének az ismerete hasznos lehet. Ezért a 6.8 ábrához kapcsolódva nem a matematikai pontosságra, hanem az egyszerűségre törekedve elmagyarázzuk három relációs művelet lényegét. Ezt azért tesszük, hogy az olvasó láthassa: mennyire megváltozott az idők során a feldolgozás lényege.



6.8 ábra: Navigálás az adatmodellben

A felhasználó azt kéri, hogy keressük ki azoknak a vevőknek a neveit, akik az X cikkszámú cikket rendelték, mert lehet, hogy készlethiány miatt a szállítások késni fognak. Ez a feladat. A megoldás pedig a következő:

Ha a készletre is kíváncsiak vagyunk, akkor be kell lépniünk a cikk egyedbe, kikeresve az X előfordulást. A belépés a *select* [kiválasztás] paranccsal történik, amelyben meg kell adni, hogy melyik egyedből történik a kiválasztás (*Select ... From*). A cikkből keresnünk kell azokat a tételeket, amelyekben a cikkszám X. A keresés ezért feltételekkel minősítendő (*Select ... From ... Where*).

A kiválasztás mindig párosul a másik művelettel. A *projection* [kivetítés] egy fogalmi művelet, amely a megvalósításban a select parancsba épül bele, jöllehet a select fogalmilag csak a tétel megkeresését jelenti. A kivetítés azt jelenti, hogy a tulajdonságok közül kiválasztjuk a számunkra szükségeseket. Vegyük észre, hogy a select *extenzionális* tartalmú (a sorok, az egyedelőfordulások közül választunk), a projection pedig *intenzionális* (az oszlopok, a tulajdonságtípusok közül választunk). Válassz ki a tételek közül azokat, amelyekben a cikkszám X és kezeljed a megrendelésszám adatot. 'Relációsul': *Select 'MSZ' From 'Tétel' Where 'CSZ = X'*. (Minden relációs nyelvnek hasonló, de picit más a szintaxisa. Ezért kérjük az értőket, hogy ne a leírási módra, hanem a lényegre ügyeljenek.)

Ha adott a megrendelésszám (MSZ), akkor rá tudunk térni a rendelés egyedre, majd onnan a vevőkóddal (VK) a vevőre - és már kész is a válasz. Az egyedek közötti mozgás - a *transzverzió* - háromféle módon valósítható meg. Az egyik a fenti többszörös select, egy a cikkekre, azt követően egy a tételre stb. A másik az ún. kombinált select, amelyet egy másik példával szemléltetünk.

Tegyük fel, hogy a rendeléstételekből kell navigálnunk a cikkek felé úgy, hogy amikor az egyik tételt elővesszük, akkor ki akarjuk keresni a cikk nevét. Ez a következő parancs-sorral történhet: *Select 'Cnev' From 'Cikk' Where 'CSZ in Cikk = CSZ in Tetel'*. A szintakszis lényegtelen; a lényeg az, hogy feltételként a közös adat - a kapcsolótulajdonság - azonos értékét jelöljük meg.

A transzverzió harmadik módja újabb gondolatoknak ad alapot. A harmadik fogalmi szintű művelet a **join** [összekapcsolás]. Ez azt jelenti, hogy két olyan táblából, amely közös adatot tartalmaz (cikk és tétel) egy harmadikat képezünk úgy, hogy az egyik tábla minden sorát kiegészítjük a másikéval akkor, ha a közös tulajdonság értéke azonos. Tehát képezünk pl. egy olyan új 'tétel-1' táblát, amely a cikkekre nézve már nemcsak a cikkszámot, hanem a megfelelő cikk nevét, árát, raktárát stb. is tartalmazza.

A join alkalmazása közel halálos bűn, mert iszonyatosan lassú művelet, ami a természetéből fakad. Éppen ez vezet bennünket a következő elmélkedésre. Az összekapcsolás azért lassú, mert tipikusan **kötegetelt** művelet, amelynek igazából nincs helye a **tranzakció-orientált** feldolgozáson belül. No, most végre tisztán elmondhatjuk, hogy mi is az a tranzakció-orientált!

A feldolgozás közben nem közömbös a navigálás **iránya**. Ha egy fölérendelt egyedtől lefelé megyünk az alárendelt egyedbe (a cikktől a tételig), akkor az 1:N kapcsolati fok miatt több találatunk lesz (lehet). Ha az alárendeltől ballagunk a fölérendelt felé (a tételtől a cikkig), akkor mindig csak egy találatunk van. Ezért a lefelé haladás mindig kellemetlenebb és mindig egy döntéssel jár.

A valódi tranzakció-orientált feldolgozás lényege az, hogy a tranzakciót végigvezetjük a teljes adatbázison. Példánk esetében amikor az X cikk első tételéhez érünk, akkor - képletesen szólva - bóját rakunk le és azonnal tovább navigálunk a megrendelésszám alapján a rendeléshez, majd a vevőhöz. Ezek után vissza kell térnünk a bójához, áttérve a következő tételre, amit ismét végigvezetünk. Tehát a tételhez kötötten egy feldolgozási **ciklust** szervezünk. Valahányszor lefelé kell haladnunk a modellben, annyiszor van szükségünk ilyen ciklusra.



6.9 ábra: Ciklus (a navigálásban)

Az alternatív megoldás az, hogy kikeressük az X cikk összes tételét és azokból **köteget** képezünk. Ezt tesszük, ha a join műveletet alkalmazzuk. Természetesen a ciklust ezzel sem kerüljük el, csak azt másképpen kell megszerveznünk. Adott esetben ez az utóbbi megoldás is indokolt lehet, de ekkor már nem beszélhetünk valódi tranzakció-orientált feldolgozásról. Azért nem, mert a kapcsolati láncot nem fűzzük végig, hanem azt megszakítjuk azáltal, hogy egy nem-kapcsolatra épülő, kötegelt átmeneti állományt képezünk.

Mivel ez a könyv nem programnyelvet akar tanítani, a parancs-szintű kifejtést itt befejezzük, és áttérünk az átfogóbb dolgokra.

6.5 FELDOLGOZÁSTERVEZÉS

A hetvenes években terjedt el a **strukturált programozás** gondolata (Jackson, Warnier és mások munkája nyomán). Tiszteletben tartva e megközelítés elveit, rá kell mutatnunk arra, hogy az adatbáziskezelés némileg módosította a korábbi programtervezési elképzeléseket. Ez természetes is, mert a **strukturált tervezés** [structured design] még a procedurális programozás korában született, ma pedig már a deklaratív rendszeralkotás idejét éljük.

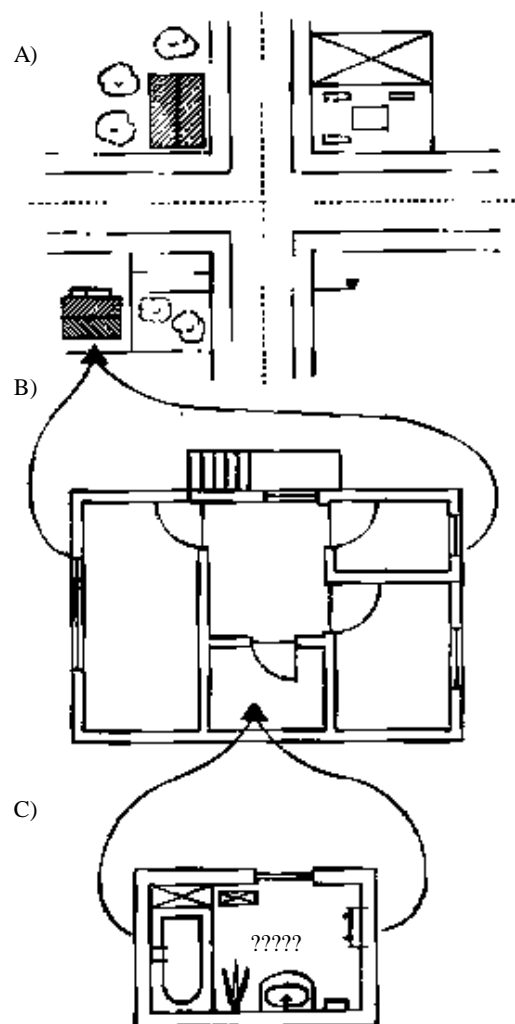
Időközben történt egy másik fontos változás is: elterjedtek az osztott kezelésű adatbázisok. Mármint - ismételjük, hogy minden tiszteletet fenntartva - a régi elvek az új körülményekre nem alkalmazhatóak. A strukturált programozás - mint a neve is mutatja - arra nézve adott irányelveket, hogy **egyetlen** programot miképpen kell célszerűen megírni, vagyis annak művelettrészeit szépen, okosan elrendezni. A strukturált tervezés arra tanított meg bennünket, hogy mely logika szerint kell elrendezni **több** programot, de a lényeg ugyanaz maradt: a program. A bemenetből kiindulva, adott műveleteket végző, kimeneteket produkáló dolog, amit a be- és kimenet szemlélete uralt - és nem az adatbázis. Ebben a nézetben az adatbázis is csak egy hol be-, hol kimenetként szereplő tényező.

A mi szemléletünk egy picit más. A strukturált gondolat elemeit nem elvetve, hanem azokat kiegészítve óhajtjuk bemutatni a saját koncepciónkat úgy, hogy előbb keressük a közöset, majd kitérünk az eltérésekre.

Természetesen vannak 'örök' érvényű princípiumok. Ilyen például a **fentről-lefelé való megbontás** [top-to-bottom] elve. Ez nem újdonság: a mérnöki tudományban már régóta alkalmazott dologról van szó.

Vannak olyan kollégák, akik még ma is az alulról-felfelé történő építkezésben [bottom-up] hisznek. „Az ördög a részletekben rejtőzik.” - mondják és szeretnék előre látni az apróságokat, mielőtt a nagyobb dolognak nekivágnának. Ma már egészen biztosan elmondhatjuk, hogy tévednek. Nem csak azért, mert pl. a házat sem a fürdőszobával kezdjük tervezni. Hanem azért is, mert amit ők 'részletnek' tekintenek, az ma már - az E 6.2 elvnek megfelelően - előre deklarálható illetve deklarálendő dolog, hála az adatmodellezésnek.

A tévedés elvi alapja nem ismeretlen. A hetvenes években 'kis' rendszereket készítettünk. Pontosabban szólva rendszernek a megírandó programhalmazokat neveztük. Valójában nem elrendezett feldolgozási feladatokat mérlegeltünk (bár azt állítottuk), hanem gondolataink a megvalósítás apró technikai részletei körül forogtak, ami az akkori eszközkorlátok miatt nem is volt csoda. Ám nem szabad a szükségből erényt kovácsolni. A zseniális részletmegoldások nem pótolhatják a magasabb szintű átgondolást.



6.10 ábra: Top-to-bottom design (tervezés fentről lefelé)

T 6.4 Vállalatunkban volt egy igen tehetséges programozó. Roppantul értett a részletekhez. Az általa készített 'rendszer' viszont állandó bajok forrása. Azért, mert sem a teljes adatbázishoz, sem a többi feldolgozáshoz nem illeszkedik.

A fentről-lefelé való megbontás koncepciója a **fekete-doboz**éval párosul:



6.11 ábra: Fekete doboz

A *fekete doboz* [black box] nem más, mint az egész és a rész sajátos viszonya, amely meglehetősen dialektikus. Már tudjuk azt, hogy mi az egész, de még nem látjuk a részleteket. Például bizonyosak vagyunk abban, hogy szükségünk lesz az anyag egyedtípusra, de annak tulajdonságait még nem ismerjük. Ámde az a tudásunk, hogy az anyag egyedtípus elmaradhatatlan, a részletből fakad. Például abból, hogy van cikkszám. Ezért a fekete doboz sohasem teljesen sötét - hiszen akkor nem is lehetne beszélni róla.

A 6.10 ábra szerint az első szintű fekete doboz lassan elszürkül. Ez azt jelenti, hogy lesznek fehér részei, miközben újabb - második szintű - fekete dobozokra bukkanunk. A fokozatos megközelítésnek ez az elve mindig követendő. Csak az a baj, hogy a fent és lent, a szint meghatározásában tévedhetünk el. Így például a strukturált programozás elvei szerint az adatkezelésre a *legalacsonyabb* szinten kell sort keríteni, holott - mint meglátjuk - az adatbázis éppen azt sugallja, hogy az ismeretet a *legmagasabb* szinten kellene megragadni.

A régi elvek szerint gondolkodók először a vezérlést határozták meg. Maga az egész strukturált programozás se többet, se kevesebbet nem jelent, mint éppen a vezérlési műveletek helyes elrendezését! Az adatkezelő és -előállító műveletek e gondolkodásban csak részletekként - a vezérlés kivilágosodó dobozában lévő és későbbi kifejtésre szoruló fekete dobozaiként - szerepeltek. Holott az igazság e szemléletnek éppen az ellentettjében rejtőzik: az adatkezelő műveletek logikus sora határozza meg a vezérlést.

A 6.8 ábra feladatának a kiegészítésével és új, más szemszögű magyarázatával fogjuk igazolni a fenti kitételeket. Tegyük fel, hogy a feladat így módosul: Kérjük azon vevők neveit, akik az X cikkből május tizennegyediké előtt rendeltek úgy, hogy látni akarjuk az azévi, de a megadott dátum előtti rendelések vevőnkénti összértékét is az X cikkekre vonatkozóan. Magyarul: csak a dátum előtti tételeket kell számbavenni, de azok értékét vevőnként összegezve kell kimutatni. Tehát a bemenet kettős: cikkszám='X' és rendelésdátum<'május 14'. A kimenet kettős: vevőnév és szelektív rendelésösszeg.

A korábban leírt navigációs útvonalról - cikk/tétel/rendelés/vevő - nem lehet letérni. Ha például a vevő oldaláról indulnánk ki, akkor számos olyan vevővel és rendeléssel kellene foglalkoznunk, amelynek semmi köze sincs az X számú cikkhez. Volt egy fekete dobozunk (szelektív X cikk lekérdezés), amelynek a belsejét még nem láttuk, de - és enélkül fekete doboz nem létezhet -

a bemenet és a kimenet adott volt. (Ha a szándék - kimenet - és annak alapja - bemenet - egyaránt ismeretlen, akkor nem a 'fekete doboz', hanem a 'töksötétség' kifejezés a helyénvaló...) A doboz szürkévé vált, amint a bemenet és a kimenet felmérése után az *egyedek kezelésének a sorrendjét* felmértük. (NB.: Bizonyos esetekben - így példánkban is - a bemeneti, máskor viszont a kimeneti egyed típus a fogódzó pont. A navigálás valahol belép az adatbázisba és valahol elhagyja azt.)

A fentiek után már világos, hogy a vezérlést és a vezérelt műveletrészek sorát az egyed-típusok elrendezése és a belépési pont határozza meg. Szó sincs arról, hogy önfejlően készítünk egy programtervet, amelyben az egyik egyed típusról a másikra ízlés szerint váltogatunk! Bár még messze nincs kész a programterv, a fekete doboz már szürkévé vált úgy, hogy a vezérlési műveletek javarészét - így tehát a feldolgozás makrostruktúráját - a hozzáférési lehetőség szabta meg.

A fiatalok talán még nem ismerik a **programvázlat** [program suit] fogalmát. Azért nem, mert sokan elfeledkeznek a fekete doboz lényegéről, azt csak kvázi jelszóként emlegetik, és azonnal a részletekben gondolkodnak - keretek helyett. A programvázlat olyan szintaktikailag és szemantikailag helyesen megírt, nem csak lefordítható, hanem futtatható művelet sor, amely alkalmas gondolkodásunk helyességének a tesztelésére. A 6.8 ábrával kapcsolatos eredeti feladat pontosan ugyanazt a navigálást követeli meg, mint a kibővített. Ezért 'leprogramozhatjuk' és tesztelhetjük magát a navigálást anélkül, hogy a részletekre ügyelnénk (azaz a kiválasztási ismérv érvényesítése, a kijelzendő adatok tekintetbe vétele nélkül.)

No de lépünk tovább! A *szelekció* és az *adatelőállítás* helye adott, ha ismert maga a számítási képlet, amit előre - és nem a program 'kitalálása' során - kell megadni. Az X cikkből megrendelt tételek összértékéhez ismerni kell az árat (legyen az mondjuk a cikk egyedben lévő egységár) és a megrendelt mennyiség adatot (ami nyilván a tételben van). Mikor tudjuk kiszámítani a tétel értékét? Ha mindkét számítási tényező már a rendelkezésünkre áll. Mivel először a cikket érjük el, onnan megkapjuk az árat. A mennyiséget viszont a tételnél tudjuk meg: tehát itt a kellő pillanat a tételérték kiszámítására. Amikor pedig a rendelés felé lépünk, akkor derül ki, hogy a dátumnak megfelelően a tételértéket bele kell-e számítani a kijelzendő végső összegbe, vagy sem.

A karácsonyfa feldíszítéséről és kivilágításáról van szó. Aminek - talán nem is annyira meglepő módon - az a feltétele, hogy legyen karácsonyfa. Az előzetes programvázlat fokozatosan kibővíthető a már nem-meghatározó részletekkel. Ezért úgy véljük, hogy a programtervezésnek nem a hierarchikusan strukturált programszerkezeti ábra a megfelelő alapja, amelyben csak éppen a lényeg - az adatokkal való kapcsolatot - nem lehet tükrözni. Az adatkezelési gráf a célszerű alap. A 6.8 ábra voltaképpen egy ilyen gráfot mutat.

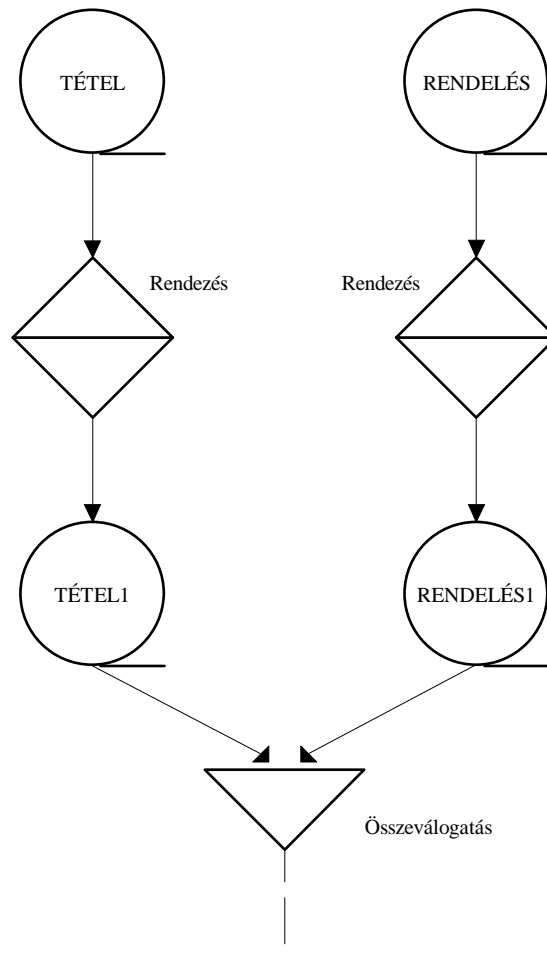
D 6.10 Adatkezelési gráfnak az adatmodell megfelelő részének a kezelési (navigálási) útvonal megjelölésével kiegészített diagramját hívjuk.

Azért nem lenne helyes, ha a mosdóvízzel együtt kiöntenénk a gyereket is. Meg kell mondanunk, hogy a strukturált programozásnak a modularításra, a célszerű elrendezésre stb. vonatkozó alapelvei természetesen ma is érvényesek. Csak az a részelt avult el, miszerint az adatkezelési eljárásokat a hierarchiának a legalacsonyabb szintjén kell megfogalmazni.

Az éles szemű olvasó nyilván észrevette, hogy a navigálás koncepciója nem is annyira nagy újdonság, mint amilyennek azt a fentiekben beállítottuk. Nem állt szándékunkban a félrevezetés, mert tényleg van új momentum is a navigálásban. A régre pedig azonnal rávilágítunk.

A hetvenes években a számítástechnikai kellékeket árusító boltokban lehetett beszerezni a *rendszer- és program-folyamatábrák* rajzolását elősegítő cellulóz-sablonokat. Manapság ugyanerre a célra jópofa kis programok szolgálnak. (Azt csak futtában említjük meg, hogy a gyakorlatban egyik ábrafélét sem rajzolták meg elvileg helyesen. Azért nem, mert a csakis a műveletek felvázolását célzó programábrába eszközjeleket, a nagy összefüggéseket tükröző rendszerábrába pedig - ad-hoc módon - műveleteket, pl. feltételeket tettek.) Program-folyamat-ábrát ma már

nemigen használnak; különben is az annyira a részleteket tükrözi, hogy ismertetése semmiképpen sem való a könyvünkbe. A rendszer-folyamat-ábra helyét pedig átvette egy rokon célú, de ki-fejezőbb technika.



6.12 ábra: A régi rendszer-folyamatábra

Az új technikát **adatfolyam** [data-flow] **diagram**nak nevezik. Az ismertetés előtt legyen szabad kitérnünk ezeknek az ábrázolási megoldásoknak a céljára. Egyúttal azt is bevalljuk, hogy saját ‘fentről-lefelé’ elvünket ezúttal magunk is megsértettük, mert ‘alulról-felfelé’ jutottunk el az indokláshoz. Ennek didaktikai okát mindjárt megvilágítjuk.

Az információs rendszer tényezőit össze kell hangolni. Tényezőcsoportokon belül is, azok között is. Az adatrendszer kialakítására a modellezés szolgál (ld. az előző fejezetet). Az adatmodell a feldolgozásoktól függetlenül is kialakítható. A vevő, a rendelés, a tétel, a cikk viszonya nem függ attól, hogy miképpen kell majd az ismereteiket feldolgozni. Ennek a gondolatmenetnek a mintájára régen azt hittük, hogy a feldolgozásrendszer is önállóan modellezhető.

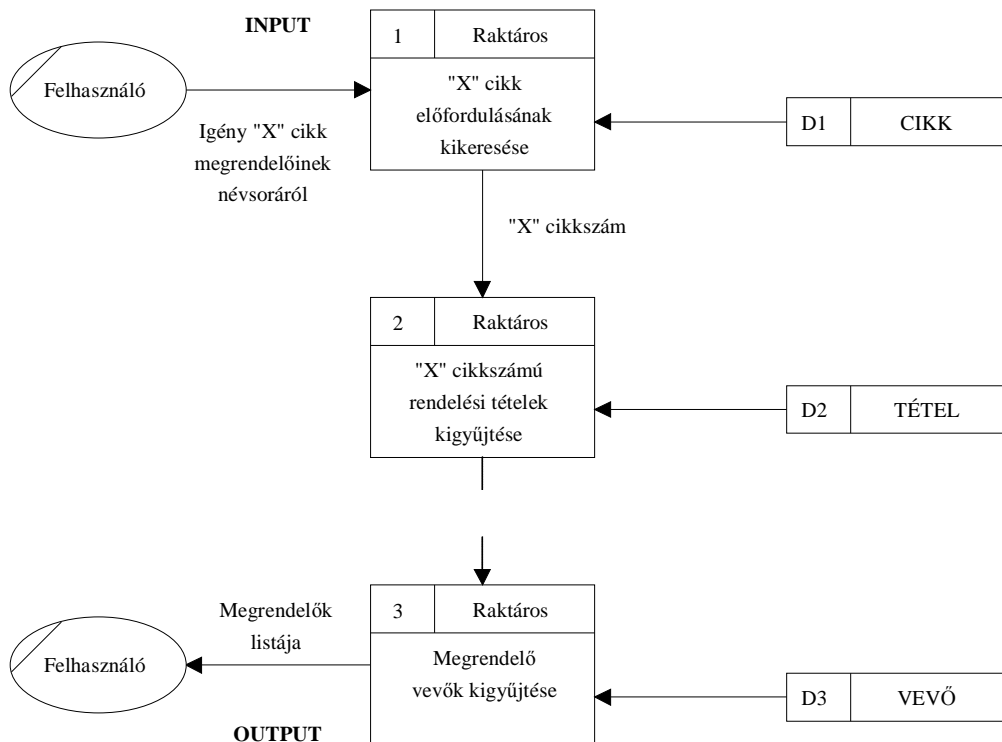
Nem az. A feldolgozásrendszer modellje csak az adatmodell kialakítása után és csakis azzal összefüggésben építhető fel. Ennek a tételnek az igazságát majd az alábbiak is igazolni fogják. A

fentiekben éppen azért kellett előbb egy picit a mélybe ásni, hogy az olvasó legalábbis megérezze az üzenetünk lényegét. Most pedig térjünk vissza a feldolgozástervezés csínjaira.

A feldolgozás események és tevékenységek láncolata. A tevékenységek az adatokra vonatkoznak ('adatfeldolgozás'). Az események pedig olyan sajátos tényezők, amelyek kiváltják/lezárlják a tevékenységeket, de ugyanakkor maguk is adatokkal leírható jelenségek. A 2.6 ábra mutatja ezeket az összefüggéseket. A feldolgozás legelső kiváltó mozzanata a **bemenet**(i esemény), legvégső lezáró momentuma a **kimenet**(i esemény). A bemenet és a kimenet maga is olyan elem, amely tevékenység(ek) tárgya. Ezért az adatmodell, a bemenetek és a kimenetek birtokában a feldolgozástervezőnek el kell rendeznie e tényezőket illetve a rájuk vonatkozó tevékenységeket. Végeredményben ki kell alakítania a tevékenységek olyan **logikai** sorrendjét, amely egyben a végrehajtás **időbeli** rendjét is tükrözi. (A 2.6 ábra képzeletbeli 'vízszintes tengelye' az idő koordinátának felel meg.)

A feldolgozás részét képező adatkezelési műveletek során az adatok az egyik tényezőtől a másikba vándorolnak: például a bemenetről az adatbázisba, majd onnan a kimenetre (ld. 6.6 ábra). Ezt a vándorlást többféle szinten és módon kell és lehet megfogni. Az adatfolyam a feldolgozástervezésnek egy olyan hasznos eszköze, amely egy bizonyos módon, az absztrakció középső szintjén fogja meg a bemenet - feldolgozás - adatbázis - feldolgozás - kimenet lánc lényegét.

Először a 'középső szint' kitéltet kell megmagyaráznunk. A tervezésnek van 'magas' szintje, amire már a fekete-dobozzal kapcsolatosan impliciten utaltunk. A lényegét majd a következő pontban fejtjük ki éppen úgy, mint a 'mély' szint részleteit. Az utóbbival kapcsolatosan itt csak annyit jegyzünk meg, hogy az adatfolyam diagram a vezérlési és adatkezelési műveletek tükrözésére szolgál, az adatelőállítási műveletekre nem való.



6.13 ábra: Adatfolyam-diagram

A középső szint maga is kettős, mint ahogyan az adatmodell-diagram is két szinten tükrözhető. Az 5.4 ábra megfogalmazható lett volna úgy is, hogy csak az egyed- és kapcsolattípusokat tüntetjük fel, a tulajdonságtípusokat nem. Analóg a helyzet az adatfolyam-diagramnál. A 6.13 ábrában csak a tevékenységeket és a nagyobb adategységeket vázoltuk fel. Viszont készíthettünk volna olyan ábrát is, amelyen a kezelt adatok (tulajdonságtípusok) is megjelennek. Csakhogy a két ábrázolási módra (5.4 és 6.13 ábra) közösen jellemző, hogy nemigen kívánatos a részletek megadása, mert azok által az ábra zsúfolttá, áttekinthetetlenné válik. Pl. minél több a részlet, nyilván annál nagyobb az ábra, tehát több lapos lesz és a benne való keresgélés nem túlzottan vonzó. Ezért - legalábbis papíron - mi az áttekintő ábrák hívei vagyunk, noha kétségtelen, hogy a számítógépen a részlet ablakának a megnyitása nagyon tetszetős és praktikus lehetőség.

Az adatfolyam-diagram a feldolgozástervezés hasznos eszköze. Ennek dacára a szerző saját gyakorlatában az adatkezelési gráfot részesíti előnyben (6.8 ábra). Kisebb részben azért, mert azon feltüntethetők az adatelőállítási műveletek is (bár éppen ezt a vonatkozó ábra nem mutatja). Nagyobb részben azért, mert az adatfolyam-diagram nem mutatja magát az adatmodell-részletet, amit pedig nem árt látni pl. a 'felfelé/lefelé' való navigálás eltérő hatása, az alternatív útválasztási lehetőségek mérlegelése stb. miatt. Szubjektíven az az érzésünk, hogy a gráf az adat és a tevékenység viszonyát egy fokkal jobban (mélyebben és közvetlenebb módon) tükrözi, mint az adatfolyam ábra.

6.6 MAKRO- ÉS MIKROTERVEZÉS

Az adatfolyam és az adatkezelési gráf a feldolgozásrendszer középső szintű tükrözésére, modellezésére alkalmas. Ma a magasabb szintű makrotervezéssel és az alacsonyabb szintű mikrotervezéssel nem sokan törődnek. Többek között azért nem, mert a fejlesztőeszközök e területeken nem nyújtanak támogatást. Ezért az alább elmondottakat nyugodtan lehet hiánypótlásnak is tekinteni. Az alábbiakban még a közepes szintű tervre is vissza fogunk térni.

6.6.1 Makrotervezés

A **makrotervezés** a tevékenységek absztrakt rendszerének a megalkotását jelenti. A 2.7 ábra kapcsán elmondtuk, hogy a rendszer rendszerrészekre, ezeken belül funkciókra, majd tevékenységekre tagolandó. Az olvasó nyilván igen meg fog lepődni azon, hogy ilyen lebontásokat a gyakorlatban nemigen használnak. A rendszerrészek - amiket egyébként is 'rendszereknek' hívnak - meghatározása az SOP [Sit Off the Pant = hasraütés] módszerrel történik. Továbbá a szerző ma nemigen találkozik olyan dokumentációval, amelyben lenne egy funkcionális áttekintő diagram. „Kapcsold be a gépet, és...” - ez a dokumentáció.

A tudatos tervezésben a fentről-lefelé elvet követjük. Ezért először nekünk is azt kell megvizsgálunk, hogy miképpen történik viszonylag jó és megbízható módon a rendszerek **rendszer-részekre** való megbontása. Manapság ugyanis a részeket szubjektíven és rosszul határozzák meg, holott létezik már objektív és jó megoldás is.

	1	2	3	4	5	6
VEVŐNYILVÁNTARTÁS	M	M	-	-	-	-
CIKKNYILVÁNTARTÁS	-	-	M	M	R	R
RENDELÉSNYILVÁNTARTÁS	M	R	R	R	M	R
TERMELÉSI TERV	-	-	R	-	R	-
KÉSZLETGAZDÁLKODÁS	-	-	R	M	-	M
DURVAPROGRAM	-	-	R	-	R	-

1 = VEVŐ	4 = CIKK-ÁR
2 = SZERZŐDÉS	5 = RENDELÉS
3 = CIKK	6 = KÉSZLET

6.14 ábra: Funkció-egyed mátrix

A 6.14 ábra egy ún. **funkció-egyed mátrixot** mutat. Ebben a táblázatban az egyik irányban a funkciók, a másikban a nagyobb adategységek jelennek meg. Nem téved az olvasó, ha az adat-egységet az egyedtéppussal azonosnak tekinti. Csak azért nem használtuk ezt a kifejezést, mert az ilyenfajta mátrixok már azt megelőzően is alkalmazhatók, mielőtt az egyedeket pontosan kijelöltük volna. (Az természetesen közömbös, hogy a sorok vagy az oszlopok tartalmazzák-e az adategységeket.) A sor és az oszlop találkozásánál feltüntetett betű azt mutatja, hogy a funkció használja-e az adategységet, vagy sem ('-'), és ha igen, akkor milyen módon ('M' - karbantartás, 'R' - visszakeresés).

Az ilyen mátrixokból sok minden kiderül az okos informatikus számára. Így például láthatja, hogy a durvaprogramozási funkciónak előfeltétele a cikk- és a rendelésnyilvántartás. Ugyanis ahhoz, hogy a cikk- és a rendelésadatokat ez a funkció visszakereshesse, azokon előbb karbantartást kellett végezni. (Itt nem árt visszalapozni a T 3.2 történetre.) A mátrix a felelősség ellen is megvéd. Ha a mátrix valamelyik oszlopában vagy sorában csupa '-' bejegyzés szerepelne, akkor az azt jelezné, hogy valamelyik funkció nem csinál semmit ill. valamelyik adategységet nem használják. Még tovább megyünk. Ha az egyszerű 'M' betű helyett a karbantartás módjára (bevitel, módosítás, törlés) utaló jelölést teszünk, akkor láthatjuk az adategységek korábban emlegetett életciklusát. Mert mit is tudnánk kezdeni egy olyan egyeddel, amelyre nem vonatkozik például beviteli feladatú funkció? Hogyan fogjuk lekérdezni azt, ami nincs is?

A 6.14 ábrában szándékosan az egyszerűbb megoldást alkalmaztuk. Ugyanis azon alapul az IBM által kimunkált **BSP** [Business Systems Planning - vállalati rendszertervezés] technika. A részletek helyett csak a lényegét magyarázzuk el. utalva arra a korábbi kitételünkre, hogy a tervezés és az elemzés egymástól elválaszthatatlan momentumok. Nos, a BSP egy igen jó elemzési módszer. A 6.14 ábra mátrixából a visszakereséseket kivéve és a táblát átrendezve egy új képet kapunk:

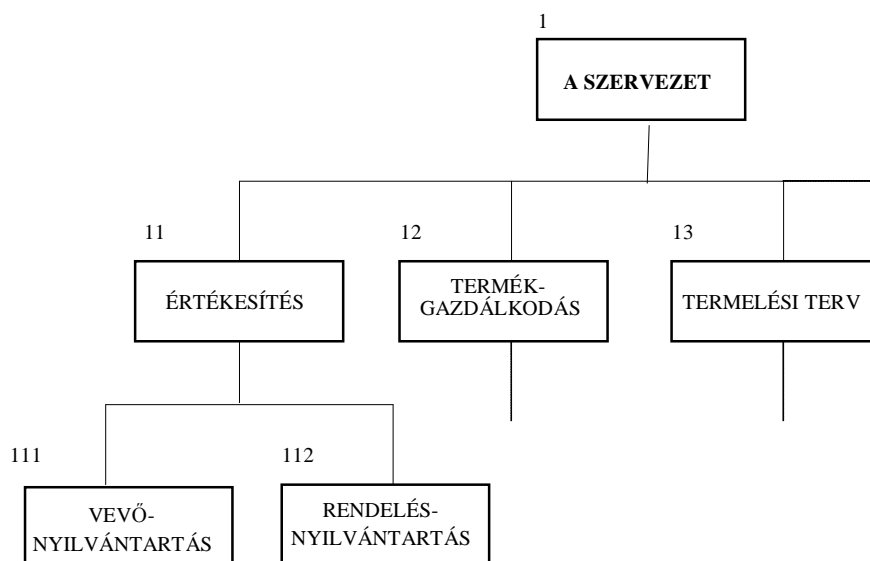
	1	2	5	3	4	6
VEVŐNYILVÁNTARTÁS	M	M				
RENDELÉSNYILVÁNTARTÁS	M		M			
CIKKNYILVÁNTARTÁS				M	M	
KÉSZLETGAZDÁLKODÁS					M	M
TERMELÉSI TERV						
DURVAPROGRAM						

1 = VEVŐ	4 = CIKK-ÁR
2 = SZERZŐDÉS	5 = RENDELÉS
3 = CIKK	6 = KÉSZLET

6.15 ábra: Klaszterizált funkció-egyed mátrix

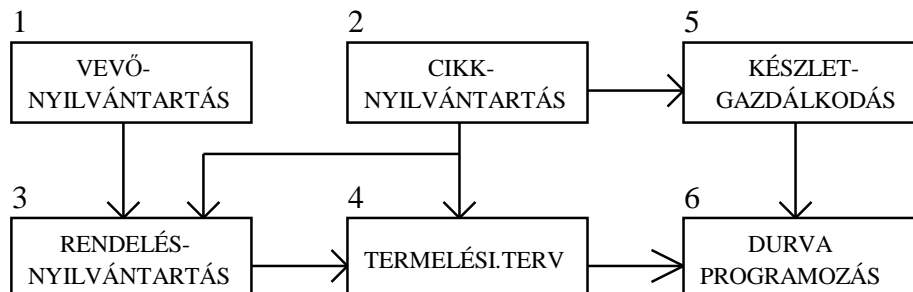
Az átrendezésre a klaszter-analízis szolgál. A mátrixból most már világosan látható, hogy bizonyos funkciókörök bizonyos adatkörökhöz kapcsolódnak. (Az elrendezés nem mindig ennyire tiszta, de az elválasztódás a legtöbbször éles.) A felhasználók - vezetők, alkalmazók, fejlesztők - gondolkodhatnak *szubjektíven* elképzelt rendszerrészekben, de a mátrix *objektíven* mutatja a ténylegeseket. Ha pedig az előző ábrára is figyelünk, akkor még a sorrendeket is meg tudjuk adni.

A minifeladat funkcionális hierarchiája a következő:



6.16 ábra: Feldolgozás-hierarchia

A BSP arra alkalmas rész módszer, hogy kimutassa a valós rendszerrészeket (a szervezeten belüli első szint) illetve feltárja a feldolgozásrendszer mélyebb (ld. a további szinteket) elrendezését. Mindez a funkciók makroszintű szerkezetének a viszonylag objektív, ámde *statikus* képét eredményezi. Azonban a tervezőnek a *dinamikus* összefüggésekre is gondolnia kell, mivel a feldolgozások időben - az egyik a másikat követően - zajlanak le.



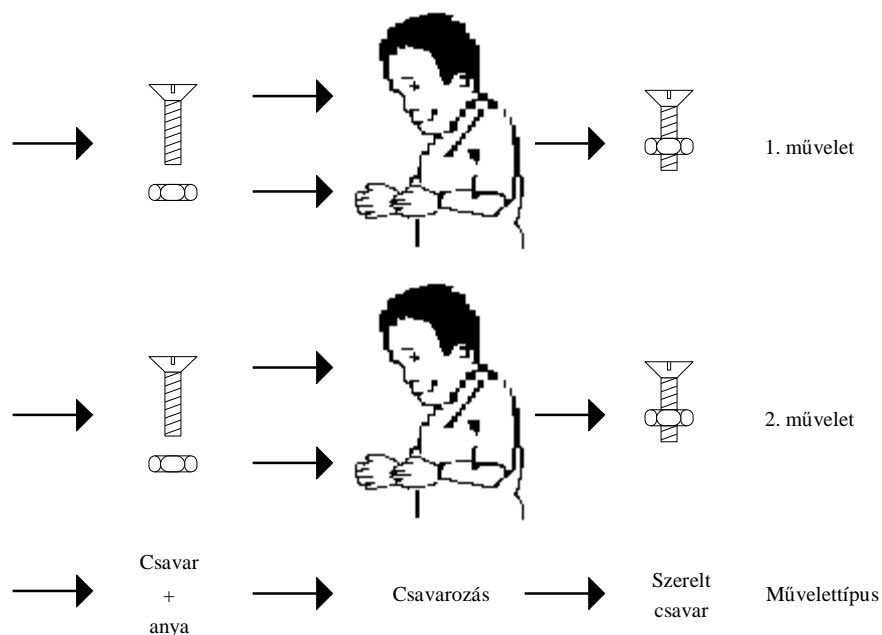
6.17 ábra: Feldolgozás-háló

A 6.16 ábra által mutatott feldolgozás-hierarchia csak 'élettelen' struktúra. Olyan, az IR *fejlesztéséhez* szükséges absztrakció, ami a feladat-lebontásnak és -kiosztásnak a nélkülözhetetlen feltétele. Azonban az IR-t majd *működtetni* is kell. Magyarul: az egyik feladatot végre kell hajtani a másik után. Az időbeli elrendezés, az 'élő' szerkezet tükrözésére alkalmas a 6.17 ábra hálójája.

6.6.2 Mikrotervezés

A számítástechnikusok egy része nem is gondol arra, hogy a feldolgozás az adatfolyamnál nemcsak magasabb, hanem mélyebb szinten is modellezhető. Bár nap mint nap használják a 'modul' vagy éppen az 'objektum' kifejezést, néha úgy tűnik, hogy nincsenek egészen tisztában gondolataik gyökereivel.

Az információs rendszerben nemcsak a jelenségekre vonatkozó ismeretek (vö. egyed- és tulajdonságtípus), hanem a tevékenységek egységei is tipizálhatók. A mindennapi életben ezt a tipizálást a fejlesztők ösztönösen végzik. A tudatosság - vagyis az elemzésen alapuló tervezés, tehát a modellezés - még nem jellemző ránk. Ne is hivatkozzon most az olvasó az újabb objektum-orientált eszközökre, amelyek egyébként tényleg arra hivatottak, hogy az ismétlődő és a 6.18 ábrával szemléltetni kívánt rutinszerű feladatokat mindig azonos módon oldjuk meg! A probléma ugyanis nem ennyire egyszerű.



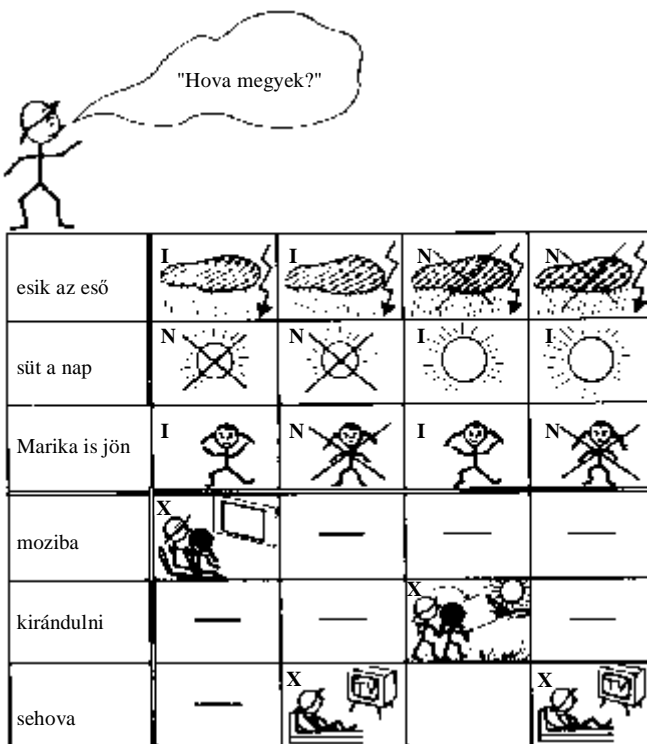
6.18 ábra: Művelettípus

Ha ma bekapcsoljuk a gépünket és megírunk rajta 'objektumként' például egy hibakezelő rutint, semmi akadálya sincs annak, hogy holnap újra bekapcsolva a gépet 'objektumként' kreáljunk - egy másik hibakezelő rutint. És ne tessék most azt mondani, hogy ez nem így van a gyakorlatban! Ugyan hányszor, de hányszor fordul elő velünk, hogy az elvileg azonos célú (pl.

számla) dokumentumokat ill. táblázatokat teljesen másként fogalmazzuk meg holnap, mint tegnap!

Történelem-tanítónk, Soma tanárúr mondását kell itt idéznem: „Édes fiam, a fejedben legyen világosság!”. Mondta ezt akkor, amikor a ‘hardver’ nem igazán jól működött, vagyis gyenge volt az osztályteremben a világítás. A szoftver sem jól működik, mert a tervezőeszközök nem adnak módot a tipizálásra illetve nem kényszerítik ki azt. Ergo: a fejünkben kell azt végrehajtani.

A standard, újrafelhasználható művelettípusok elgondolása és azok roppantul tudatos alkalmazása a mikrotervezésnek csak az egyik oldala. Van még több is.



		I	I	N	N
Ha	esik az eső				
és ha	süt a nap				
és ha	Marika is jön				
akkor	moziba		—	—	—
megyek	kirándulni	—	—		—
	sehova	—		—	

6.19 ábra: Döntési táblázat

A feldolgozásokban - amint azt korábban kifejtettük - adatkezelő, -előállító és vezérlési műveletek váltakoznak. Ki tudja miért, de a fejlesztőkre jellemző az, hogy figyelmüket az első műveletfajta köti le a tervezés során.

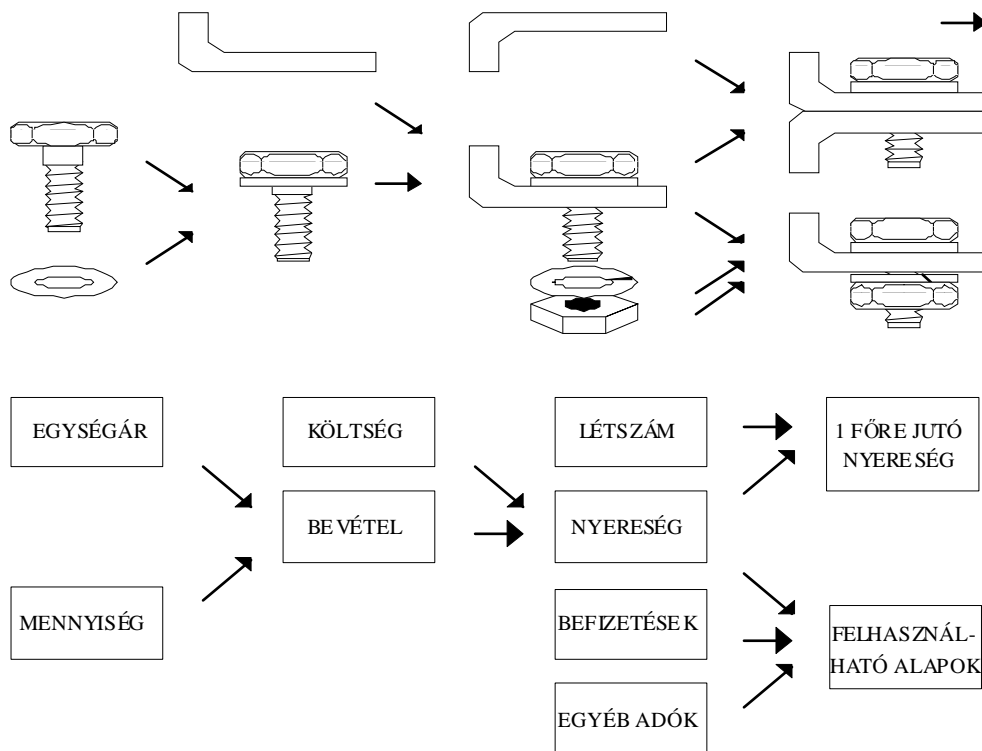
Nemcsak az ismeretekre, hanem a tevékenységekre vonatkozóan is optimum-kritériumokban (vö. 4.5 pont) kellene gondolkodnunk. A tevékenység is lehet nem valósághű, nem érthető, nem egyértelmű, nem minimális és nem teljes. Így hát elvileg feladatunk lenne a feldolgozásrendszer optimalitásának a vizsgálata is. A vezérlés egyértelműségének és teljességének az ellenőrzésében jelenthetne komoly segítséget a **döntési táblázat** (6.20 ábra), ami - érthetetlen okokból - ma egyetlen fejlesztési segédletben sem jelenik meg elemzési rész-eszközként.

A programok végrehajtási logikája **szekvenciális**, és ennek megfelelően maga a vezérlés is az. Ezzel szemben az emberi gondolkodásmód, tehát az elvégzendő tevékenység is elvileg **párhuzamos**. A türelmetlen fejlesztő már a végrehajtásra gondol a fejlesztés során is. Ezért - valljuk be - előfordul, hogy egy lehetőségéről, egy feltételkombinációról, egy programágról egyszerűen elfeledkezik. Nemcsak a mikroszinten (bár ott jellemzőbb), hanem a makroszinten is meg-

történik, hogy a feltételeket és következményeiket együttesen és párhuzamosan nem gondoljuk át. A szerző régen (25 éve) külön könyvet írt a döntési táblázatokról. Most csak arra kívánja felhívni a figyelmet, hogy a tevékenység-elemzésnek ezt az olykor nagyon hasznos eszközét ma sem kellene kifelejteni az eszköztárunkból.

Az egymással összefüggő tervezés és elemzés egy másik mikroszintű, szintén nem kellőképpen kiaknázott technikai eszköze a **precedencia-háló**. Itt ismét egy olyan régóta ismert tervezési/elemzési részmodszerről van szó, amit a ‘korszerű’ technológiákból úgymond kifelejtettek.

Tetszik vagy nem tetszik ez a megfogalmazás, a felhasználóknak szóló végső feldolgozási kimenet egy gyártmány, egy termék. Egy olyan produktum, amit a feltételezett ismereti félkész-termékekből, alkatrészekből és nyersanyagokból kell összeállítani. A feldolgozástervező az az informatikai technológus, aki látja, érti és elképzei azt, hogy az ismeretdarabokból miképpen lehet elővarázsolni azt a végső kimenetet, amely értelmezhető adatok együtteseként majd a felhasználói fejekben információvá gyűrődik.



6.20 ábra: Előzmény-háló

A feldolgozástervezők az adatfolyamra koncentrálnak, a vezérléssel és az adat-előállítással kevesebbet törődve. Az előzmény-háló az előállítás tervezésének és elemzésének a roppantul hasznos eszköze.

Ahhoz, hogy gépkocsi szülessen, el kell készíteni a karosszériát, a futóművet, a meghajtó berendezéseket. A motorhoz szelepek, ezekhez rúgók szükségesek. Alátét, tömítés, gyűrű - végül pedig alapanyag. Nincs ez másként az informatika terén sem. Ahhoz, hogy megismerjük a rendelésállományunk összértékét, kell tudnunk, hogy egy-egy vevő milyen értékben rendelt tőlünk. Ami nem tudható meg, ha nem ismerjük az egyes feladott rendeléseinek az értékét. Ehhez viszont

a rendelőtétel értéke szükséges, hogy a gondolatsorunk végén eljussunk a két 'alapanyaghoz' - az árhoz és a megrendelt mennyiséghez.

Sohasem születik gépkocsi, ha nincs tömítőgyűrű a motorszelepeknél. Ha az ár nem áll rendelkezésre, rendelőállományunk összértékét nem tudhatjuk meg. Előbb van az ár, csak azután van az összérték. Az előbbi az utóbbi előzménye, latin szóval *precedense*.

Az olvasó most megkérdezi, hogy miért kell ezt a teljesen logikus dolgot így és ennyire hangsúlyozni? Hiszen mindez kézenfekvő!

Nem az. A legkorszerűbbnek hirdetett fejlesztőeszközökben hiába is keresné az olvasó a precedencia-háló nyomát. Nincs, nem létezik, nem fontos. Ez baj és kényelmetlenségeket okoz.

Bár nem gyakori, de nagyon kellemetlen jelenség az **adateciklus**. Az X adatot másokból származtatjuk úgy, hogy az egyik származtatási tényező maga az X. (Az ugyanazt a tárváltozót hanyagul más-más célra felhasználó programozókat persze ez az 'apróság' nem érdekli.) Nem értjük, hogy a jól-megírt programunk miért nem működik, mert a precedencia-háló hiányában nem látjuk a ciklust. Ez a kisebbik baj.

A háló **megszakadása** (törlés) vagy **változása** (módosítás) jelenti a tényleges gondot. Az informatikusok nem eléggé figyelnek arra, hogy az ismeret 'anyagai, alkatrészei, szerelvényei, félkésztermékei' a teljes IR-ben kvázi szétszóródnak. A gépkocsi-gyártásban természetes, hogy a végső összeszerelést végző szakember nyugodtan hagyatkozik az előkészítő munkálatokért felelősök tevékenységére, és ha a technológia változik, akkor új eligazítást kap.

Az informatikában egyelőre más a helyzet. Az ismeretek előállítási műveletei a tevékenységi funkciókban szétszórtak. Az A1-es funkció előállítja és tárolja az adatbázisban a befizetett díj adatot (nem is annyira egyszerű algoritmus szerint, mert ha túlfizettek, vagy nem elegendőt, akkor el kell rendezni a többletet illetve a hiányt időszakonként is). Az A2-es funkció arra épít, hogy a túlfizetéseket és a hiányokat majd úgy összegzi, hogy... Előzmény-háló nincs. Az A1-es funkció algoritmusát megváltoztatják, ámde az A2-es funkció készítőjét erről egyáltalán nem értesítik. Tehát 'továbbketyeg' az A2-es tevékenységsor mindaddig, amíg valaki észre nem veszi, hogy az eredmény logikailag hibás.

Ha a precedencia-háló valamelyik tételének az értelme megváltozik, akkor ez mindig a háló újradefiniálását igényli. Ez egy autó készítésében régen világos. Az ismeretek előállításában pedig még ma is közömbös...

6.6.3 Feldolgozás-konzisztencia

Az adatrendszer tervezésének (ld. előző fejezet) régóta ismertek az elemzési lehetőségei és szabályai. Ez a kitétel a feldolgozásrendszer bizonyos aspektusait illetően is igaz. Azonban az előző pontban ismertetett elemzési részmódszereket sokan nem ismerik, a közkézen forgó eszközök pedig nem támogatják. Egyedül talán az absztrakció középső szintjén, tehát az adatfolyamhoz kapcsolatosan látszik valóban egészséges törekvés az elemzés és tervezés együttesének a kialakítására.

Az adatfolyam dinamikusan - tehát az időbeli sorrendnek megfelelően - írja le a tevékenységek és az adategységek (bemenetek, adatbázis egyedek, kimenetek) szervezett együttesét. Első szinten valóban csak az **adategységekkel** törődve, ám a másodikon már az adattételek **mélységéig** hatolva. Olykor-olykor - például az összetett feltételek esetében - a helyes sorrend megtalálása nem könnyű, máskor pedig többféle sorrend-változat is egyaránt alkalmasnak látszik.

Az adatfolyammal tükrözött feldolgozásnak az a feladata, hogy a vonatkozó bemenet(ek)ből, az adatbázis tételeit felhasználva (most kivételesen ide érthetjük az átmeneti állományokat is) előállítsa a kért kimenet(ek)et. A tervezőnek emiatt kettős a teendője. Egyrészt meg kell győződnie arról, hogy a kimenet egyáltalán produkálható-e a bemenet és az adatbázis alapján. Másrészt meg kell néznie azt is, hogy a bemenet minden tényezője értelmes felhasználásra kerül-e. Erre a két

elemzésre sem az előzményháló (ld. 6.20 ábra), sem a makroszintű adatfolyam-ábra (ld. 6.13 ábra) nem alkalmas. Az előbbi nem tárja fel, hogy honnan való az adattétel, az utóbbi pedig azt nem árulja el, hogy mire használjuk azt.

A hetvenes években a rendszerdokumentációk nélkülözhetetlen része volt az ún. *input-output* mátrix. Ez mutatta, hogy a bemeneti adatot honnan vesszük és az miként és hol jelenik meg a kimeneti ismeretek között. Mi tagadás, egy kicsit sablonos részmódszerről volt szó. Azért, mert az 'ebből a bemenetből az abba a kimenetbe' logika képes volt ugyan az adatkezelési műveletek tükrözésére, de a vezérlési és adatelőállítási művelet a mátrixban nem volt nyomkövethető. Pl. a kimenetben megjelenhet olyan adat is, ami számított, tehát sem a bemenetben, sem az adatbázisban expliciten nem szerepel.

BEMENET	VEVŐ-LEKÉRDEZÉS	KIMENET
Cikkszám='X'	CIKK Cikkszám='X' CIKK > Egységár TÉTEL Cikkszám='X' TÉTEL > Mennyiség <i>Tételérték=Egységár*Mennyiség</i> TÉTEL > Rendelésszám RENDELÉS Rendelésszám RENDELÉS Dátum<'Y' <i>Rendelésérték=ΣTételérték</i> RENDELÉS > Vevőkód VEVŐ Vevőkód <i>Vevő-összérték=ΣRendelésérték</i>	Vevőnév Vevő-összérték

6.21 ábra: Bemenet-tevékenység-kimenet tábla

A 6.21 ábra a korábbi navigációs feladatunk kibővített változatát tartalmazza táblázatos formában. Arra szolgál szemléltetésül, hogy az egyszerű input-output mátrix helyett kifejezőbb és az alaposabb elemzést támogató megoldást is lehet találni. A táblában '|' mutatja a szelekciót, '>' a projekciót, dőltbetű a számítást. Persze összetettebb esetben minősítéseket is lehet/kell alkalmazni. Ha netán a táblázatot még a vezérlés egyéb elemeire (pl. ciklus) utaló megjegyzésekkel ill. jelölésekkel is ellátnánk, akkor egészen használható kis részeszközt kapnánk a feldolgozások középszintű elemzéséhez. A lehetőségek megítélését azonban az olvasóra bízunk. (Mellesleg az ábra hibás. Lásd a 612 feladatot.)

6.7 FELDOLGOZÁS ÖSSZEFOGLALÓ

Az információs rendszer feldolgozásvetületét alkotó tényezőket rendszerként kell felfognunk az IR-en belül. A feldolgozásrendszert alkotó tevékenységek és események ugyanúgy elképzelések termékei, vagyis *absztrakciók*, mint maguk az adatok. Ezt jól mutatja, hogy két cégnél a teljesen ugyanazt (!) a célt szolgáló értékesítési rendszerrészt nem-teljesen azonos módon építik fel, valamint nincs két olyan fejlesztő, aki a számlakarbantartási tevékenységet illetve azon belül pl. a végösszegezés-számítási algoritmust teljesen ugyanúgy oldaná meg.

A feldolgozásokat mindig kétféle *módon* kell megközelíteni. Szükség van arra is, hogy a feldolgozásokat önállóan (szerkezetileg - hierarchikusan, időbelileg - dinamikusán) modellezzük

a fejlesztésben például a feladat célszerű megosztása érdekében, a működtetésben például a feldolgozások ütemezése kedvéért. Ámde igény van arra is, hogy a feldolgozásokat minden szinten az adatokkal létesített viszonyaikban szemléljük, mert a feldolgozásnak tárgya és - az esemény mindig ismeret formáját ölti - belső résztényezője az adat. E passzusban a 'modellezzük' és a 'minden szinten' kitételekre kell jobban odafigyelnünk.

A feldolgozások absztrakt *hierarchiát* alkotnak. Ennek a csúcán maga az IR áll. A *teljes rendszer* mindig fogalmi, külső és belső aspektusból vizsgálendő. A 6.1 pont mutatta be ezeket a szempontokat. Minden feldolgozásnak generikusan az az elvi feladata, hogy az egyik *külső* nézetből kiindulva, a *belső* ismereteket felhasználva a másik külső nézetet előállítsa a közös *fogalmi* kép alapján. Ezen az elméleti síkon a hangsúly az utóbbi kitételen van: a számítógépen tárolt és az azon kívüli ismeretek egyetlen (!) feldolgozás tárgyai. A legtöbb probléma abból adódik, hogy a gépi és a manuális résztvekenységeket egymástól elkülönülten, ráadásul nem is egy közös fogalmi kép szerint alakítják ki.

Ez az elméleti és általános logika a gyakorlatban a 6.2 és 6.3 pontok által leírt konkrét módokon valósul meg. Az egyik külső nézetet a tevékenységet elindító *bemeneti*, a másikat az azt lezáró *kimeneti* esemény jelenti. Azért beszélhetünk nézetről, mert a bemenet és a kimenet mindig adatok formájában testesül meg úgy, hogy ezek az adatok az adatbázisban tároltakra vonatkoznak illetve abból származnak (vö. a karbantartás, visszakeresés és adatelőállítás mozzanataival.) Nem fogalmi, hanem logikai szintű kategória az *átmenet*, amely természetesen meg kell, hogy feleljen a fogalmi képnek. Mégsem ilyen szintű, mert mindig a fejlesztő által alkalmazott kompromisszumok tárgya maga az is, hogy lesz-e átmenet illetve az is, hogy ha igen, akkor melyik feldolgozási ponto(ko)n.

„Aki szelet vet...” A bemenet az adatbázison belül kisebb-nagyobb „vihart arat”. A fejlesztő külön feladata, hogy meghatározza a tranzakció kalandos útját az adatbázis tengerén. Erre a célra a *navigáció* szolgál, amit az adatkezelési-gráf segítségével tükrözünk (6.4 pont). A navigálás ma már nem teljesen ismeretlen koncepció. Egyrészt azonban még nem terjedt el széleskörűen, másrészt pedig sokan összetévesztik a hagyományos *adatfolyam* lényegével.

A régi feldolgozástervezési részmódszerek (ld. 6.5 pont) mindegyikét ma is hasznosnak és nélkülözhetetlennek tartjuk. Csak arra kívántuk felhívni az olvasó figyelmét, hogy a hagyományos makro- és középszintű tervezési módszerekből vagy hiányzik az elemzési momentum, vagy az nem eléggé erős ill. mély, vagy időközben részben meghaladott elveken alapul. Pl. a strukturált programozás a legmélyebb szinten foglalkozik csak az adattal, holott ezt a legmagasabban kell tenni. Az adatfolyam ma is kellemes módszer, de nem eléggé mély. Végül pedig a mikro-szintű tervezés nem kap elegendő figyelmet.

A fentiekben kiemeltük a 'modellezzük' szót. A modellezés roppant tudatos, elemzéssel párosuló tervezést jelent. Makro-, mikro- és középszinten egyaránt léteznek ma már elemzési részmódszerek. A 6.6 pontban ezek közül néhányra kívántuk felhívni a figyelmet. Maguk a *rendszerreszek* nem pusztán a szubjektív akarat, hanem az objektív mérlegelés eredményei is lehetnének. A feldolgozási statikus *hierarchia* és dinamikus *háló* a szokásosnál tudatosabban is kialakítható (makroszint). Az ellentétes oldalon (mikroszint) is adódnak még ki nem használt lehetőségek főleg a *vezérlési* és az *adatelőállítási* műveletek elemzésére. Végül: a középső szinten az *adatkezelési* műveleteknek az előzőekkel való harmóniája az eddigieknél pontosabban is vizsgálható lehetne.

Mára már lejárt a régi nagy egyéni felfedezések kora. Nincsenek már egyedül dolgozó és átütő sikereket elérő tudós szakemberek. A sikerhez csapatmunkára, eszközökre, időre és - módszeres tevékenységre van szükség. Az információs rendszer feldolgozásrendszere sem nélkülözheti a mélyreható elemzést, vagyis a modellezést. Erre kívántunk rávilágítani könyvünknek ebben a fejezetében.

ELLENŐRZŐ KÉRDÉSEK - 6

Az olvasó vizsgálja meg, hogy mennyire érti a feldolgozásrendszer kettős - önmagában véve abszolút és az adathoz képest relatív - természetét. Mérlegelje, hogy miért van szükség a tevékenységek többszintű vizsgálatára. Végül pedig gondolkodjon el azon, hogy mi az elemzés szerepe a feldolgozástervezésben.

- 601 Ön kap egy számlát a közértben. Nagyobb helyeken ezeket a számlákat számítógépen is vezetik. A megkapott papír informatikai szempontból hiányos. (Gyakorlatilag nem az!) A három - külső, belső, fogalmi - sémában gondolkozva saját szavaival mondja el, hogy
- e példában mi jelenti a belső és a külső aspektust
 - miért hiányos elvileg a megkapott papír, gyakorlatilag miért nem az
 - ezekre tekintettel mi a feldolgozástervező feladata
 - mi teremti meg az egész feldolgozás összhangját.
- 602 Árulja el, hogy az előző példában a cikknév bemeneti(1), kimeneti (2), átmeneti (4) vagy törzsadatként (8) szerepel-e? Egyszerre több is lehet! Adja össze válasza pontszámait és mondja meg az eredményt. (Bináris összeadásról van szó, tehát különböző feltételezések esetében az összeg mindig eltérő lesz. Pl. bemenet és törzs = 9.)
- 603 Amikor Ön a közértben a számlát megkapja, milyen módú a feldolgozás? Lehet-e másféle? Amikor és ha a közértes este mérleget készít, milyen módú a feldolgozás? Lehet-e másféle?
- 604 Tekintettel az árubeszerzési, számlaadási, adózási, bérleti stb. feladatokra Ön szerint hány információs rendszere van a közértesnek? Ha csak a felsorolt négy feladatra gondol, hány rendszerrészt alkotna Ön és melyek lennének azok? Vigyázat, a kérdések becsapósak!
- 605 Saját szavaival mondja el, hogy mi szükséges a rendszerrészek objektív meghatározásához és milyen eszköz szolgál erre a célra!
- 606 Saját szavaival mondja el, hogy mire szolgál a fekete-doboz és a feldolgozási hierarchiának melyik szintjén érdemes azt alkalmazni!
- 607 A 604-es feladatra visszatérve fogalmazza meg, hogy a feldolgozás miért absztrakció! Gondoljon arra, hogy az árubeszerzés és a számlaadás mint valós feladat milyen informatikai feltételhez kötött. Mi bennük a közös? Mihez kapcsolódik mindig a feldolgozás? Segítségként: milyen művelet kell, hogy megelőzze mind az árubeszerzést, mind a számlaadást? Végül: hogyan nevezzük az ismeretkezelés dinamikus rendjét tükröző eszközt?
- 608 Vessen egy pillantást a 6.8 ábrára! Válaszoljon a következő kérdésekre, amelyek így kezdődnek: „Változik-e a navigációs útvonal akkor, ha...” és így folytatódnak:
- csak a Z mennyiséget meghaladó rendeléstételek az érdekesek
 - csak a Q helység vevőire vagyunk kíváncsiak
 - szükségünk van a számlákra is.

- 609 A szerző olyan eszközt szeretne látni, amely képes a navigációs útvonalak tárolására és kezelésére is. Vajon miért? Gondoljon arra, hogy nemcsak az adatok, hanem a feldolgozások terén is vannak átfedések!
- 610 Ön szerint teljes-e és helyes-e a következő gondolatmenet? Milyen eszközzel vizsgálná a konzisztenciát? „Ha az áru külön adóval terhelt, akkor az A rutint kell végrehajtani, ha kedvezményt is nyújtunk, viszont a B rutint, ha kedvezményt nem adunk. Ha külön adóval nem terhelt és kedvezményt nem adunk, akkor a C rutin a feladat. Ha viszont külön adóval terhelt, de a szállítónk X típusú, akkor az A rutint kell alkalmazni. Y típus esetén mindenképpen a B-t.”
- 611 Mi köze van a mélyszerinti előzmény-hálónak (6.20 ábra) a feldolgozottság fokához (vö. 2.7 ábra)? Mondja el saját szavaival!
- 612 Mire alkalmas a bemenet-tevékenység-kimenet tábla (6.21 ábra)? Vajon észrevette-e, hogy az ábra rosszul megfogalmazott? Hol?

7. AZ IR KÖRNYEZETRENDSZERE

Az előző két fejezetben az IR adat- és feldolgozásvetületéről volt szó. Ezért úgy logikus, hogy most a környezetvetülettel foglalkozzunk. Azonban a fenti cím nem egészen pontos. Az IR **környezetrendszerébe** a felhasználók, az erőforrások és a szabványok tartoznak. Mi azonban nem óhajtunk részletesen foglalkozni a pénzzel, az idővel és az eszközzel. Az előbbi kettő a projekt-menedzsment tárgykörébe tartozik. A harmadik pedig annyira változatos, hogy rendszer szinten, egy ilyen átfogó célú könyvben nem lehet megfogni. (Arról nem beszélve, hogy ez felesleges is lenne, hiszen az eszközöket illetően nem panaszkodhatunk szakkönyvhiányra.)

A fennmaradó két tényezőre - felhasználó és szabvány - közösen jellemző, hogy az IR legkritikusabb elemei. Sajnáljuk, hogy ezt kell mondanunk, ám tény, hogy az információs rendszereink valamiképpen **nem emberiek**. Ennek alapvető okát abban látjuk, hogy a számítástechnikát ekként - technikaként - fogják fel a legtöbben. Magyarul: az emberhez nem értenek annyira, mint a technikához. A másik baj az, hogy amennyire jók a magyar szakemberek a tervezésben, annyira gyarlók a szervezésben. Ezért nem kerülhetjük el, hogy a munka- és projekt-szervezés egyes mozzanataira kitérjünk.

A szervezés gyengeségének a legfőbb oka a szabványok hiánya illetve azok nem megfelelő volta, továbbá az, hogy nálunk a szabvány 'írott malaszt'. Nem tudunk jó szabványokat készíteni, a szabványrendszer nem szervezett együttes, az aktualizálással pedig nem sokat törődünk. A szabvány olyan nálunk, mint a KRESZ, amit kelleni, de barmok nem tartják be, nem pedig javunkat szolgáló eligazításnak tekintünk. Különösen vonatkozik ez a kitétel a dokumentációkra, amikről éppen ezért külön pontban kell megemlékeznünk.

Ennek a fejezetnek a célja kettős. Egyrészt tárgyszerűen ismerteti az IR emberi környezetével, szabványrendszerével és a rendszerdokumentációval kapcsolatos tudnivalókat. Másrészt megpróbálja felhívni a figyelmet az IR bizonyos emberi mozzanataira.

Az információs rendszerek nem mindig sikeresek. Hiába hiszi azt a fejlesztő, hogy ő jól alkotott, ha az utca embere joggal elégedetlen a megoldásokkal. Több türelemre és belátásra lenne szükség. Meg kellene végre érteni, hogy a fejlesztés nem kín, hanem közös emberi alkotó munka. A szabvány nem kényszer, hanem a közös megértés alapja. A dokumentációt pedig nem muszájból kell megírni...

7.1 AZ EMBERI KÖRNYEZET

7.1.1 Pszichológiai mozzanatok

Korábban az információs rendszer felhasználói közé soroltuk az ún. végsőtől kezdve az alkalmazási felhasználón át a fejlesztői és a vezetői szerepet betöltő embereket egyaránt. Kifejtettük, hogy a felhasználó nem abszolút, hanem **relatív** fogalom. Azért az, mert ugyanaz a valaki eltérő szerepeket tölthet be ugyanazon ismeretek vonatkozásában. Például ma fejlesztő,

holnap pedig végső-felhasználó ugyanabban az információs rendszerben. Amit az egyik minőségében pontosan ő akart, éppen az zúdul vissza a fejére - a másik szerepkörében.

Érdemes megfigyelni, hogy a szerepváltás milyen hatásokkal jár. Mindaz, ami addig szép és jó volt, az hirtelen csúnya és rossz lesz - persze mások miatt. Az oktatás egyik komoly hibája, hogy nem készíti fel a leendő informatikusokat a fejlesztés és a működtetés **pszichológiai mozzanataira**. Így az alkalmazó és a fejlesztő nem képzeleli magát a végső-felhasználó bőrébe; az utóbbi nem figyel az előbbi lelkivilágára sem; ő maga pedig csak jelzősen, *szakemberként* szerepel a vezetők gondolataiban. Végeredményben az informatika éppen az információk természetéből miatti elvileg az empátiára kellene, hogy épüljön, ámde gyakorlatilag éppen a ráérzés, az együttérzés a legnagyobb hiánycikk.

Persze a **végső-felhasználót** olykor nem könnyű megérteni. Ma ő az, aki nem tölti ki pontosan az adatait, holnap pedig éppen ő fog a leghevesebben tiltakozni azért, mert azokat az Y papíron nem korrekten látja vissza.

T 7.1 Amikor az ÁB bevezette a kötelező biztosítási módozatot, mintát küldött ki a remélt végső-felhasználóknak (ügyfeleknek). A minta elmondta, hogy 'Kovács Ferenc kocsija Skoda, színe zöld' stb. és az adatokat ilyen-féle módon kell kitölteni. Napokon belül az ÁB több ezer levelet kapott, hogy 'én nem vagyok Kovács; az vagyok, de nem Ferenc; kocsim nem Skoda; Skoda, de nem zöld' stb. Ez semmi: több ezer jelentkezés pontosan a mintában felsorolt adatokat tartalmazta.

Az **alkalmazói-felhasználó** adott esetben még nehezebb dió. Jellemző módon őt kizárólag csak az általa látott dolgok érdeklik, és az ismereteket másokkal is e szűk látásmódnak megfelelően óhajtja elrendeztetni. A vezető sokszor belemegy ebbe a játékba - és ekkor születnek a párhuzamos 'rendszerek'. Az alkalmazóról azt is tudni kell, hogy a negatív és a késedelmes akarat megtestesítője. Sokszor nem is érti pontosan, hogy mit is szeretne (pozitív akarat), de annyi biztos, hogy nem azt, amit a fejlesztő és amit a többi alkalmazó (negatív akarat). Jellemző rá, hogy a fejlesztés vége felé jelenti ki, hogy mégsem úgy gondolta vagy akkor ad meg egy olyan információt, ami az addig hitt képet teljesen felborítja.

T 7.2 Amerikában a Byron & Jackson cégnél a fejlesztők egy olyan pilóta-rendszert készítettek, amelynek a mélyén - titokban - egy alkalmazási statisztikai program futott. Majd az X fejlesztő szólt az Y felhasználónak, hogy nézze meg az alkalmazást, mert már a Z felhasználó is erre készül. Y - akinek addig esze ágában sem volt a rendszerrel törődni, hanyatt-homlok rohant a géphez... A trükk bevált: a statisztikák alapján a fejlesztők a rendszert kicsit átpofozták - és lett egy jó kis alkalmazásuk.

Hát igen, ez a negatív akarat: Y addig nem, de nehogy Z... A féltékenykedés a **fejlesztőket** is megmértelyezi, aminek az oka a következő alponthan ismertetett rossz munkaszervezés. A fejlesztő az 'ő' rendszerén majomszeretettel csüng. Mindentől óvja - főleg az értő szemektől. Ebből azután egészen kellemes - bár roppant gyerekes - társasjáték kerekedik. Az A fejlesztő megpróbálja a B elől elrejteni a dolgait, az pedig a saját munkája helyett inkább keresésre indul.

Végül ne feledkezzünk el a **vezetők**ről sem. Akinek az Istenke hatalmat adott, annak nyilván tudást is - vélik egyesek. A magyar számítástechnikában sajnos általánosan jellemző az akarnok, a szakmailag megalapozatlan döntések sora. Az információs rendszernek egyik peremfeltétele és egyben kritériuma a **stabilitás**. Az IR nem óra, amit bármelyik órás reparálni tud. Az IR-nek része az ember, a maga információk iránti roppant érzékenységgel és ismereti háttérrel. A jó

informatikusnak időbe telik beletanulnia ebbe az objektívan szubjektív miliőbe. Ezért nem váltható fel úgy egyik napról a másikra, mint az órás. Ehhez képest meglepő, hogy hazánkban igen magas az informatikai *fluktuáció*. Vajon mi - pontosabban: ki - ennek az oka?

A felhasználók *csoportokat* alkotnak. Nemcsak a formailag elismert szervezet (pl. a cég részlegei) tekintendő csoportnak. A végső-felhasználóknak is vannak szociális, műveltségi stb. rétegei. Ehhez képest eléggé meglepő, hogy az itthoni információs rendszerek mennyire sablonosak. Az evidens, hogy pl. egy országos cég nem készíthet mindenki számára 'testreszabott' rendszerrészeket. Azonban az mégis furcsa, hogy papírjaik megértését és a kooperációt ugyanazon a szinten várják a kétdiplomás fiattól, mint a nyugdíjas szövmőntől...

A sajnálatos tény az, hogy *rendszerpszichológiát* ma nem tanítanak sehol sem a leendő informatikusoknak. Ennek a könyvnek nem lehet az a feladata, hogy pótolja ezt a hiányosságot; csak magára a tényre óhajtotta felhívni a figyelmet.

Intermezzo

A szerző több mint 30 műtéten esett át tűzbalesetét követően. Ez mondhatni magánügy. Az viszont már nem az, hogy egyetlen egyszer sem érdekelték az orvosait a saját tapasztalatai, a magánvéleménye, a sérült - mint ember. Pedig lett volna mit elmondania 'profi-betegként'. (A teljes igazság az, hogy akadt egy kivétel is.)

Az orvosok egy részének az arroganciája közismert. Az informatikus éppúgy emberekkel foglalkozik, mint az orvos. A dedukció logikus: az informatikusok jó része - *arrogáns*. (A szerző csak tudja, mert ő is átesett ezen a betegségen.) Gügyögősen, lekezelően, 'ezt-úgy-sem-tetszik-érteni' stílusban foglalkozunk a felhasználókkal. Ne tegyük, mert elvileg az informatika is - hivatás, tehát más emberek életének a jobbítására szolgáló feladatvállalás.

7.1.2 Munkaszervezési mozzanatok

Az IR életének három nagy fázisa van: a fejlesztés, a működtetés és a változás. Mindhárom szakaszban különböző képzettséggel rendelkező (szak)embereknek a munkáját kell összehangolni. (Az se ugorja át ezt az alpontot, aki a saját maga számára illetve kisvállalkozásban óhajt fejleszteni. Nem kizárt, hogy a részére is akad mondanivalónk.)

Az informatikai munka megszervezésében számos probléma mutatkozik. Ez az alpont arra szolgál, hogy ismertessük e negatív jelenségeket és bemutassuk azok kiküszöbölésének a módját.

Nagyon sok helyen szervezetiileg is szétválasztják egymástól az ún. *gépi* és az ún. *manuális* 'szervezést'. Tehát külön szervezeti egységet hoznak létre az ún. gépekből és az ún. ügyvitelekből. Ez a megoldás több szempontból rossz. Elvileg azért, mert a 'külső' és a 'belső' nézet (ld. 6.1 pont) teljes harmóniában kellene, hogy álljon egymással a 'fogalmi' aspektuson keresztül (aminek viszont a legtöbb helyen egyáltalán nincs gazdája, mint arra majd kitérünk). Mármint az ügyviteles nem ért a géphez, a számítógépes a papírhoz. Hogyan lesz a kettőből - gép és papír - harmonikus rendszer? Vagyis a valódi probléma a 'félműveltség' - amit viszont a besorolás mai módja alapoz meg. A végeredmény: senki sem látja egyben, elejétől végéig a feldolgozási láncot. Gyakorlatilag a két szervezet léte csak arra ad alkalmat, hogy a kétféle társaságot összeugrasszák.

Néhány helyen már belátták a helyzet tarthatatlanságát, és egy 'informatikai' részlegen belüli egységekként működtetik a 'számítástechnikát' és az 'ügyvitel-szervezést'. Ezzel persze a problémák nem oldódnak meg, csak a felelősség és az ellentétek elsimításának a terhe helyeződött egy szinttel lejjebb. Mert, persze, a helyes megoldás a valódi projekt-szervezés lenne. Ugyanis sohasem a funkciót kell a szervezethez igazítani, hanem éppen megfordítva.

Csak hogy az a baj, hogy épp a funkciót nem látjuk helyesen. Az informatika az ismeretek megismerésének, elrendezésének és kezelésének a tudománya; nem a papíroké és nem a számítógépé. Az képtelenség, hogy a kódokat, azonosítókat, egyéb adatokat a papírokon más alakítsa ki, mint a számítógépen. Ezek a dolgok ugyanis informatikailag egyetlen funkciót jelentenek.

A második alapvető probléma az, hogy ma sok helyütt **szervező-programozó párosban** gondolkodnak. Persze ez nem mindig páros (olykor több programozó is dolgozhat egy szervezőnek, bár ez néhol nem jellemző) és nem is az érthető munkamegosztás ellen szólunk. Elvileg a programozó inkább a számítógéphez áll közelebb, algoritmikus gondolkodású és minél előbb szeretné látni munkája eredményét. (Ez is rendszerpszichológia.) Vele szemben a szervező inkább az emberhez áll közelebb, áttekintő gondolkodású és nem zavarja, ha a munkája gyümölcse csak későbbben érik be. Éppen ezért nevetséges volt a hetvenes évek azon szemlélete, hogy csak a jó programozóból válhat jó szervező.

Nem csak mellékesen jegyezzük meg, hogy egyes helyeken - főleg a kisebb cégeknél - szervezőt nem is alkalmaznak a 'számítástechnikus = programozó' képlet jegyében. Nem akarjuk megsérteni a jószándékú programozókat, ezért nem mondjuk azt, hogy ez a képlet igaz is; tehát a 'programozó \neq informatikus' összefüggés is fennáll. Aki programot akar, azt is kap. Aki viszont rendszert, annak szervezőt is alkalmaznia kell. A programozó és az alkalmazói-felhasználó ugyanis szükségszerűen nem érti meg egymás szavát. A kettejük közötti tolmács szerepe a szervezőre hárulna - és ez is rendszerpszichológia.

Intermezzo

Semmi alapunk az egyik szakma és az egyik munkatárs lebecsülésére illetve a másik szakma és a másik munkatárs feldicsérésére. Fentebb nem a programozó és a szervező rangsorolásáról volt szó. A programozás és a szervezés egyaránt szükséges szakmai feladat - és ez a lényeg. A baj nem az, hogy egyesek nem alkalmaznak szervezőt, hanem az, hogy a szervezői munkát nem végzi el senki sem. Egy szót sem szólnánk, ha azt a programozónak nevezett valaki megtenné.

Egyébként döbbenetes, hogy az 'informatikát' alkalmazni óhajtó vállalatokban mennyire nem ismerik a legalapvetőbb szakmai struktúrákat és feladatokat sem. Tessék csak megnézni a Számítástechnika hirdetőit. Szervezőt keresnek UNIX üzemeltetésre; programozót adatbázis-tervezésre; programozó-matematikust a CASE bevezetésére stb. stb.

Ám térjünk vissza a 'páros'hoz'. A baj az, hogy a szervezőt ill. a programozót kölcsönösen és tartósan egymáshoz valamint a feladathoz kötik. Ha adott az X rendszerrész, akkor azon az XS szervező és az XP programozó dolgozik, míg az Y rendszerrészen az YS - YP páros. Ez a beosztás a fejlesztés természetének a tökéletes meg-nem-értéséről tanúskodik.

A vércsoportok mintájára a fejlesztési teendőket kereszteljük el 0 - A - B - AB módon! Vannak mindent átfogó feladatok (0), amikről majd később szólunk. A fejlesztésben - annak szakaszai szerint - pedig az A - B - AB képlet uralkodik. A fejlesztés kezdetén a szervezőnek; közepén a programozónak akad több dolga; a végén pedig együtt kell, hogy dolgozzanak a bevezetésen (tesztelés, átadás stb.). Ebből a képből logikusan következik, hogy a páros egymáshoz rendelésének az egyenes következménye a munkaerő-kihasználatlanság. A tervezési szakaszban a programozó 'lógatja a lábát', a kifejlesztésiben pedig a szervező. Ezen a bajon is másféle szervezéssel lehetne segíteni.

A harmadik probléma a **belső**sök és **külső**sök viszonya. A külsősök bevonása a fejlesztésbe számtalan formában történhet. Előre kell bocsátanunk, hogy nem a tény, hanem a mód ellen van kifogásunk. Vegyünk csak bátran szoftvert. Nem kell megijedni a külsős magánvállalkozó szereplésétől sem. Még az sem kizárt, hogy a teljes fejlesztéssel egy külső céget bízunk meg. Azonban a kulcsszó már elhangzott.

Mindig valóban *teljes* fejlesztésben kell gondolkodni. Már pedig sokan nem ezt teszik. A programcsomagot illeszteni kellene nemcsak a többi szoftverhez, hanem az ügyviteli dolgokhoz is. A külső bedolgozó nem lehetne 'magányos farkas', mint az ma annyira jellemző, hanem csak egy más formában alkalmazott munkatárs, azaz a többi dolgozónak a társa, és nem vetélytársa vagy ellenlábas. Az egyetlen célszerű megoldási mód ismét csak a projekt-szervezés.

Végül fel kell hívnunk a figyelmet a *fejlesztés* és az *üzemeltetés* ellentétére. A szerző még sohasem látott olyan rendszert, amely hibamentes lett volna és/vagy amelyen soha nem kellett változtatni. Az is rendszerpszichológia, hogy értjük az IR ilyen természetét. A megértő türelem helyett viszont az jellemző, hogy a két 'parti' - a fejlesztők és az üzemeltetők - a hibák fellépésekor egymásra mutogat. Ez csak az egyik baj. Nem ritkán az is előfordul, hogy felcserélődnek a szerepek - a fejlesztő üzemeltetési feladatot is ellát, viszont az üzemeltető itt-ott egy kicsit 'belenyúl' a rendszerbe, tehát fejlesztést végez.

Az IR fejlesztésének szinte minden résztvevője ma skizofrén állapotban van. A saját szerepeket nem látják, nem vállalják, viszont a másokét átveszik. Nincs értelmes informatikai munkamegosztás. Pedig még a vezető-beosztott; az X és Y szakterület; az 'informatika' és a szakterületek ellentéteiről nem is szóltunk. A nagy kérdés az, hogy milyen módon oldhatók fel ezek az ellentmondások?

7.1.3 A rugalmasság és a szilárdság mozzanatai

Amilyen kifejezést az informatikában rosszul lehet használni, azt rosszul is fogjuk használni. (Ezt mondhatta volna Murphy is, de nem ő tette.) Az egyik gyakran félrehasznált szavunk a **projekt**.

T 7.3 Egy programozó ismerősöm elújságolta, hogy legújabbán az X projekten dolgozik. Kivel? - kérdeztem. Hát az Y szervezővel. - volt a válasz. És még kivel? - érdeklődtem. Hogyhogy és még...?

A 'projekt = programozási', jobb esetben a 'projekt = szervezési+programozási' feladat képletek roppant hamisak, de így terjedtek el a köztudatban. Ezért bár ez a könyv nem a projekt-menedzsmentről szól, nem kerülhetjük el, hogy pár szó erejéig ki ne térjünk a projekt-szervezés néhány momentumára. Kezdjük azzal a tárgyalást, hogy kijelentjük: a projekt viszonylag hosszú ideig tartó, sok emberi és eszközbeli erőforrásnak az összehangolását igénylő fejlesztési feladat. Értsd: nem programozási, szervezési és programozási, hanem *fejlesztési* munka.

Egy szervező és egy programozó nem alkothat projektet, mert a szervezés és a programozás együtt még nem fejlesztés. Hol marad az üzemeltetés kialakítása? A többi feladattal való összehangolás? Az erőforrások előteremtése? A minőség ellenőrzése? A változtatás menedzselése? A felhasználó hozzárendelése?

Induljunk ki abból, hogy minden vállalat erőforrásai szűkösek. Ezért az egyes fejlesztési feladatok nem függetlenek egymástól. A projektek versenyeznek az erőforrásokért, amelyek *közös használatával* a vállalat megtakarítást ér el, ám szétaprózásával felesleges többletekre lesz szüksége. A projekt tehát nem lehet sem két ember ügye, sem az informatika és/vagy a felhasználó belügye: vállalati szintű dologról van szó.

Ezért a projektek felett kell, hogy létezzen valamilyen együttes szervező erő. A szakirodalom *kormányzó bizottságnak* [steering committee] hívja a felelős elnökből, szakterületi képviselőkből és informatikai szakemberekből álló első szintű irányító szervezetet. Nálunk is próbálkoznak hasonló megoldással, ámde nem túl sok sikerrel. A sikertelenség okai a következők. Ad 1) Az elnök mindig ugyanaz és vagy a szakterület, vagy az 'informatika' vezetőjeként elkötelezett az egyik irányban. Ad 2) A tagok nem egyenrangúak; szavuk az erőssorrendben fog a latba esni. Ad

3) Nem a projekt valódi vezetője (ilyen nincs is), hanem mindig a szervezési/programozási részleg első embere képviseli az 'informatikát'.

A magyar vállalatokban nem akarják megérteni, hogy ez a bizottság nem egy szűkebb körű 'igazgatói tanács'. Ezért nem szakmai, hanem vezetői testületként működtetik azt. Abban a tévedésben élnek, hogy az egyes szakterületekhez ill. az informatikához azok vezetői értenek legjobban. Ezért az ülések nem érdemi, hanem presztízs jellegű viták jegyében telnek. Az erőforrások elosztása így nem a feladatokra, hanem voltaképpen a szervezeti egységeknek történik. Nem kell ezt a témát tovább ragoznunk: a lényegét úgymint érti mindenki.

A konkrét fejlesztési feladathoz igazi **projekt-csoportot** [project team] kellene szervezni. Ez a csapat - mint neve is mutatja - nem egy fix szervezet, hanem két értelemben véve is rugalmas. Egyrészt az összetétele a projekt fázisai (minőség) és a munka volumene (mennyiség) szerint időről-időre megváltozik. A projekt elején és végén az ügyvitelszervezők; a vége felé az üzemeltetők; menet közben ilyen-olyan (belső vagy külső) szakértők; most több programozó stb. bevonása szükséges. Másrészt a személyek projektek között is cserélődnek. A szervező és a programozó nincs 'madzaggal' egymáshoz kötve. Ma együtt dolgoznak, de már holnap a szervező a másik projekt résztvevője lesz vagy éppen fordítva.

Ezt a munkamegosztást nevezik az informatikusok projekt-szervezésnek. Nem azt, amikor minden projektnek mindig a számítástechnikai részleg első embere a vezetője és mindig ő képviseli a feladatot az előző bizottságban. Nem azt, ahol a szervezők és a programozók egymáshoz vannak láncolva. A merev informatikai szervezeti felépítésnek a projektbe való átvetítése nem megoldás. Az ún. **mátrix-szervezésről** nálunk még csak mesélnek, azt nem alkalmazzák.

A mátrix-szervezés azt jelenti, hogy a szakemberek egyrészt merev szervezeti egységekhez tartoznak az általános szakmai, munkaügyi stb. dolgokat tekintve, másrészt rugalmasan változó projekt-egységekhez kapcsolódnak, ahogyan azt a konkrét fejlesztési feladatok megkívánják.

A fenti 0 - A - B - AB képletnél utaltunk rá, hogy vannak 'általános véradók', tehát olyan szakemberek, akik az átfogó, a projektnek nem egyetlen szakaszához kötődő feladatokat (0) látják el. Az ilyen (a civilizált informatikájú környezetben már 20 év óta ismert) szerepkörök nálunk még ma sem találták meg a helyüket.

Adatadminisztrátornak vagy **adatgazdának** a vállalati főinformatikust hívjuk a főmérnök illetve az anyaggazda mintájára. Az informatikus az ismereteknek a megismeréséhez, elrendezéséhez és kezeléséhez értő valaki. Az adatgazda tehát az a személy, aki a vállalat egészének a szintjén teszi ezt. Magyarországon ez a szerepkör két ok miatt nem létezik. Egyrészt a vállalatot senki sem óhajtja egy informatikai lényegként szemlélni. A 'rendszernek' titulált rendszerrészek - sőt, olykor csak funkciócsoportok - felelőseit **rendszergazdának** hívják. Szerepében a működtetésen van a hangsúly, nem a fejlesztésen. Feladata technikai jellegű - és nem tartalmi. A rendszergazdák - belőlük több van, szemben az adatgazdával - helyzetüknél fogva csak egymással össze nem függő részleteket láthatnak a cég egészének az informatikájából, azokat is az aktuális technikai burokból.

A másik ok, ami miatt nem létezik nálunk adatadminisztrátor egy általánosan elterjedt és téves nézetben lelehető fel. Magyarországon összetévesztik egymással a szakembert és a menedzsert. Azt hiszik, hogy a jó szakemberből jó vezető lesz és megfordítva, ha valaki jó vezető, az bizonyára egyben jó szakember is. Ergo: ha az informatikai vezető jó menedzser, akkor bizonyára jól fognak menni az informatikai szakmai dolgok, tehát felesleges a főinformatikus mint olyan.

Itt nyilvánvaló szereptévesztésről van szó. Az informatikai vezetőnek a dolga az erőforrások menedzselése, a koncepciók kialakítása, általában a makroszintű dolgok elrendezése. A főinformatikusnak - megtévesztő neve ellenére - éppen ellenkező előjelű a feladata. Ismernie kell a rendszer - adatok, feldolgozások stb. - részleteit, hogy azokat összehangolhassa.

A következő pontok mind az adatadminisztrátor feladataival függenek össze. Ezt az alpontot két speciális feladat ismertetésével zárjuk le. A szabványokat az IR igen fontos tényezőinek

tekintettük. Minden szabvány annyit ér, amennyire azt betartják. A magyarok híresek arról, hogy csodálatos terveket szőnek, amiket azután sohasem valósítanak meg. (Ld. a magyar törvényhozást és -ellenőrzést.) Így van ez a szabványok területén is. A szabványt megalkotják, majd azt a kutya sem veszi figyelembe. A **minőségellenőrzés** [quality control] feladata lenne az, hogy a fejlesztés és a működtetés során betartassa az előírásokat. Márpedig nálunk ilyen részleg vagy nem is létezik, vagy senki sem veszi komolyan azt, amit mond.

A fejlett cégeknél szabványos eljárás a **változáskezelés** [change management]. Az IR változik. A felhasználó nem ért az ismeretekhez. Ezért egy-egy nüansznai módosítás esetén is fejlesztésért kiált, viszont a valóban mélyreható változáskor ő csak apróbb karbantartási feladatot lát. Példa: Szükség lenne egy új kimenetre, tehát fejlesztésre (megoldása akár félórát is igénybe vehet). Ellenpélda: Csak az a gond, hogy egy karakterrel meg kell növelni az azonosítót (ez az 'apró' dolog több hétig is eltarthat). Nálunk egészen egyszerűen nincs józan mechanizmus arra, hogy az elkerülhetetlen változtatási igényeket mikor kell projekt-mederbe terelni (újrafejlesztés) és mikor elegendő egy kis kiegészítés (karbantartás). Ha pedig valaki mégis látja, hogy a változtatás lényeges, senkinek eszébe nem jut a projektet újraéleszteni. Az összefüggésekre való tekintet nélkül 'sutyiban' fogják végrehajtani a változtatást - egy dolgot javítva, kettőt rontva.

T 7.4 Egy programozó kitalálta, hogy az azonosítóban kellene még azt az ismeretet is feltüntetni, hogy... Majd 'véletlenül' kiderült, hogy emiatt a mások által fejlesztett adatbázist is át kell szerkeszteni. Projekt nem volt, a megbeszélés pedig annyiból állt, hogy ezentúl pedig így lesz és pontum.

Az adatadminisztrátor, a minőségellenőr és a változásmenedzser szerepe nem független egymástól. A fejletlen kapkodás csak tudatos renddel kerülhető el...

7.2 A SZABVÁNYRENDSZER

7.2.1 A szabványokról általában

A **szabványok és szabványos eljárások** [standards & procedures] minden IR nélkülözhetetlen környezeti tényezői. Az informatikusnak mindig gondolnia kell arra, hogy az IR nem zárt. A cég információs rendszere ezernyi szállal fűződik a külső környezethez, ezért tekintetbe kell venni a **külső** konvenciókat. Természet szerint ezek többfélék lehetnek. A kötelező **előírásokon** (törvények, rendeletek - vö. például a könyveléssel) kívül érdemes figyelni az **ajánlásokra** (pl. ISO, IFIP stb.) is. Azt sem szabad elfelejteni, hogy léteznek olyan többé-kevésbé elterjedt **szokásokon** alapuló megoldások, amelyek vagy egy-egy szakmán belüliek (ld. például a CASCO kék-sárga papírját), vagy általános érvényűek (pl. számlák).

A külső szabványok rendkívül sokfélék. A kötelező előírásokat kivéve olykor többféle ajánlás illetve szokásos megoldás közül válogathatunk. Azt is tudnunk kell, hogy az általunk megvásárolt **eszközök** is szabványok hordozói nemcsak abban a tekintetben, hogy maguk is megfelelnek bizonyos megegyezéseknek, hanem abban is, hogy alkalmazásuk esetén adott megoldásokat kényszerítenek ránk. Ez a kitétel még a legegyszerűbb eszközökre nézve is igaz.

Az IR felhasználóknak ki kell alakítaniuk a szabványok **belső** rendszerét. Ez a feladat a 'kényelmes/kellemes kényszer' kategóriájába tartozik. Minden tudatos emberi tevékenység a kommunikáción alapul, amihez pedig nélkülözhetetlen a közös nyelv használata. ('Nyelv' alatt itt

a közlés általános eszközét kell érteni, nem pusztán az írott és kimondott szavak együttesét.) Az információs rendszer sokszor azért nem sikeres, mert a résztvevők „nem beszélnek közös nyelvet”.

A belső szabványok kialakítása több dolgot jelent. Egyrészt ismernünk kell a kötelező külső előírásokat. Másrészt ki kell választanunk és adaptálnunk kell az ajánlások/szokások közül a számunkra megfelelőeket. Harmadrészt ‘feltalálói’ munkát kell végeznünk, vagyis kutatnunk kell a szabványosításra alkalmas, de az előzők által le nem fedett tényezőket.

Ezzel a munkánk még korántsem fejeződött be. Az IR-en belül a szabványok is saját rendszert alkotnak. A **szabványrendszer** maga is igen komplikált dolog. Egyrészt - mint látható - a szűkebb és tágabb külső környezet egymással sem mindig tökéletesen harmonizáló meg-egyezéseit kell átvennünk úgy, hogy azok és a saját belső meg-egyezéseink összhangban álljanak egymással. Másrészt a szabvány emberekhez és eszközökhöz kötött. A mérnök másféle konven-
ciókban gondolkodik, mint a közigazda. A relációs és az objektum-orientált adatkezelők alap-
fogalmai eltérőek, jóllehet azok készlete az egyik legfontosabb konvenció.

Harmadrészt gondolnunk kell arra is, hogy a szabványok bizonyos tekintetben maguk is absztrakciók, hiszen ‘leírt’ dolgok. Ebből következően van **tartalmuk**, azt hordozó **formájuk** és - mivel nem függetlenek egymástól - **szerkezetük**, azaz elrendezésük. Ezt a kitélt megfordítva azt is mondhatjuk, hogy a konvenciók az IR tartalmi, formai és szerkezeti aspektusaira vonatkoz-
hatnak.

Ezzel persze a körképet még korántsem vázoltuk fel a maga teljességében. Az osztályozás legcsekélyebb szándéka nélkül rá kell mutatnunk, hogy szabályozni kell a rendszer **életciklu-
sának** a szakaszait (fejlesztési/működtetési szabványok); a fejlesztés kétféle **fonalát** (tartalmi munka és projekt-menedzsment); továbbá az IR két **síkját** (alkalmazói és fejlesztői szabványok). Az utóbbi páros kapcsán emlékeztetünk arra, hogy vannak speciális és általános konvenciók. Például az alkalmazásban a lakcímetek úgy kell írni, hogy... és viszont a tervezésben az adat-
neveket úgy kell írni, hogy... (Az utóbbiak a metasabványok.)

Az alpont lezárásaként kell utalnunk arra, hogy a szabványok kialakításához több feltétel összehangolása szükséges. Először is meg kellene alkotni magát a ‘szabvány-konvenciót’. Ez írja le a szabványokról szóló tudnivalókat. Mire kell itt gondolni?

Ma a ‘diktált’ ún. szervezeti és működési szabályzatokkal (SZMSZ) próbálják pótolni a valódi konvenciók hiányát. A ‘diktált’ két dolgot jelent. Az egyik az, hogy az értelmes szakmai meg-egyezések helyett az SZMSZ valóban diktátum. A másik az, hogy az SZMSZ - tisztelet a kivételnek - ‘tollbamondott’, ‘lediktált’ szöveg. Egyrészt nem egy feszes szerkezetű, kötött tartalmú és mindig azonos formátumú ‘közlöny’, hanem bizonyos gondolatok szubjektív meg-
fogalmazása. Másrészt nem tükröződik benne a más szabványokkal való összefüggés, holott a szabványok rendszert alkotnak, tehát a közöttük lévő kereszthivatkozások igen lényegesek. Harmadrészt az SZMSZ-ek halmaza ad-hoc, amennyiben nem előre elhatározott és rendszerezett elgondolás terméke, hanem a ‘ja, most ezt jó lenne szabályozni’ gondolatroham jegyében szü-
letik. Negyedrészt mindennek pedig az az oka, hogy nincs főinformatikus, nincs minőségellenőr ill. változásfelelős (ld. az előző alpontot). Nincs ‘szabványgazda’; az új belépő nem azzal kell, hogy kezdje a munkáját, hogy megismerkedik a ‘belső hagyományokkal’ stb.

Magyarul és összefoglalóan: A hazai vállalatokban vannak előírás-kísérletek, szabvány-
rendszer viszont nem létezik.

7.2.2 Milyen szabványokra van szükség?

Természetesen arra nincs lehetőség, hogy az IR ezernyiféle tényezőjére és az azok össze-
függéseire vonatkozó szabványosítási alkalmakat és feladatokat mind felsoroljuk. Ezért az
alábbiakat csak válogatásnak tekintse az olvasó.

Az információs rendszer fejlesztése nem akkor kezdődik, amikor nekivágunk egy-egy konkrét IR kialakításának. Mielőtt ezt megtennénk, már szabványokra van szükségünk. Most ne beszéljünk azokról a cégekről, amelyekben egyáltalán nem végeznek rendszertervezést, vagy pedig - és rengeteg ilyen vállalattal lehet találkozni - azt mindig ad-hoc módon, 'szervezője válogatja' alapon teszik. Az egészséges informatikájú helyeken már a fejlesztéseket megelőzően kialakítják a szükséges szabványokat, amelyek több kategóriába esnek.

A 4. fejezetben beszéltünk a *fejlesztés módszeréről*, aminek minden kelléke a szabványosítás tárgya kellene, hogy legyen, mégpedig egymásra épült módon. Az nem lehet, hogy az alapkoncepciónk és az azt feltételező struktúra illetve terminológia ellentmondásban álljon pl. az alkalmazott fejlesztési eszközzel. Az sem megoldás, hogy a szoftvereladók, barátok, külső szakemberek minden új ötletét kritika nélkül úgy vesszük át, hogy a megoldásaikat nem egyeztetjük a szabványainkkal. Főleg új eszközök vásárlásakor vagyunk hajlamosak arra, hogy a régi dolgokat félresöpörjük és implicit szabványként - azért rejtetten, mert tudatosan nem véghezvük el a szabványrendszer karbantartását - kezdjük használni pl. az eszközhöz kötődő fogalomrendszert.

A módszertani konvenciók maguk is sokrétűek. Ide tartoznak a *tervtermékre* és a *tervezési eljárásra* [design product & procedure] vonatkozó megkötések. Ne feledjük, hogy a fejlesztés maga is rendszer! A tervek fejlesztési adatbázist alkotnak, amit a fejlesztési események kapcsán meghatározott erőforrásokkal a fejlesztők dolgoznak fel. Mindezeket a tényezőket szabályozni kell.

Intermezzo

Itt egy általános ellenpéldát kell bemutatnunk. Az átlagos magyar vállalatban a fejlesztés a következő módon zajlik:

Valamelyik felhasználó (az persze nincs szabályozva, hogy ezt egyáltalán ki teheti meg); közvetlenül (a projekt- ill. a változásmenedzser kimarad a láncból); telefonon (vagyis írásban történő rögzítés nélkül); megkeresi a fejlesztőt (vagyis azt az emberkét, akit ismer és akit illetékesnek hisz); előadja neki a tökéletesen zavaros vágyait (mert még az elképzeléseit sem vetette papírra és fontolta meg); mire a fejlesztő addigi munkáját (engedély nélkül) megszakítva nekilát az ötlet megvalósításának (szintén engedély nélkül); X napi munka után pedig közli vele a felhasználó, hogy mégsem úgy gondolta...

Mint fentebb már említettük, a fejlesztésben két fonal húzódik végig. Az egyik az érdemi munkáé, az elemzésé, tervezése és megvalósítása. A másik a *projekt-vezetése*. Ezért projekt-szabványokban kell rögzíteni azt, hogy ki, mikor, milyen formában és kivel kommunikálhat a fejlesztés során. A fejlesztés időt, eszközt és pénzt igénylő munka. Az képtelenség, hogy vagy nincs is projekt, vagy annak a kereteit bárki megkerülheti. Nap mint nap tapasztalható, hogy az ilyen fejtelten kapkodás mennyi bajt szül. Az alkalmazási-felhasználó és a fejlesztő egymásra mutogat - ezt akartad, nem ezt akartam -, a fejlesztési vezető pedig az egészről semmit sem ért.

A *projekt-megvalósításban* is sokféle konvenciót lehetne alkalmazni. Előbb a kivitelezésre térünk ki, és majd alább foglaljuk össze az elemzés és a tervezés szabályozási lehetőségeit. A fejlett környezetekben széleskörűen használnak ún. *program(ozás)-technikai* szabványokat. Nem csak a képernyő és az írásos kimenetek egységes elrendezése, a menük és a help kezelése, a funkciógombok hozzárendelése stb. tartozik a szabályozás körébe. A programok szerkesztése, az utasítások használata, végeredményben a megírási mód maga is egyezményes.

T 7.5 A szerzőt megkérdezték, hogy vajon miért nem képes lefutni az adott számítógépen a 6000 soros - Clipperben megírt - 'rendszer'.

Rendszer? Egyetlen monolit programként? Készítőjének fogalma sem lehetett a programozástechnikáról, például a szegmentálásról! Ellenpélda: a hetvenes évek közepén a Sz...ben a fejlesztési főosztályvezető sorra kirúgta azokat a programozókat, akik 800 sorosnál hosszabb COBOL-programot írtak. Ő tudta, hogy miért kellett ezt tennie... Másik ellenpélda: A szerző kolléganője pár éve készített egy 101 programból álló, cirka 40-50 ezer soros rendszert, ami - hála az ésszerű megoldásoknak - a legkisebb PC-n is képes volt működni.

Intermezzo

Az objektum-orientált koncepciót érdemes figyelemmel kísérnünk, mert abban a programozási szabványok csírái rejtőznek. Az objektumként megfogalmazott - és ekképpen transzferálabilis - programrészek az egységesedés felé mutatnak. Még nem szabványok, csak 'egyéni ajánlások'. Azért nem konvenciók, mert vezetési szabványok hiányában az X programozó objektumait senki más nem veszi át - még saját maga sem a következő feladatában. Az objektum ugyanis csak olyan elvi lehetőség, ami nem sokat ér, ha nem párosul a szabvány kényszerével.

Sajnos, ma a programozás a programozó magánügye. Programját mások elől rejtegeti (bizonyára jó oka van rá) és/vagy átláthatatlan megoldásokkal tűzködi teli, hogy 'vissza ne fejtsek'. Régen még létezett a 'programkönyvtáros' funkció úgy, hogy a megírt programokat a könyvtárba le kellett adni, ahol azokat ilyen-olyan szempontok szerint ellenőrizték. A programozás kérdését azzal zárjuk le, hogy a vállalatokban egyáltalán nincsenek *tesztelési* és *átadási* szabványok. A hármas tesztelés - 'unit' teszt, 'integrációs' teszt, 'átadási' teszt - koncepciójáról a legtöbben nem is hallottak. Nincsenek a tesztadatok méretére, összetettségére, a funkciók vizsgálatára stb. vonatkozó konvenciók. Így még az is előfordulhatott, hogy 'tesztelés' után (!) átvettek egy olyan rendszert, amelyben az aktualizálási menüpontra lépve a rendszer simán elszállt (nem volt megírva az a programág).

A szabványok 'legelhálásabb' területe a *tervezéstechnika*. Ennek az az evidens oka, hogy az adat és a feldolgozás - absztrakció. A normális tervező magában és magának mindig kidolgoz egy 'metódust' a rendszer tényezőinek az elrendezési, elnevezési, leírási stb. módjára. Innen már csak egy lépés, hogy ezeket az egyéni metódusokat egyeztessék - és közös szabványt alkossanak belőlük. A teljesség és a részletesség igénye nélkül felsorolunk néhány szabványosítható tényezőt:

- **Karakterek.** A szoftverszállítók ellenségei. Az ugyanis képtelenség, hogy az egyik programverzió nem ugyanazzal a betűkészlettel dolgozik, mint a másik. Az is furcsa, hogy ugyanattól a pénzintézettől háromféle (!) betűkészlettel írt bizonylatokat kapunk. Ékezet nélkülít, magyar és nem-magyar ékezeteset.

- **Adatnevek.** A különféle adathordozókkal ma az egyik fő gond az, hogy az ismeretek megnevezései finoman szólva 'ziláltak'. A papíron, a képernyőn és belül az adatbázisban ugyanazt a dolgot egymásra még csak nem is emlékeztető nevekkkel illetik. A nevek olykor a kelleténél beszédesebbek, olykor hallgatagok, no meg az is előfordul, hogy a papírokon az adatnak nincs neve, mert például dobozokat kell kiikszelni (adatnév nélkül értékek).

- **Azonosítók.** Az azonosító bizonyos értelemben maga is 'név' (vö. nominális azonosítás). A fejlesztők - a hetvenes évek lyukkártyás gyakorlatát követve - ma is az azonosítóba akarnak tuszkolni mindenféle osztályozási ismérvet. Ez pedig számos indexelési és szelekciós problémát okoz. Ráadásul az egyik tervező így, a másik pedig úgy képzei el a saját 'rendszerében' ugyanannak a jelenségnek az azonosítását. Ez vagy kapcsolhatatlanságot eredményez, vagy mindenféle 'okos' trükkkel lehet csak összekapcsolni az ismereteket. Ezeket a gondokat elkerülendő az azonosítók általános használatára vonatkozó szabványokat kell kidolgozni.

- **Kódok.** A kódok az egyértelműség és a tármegtakarítás hasznos eszközei. Ma már nem a régi módon használjuk őket; legalábbis a tranzakció-orientált kezelés során nem a kód, hanem a

valós tartalom bevitelét várjuk az alkalmazótól. Ezt mindenféle segédtablák - kód/megnevezés párosok gyűjteményei - segítségével érjük el. A kezelő megnevezést választ, a rendszer kódot tárol. Ez így rendjén is lenne, ha... Ha ezek a segédtablák egységesek - szabványosak - lennének. Ámde sokszor nem azok. Például ahány 'rendszer' építenek, annyi különféle település segédtablát használnak. Hol a '01' jelenti a Budapestet, mint 'megyét', hol a '20'. (A 19 megye mellett sokszor Budapestet is kiemelt fontossággal kell kezelni. A 'Megyekód' adatnév ilyenkor hazudik, mert Budapest nem megye. Ám egyelőre feledjük el, hogy az adatnévnek a valós tartalmat kellene tükröznie. Itt és most a kódok egységes jelentéséről, szerkezetéről és konkrét tartalmáról van szó.)

A feldolgozási jelek is kódok. A kezelőgombok is azok. Az képtelenség, hogy pl. az adatbevitelt hol az 'A' vagy a 'B' betű, hol az F3 vagy az F4 gomb jelzi. A kódokat a normális ember kerüli. Szemben a mindent rövidítő amerikaiakkal az európai a kerek beszédet kedveli. Ha tehát mégis kódokra kényszerítjük, akkor legalább ügyeljünk azok tisztaságára és egyértelműségére.

Az információs rendszer terve dokumentációban kerül rögzítésre. Kézenfekvő tehát, hogy normális vállalat nem létezhet **dokumentációs-szabvány** nélkül. Ám amit el lehet rontani, azt el is rontjuk. Nem egy olyan vállalat létezik, amelyben előírják azt, hogy a dokumentációt milyen *eszközzel* kell készíteni. Azt viszont nem, hogy mi legyen az iromány *szerkezete, tartalma és formája*. Hogyan kell felépíteni a dokumentációt; az egyes fejezetek miről szólnak; milyen módon történjen az ábraszámozás, a kereszthivatkozás, a kiemelés stb. stb.

Ezt a pontot most le kell zárunk. Lehet, hogy egy kicsi csalódást okoztunk, de az olvasónak meg kell értenie, hogy nem volt feladatunk a konkrét konvenciók ismertetése. „Ízlések és pofonok...”. A szabvány - belügy. Csak arra hívhattuk fel a figyelmet, hogy a szabvány „Sine qua non.”. Nem létezhetnek információs rendszerek szabványok - sok-sok megegyezés - nélkül.

7.3 RENDSZERDOKUMENTÁCIÓ

Ez a könyv nem lenne teljes, ha egy valóban nagyon rövid pont keretében nem térnénk vissza a dokumentációk kérdésére. Ezt meg kell tennünk annak ellenére is, hogy a dokumentálás valójában a projekt-menedzsment témakörébe tartozik, amivel pedig ez a mű nem óhajt foglalkozni.

Mint már említettük, a fejlesztésben két fonal húzódik végig: a megalkotásé és a menedzselésé. Ezért eleve szét kellene választani egymástól az adminisztrációs papírokat, vagyis a **projektdokumentációt** és a fejlesztési munka végeredményét rögzítő köteteket, a valódi **rendszerdokumentációt**. Meg kell jegyeznünk, hogy a teljes fejlesztési dokumentáció nem merül ki az előző kettőben, mivel a munka során születnek olyan dokumentációs részek is (pl. változatok, vázlatok), amik nem kerülnek a végső tartalmi leírásba.

Általában a rendszerdokumentáció három kötetet szokott felölelni. Persze itt a 'szokott' kifejezés enyhe túlzás, mert igen sokszor dokumentáció egyáltalán nem készül és főleg nem három kötet; a legtöbbször pedig utólag írják meg, kényszer hatására és a tényleges IR-re csak távolról emlékeztet.

A három kötet a következő: Az első a **tervdokumentáció**. Ez nem más, mint a rendszerrész teljes strukturális leírása, amit alább bővebben is kifejtünk. Ez az alapja a másik kettőnek, és éppen ezért ezt kell először elkészíteni.

A második a **felhasználói dokumentáció**. Ez önmaga sem egyetlen, homogén dolog. Célja az alkalmazási- és a végső-felhasználók eligazítása. Amikor például tájékoztatót kapunk a banki ügymenet megváltozásáról, akkor ezt az eligazítást is bátran a felhasználói dokumentáció részének tekinthetjük, hiszen mi vagyunk a végső-felhasználók. Ez a kötet nem homogén. Annyi

plusz egy fejezetet kell tartalmaznia, ahányféle alkalmazási-felhasználója van a rendszernek. Ugyanis ha az egyik ügyintéző csak az X-féle esetekkel, anyagokkal stb. foglalkozik, akkor felesleges őt a másik ügyintézőre tartozó dolgokkal terhelni. A plusz egy fejezet - a 'fejezet' itt képletesen értendő - a közös technikai tudnivalókat írja le.

A harmadik kötet az **üzemeltetési dokumentáció**. Ez a működtető személyzet eligazítására szolgál. Felépítési logikája az előzőéhez hasonló, de egy picit még összetettebb. Azért, mert az üzemeltetésben sokféle funkció létezik. Nemcsak a rendszerrészek, hanem a működtetési szerepek (pl. rendszergazda, operátor stb.) szerint is külön fejezetekben kell kifejtetni a tudnivalókat.

Most térjünk vissza a tervdokumentációra, ami számos félreértés tárgya. Nem regényről, nem szakkönyvről van szó. Ehhez képest meglepő, hogy a szigorúan tárgyilagos, zömében táblázatokból és diagramokból álló valódi dokumentáció helyett olykor 'elbeszélő' stílusban készítik a rendszerleírást. A **formailag** rossz megoldások egyébként is divatosak. Pl. nincsenek kereszt-hivatkozások; a listák sorrendje ad-hoc (az egyetlen célszerű ABC-szerinti helyett); sőt

T 7.6 Egyszer a kezembe kaptam egy dokumentáció egyetlen példányát. 'Véletlenül' leejtettem és széthullott. Nem volt oldalszámozva... Nem volt tartalomjegyzéke... Nem volt fogalomjegyzéke...

A tervdokumentációk **szerkezete** csak ritka esetben megfelelő, **tartalma** pedig sokszor hiányos. Ez azért van így, mert nincs dokumentációs etalon, szabvány. Pedig a képlet egyszerű. A kötetet mindig **bevezetővel** kell kezdeni. Ez a célt és az átfogó képet mutatja be. Ehhez definiálni kell a rendszer tartalmát; kijelölni annak határait; leírni a belső és külső környezet legfontosabb összefüggéseit és felvázolni a rendszer makrostruktúráját (tagolását). Minderről az 1. fejezetben volt szó. A bevezetőhöz tartoznak a 3.1 pontban leírt elemzési szempontokra vonatkozó döntések is. Most csak egyet emelünk ki: 'Hová fejlődik?'. Vagyis nemcsak azt kell leírni, hogy a rendszer most mit és hogyan tartalmaz, hanem azt is, hogy milyen irányban és módon fogjuk továbbfejleszteni.

Mivel a rendszer szervezett együttes, evidens, hogy a dokumentáció 'testében' a tényezőket és azok összefüggéseit kell leírni. A makroszerkezet szerint **adat-, feldolgozás- és környezetvetületi** fejezetekre van szükség, mégpedig ebben a logikus sorrendben. Ellenpélda: A mai dokumentációk a számítógépes eszközök ecsetelésével kezdődnek és egyébként is millió helyen utalnak a gépekre. Ezért ha az eszköz változik, az egész dokumentációt újra kell írni ahelyett, hogy csak az utolsó fejezetek egyikét cserélnék ki benne. Mivel az adat- és a feldolgozás absztrakció, törekedni kell arra, hogy a leírás a 6.1 ábra szerinti aspektusok és szintek szerint történjen. Ez megint a dokumentáció karbantartása miatt fontos: ha egy megvalósítási részlet megváltozik, akkor azt csak valóban egy helyen és minimális erőfeszítéssel kelljen átvezetni a dokumentáción.

Már csak egy dolog maradt hátra. Ismételten fel kell hívni a figyelmet arra, hogy a dokumentáció közkincs és annak a szervezet a tulajdonosa.

T 7.7 Fiatal külsős programozók 'rendszert' készítettek egy cég részére. Majd hetekig vitatkoztak azon, hogy kié a dokumentáció. Át kell-e azt nekik adni a cégnek, vagy sem?

Ha veszünk - mondjuk - egy tévét és nem kapjuk meg az áramköri rajzát, akkor el se vesszük a készüléket. Ha a lakásunkat felújítjuk, akkor a mérnök át fogja adni nekünk a műszaki leírásokat. A számítástechnikában viszont terjed - mert a cégek engedik - egy általános félreértés. A szerzői jogra apellálva egyes programkészítők nem adják át a rendszerdokumentációt a felhasználónak. No és mi történik akkor, ha például a programkészítő kft tönkremegy? Ott áll a cégünk egy teljesen 'fekete doboz' termékkel, amit úgy ahogyan van kidobhat, hiszen se javítani, se bővíteni, se átalakítani nem tudja.

A rendszerdokumentációt mindig meg kell követelni. Ha szállított termékről van szó, akkor a garancia és a szerzői jog védi meg a szállítót attól, hogy a vevő illetéktelenül használja a dokumentációt. Ha belenyúlok a tévébe, elveszítem a garanciát. Ha belenyúlok egy szállított rendszerbe, ugyanez történik. Ezért meg fogom gondolni a beavatkozást, de a tévé az enyém... Ha lemásolom egy tévé áramköri rajzait azért, hogy azokat én is tévékészítésre használjam, megsértem a szerzői jogot. Ugyanez vonatkozik az IR tervére is.

A rendszerpszichológia körébe tartozó megjegyzéssel zárjuk ezt a pontot. A számítástechnikusok roppant rátartiak. Azt hiszik, hogy valamit csak ők képesek megcsinálni. Gondolkodásuk kétszeresen is illogikus. Egyrészt azért, mert mind azt hiszi, ergo mindegyik képes. A valódi profi bármilyen rendszert vissza tud fejteni... Másrészt azért, mert éppen a felhasználótól nem kellene félni, hiszen ő úgysem képes - tehát nyugodtan átadható neki a dokumentáció.

7.4 KÖRNYEZETI ÖSSZEFOGLALÓ

Ez a fejezet szükségszerűen más felépítésű és hangulatú, mint a korábbiak. Rövidebbre is sikerült amazoknál. Mindennek két oka van.

Egyrészt az adat és a feldolgozás olyan absztrakció, amelynek a célszerű végrehajtásában nem lehet figyelmen kívül hagyni az emberi momentumokat sem, de elsősorban a fejlesztő **technikai tudásán** múlik a tevékenység sikere. Ezzel szemben nem a fejlesztő határozza meg a környezet tényezőit. Ő úgymond csak belecsöppen egy fejlesztési szervezetbe, ahol a munka- és projektservezés, a szabványok és a dokumentálási rend olyanok, amilyenek. Azaz minden leendő informatikusnak számolnia kell a vágyai és a valóság mindennapos ellentéteivel. Persze a helyzetének az áttekintéséhez látnia kellene egy 'etalont'. Nevezetesen azt, hogy a környezetben mi lehet jó és mi rossz. A szerző olykor meglepődik, amikor fiatal informatikusok a képtelen fejlesztési körülményeiket úgy mutatják be, mintha az lenne a világon a legtermészetesebb mód. Ezt a fejezetet ezért egy picit ébresztőnek is szántuk: azt akartuk elmondani, hogy lehet másként is.

Másrészt arra is fel kell hívnunk a figyelmet, hogy a 'környezet' nem pusztán objektív dolgokból áll és nem kell annak minden tényezőjét ténynek venni. A környezet 'miliőt' is jelent. A fejlesztés mindig is 'idegmunka' volt. Ezért aligha csoda, hogy a fejlesztési részlegeken a szokásosnál gyakoribbak a súrlódások. Ez egy tény, ám manapság egyre sűrűbben tapasztaljuk a fejlesztők keserűségét, ami sajnos egy másik tény. Nincs idő, feszített a tempó, zord a vezető, csacsi az alkalmazói-felhasználó - ezek napi panaszok. Ezekre való hivatkozással azután a fejlesztő úgy véli, hogy elkerülheti a módszeres munkát. Pedig téved. Nem kell több idő a jó munkához, mint a rosszhoz. Senki sem gátolhat meg senkit abban, hogy tisztességes dokumentációt készítsen és abban sem, hogy szabványokat alkalmazzon, még ha azok nem is válnak (azonnal) elismertté.

A fejlesztéshez nem elegendő a technikai tudás: érteni kell az **emberi miliőt** is. Mármint ez az a szubjektív dolog, amiről a fiatal informatikusoknak a legtöbb iskolában nem beszélnek. Ezért azután a tapasztalatlanokat meglepetések érik. Ezt a sokkhatást elkerülendő ennek a fejezetnek az egyik rész célja a szemnyitás volt. Természetesen az IR teljes lelkivilágát nem óhajtottuk kitergetni; ezért a fejezet a szokásosnál rövidebb.

Az ellenőrző kérdések ezúttal elmaradnak. A fejezetben kevés a tárgyyszerűen megtanulandó dolog. Inkább a szemléletformálásra törekedtünk; szemléletből pedig nem lehet 'vizsgáznai' az iskolákban - csak az életben.

FELADATMEGOLDÁSOK

MEGOLDÁSOK - 1

- 101 A helyes gondolkodási sorrend a fogalomtól halad a megvalósításon át az eszközökig. Ezért a sorrend: 3-4-2-1.
- 102 Mint a neve is mutatja, a szövegmintának van rajza, vannak színei, van szövegmódja stb. Mindezek a dolgok szervesen összefüggnek egymással. A logikus válasz: I.
- 103 Kap Ön fizetést? Munkaeszközöket? Gondolja meg! A jó válasz: V.
- 104 Általában és elvileg persze, hogy a lakás a rendszer. A festő számára viszont csak a hálószoa az. A helyes válasz: H.
- 105 A festőnek nem csak a hálósobával kell törődnie. A jó válasz: N. A szoba határait a falak és a nyílászárók képezik. Az ablakokat és ajtókat kívülről is be kell festeni, mert az erkély és az előszoba a háló közvetlen külső környezete és hogyan nézne ki, ha...
- 106 A kisvállalkozó számára maga a vállalkozás a rendszer. Ezért az első minősítés: H, mert nem lehet két rendszerről beszélni. A harmadik tételre I a megfelelő válasz, mert két rendszerrészből van szó. Azt viszont ne mi döntsük el, hogy ezeket a kisvállalkozó miként rendezi el: tartalmazást vagy bennfoglalást alkalmaz-e tagolási módszerként. A másik két tétel lehetséges, de nem abszolút megoldást mutat. Esetükben a válasz: L.
- 107 Az eszköz nem tagolási ismérv, ezért a második tétel R. A másik kettő jó megoldás (J). A tagolás ugyanis lehetséges, de nem kötelező megoldás.
- 108 Ha egy X rendszert X1 és X2 részekre tagolunk, akkor ezt kétféle módon tehetjük. Úgy, hogy azok 'kitöltik' az X-et, ami szükségszerűen a két rész átfedésével jár. Például a hazai és az export rendelésekben is lesz egy-egy anyagállomány. Ez így nem jó megoldás. A másikféle módszer jelenti a helyes gondolkodásmódot. Az X tagolásánál mindig kell, hogy legyen egy olyan X' rendszerrész, amely a közös tényezőket tartalmazza. Ez a redundancia elkerülésének, az ismeretek integrálásának a titka.
- 109 Barátomnak egy rendszere sincs, mert a rendszer nem eszközkötött kategória. A számlán (Excel) mindig másként írja a vevő nevét, címét stb., mint a levelekben (Word). A számla kezelésekor pedig nem tudja előkeresni az anyagadatokat (Clipper). Hol van itt a 'szervezett együttes'?
- 110 'Minden mindennel összefügg.' Ezt az elvet nem feledve, a helyes válasz a következő: E-I-V-M. Persze, hogy kihat a gép állapota az ismeretekre, de a gépkarbantartás elsősorban műszaki tevékenység, amint az átszervezés elsősorban valós. A bizonylat tipikusan információs tényező. Viszont az új ismeret bevezetése mindhárom szférát érinti. Az ismeretet meg kell fogalmazni (IR) és gépen kell tárolni (ER) azért, hogy használják (VR).

- 111 A mai gyakorlati számítógép-alkalmazásokat az átfogó informatikai helyett a szűk technikai szemlélet jellemzi. Nem feladatokban és azokat felölelő, jól tagolt rendszerekben, hanem alkalmazandó eszközökben gondolkodunk. Ezen a szemléleten kellene mielőbb változtatni. Amíg nem késő, amíg nem költünk rengeteget használhatatlan 'rendszerekre'.

MEGOLDÁSOK - 2

- 201 A helyes válasz: I. Maga a leírt ismeretsor csak adat, mert értelmezhető - érzékelhető, észlelhető, felfogható - közlés. Azonban Ön ennek olvastán rögtön egy képet alakított ki magában. Kicsi, nagy, akkora, mint... Ez a következtetés viszont már információ.
- 202 A szerző férfiasan bevallja, hogy csak sejti, de nem érti a 'profilaxis' szó lényegét és lusta megnézni azt a szótárban, meg nem is érdekli. A helyes válasz a számára: S, mert a közlés neki nem értelmezhető. Ami nem zárja ki, hogy egyes olvasóknak az. Az pedig bizonyos, hogy a reklám célja a kódosítás - a nem-értelmezhető 'információk' közreadása.
- 203 Mivel a 'desszin' lényege nem közismert, a legtöbb olvasó valószínűleg csak a második fázisig jutott el. Számukra a helyes válasz: 2.
- 204 A helyes sorrend: 2-1-4-3. Az értékesítés osztályvezetője közel áll a dolgokhoz, ezért ő ért belőlük a legtöbbet. A főkönyvelő számára sem újdonság a számla lényege, de a konkrét dolgoktól már távol áll. Nem ismeri pl. annyira az eladott dolgot. Minden pénzügyekben érintett ember konyít valamit a számlákhoz, ezért a bérügyes sem elveszett egy számla láttán. A kiállító részére viszont nem születik információ – hiszen ez már megtörtént, amikor kiállította a számlát.
- 205 Ugye, hogy bajban van? A határozatlanság csakis két feltétel esetében használható fogalom. Akkor, ha valamit meg akarunk ismerni. Ha nem akarunk megismerni valamit, akkor is nyerhetünk információt. De annak mértéke meghatározhatatlan. Különben pedig kellene, hogy legyen egy előzetes mércénk (X, Y, Z darab halhoz képest mennyit fogtunk). Mivel az ismeret elsősorban minőség, a mérce leegyszerűsítése nem visz előrebb. Ezért a szerző a két kérdésre '0%' választ adna.
- 206 A helyes válasz: 3. Elvileg persze a tartalom a fontos, de mit érünk vele, ha nem tudjuk 'elolvasni', vagyis hiányzik az észlelés vagy a felfogás momentuma?
- 207 A helyes válasz: E-E-N-E.
- 208 A helyes válasz: I-N-E. A sokszorosítás fontos dolog, de nem soroltuk az információs tevékenységek közé. A tevékenység független az ismeret hordozójától, ezért a papírokban való keresgélés is az. Az Internetben való matatus - számítógép ide vagy oda - a legtöbb esetben nem számít információs tevékenységnek. Csak akkor az, ha a felhasználói részére fontos és a többihez kötött ismereteket eredményez, vagyis a bekapcsolás eleve valódi információs eseményt jelentett.

- 209 Az 'embernap' koncepciója jó, mert az erőforrásokat valahogyan mérni kell. A baj az, ha 'darab-darab' alapon a mennyiséget összekeverik a minőséggel. Például összeadják a szervezői és a programozói időket. A következmény evidens: valamelyik feladatra kevés idő jut.
- 210 „Tud úszni? Nem? És ha megfizetem?” A belső fejlesztőnél a külső nem dolgozik se jobban, se gyorsabban, de több pénzt kap. Kovács öt hét alatt megoldja azt a feladatot, amivel Szabó hónapokig kínlódna.
- 211 A beviteli idő általában semmit sem változik (2) attól, hogy maga a gép gyorsabb. A kezelési idő fő meghatározója maga az ember; az, hogy mennyire gyorsan 'gépeli'. Ha a bevétel összetett számítást is igényel, akkor némi - nem érdemi - gyorsulás várható. Lassulás is bekövetkezhet akkor, ha 'nagyobb a gépünk' alapon azt 'kitömik' - és ezzel megnövelik a lemez-hozzáférési időt.
- 212 Ön a végső-felhasználó, a könyvtáros pedig az alkalmazói-felhasználó.
- 213 A helyes válasz: I. Önmagában véve az, hogy van szabvány, mindig jó. Azonban a szabvány célja az, hogy a felhasználó életét megkönnyítse. Mármint az új csekkszám sem a végső-, sem az alkalmazói-felhasználó életét nem könnyíti meg, sőt.
- 214 A helyes válasz: N. A pénz arra való, hogy ekként funkcionáljon. „Ha a só elveszíti az ízét...” Nem ér semmit a pénzként nem használható pénz. Az ismeretek szintjén fogalmazva: mindennapi életünket sokszor nem maga a valóság, hanem a róla alkotott kép vezérli. Gondoljon csak az olvasó például az alaptalan féltékenykedésre...

MEGOLDÁSOK - 3

- 301 A kérdés valóban becsapós volt. A helyes válasz ugyanis az, hogy az öt felsorolt tényezőt csak egymással összhangban, együtt szabad szemlélni. Nincs sorrend! Például a cél nem lehet fontosabb a tartalomnál, mert anélkül nem is létezhetne.
- 302 A cikksorozat elkészítéséhez az szükséges, hogy Ön lássa elrendezetten a dolgokat. Gyakorlatilag kis többlet-feladatot jelent, de elvileg közömbös, hogy az ország melyik részéből és milyen eszközön kapja meg a cikkeket. A feladat - egy. A 'rendszer' - a cikksorozat - is egy. Az a meghatározó dolog.
- 303 A szervezeti átalakítások a feladat lényegét nem módosítják. Továbbra is rendelkezéseket, számlákat stb. kell kezelni. Csak akkor van baj, ha az éppen aktuális szervezethez 'idomítják' a megoldásokat. Ma erre, holnap vissza.
- 304 Az adat a legstabilabb vetület (1). A környezet már változékonyabb (2). A feldolgozás pedig nagyon variábilis (3). Ugyanazon ismeretek kezelésére ugyanazon eszközök mellett is mindig találunk újabb és újabb módokat. A fizikai függetlenség elvének a betartása mentesítene bennünket az adat- és a környezetvetület viszonyában beálló változások hatásaitól. A logikai függetlenség elvének a betartása az adat- és a feldolgozás viszonyainak a változásaitól.

- 305 Ez és a következő feladat összefüggő választ kíván. Azért, mert az adat és annak típusa kétféle jelentést hordoz. Most csak arra utalunk, hogy az adattípus a tárolási egységnek és annak kezelésének az egyik eszköze. A 'duplapontos' - SQL szabvány szerinti - numerikus adat éppen úgy szám, mint a nem 'duplapontos'. A felhasználónak elvileg semmi köze sincs ahhoz, hogy a kezelő az egyiket így, a másikat úgy bazírgálja a gépen, tehát fizikai szinten. (A válasz: 3). Éppen elég baj, hogy ezt a dolgot a logikai - tartalmi - adatbázis-tervben és így kell megadni, minden létező függetlenségi elvet megsértve.
- 306 A helyes válasz: 1-2-3. Az adattípust a mai adatkezelőkben igen zavaros módon használják. Olykor csak a tárolás egységét jelöli ki (vö. az előző feladattal), máskor viszont 'eszmei' tartalmat is tulajdonítanak neki. Erre épül például a validálás. A dátum olyan fogalom (1), amely tartalmilag ellenőrzendő (2) és egyben sajátos fizikai tárolást (3) feltételez. Nem az informatikus, hanem a számítástechnikus hibája, hogy a típus hol ezt (ld. előző feladat), hol azt jelenti...
- 307 A 'legmagasabb végzettség' és az 'előző munkahely' közötti különbség lényeges, amennyiben a fogalmakat megfelelően értelmezzük. Valakinek lehet több 'legmagasabb' végzettsége, de 'közvetlen' előző munkahelye csak egy. Ezért a személy adatsorában nem jelenhet meg az előbbi, de ott nem kivetendő az utóbbi ismeret.
- 308 A különböző szoftverek és változataik használata a fizikai függetlenség elvét éppen úgy sértheti, mint a logikaiét. Az ábrázolás - fizikai paraméter. Ha a magyar 'Á' betű itt ilyen, ott pedig más, az baj. Még nagyobb gond, hogy a különböző kezelők - egymásra nem épülve - megengedik azt is, hogy az ismeretek tartalmilag is eltérőek legyenek. A tartalom a formára vezethető vissza. Ha a vevő nevét átírom Excel-ben, az automatikusan nem íródik át a Word-ben. Tehát az adat és a program nem független.
- 309 Tökéletesen közömbös, hogy a három barát közül ki és milyen eszközzel vezeti a nyilvántartást (fizikai szint). Az is lényegtelen, hogy milyen rend szerint (logikai szint). Szemben az előző feladattal, nem gondolkodnak egy rendszerben, hanem csak közönsnek szánt ismeretekben. Ezért csak az a lényeg, hogy ismereteik veleje (fogalmi szint) azonos legyen. Például az egyik ne képzelje el a lakcímet irányítószám nélkül, miközben a másik azt is a cím részének tekinti. Nincs szükség direkt kommunikációra - tehát a megoldás egyeztetése is felesleges. A tartalomé nem az.
- 310 Én - mondjuk - Word-ben, egyik barátom Excel-ben, a másik Clipper-ben vezeti az előző feladatban említett ismereteket. Az részletkérdés, hogy e 'gusztusunk' megegyezik-e az eszközt illetően. Attól függetlenül három szint lesz a feldolgozásban. Fogalmi - be kell vinnünk és változtatnunk kell az adatokat. Logikai - eszközeink más-más módon biztosítják a számunkra a kétféle művelet lehetőségét. Fizikai - az első két eszköz esetén a kezelő szűrja el az ismeretek tárolását (ha kiakad), Clipper-ben ezt saját magunk tehetjük meg. A három szint látható, a három szint él - nemcsak az adat-, hanem a feldolgozás-vetületben is.
- 311 Ez és a következő öt feladat összefügg. Minden alkotáshoz három tervet kell készíteni. A hálózobánál is szükség van a nagyjából mit, mikor, mivel, kivel, mennyiért tervezetre (plan). A pontos felmérés után készül el a részletes elrendezés terve (design). Végül a száradással és egyébbel kalkulálva ütemezést (schedule) képzelünk el.

- 312 Minden projektben (a lakástatarozás is az) szükség van vezetőre, mondjuk így: fő-vállalkozóra. A vállalatokban még csak-csak kineveznek projekt-vezetőt, de az kevés. Az IR ugyanis egy, a projektek száma viszont több. Ez a dilemma csak úgy lenne feloldható, ha főinformatikust alkalmazna a cég. Ám ezt nem teszi.
- 313 A design - elegáns terv. A szobát is meg lehet csinálni szépen is, meg hanyagul is. A mestertől az ember nemcsak azt várja, hogy elvégezze a munkát, hanem azt is, hogy a jó megoldásra felhívja a figyelmünket és óvjon a rossztól. Egy példa: Ha rossz festéket erőltetünk és mondjuk nem eléggé szedik le a régit, a festék két hét múlva lepereg.
- 314 Célszerű írásban kérni a kalkulációt és a tervet (dokumentáció). Így azt meg lehet beszélni a családdal, rendelkezésre áll az ellenőrzés alapja stb.
- 315 Jobb, ha a dolgok egy kézben vannak. Mondanivalónk lényege az, hogy a tervezés és a szervezés szétválasztása általában nem kívánatos. Ezzel az elvvel szemben a mindennapos informatikai gyakorlat más. A rendszer-szervező mindent csinál, csak nem szervez...
- 316 A 312 megoldásban már utaltunk a lényegre. Minden hosszabb időt, több embert, sokféle erőforrást feltételező és így azok összehangolását igénylő feladat - projekt. A lakástatarozás éppen úgy, mint az IR-fejlesztés. Ezért az utóbbit illetően nem kell kitalálni a spanyolviaszt. A fejlesztés, tervezés, dokumentáció, szervezés, kivitelezés stb. mindennapos józan gyakorlatát kellene csak átvenni.

MEGOLDÁSOK - 4

- 401 Ön koncepciózus ember? Ha igen, akkor első dolga, hogy elgondolkodjon az alap-koncepción. Tehát a helyes válasz: 2.
- 402 Az Excel és a valódi adatbáziskezelő 'táblázat' fogalma nem azonos. Ez a terminológiai zavar egyáltalán nem az elméleti szintre tartozik. Ön azt fogja hinni - joggal -, hogy a kétféle táblázat egymásba konvertálható. Nem az! Ön előbb-utóbb a zavaros fogalmak gyakorlati áldozata lesz.
- 403 A megfogalmazási mód lehet laza, vagy kötött. A szövegszerkesztő és a papír 'mindent elvisel'. A tervet le lehet írni pongyolán, a szakkifejezések elhagyásával vagy rossz használatával. A kötött megfogalmazás a pontosságon keresztül bennünket segít. Addig nincs értelme kritériumra, algoritmusra, dokumentációra, sőt eszközre sem gondolni, ameddig nem látjuk át a 'tervezési nyelv' lényegét.
- 404 Vegye észre, hogy a legközönségesebb, a leggyakoribb fogalmaink - azok pontos behatárolása nélkül - mindig homonimák. Például a szerző látott olyan tervet, amelynek egyik részében a 'cím' laccím volt, a másikban pedig titulus (dr., prof. stb.) Az egyértelműségi hiány megkérdőjelezheti a valósághűséget és az érthetőséget is. Nagyobb baj az, hogy a minimalitás érdemben nem vizsgálható. Ha az adat nem egyértelmű, akkor nem dönthető el, hogy redundanciát jelent-e a többszörösség, vagy sem.

- 405 Egyazon jelenségre számos esemény vonatkozhat. Különböző bemenetek és kimenetek lesznek a vele kapcsolatos tevékenységek alapjai. Ma még csak azt akarjuk tudni az anyagról, holnap pedig már mást is. Az IR legstabilabb vetülete az adat. Ergo: az a 'kályha'.
- 406 A dokumentáció - több funkciója mellett - a résztvevők tájékoztatásának az eszköze. Ezért mindig meg kell döbennem azon, hogy meghívnak egy megbeszélésre anélkül, hogy annak a tárgyát pontosan ismerném. (Az már csak hab a tortán, hogy a kérdéses értekezleten ötünknek négyféle 'verzió' volt a kezében. Az enyémben semmi, a többiekében más.)
- 407 A módszeresség hiányzik. Semmi akadálya annak, hogy ugyanazt (!) a rendszert én az egyik gépen így, másvalaki úgy dokumentálja. Ezért állítottam - joggal -, hogy a fejlesztési eszközök módszert nem nyújtanak. Legalábbis én a feladatban leírt esetben nem láttam a módszer nyomait.
- 408 Megegyezősek, konvenciók, azaz szabványok szerint nem lehet rendszert készíteni. A rendszer közös ügy. Mindenkinek egyformán kell értenie. Dokumentációja nem csak a megvalósításra és az önigazolásra való (én elkészítettem...). Nevezhetem-e 'krumplinak' a tízes számot? Megtehetem. Az olvasó derül. Az író pedig azokon mosolyog, akik a rendszer ürügyén pontosan ugyanígy járnak el.

MEGOLDÁSOK - 5

- 501 A válasz: 2. A rekord, a reláció, az objektum stb. logikai szintű kezelési és tárolási egységek. Az adatmodell tényezői fogalmi szintű tükrözések. Ezért érvényüket nem veszítik attól, hogy logikai megvalósításuk milyen módon történik. Itt eléggé zavaró, hogy egyesek pl. 'relációs adatmodell'-ről beszélnek. Az ugyanis nem egy konkrét modell - nem alkalmazási tükörkép. Mivel a kezelést is felöleli, az 'adat' jelző megtévesztő. Az lenne a helyes, ha 'relációs metamodellt' említenének, mert hiszen a informatikai tényezők absztrakt elrendezésének egy módjáról van szó.
- 502 A helyes válasz: 2 és 4. Az 1 ki van zárva, és nem csak azért, mert egy gépkocsinak rengeteg alkatrésze lehet. Még az sem elegendő ok, hogy az alkatrész nem ugyanaz a jelenség, mint a gépkocsi, hiszen más ismeretek írják le. A valódi ok a következő: Ha az én gépkocsimnak van egy pl. féktárcsa tartozéka, akkor ez a 'tartozék' kifejezés megtévesztő. Hiszen semmi akadálya sincs annak, hogy ezt a féktárcsát a barátom kocsijára szereljék fel - és ne az enyémre. A kocsi és az alkatrész tehát nem egy, hanem két egymással viszonyban álló dolog. Néha a viszonyt magát is ismeretekkel kívánjuk leírni (mikor került új féktárcsa a kocsira). Ezért a 4-es válasz is helyes megoldás.
- 503 A helyes válasz: M. Lenne külön tulajdonos egyed és a kocsinál vezetni kell - tulajdonoságként - hogy ki a tulajdonosa.
- 504 A helyes válasz: 2. A tulajdonság a felhasználói ismeretekre fenntartott fogalom. A zár is adat, de a cikket mint valós dolgot nem jellemzi.

- 505 Az azonosító egy nagyon sok gondot okozó tulajdonság. Pl. a Rendszám nem alkalmas a gépkocsik azonosítására, hiszen időben és térben - egy magyar és egy nem-magyar kocsinak lehet azonos a rendszáma - nem egyedi. Pl. új rendszámot kap a kocsi, amitől ő még nem másik dolog.
- 506 A deskriptív-ből képzett nominatív azonosító - pl. összekapcsoljuk a Rendszám és az Alvászám tulajdonságot a kocsi behatárolására - baja az, hogy hosszú. Ezért kettős azonosítást szoktak alkalmazni. A papírokon ez a hosszú azonosító jelenik meg (amit át kell írni, ha pl. alvázcsere történik), a számítógépen viszont - a Rendelésszám mintájára - egy belső, mesterséges, egytagú nominatív kulcsot használnak. Elvileg tehát a kocsi érdemi adatai, gyakorlatilag ez a 'pótazonosító' a kapcsolat eszköze.
- 507 Az M:N fokú viszonyt nem kapcsolattal tükrözzük. A rendelés és a cikk közötti viszony a rendeléstétel, amit ismeretekkel akarunk leírni. Ezért definíció szerint egyeddel tükrözendő. Egyébként sem lehet a rendelés és a cikk között direkt kapcsolatot létrehozni. A viszony mindig egy közös tulajdonságon alapul. A két egyedben pedig nincs ilyen. Ha a cikkbe beletennénk a rendelésszámot, a rendelésbe a cikkszámot, akkor ez ad 1) hatalmas redundanciát jelentene, ad 2) rengetegszer ismétlődő adatot kapnánk. Hány rendelés vonatkozhat egy cikkre egy évben?
- 508 A válasz: 2. Másutt az Ön cége egyedelőfordulás, nem -típus. 'Otthon' sem lehet típus, hiszen csak egy előfordulása van, tehát nem osztály.
- 509 A válasz: 2. Nem fogjuk a tanár egyedtípus minden előfordulásában megismételni, hogy a foglalkozás = tanár. Ezt a jellemzőt maga a csoport-képző ismerv - az egyed neve - már elárulja.
- 510 Az adatbázis az adatmodell szerint szervezett együttes. A Word-ben nem határozható meg adatmodell - tehát adatbázis sem.
- 511 Az objektum-orientált gondolkodás picit féloldalas. Már 1970-ben történt kísérlet arra, hogy az 'alapegyedet' (személy) és annak 'kiegészítőit' (pl. nyelvtudás, pótlékok, végzettségek, bérek stb.) egy 'fa'-ként kezeljék. A vonatkozó adatbáziskezelő (SYSTEM 2000) azóta letűnt a porondról. Azért, mert az adatmodell viszonyai hálósak - és nem hierarchikusak. Tehát nem pusztán az a gond, hogy óriási objektumokat kapunk. Hanem az is, hogy féloldalassá kényszerül a nézetünk. A nyelvtudás nemcsak a személyhez tartozik, hanem a nyelvhez is, hiszen viszony.

MEGOLDÁSOK - 6

- 601 A számítógépen tároljuk a számlát (belső aspektus). Kapunk egy papírt (külső aspektus). Ez a papír elvileg hiányos, mert nem szerepel rajta az 'áru neve' megjelölés. Gyakorlatilag nem az, mert mi jól tudjuk nélkül is használni. Ámde a belső és a külső között a fejlesztőnek összhangot kell teremtenie. Vagyis a feldolgozáshoz tudnia kell, hogy a 'só, cukor stb.' - a gépen tárolt 'áru neve' tulajdonság egy-egy értéke. A tartalmi és formai elrendezés (logikai és fizikai szint) kérdései természetesen csak azután vizsgálhatók, ha tudjuk: a számlán árunevek (fogalmi szint) vannak. A fogalmi séma teremti meg a belső és külső aspektus összhangját.

- 602 A helyes válasz: 10. A bemenetben a cikkszám - és nem a -név - szerepel. Átmenet nem képződik. A cikkszám alapján keressük ki a tárolt nevet, és azt megjelenítjük.
- 603 A közértes számlakészítés mindig tranzakció-orientált módú; más nem is lehet. Az általam vett dolgoknak semmi közük sincs mások áruhoz. Sőt, a 'só' árának és nevének a kikeresésével sem várjuk meg a 'cukorét'. Az esti mérlegkészítés viszont éppen hogy köteget; más nem is lehet.
- 604 Minden cégnek csak egy információs rendszere van. Ugyanis a számla nem független az árutól, az adó a számlától, az összes kiadásnak pedig az adó és a bérleti díj egyaránt része. A rendszerrész absztrakció, ami nem közvetlenül a látható, valós feladatok szerint épül fel. Ezért a 'fogalmunk sincs' a helyes válasz a rendszerrészek számát illetően. Lásd a következő feladat megoldását.
- 605 A funkció-egyed mátrix (6.14 ábra) a rendszerrészek objektív kijelölési eszköze. Azért az, mert rendszerrésznek a rendszer egymással viszonylag szorosabban kapcsolódó elemeinek az együttesét tekintjük. A kapcsolat szorossága pedig nem szubjektív megítélés tárgya. Mi köze van például két olyan feldolgozásnak egymáshoz, amelyek egyetlen közös ismeretet sem használnak?
- 606 A fekete-doboz a feladat és a megoldás ellentmondásának az elkerülésére szolgál. Még nem tudjuk, hogy mi a megoldás, de már ismerjük magát a feladatot és azt el is kell rendeznünk. A közértes példában feladat lesz a cikk és a partner nyilvántartása. Az előbbibe már beleváltunk, a másikba még nem. Ám tudjuk, hogy a két dolog majd összefügg. A fekete-doboz az absztrakció minden szintjén alkalmazandó elv.
- 607 Az árubeszerezés és a számlázás közös feltétele az árnyilvántartás. A 604-es kérdés 'valós' tevékenységként ezt nem is említette, de az informatikai feladat nyilvánvaló. Az árnyilvántartási funkciónak időben meg kell előznie a beszerzést és a számlázást. A funkciók időbeli elrendezését a funkció szintű precedenciaháló mutatja.
- 608 A navigáció az első esetben nem változik. A második esetben igen, mert a vevőt fogjuk kiindulási pontként választani. Nem fogjuk X ezer vevő rendeléstételeit átnézni akkor, ha a Q helységben csak pl. öt vevőnk van. A harmadik eset új helyzetet mutat, mert megváltozott az almodell. Nem is tudjuk, hogy a számla a modellben hol található. A navigáció változik, de a korábbi útvonal egyes részei talán nem módosulnak.
- 609 Egy célszerű eszközzel a navigációs utak átfedései elemezhetők lennének. Például kiderülhetne, hogy az X navigáció az Y-t tartalmazza vagy azzal nagymértékben átfed. Ergo: sokkal kevesebb programozási munkára lenne szükség, ha látnánk a közösen alkalmazható eljárás-részeket.
- 610 Tipikus problémák a következők. Ad 1) A feltételek nem teljesek. Nem mondtuk el, hogy mi van akkor, ha az áru külön adóval terhelt, de nincs kedvezmény. Ad 2) A feltételek ellentmondóak. Az 'Y típus esetén mindenképpen a B-t' ellentmond annak, hogy csak kedvezmény esetén. Ad 3) A feltételek köre inkonzisztens. Mindezek a problémák a döntési táblázat segítségével kiszűrhetők. Ha annak hiányzik valamelyik feltétel-oszlópa vagy -sora illetve a tevékenységek különböző feltételek esetében is azonosak, akkor gondolkodásunk hibás vagy az lehet.

- 611 A mélyszintű előzmény-háló (6.19 ábra) a származtatási műveleteknek a teljes elrendezését mutatja. Ha nem a legelső alapadatokat tároljuk a gépen illetve nem a legvégső származtatottakat jelezzük ki a végső kimeneteken, akkor a feldolgozottsági fok alacsony, mert a háló előlről és/vagy hátulról megszakad.
- 612 A bemenet-tevékenység-kimenet tábla (6.20 ábra) a feldolgozás logikai végrehajthatóságának az elemzésére alkalmas. Az ábra rossz, mert nem mutatja, hogy honnan vettük a Vevőnév adatot. A VEVŐ | Vevőkód sort a VEVŐ > Vevőnév sornak kellett volna követnie - és csak ebben illett volna a kimenetben megjelölni a Vevőnév ismeretet.

FOGALOMJEGYZÉK

Az alábbiakban felsoroljuk a könyvben található legfontosabb fogalmakat és azok feltalálási helyeit. A vastagon szedett oldalszám arra utal, hogy a definíció azon a lapon szerepel. Egyébként nem-definiált, de alapvetően fontos dologról van szó. Megjegyezzük, hogy aki e fogalomtár tételeinek a lényegét ismeri, az közel áll ahhoz, hogy jó informatikus váljék belőle.

‘Amiről szó van’ (UoD)	23	Értékhalmoz	137
Ábraorientált (tervezés)	113	Értelmezés	42
Ábrázolás	83	Érzékelés	39
Adat	37	Esemény	46, 48
Adatmodell	147	Eseményadat	167
Adatbank	152	Eszköz	51
Adatbázis	144	Eszközkörnyezet	31
Adatbázis-szervezés	160	Eszközrendszer	30
Adatbázis-tervezés	124	Észlelés	39
Adatbáziskezelés	150	Fejlesztési módszer	103
Adatelőállítási művelet	45	Fejlesztő	58
Adatfeldolgozás	44	Fejlődés	73
Adatfolyam diagram	182	Fekete doboz	124, 179
Adatkezelési gráf	180	Feldolgozási fok	44, 51
Adatkezelési művelet	45	Feldolgozási mód	165
Adatmodell	145	Feldolgozásrendszer	159
Adatmodellezés	161	Feldolgozásvetület	81
Adatrendszer	132	Felfogás	40
Adatvetület	80	Felhasználó	55
Alapkonceptió	105	Felhasználói dokumentáció	204
Alkalmazás	27	Fizikai adatfüggetlenség	82
Alkalmazási felhasználó	57	Fizikai szint	83
Alkalmazási környezet	31	Fogalmi séma	162
Állapot	47	Fogalmi szint	84
Állomány (fájl)	150	Formaorientált tervezés	113
Állománykezelés	150	Funkció	49
Állományszervezés	160	Funkció-egyed mátrix	184
Alrendszer	18	Funkciócsoport	49
Átmenet	164	Hardver	32
Azonosító	139	Idő	47, 52
Belső esemény	49	Információ	41
Belső környezet	16	Információs rendszer	13, 22, 35
Belső séma	162, 163	Informálás	38
Bemenet	164	Informatika	28
CASE	126	Informatikus	28
Cél	69	Integrált	36
Dokumentáció	88, 96	Ismeret-előállítás	170
Döntési táblázat	187	Kapcsolat	141
Egyed	135	Kapcsolattípus	141
Egyed-előfordulás	136	Kimenet	163
Egyed típus	136	Kivételek	72
Életciklus (egyed)	46, 167	Kivitelezési terv	93
Életciklus (rendszer)	90	Kommunikálás	38
Előfordulás-halmaz	136	Komplex	36
Erőforrás	51, 52	Konvenció	14, 60

Korlát	145	Rendszertervezés	94
Környezet	31	Részrendszer	19
Környezeti vetület	81	Séma	162
Környezetrendszer	194	Specifikáció	97
Kötegelt feldolgozás	165	SQL	173
Külső esemény	49	Struktúra	70
Külső környezet	16	Strukturált programozás	177
Külső séma	162, 163	Strukturált tervezés	79, 177
Logikai adatfüggetlenség	81	Szabvány	60, 64
Logikai szint	85	Szabvány és szabványos eljárás	61, 200
Megfogalmazási mód	111	Szabványrendszer	201
Mennyiség	42	Szervezés	98
Meta információs rendszer	111	Szervezeti egység	59
Metaadat	171	Szint	83, 159
Metamodell	108	Szoftver	27, 32
Mező	150	Szövegkezelő rendszer	152
Minőség	42	Szövegszerkesztés	151
Modell	99	Szövegszerkesztő	114
Módszer	103	Táblázatkezelő	152
Módszertan	103	Tartalom	70
Negyedik generációs nyelv	114, 127	Terminológia	110
Nézet (view)	148	Terv-termék	94
Nyomtatási-kép állomány	166	Tervdokumentáció	204
Objektum-orientált	95, 154	Tervezés	98
Optimum-kritérium	98	Tervezési algoritmus	121
Orgver	32	Tervezési kritérium	115
Pénz	52	Tervezési nyelv	112
Precedencia-háló	188	Tervezési nyomtatvány	112
Prioritási sorrend	69	Tervezési-folyamat	94
Programfejlesztés	114	Tevékenységi	44
Programvázlat	180	Törzsadat	167
Projekt	198	Tranzakció	165
Projekt-menedzsment	104	Tranzakció orientált feldolgozás	166
Projekt-terv	93	Tranzakciós állomány	166
Pszichológiai mozzanat	194	Tulajdonságérték	137
Rekord	149	Tulajdonságtípus	137
Rendszer	11, 15	Ügyvitelszervező	98
Rendszer határ	15	Üzemeltetési dokumentáció	205
Rendszer környezet	16	Valós rendszer	22
Rendszerdokumentáció	204	Változás	46
Rendszerelemzés	68, 94	Változáskezelés	90
Rendszerelemző	68	Változat	73
Rendszerrész	19, 50, 73, 183	Végső-felhasználó	57
Rendszerstruktúra	108	Vetület	80
Rendszerszervezés	14	Vezérlési művelet	46
Rendszerszervező	12, 97	Vezető	58
Rendszertagolás	18	Vezetői információs rendszer	36
Rendszerterv (design)	91	Visszakeresés	169
Rendszerterv (plan)	91		