tanulmányok 142/1983

MTA Számitástechnikai és Automatizálási Kutató Intézet Budapest

MAGYAR TUDOMÁNYOS AKADÉMIA
SZÁMITÁSTECHNIKAI ÉS AUTOMATIZÁLÁSI KUTATÓINTÉZETE

DIALÓGUSSAL VEZÉRELT INTERAKTÍV GÉPÉSZETI CAD RENDSZEREK ELMÉLETI ÉS GYAKORLATI MEGFOGALMAZÁSA

Irta:

PIKLER GYULA

Tanulmányok 142/1983

A kiadásért felelős:

DR VÁMOS TIBOR

Főosztályvezető:

DR SOMLÓ JÁNOS

ISBN 963 311 151 X

ISSN 0324-2951

MSZH Nyomda

TARTALOMJEGYZÉK

1. BEVEZETÉS

Napjaink egyik legégetőbb megoldandó problémája a termelés állandó növelése a gyártmányok minőségének javitása és a termelékenység fokozása, mindez a gyártmányféleségek változtatása mellett. Ezeknek a feladatoknak egyik megoldása a tervező és a termelőmunka automatizálásában rejlik. Szerte a világon sok helyen foglalkoznak az integrált anyag és adatfeldolgozó rendszerek kutatásával. Ez az irányvonal komoly segitséget nyujthat a tervezés és gyártás automatizálásának rugalmas összehangolásában.

A tervezőmunka és a gyártás automatizálása a számitástechnikával összefonódott. Az intelligens számitógépes perifériák /grafikus display-k/ megjelenése a gépészeti tervezőmunka automatizálásában fordulópontot jelentett, mivel az intellignes perifériák alkalmazása lehetővé teszi az ember /tervező/ és a számitógépes rendszer közvetlen kapcsolatának megteremtését.

Magyarországon 1971-ben készült el az első grafikus display /GD'71/ [1], amely megteremtette a feltételét az interaktiv tervező rendszerek létrehozásához szükséges kutatások meginditásának.

E kutatómunkák keretén belül 1972-ben elkészült a "Mini-számitógépes interaktiv alkatrészprogramiró rendszer" [2] NC szerszámgépek automatikus programozásához és 1978-ban az "Interaktiv sajtolószerszám tervező rendszer" /ISTER/ [3,4]. Az ISTER sikeres ipari bevezetése után nyilt lehetőségünk ar-

ra, hogy elméletileg megalapozzuk a rendszerben megvalósitott elveket, kidolgozzuk a célorientált interaktiv gépészeti tervező rendszerek generálásához szükséges általánosan használható elvet és eljárásokat.

A tanulmány a fenti témákban folytatott sok éves kutatómunkánk összefoglalása.
A kutatómunka eredményeit két területre korlátoztuk. Az egyik az ISTER létrehozásában realizálódott, a másik az ISTER-ben megvalósitott rendszertechnikai elvek és megoldások általánositásában, továbbfejlesztésében jelentkeztek.
A tanulmány terjedelme nem engedi meg, hogy az ISTER-t részletesen ismertessük, igy csak röviden összefoglaljuk. [146, 161, 162]
A dolgozatban a kutatásaink második részét ismertetjük. Ennek célja, módszerek és software eszközök kidolgozása, amelyek a gépészeti tervezés automatizálását segitik, azaz célorientált /adott tervezési területre/ interaktiv tervező rendszerek létrehozásához nyujtanak segitséget.

Ferenczy Jenő kezdeti eredményeire [5] támaszkodva az ISTER kutató, fejlesztő munkáit 1974-ben kezdtük el az MTA SZTAKI-ban.

A rendszer tervezési munkái közösen folytak az Egyesült Izzó és Villamossági Gyár RT /EIVRT/-vel és azt az EIVRT-n kivül az OMFB is finanszirozta. A cél az volt, hogy fejlett számitástechnika alkalmazásával, olyan tervező rendszert hozzunk létre, amely a sajtolószerszámok tervezéséhez és gyártáselőkészitéséhez nyujt segitséget. Általánosan meg-

fogalmazva, egy viszonylag szük területen CAD/CAM /Computer aided design/Computer aided manufacturing/ rendszert kellett létrehoznunk. Gépészeti szempontból a feladatot három részre tagolva valósitottuk meg. Ezek a következők:

1. A sajtolandó alkatrészek geometriai és egyéb adataiból kiindulva a szerszám megtervezéséhez szükséges alaptechnológiai müveletek /kivágás, huzás, hajlitás stb./ meghatározása, azokra jellemző paraméterek előállitása, valamint az alaptechnológiai müveletek összerendelése. Ebben a részben történik az alkatrészek legyártásához rendelhető szabványos felépitésü szerszámok és a sajtológépek kiválasztása, valamint a sávtervezés.

2. A sajtolandó alkatrészek legyártásához a szabványos felépitésü szerszámok megtervezése a sávterv ismeretében, és a sajtoló technológiák paramétereinek segitségével. Ha az alkatrészek nem gyárthatók le szabványos felépitésü szerszámokkal /pl. sorozatszerszám/, akkor a rendszer segitségével, annak szolgáltatásaival az egyedi tervezések is elvégezhetők.

3. A megtervezett szerszámok alkatrészeinek legyártásához gyártástechnológiai tervezés. A tervezés kiterjed az NC szikraforgácsológép vezérléséhez vezérlő lyukszalag készitésére, optikai köszörüléshez pantográfrajzok előállitására és forgácsolási müveletek meghatározására a forgácsoló szerszámgépek kiválasztásával együtt.

Rendszertechnikai szempontból, dialógus rendszerrel vezérelt tervező rendszert alakitottunk ki. Ennek

lényege az, hogy az ember és a gép közötti párbeszédet végrehajtó dialógus rendszer [146] gondoskodik arról is, hogy a program futása arra a programszegmensre kerüljön, amelyhez a tervezőtől bekért adatok tartoznak. Ezt ugy realizáltuk, hogy a programszegmenseket egymáshoz képest párhuzamosan füztük fel. A párhuzamosan felfüzött programszegmenseket felülről egy vezérlő program fogja össze, alulról pedig a dialógus rendszerhez csatlakoznak. A programszegmensek funkcionális szempontból ugy csoportosithatók, hogy három alrendszert alkotnak, mégpedig geometriai, technológiai és dokumentációs alrendszert. A kétdimenziós geometriai alrendszer gondoskodik arról, hogy a tervező a tervezéshez szükséges geometriai adatokat előállitsa és a rajzokat elkészitse a display felhasználásával [162]. A technológiai alrendszer azokat a programszegmenseket tartalmazza, amelyek a szerszám megtervezéséhez szükségesek [161].
A dokumentációs alrendszer gondoskodik arról, hogy a tervezés dokumentációi elkészülhessenek /összeállitási, mühely, pantográf rajzok; vezérlő lyukszalagok; számitások eredményeit tartalmazó listák; darabjegyzékek stb./.

Az ISTER-t felhasználás szempontjából ugy alakitottuk ki, hogy az a tervezőmunkához különböző szolgáltatásokat biztositson és ezeket a szolgáltatásokat a tervező aktivizálja. Ezzel elértük azt, hogy a tervezéshez szükséges kreativ tevékenységet a konstruktőr meg tudja valósitani, szemben a kötött tervezési sorrendet megkövetelő rendszerekkel.

Kutatásaink második részét, az ISTER-ben megvalósitott elvek és módszerek igazolását, valamint azok továbbfejlesztését tárgyaljuk. A 2. fejezet a gépészeti konstrukciós tervezés történeti áttekintését foglalja össze, bemutatva irodalmi hivatkozások alapján a fejlődési irányzatokat. A 3. fejezet az interaktiv számitógépes tervező rendszerek fejlődését mutatja be és azoknak az általános

problémáit tárgyalja. A már müködő interaktiv rendszereknek e fejezetben történő csoportositása megadja a lehetőséget arra, hogy tisztázzuk a dialógus rendszerek helyét és szerepét az interaktiv rendszerekben. Itt foglaljuk össze a dialógus rendszerek kutatásában eddig elért eredményeket is. A 4. fejezet az ember és a számitógépes rendszer kapcsolatával foglalkozó elméleti kutatásokat tekinti át. Az 5. fejezet az előző fejezetben összefoglalt kutatások eredményeinek a felhasználásával, valamint az ISTER-ben megvalósitott rendszertechnikai elvek általánositásával kidolgozott elméleti modellt ir le. A modell problémaorientált interaktiv tervező rendszerek létrehozásában és azok alkalmazásában lejátszódó folyamatokat modellezi. Az elméleti modell segitségével tisztázott rendszerfogalmak és kapcsolatok teremtették meg a feltételét annak, hogy a problémaorientált /célorientált/ interaktiv gépészeti tervező rendszerek létrehozásához módszereket, automatikus eljárásokat dolgozzunk ki. A 6. fejezet konkrét software megoldásokat és uj módszereket ir le célorientált tervező rendszerek létrehozásának megkönnyitéséhez. Itt definiáljuk a tervezési folyamatok megtervezésére szolgáló dialógus gráfot, a tervezési folyamat és az abban lévő dialógus leirására alkalmas DL /Dialogue Language/ nyelvet. Foglalkozunk a célorientált rendszer vezérlő programját előállitó dialógus generátorral és az ember-számitógép kapcsolatot megvalósitó dialógus processzorral. A 7. fejezetben foglaljuk össze, hogy a tanulmányban javasolt eljárások, módszerek és programszegmensek felhasználásával hogyan lehet létrehozni interaktiv célorientált tervező rendszereket.

Ezt a helyet szeretném felhasználni arra,hogy köszönetet mondjak dr. Somló Jánosnak, aki szakmai segitségen kivül lehetővé tette számomra, hogy ezt a tanulmányt elkészitsem.

Köszönetemet fejezem ki dr. Hatvany Józsefnek az ISTER munkáinál felmerült, különböző jellegü nehézségek áthidalásában nyujtott segitségért. Itt köszönöm meg Turai István, Holló Krisztina, Déry Gábor, Hargitai Zsuzsanna közvetlen munkatársaimnak azt a lelkes és odaadó munkájukat, amelyet az ISTER implementálásában nyujtottak. Nem utolsó sorban szeretném megköszönni dr. Márkusz Zsuzsannának és Szőts Miklósnak az elméleti kérdések tisztázásában nyujtott segitségüket. Köszönettel tartozom Krammer Gergelynek az ISTER létrehozásának idején velem folytatott segitőkész szakmai konzultációkért.

2. A PROBLÉMÁK FELVETÉSE. TÖRTÉNETI ÉS IRODALMI ÁTTEKINTÉS

A gépészeti tervezés automatizálásának kérdéseit az elmult évtizedek /30-40 év/ rohamos technikai fejlődése vetette fel. A fejlődést többek között a számitástechnika megjelenése és alkalmazása tette lehetővé. A tervezési módszerek fejlődése ma már nem választható el a számitástechnika fejlődésétől, hiszen e két terület egymás kölcsönhatásával fejlődik.

A változások tükrében összefoglaljuk az automatizálás legfontosabb kérdéseit és irányzatait. A kutatási területeket két oldalról közelitjük meg. Az egyik a gépészeti tervezés, a másik a számitástechnika fejlődés irányai.

2.1. A gépészeti tervezés fejlődésének irányai

A 2. fejezetben emlitett, a fejlődés folyamán megnövekedett követelményeket a hagyományos tervezői tevékenységekkel már nem lehetett kielégiteni. Uj utakat és módszereket kellett keresni. Az intenziven megindult kutatások egyik legalapvetőbb kérdése az volt, hogy mit értsünk tervezésen. Erre a problémára nagyon sok, különböző területeken dolgozó kutató próbált választ adni [6, 7, 8, 9, 10, 11]. A válaszokból kitünik, hogy a probléma, annak megközelitésétől függően, más és más módon jelentkezik. Mi a továbbiakban a tervezésen azt a munkafolyamatot értjük, amelynek eredményeképpen uj gyártmányok, berendezések müszaki tervei jönnek létre.

Mielőtt a tervezési módszerek kutatásának eredményeivel és a jelen problémáival foglalkoznánk, röviden áttekintjük a gépészeti tervezés multját és fejlődését.

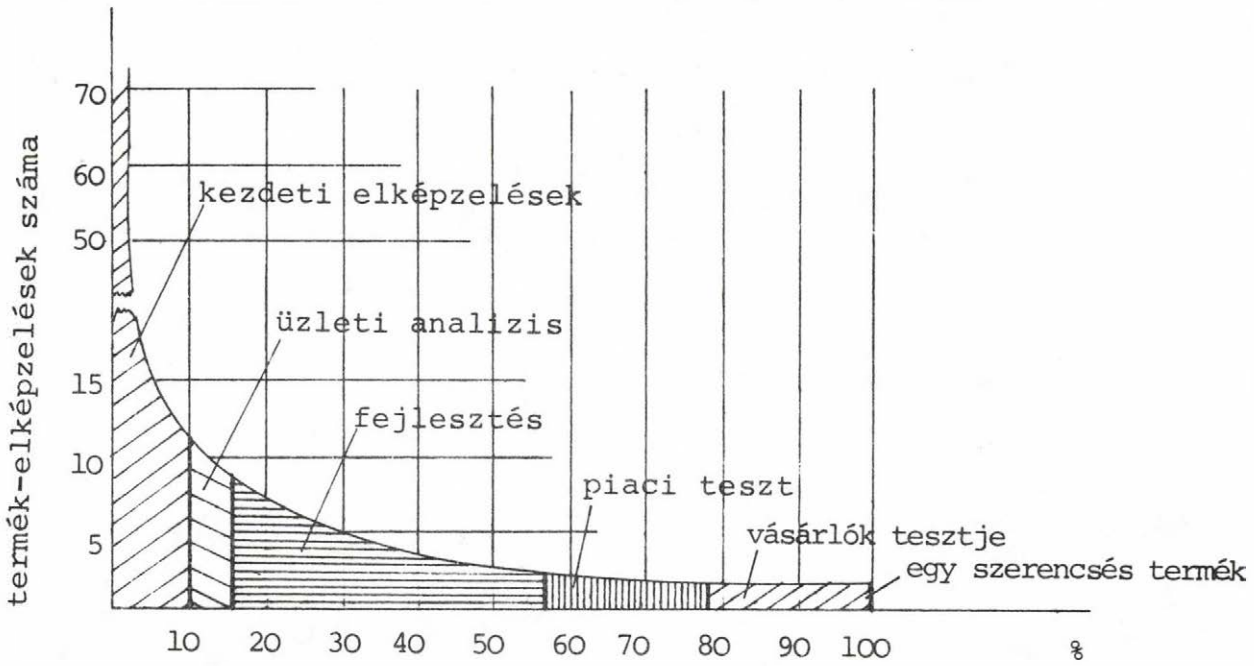
Jones [12] a századforduló előtti korszakot "Craft Evolution"-nak nevezte, amelyre az volt a jellemző, hogy egy berendezés tervezésének és készitésének folyamatai párhuzamosan folytak és mindkét tevékenységet egy személyben a mester végezte. A valóban müvészi alkotásokról külön tervek általában nem is készültek.

A századforduló körül indult el a modern iparosodás. Ez magával hozta a tervezésnek a berendezések készitésétől való különválását. Ekkor kezdődött el a rajzolással történő tervezés, amely forradalmasitotta a tervezés folyamatát és uj lendületet adott az ipar fejlődésének. A leglényegesebb változások a következők voltak:

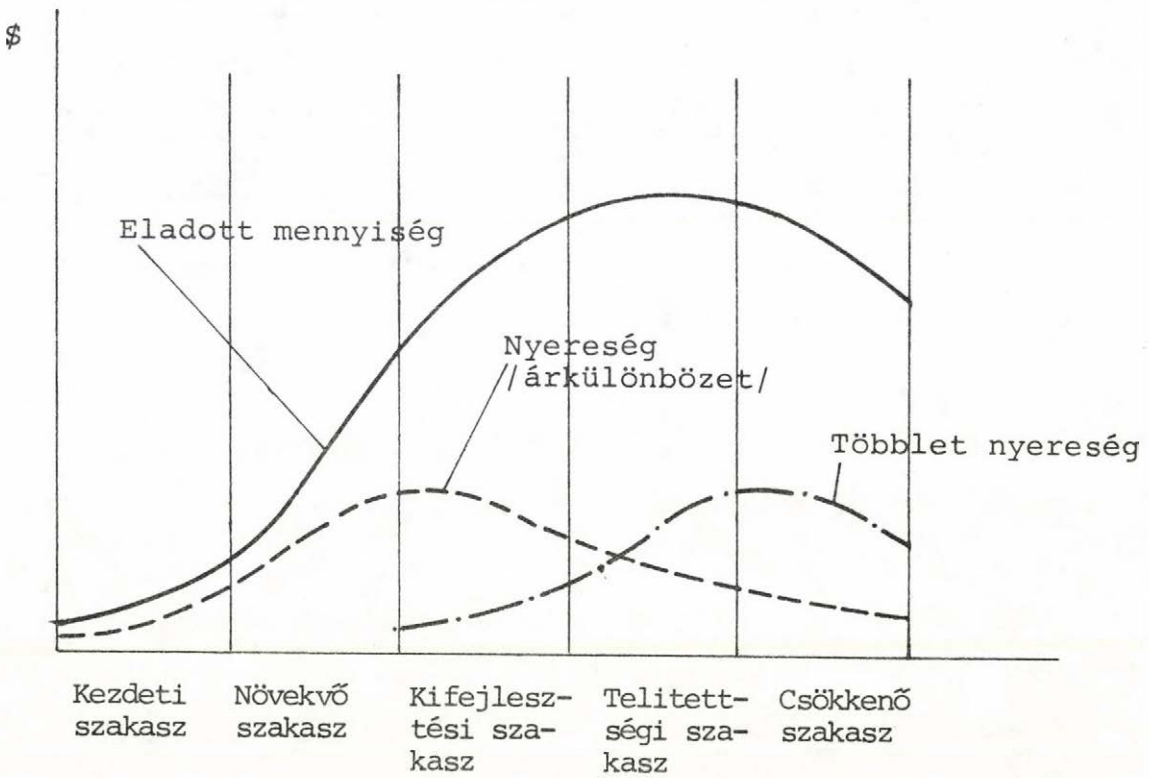
1. az uj tervezési módszer lehetővé tette, hogy a tervezés szétváljon a termeléstől;
2. a rajzokon levő méretek és türések lehetővé tették, hogy egy gépet alkatrészeire bontva más és más személyek készitsenek el, igy a tervezői munkát fel lehetett osztani;
3. a rajzolással történő tervezés lehetőséget adott arra, hogy a tervező egy feladat megoldására több változatot hozzon létre /próbálgasson/, és a legjobb megoldást vigye a termelésbe.

E fejlődési szakasz elején a tervező függetlenné vált. A tervezésen kivül irányitotta az általa tervezett berendezés/ek/ gyártását is.

A fejlődés későbbi szakaszában, mind a gyárakon belüli, mind az azokon kivüli követelmények egyre nehezebb feladatok megoldása elé állitották a tervezőket. Ahhoz, hogy a nagy sorozatban gyártott termékek gazdaságosan termelhetők és jól eladhatók legyenek, a termékek kiválasztását és megtervezését különböző szinteken kellett megvizsgálni. Hill [13] szerint az ellenőrzési területek a következők:



1. ábra. A termékre forditott idő



2. ábra

- az uj termékekre vonatkozó elképzelések elemzése /megfelelők kiválasztása/;
- üzleti analizis;
- a kiválasztott termék kifejlesztésének analizise;
- a nullszéria elkészitéséből adódó tapasztalatok elemzése;
- piaci lehetőségek megvizsgálása;
- a vásárlók tesztje, a várható fogadtatás.

Az 1. ábra mutatja, hogy a kidolgozott termékek változatainak száma hogyan csökken az egyes tesztek alatt, a termék előállitására forditott teljes idő függvényében. A "BOOZ, ALLEN and HAMILTON INC."-től átvett ábrák azt is mutatják, hogy a termék előállitására forditott összidőből hány százalékot tesz ki a tervezési és gyártási idő. Egy uj termék vagy gyártmány kiválasztásánál figyelembe kell venni az élettartamára jellemző görbéket is, amelyet a 2. ábra mutat [13]. A két görbéből /1. és 2. ábra/ egyértelmüen leolvashatók a következők:

- a tervezők fantáziája korlátok közé szorult;
- az elképzeléseket sok szempontból tesztelték, és ez korrekciókhoz vezetett;
- a maximális nyereség elérésére való törekvés mint kényszer lép fel;
- a gyártmány "életgörbéjéből" adódó időtényezők mint befolyásoló körülmények lépnek fel.

Az utolsó következtetés figyelmeztet arra, hogy ujabb gyártmányokat kell kifejleszteni, tervezni, mielőtt a meglevők elavulnak.

Az 1960-as években a gépészeti tervezési módszerek fejlődésének a széles körben elterjedt számitástechnika adott uj lendületet. Kezdetben a számitógépeket olyan eszközként használták, amelyekkel bonyolult tervezési számitásokat lehetett elvégezni rövid idő alatt. Ez a periódus

a számitási eljárásokat, módszereket fejlesztette elsősorban, például az optimizálást. A bő irodalomból csak néhányra hivatkozunk [14, 15, 16, 17, 18, 19].

Az 1960-as évek végén és a 70-es évek elején a számitógépes tervező rendszerek kezdtek szélesebb körben elterjedni, amelyek a tervezési folyamatban a különböző részfeladatokat megoldó programokat kapcsolták össze. A fejlődésnek ezt a szakaszát külön fejezetben taglaljuk /2.3. fejezet/.

2.2. A számitástechnika és a számitógéppel segitett tervezés fejlődése

A számitógépek alkalmazásában a kezdeti eredmények a hadiiparban születtek. Az eredmények közül néhány. Az 1950-es évek végére dolgozták ki a MIT-ben /Massachusetts Institute of Technology/ az APT /Automatically Programmed Tools/ rendszert, az NC szerszámgépek programozásához [20]. 1959-től kezdtek dolgozni a DAC-1 nevü tervező rendszeren [21, 22] a General Motors-nál. 1961-ben fejezték be a Sketchpad első változatának kidolgozását az MIT Lincoln Laboratóriumában. Lockheed 1964-ben kezdte meg saját tervező rendszerének kidolgozását [23]. A kutatások két irányban folytak: az első a repülőgépek tervezéséhez szükséges programok kidolgozásához, másik a megtervezett alkatrészek legyártásához szükséges NC szerszámgépeket vezérlő alkatrészprogramok automatikus előállitásához kapcsolódik [24].

1959-ben a General Motors és Ford kutatóintézetei értek el kimagasló eredményeket. Itt jöttek rá arra, hogy olyan számitógépes tervező programokat kell készitени, amelyek az aritmetikai, logikai és adatfeldolgozási rutinmunkákat végzik el, az intuitiv tevékenységeket pedig meghagyják az embereknek [25, 26]. Ezt a gondolatot

fejtette ki Neumann János 1965-ben megjelent könyvében is [27].

A számitógépek széles körü elterjedése az ember-számitógép közötti rugalmas kommunikáció megoldásának problémáját vetette fel. E területen jelentős eredményeket ért el a Ross,D.T., Coons,S.A. és Mann,B.W. professzorokból álló csoport az AED /Algol Extended for Design/ koncepcióval [28, 29]. A számitógépes grafika megjelenése ujabb követelményeket vetett fel [30, 31]. Ez inditotta el az ember és a számitógép közötti interaktiv grafikus kommunikáció kutatását, amely kapcsolat addig még csak nyelvi szinten valósult meg. A MIT-ben már 1951-ben használtak sugárcsöves képernyőt légi elháritási kutatásban. Interaktiv jellegü számitásokra és programozási hibák felderitésére az illinoisi egyetemen alkalmaztak először display-t [32, 33]. Fényceruzát 1955-ben használtak először a SAGE légitaktikai program keretében. Ettől az időtől kezdve a "man-machine communication" kutatásai és alkalmazásai rohamos léptekkel haladtak előre [34, 35, 36, 37].

A fenti példákból is kitünik, hogy az 1950-es években elindult fejlődés főleg az Egyesült Államokban indult el. Az 1960-as években döbbent rá Európa a nagy lemaradásra. Egy 1966-ban megjelent NEL /National Engineering Laboratory/ tanulmány ezt irja: "Az angol gépipar számára olyan jelentősége van a számitógépes tervezés bevezetésének, hogy ezt a következő évekre egyenesen az ipari potenciál egyik központi kérdésévé kell tenni" [38]. E program keretében az Angol Technológiai Minisztérium támogatásával CAD /Computer Aided Design/ központot hoztak létre a cambridge-i egyetemen [39], ahol főleg a gépészeti tervezés számitógépes módszereinek kutatásával kezdtek foglalkozni [40]. Olaszországban a Ferranti cég a rajzoló

berendezéséhez egy CAD grafikus nyelvet fejlesztett ki [41]. A Német Szövetségi Köztársaságban Opitz professzor vezetésével az Aacheni Főiskolán indult el jelentős kutató munka [42, 43]. A Belorusz Tudományos Akadémia Müszaki Kibernetikai Intézetben /ITK/ folyó gépészeti tervezés automatizálásának intenziv kutatásairól Goranszkij, G. K. professzor tartott értékes beszámolót 1968-ban Budapesten [44].

E nemzetközi versengésbe Magyarország 1966-ban kapcsolódott be. Dr. Hatvany József irányitásával [45] az MTA SZTAKI-ban. 1967-ben indult el egy képcsöves interaktiv megjelenitő eszköz kutatása [1]. Az NC szerszámgépek programozó rendszereinek /EXAPT, 2CL/ honositási munkái ugyancsak 1967-ben kezdődtek meg az MTA SZTAKI-ban [46, 47]. A GTI-ben /Gépipari Technológiai Intézet/ dr. Horváth Mátyás vezetésével a FORTAP rendszert [48] dolgozták ki. 1969-ben az MTA SZTAKI-ban értékes tanulmány született, amely nemcsak a gépészeti számitógépes tervezésben addig elért eredményeket rendezte és foglalta össze, hanem a jövő perspektiváit és a későbbi hazai munkák koncepcióját határozta meg [49].

A fenti koránt sem teljes áttekintésből is kitünik, hogy az 1950-1960-as években széles körü kutatómunka folyt nemcsak a számitástechnika, hanem annak gépészeti alkalmazása területén is [50]. Eltekintve néhány óriási cégtől, amelyek már komoly számitógépes rendszerekkel segitették a gépészeti tervezés nagy részét [51, 52], az uj eredmények még nem tudtak széles körben elterjedni. Ennek okait a következőkben foglaljuk össze:

1. A kifejlesztett rendszerekhez szükséges hardware magas ára. A kutatásokra forditott összeg 80%-a hardware ár, 20% software ár [53]. /1960-as statisztikai adat, ma mások az arányok/.

2. Az alapsoftware viszonylagos fejletlensége, különböző nyelvek /grafikus, célorientált/ létrehozásához software eszközök nem álltak rendelkezésre.
3. A gépészeti tervezéshez szükséges programok és rendszerek nagy és költséges számitógépeket igényeltek.
4. A gépészeti tervezéshez szükséges számitógépes algoritmusok egyáltalán nem, vagy csak részben voltak kidolgozva.
5. A software kutatások magas költségei.

A fejlődés üteme az 1970-es évek elején meggyorsult, ezzel párhuzamosan a gépészeti alkalmazások is ujabb követelményeket állitottak a számitástechnika elé. Bizonyos tervezési részfeladatok esetenkénti számitógépes megoldása csak részben segitette a tervezői munkát. Hamar nyilvánvalóvá vált, hogy a számitógépi lehetőségek valóban jó kihasználását csak az integrált tervező rendszerek szavatolják. Az integrált tervező rendszerek létrehozása ugyanakkor ujabb számitástechnikai és gépészeti tervezési problémákat vetett fel. A számitástechnikai problémák közül néhányat emlitünk.

1. Az ember-számitógép közötti kapcsolat nyelvi igényeit a hagyományos és univerzális /FORTRAN, ALGOL/ nyelvek csak egy bizonyos szintig tudták kielégiteni. Nagy méretü és bonyolult felépitésü komplex programoknál a kezelendő adatok nagy száma és bonyolult strukturáltsága már megkivánta a célorientált programnyelvek létrehozását. Ennek megoldására születtek meg az olyan programgenerátorok, amelyek célorientált nyelvek forditóprogramjait képesek automatikusan előállitani [54, 55].

2. A grafika alkalmazásának problémái két szinten jelentkeztek. A programból előállitható grafikus információk kezelését soroljuk az első szinthez. A megoldások közül a legjelentősebb a grafikus szolgáltatásokkal kibővitett algoritmikus nyelvek létrehozása, és a külön egységeket alkotó grafikus rendszerek [56] kidolgozása volt. A második szinthez soroljuk az ember-számitógép közvetlen alfanumerikus és grafikus kommunikációjának a megvalósitását. Ezzel a 3.3. fejezetben foglalkozunk részletesen.
3. Az integrált tervező rendszerek megkivánják, hogy egységeik a feladattól,és ne gépi reprezentációtól függően legyenek megfogalmazva. A feladat orientáltsága a strukturált programozás elveinek alkalmazását követelte meg. Ez tette szükségessé a strukturált programozásra alkalmas nyelvek kidolgozását, és a hagyományos nyelveket strukturáltá tevő processzorok létrehozását [57, 58, 59].
4. Az integrált tervező rendszerekben használt adatok egy része adatbázis jellegü /a rendszer futása közben változatlan/, másik része a rendszer futása közben keletkezik, változik vagy törlődik. Az ilyen tipusu adatoknak a program futásának megszünésével vagy megszakadásával nem szabad elveszniük, azaz archiválódniuk kell. Az utóbbi megoldására dinamikus adatkezelő rendszereket dolgoztak ki. Ezek közül a széles körben elterjedt "Codasyl" tipusu adatbázis-kezelő rendszert emeljük ki [60, 61].
5. Az integrált tervező rendszerekkel kapcsolatban levő szakemberek között specializálódás alakult ki,amely 3 szakterületet hozott létre. Ezek a következők:
   - tervezők, akik az integrált rendszert mint eszközt használják;
   - rendszertervezők /system engineer/, akik az integrált rendszereket készitik a tervezőknek, felhasználva a számitástechnika által nyujtott alapsoftware-t;

- alapsoftware szakterület, amely az általános célu programozástechnikai eszközöket állitja elő. Ilyenek a 2., 3. és 4. pontokban emlitett programgenerátorok, grafikus alapsoftware-k, adatbázis-kezelő rendszerek, általános célu integrált rendszerek, monitor rendszerek stb. [62,63, 64, 65, 66].

A megvalósitott integrált tervező rendszerek közül csak néhányat emlitünk. Az USA haditengerészete a COMRADE [67] integrált rendszert dolgozta ki hajótervezésre. Az IPAD a légi járművek tervezésére kifejlesztett rendszer /USA/ [69]. Az aacheni müszaki főiskolán több évig dolgoztak a tengelyek, valamint a forgástestekből álló gépészeti berendezések tervezésére alkalmas rendszereken [69, 70, 71]. Magyarországon az első megvalósitott integrált tervező rendszer az ISTER volt [72, 73, 74].

2.3. A gépészeti tervezés kutatási eredményei és problémái

A gépészeti tervezés területén folyó kutatások a következő két irányban indultak el.

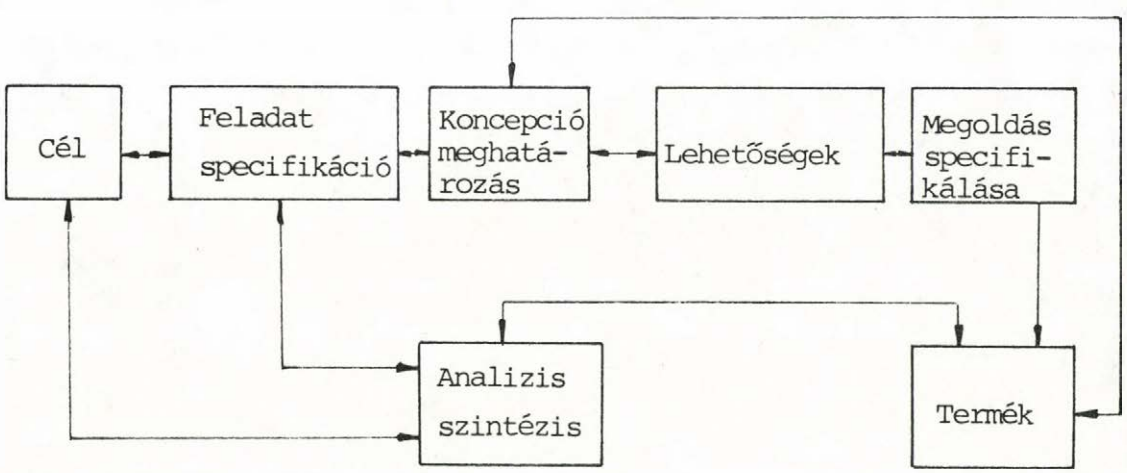
1. A tervezés tárgyának a kutatása. Ezek kiterjednek:
   - a tervezési folyamatok megismerésére;
   - a folyamatok modelljeinek megalkotására;
   - a hagyományos számitási eljárások algoritmizálására;
   - az uj számitási algoritmusok kidolgozására;
   - algoritmusok közti kapcsolatok feltárására.
2. A tervezési módszerek kutatása. A számitógépek alkalmazása uj tervezési módszerek kialakulására ad lehetőséget. A kialakitandó uj módszerek, azok hatékonysága sok összetevő függvénye [75] /a számitógépek optimális kihasználása, feladat-orientáció, ergonómiai kérdések, nyelvi problémák stb./.

A két csoportba tartozó kutatási témák mereven nem választhatók el, hiszen sok esetben függenek egymástól.

2.3.1. A tervezési módszerek kutatásának alapvető eredményei

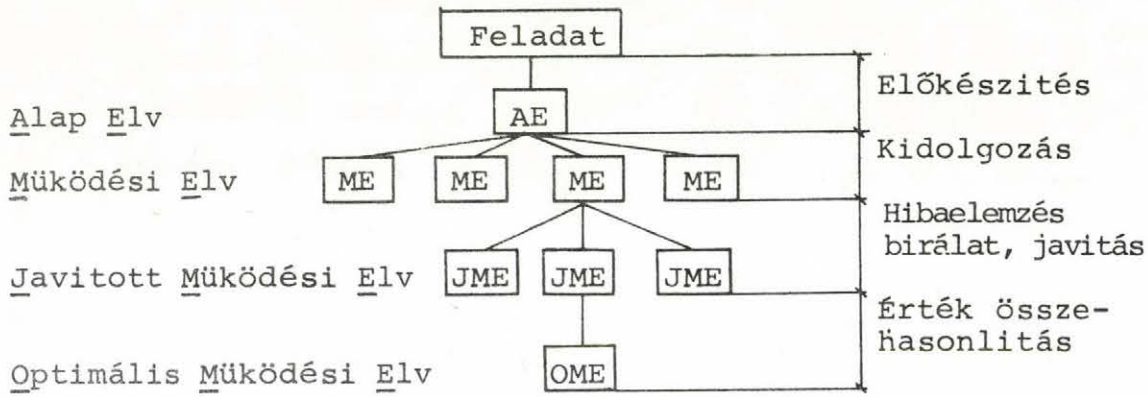
Ma már triviálisnak tünik, de korábbi kutatások egyik legfontosabb eredménye az volt, hogy a konstrukciós tevékenység felfogható folyamat gyanánt. Black már az 1955-ben megjelent könyvében [76] a mérnöki tervezési tevékenységet folyamatábrában mutatta be /3. ábra/. Az ábra jelentősége nemcsak az, hogy az egyes tevékenységeket szétválasztja, hanem többirányu és iterációs kapcsolatokat is létesit közöttük.

A számitástechnika megkövetelte a gépészeti tervezésben használt számitási eljárások algoritmizálását. Termékenyitőleg hatott az uj számitástechnikai módszerek alkalmazására a gráfelmélet eredményeinek használata. A modern mérnöki tervezési tevékenységben a gráfelmélet széles körü alkalmazását Hendley, Williams [77], W. R. Spillers [78] könyvei és a MINSK-ben /ITK/ folyó munkák [79] jól szemléltetik. A gráfok leirására kidolgoztak néhány szimbolikus nyelvet, mint például a LISP [80], SLIP [81], COMIT [82, 83].



3. ábra

A tervezés automatizálásának egyik követelménye, hogy felfedjük: milyen tevékenységek tartoznak a tervezési folyamatba. Hansen [84] a müszaki alkotás kidolgozásához tartozó tevékenységeket egy fastrukturával ábrázolja /4. ábra/.



4. ábra

A gépészeti tervezéssel foglalkozó irodalom nagy teret szentel a kreativitásnak [85, 84, 13, 12, 86]. Hansen [84] azt mondja, hogy "a szerkesztés logikus gondolkodási folyamat eredménye", de hozzáteszi, hogy "nem müvészeti eredmény". Egy másik helyen azt irja: "a tervezésre nincs eljárási utasitás, a tervező /kutató/ belső modellt épit fel magában arról, amit realizálni akar". Hill [13] ugy definiálja a kreativitást, hogy "egy szerencsés lépés, amellyel a tudásunk vagy ismereteink határát lehet átlépni. Kreativitás nélkül nem lenne fejlődés."

Jones [12] szerint "a tervező a jelen ismeretével a jövőt formálja meg a kreativ tevékenységével". A tudomány előrehaladásával egyre több ma még kreativnak minősitett tevékenység algoritmizálhatóvá válhat a jövőben [86].

A tervezés kutatásának eredményei közé kell sorolni az uj tervezési módszerek létrejöttét, amelyben nagy szerepe volt annak a felismerésnek, hogy a tervezési folyamatban a tervező sokszor csak a bemenetekkel és kimenetekkel meghatározott ugynevezett black boxokat is használ. Az emberi gondolkodásban és cselekedetben is számos ilyen elem, jelenség van. Newman [87] a jelenséget egy elektromos hálózathoz hasonlitja, amelyet anélkül használunk, hogy a belső müködését ismernénk. Ez azt jelenti, hogy használhatunk rendszereket anélkül, hogy belső szerkezetüket ismernénk. A fenti felismerés tette lehetővé, hogy a tervezés automatizálását több szinten valósitsuk meg [88]. Egy automatizált tervezőrendszer létrehozásában a következő szinteket különböztethetjük meg:

- Felhasználói szint. Ezen a szinten a tervező a rendszer felhasználásával, illetve szolgáltatásaival különböző tervezési feladatokat old meg. A tervező a rendszerről csak annyit tud, hogy milyen szolgáltatásokat biztosit, azokat hogyan aktivizálhatja, de nem kell ismernie a különböző szolgáltatások megvalósitásának módjait /black boxoknak tekinti/.
- Rendszertervezési szint. Itt figyelembe kell venni a tervezői szint követelményeit és támaszkodni kell néhány alacsonyabb szint munkáira. A rendszertervezés ma már egy kifejlődött önálló tudományágnak tekinthető. Számos irodalom foglalkozik a rendszertervezés problémáival, irányzataival, követelményeivel stb. [89, 90, 91, 92, 93].

- Alap-tervezési egységeket létrehozó szint.E szinten történik az algoritmizálható tervezési blokkok kidolgozása, amelyek épitőkövei lesznek a tervezői rendszernek. Például véges elem módszerek [94], forgácsoló szerszámok geometriájának leirása [95. 96], mechanizmusok pályáit kiszámitó algoritmusok[97]stb.

Megjegyezzük, hogy az egyes szintek nem különithetők el teljesen egymástól, azaz egymásra hatnak.

2.3.2. A tervezés modellezésének néhány problémája

Egy számitógép müködtetésében két üzemmódot különböztetünk meg, a "batch" és az "interaktiv" üzemmódot. A batch üzemmód megvalósitása egyszerübb [96, 98], a bemenő adatokra, mint egy függvény, meghatározott kimenő adatokat állit elő. Bonyolultabb programok vagy programrendszerek [99, 100, 101, 102] esetén ez az üzemmód nehézkes és sokszor nem alkalmas a kitüzött feladatok megoldására [103, 104]. A továbbiakban az interaktiv üzemmód tulajdonságait mutatjuk be kiemelve annak előnyeit és hátrányait [105, 106].

Előnyök:

1 közvetlen /on-line/ kapcsolat alakul ki az ember és a számitógép között;
2. lehetővé válik, hogy az ember beavatkozzék a programrendszer müködésébe, ellenőrizheti és közvetlenül irányithatja a tervezési folyamatot;
3. olyan tervező rendszerek is létrehozhatók, amelyekben az ember intuitiv döntéseire és kreativ tevékenységére van szükség;
4. a programrendszer számára szükséges input adatokat a tervező dialógus formában is megadhatja, tehát nincs szükség a bemenő adatok megadására szolgáló célorientált nyelvet létrehozni. Batch üzemmódban - mindenek-

előtt a felhasználói oldal támogatása miatt - ez általában alapkövetelmény;

5. grafikus display alkalmazásával grafikus kommunikáció is létrejöhet az ember és a számitógép között. Ez müszaki tervezési feladatok megoldásánál sokszor nagyságrendekkel megkönnyiti a munkát.

Hátrányok:

1. Grafikát is alkalmazó interaktiv üzemmódot a költségek miatt csak nagy teljesitőképességü programrendszerek futtatására érdemes létrehozni;
2. interaktiv /grafikus/ rendszerek elkészitéséhez, mint általában az összetett rendszerek esetében, rendszertervezői ismeretekkel biró speciális szakember szükséges;
3. interaktiv tervező rendszerek hatékony müködéséhez fejlett interaktiv periféria szükséges: grafikus display /grafikus display-k sorozatgyártása a szocialista országokban egyelőre nincs megoldva/;
4. különböző, még meg nem oldott rendszertechnikai problémák jelentkeznek /modellezés, általánosan megfogalmazott ember-gép kapcsolat stb./.

Az interaktiv, számitógéppel segitett tervező rendszerek egyik sarkalatos problémája az ember és a számitógépes rendszer közti munkamegosztás. Ennek a kérdésnek a tisztázására áttekintjük, hogy milyen követelményeket kell kielégiteni egy jól használható interaktiv rendszernek.

1. Rugalmas ember-gép kapcsolat kialakitása [107, 108]. Ezen azt értjük, hogy a számitógépes rendszer által feltett kérdések olyanok legyenek, amelyeket a tervező azonnal megért. A tervezői beavatkozás az emberi gondolkodásmódhoz közelálló, kevés manualitást igénylő legyen. Az ember is tehessen fel kérdéseket a számitógépes rendszernek, amelyekre válaszokat is kap.

2. Az ember-számitógép közös tevékenységben a tervező a kreativ képességeit ki tudja fejteni [85]. E követelmény teljesitésének egyik megoldása lehet az, hogy a rendszert ugy alakitjuk ki, hogy az a tervezői munkához különböző szolgáltatásokat nyujtson. Ezeknek az aktivizálásával segiti a rendszer a tervezői kreativitás megvalósitását.
3. Egy interaktiv rendszer használhatóságát nagyon befolyásolja a munka közben végbemenő dialógus gyorsasága. A párbeszéd sebessége mindig két részből tevődik ösz-sze, mégpedig az ember és a számitógépes rendszer /gép/ válaszolási idejéből. A rendszer használhatóságát főleg a gép válaszolási ideje /response time/ határozza meg [109]. Nagy válaszolási idő türelmetlenné, idegessé teszi a tervezőt. A válaszolási időről mint a tervezést befolyásoló tényezőről sok irodalmi tanulmány jelent meg [110, 111, 112, 113].

Az interaktiv rendszereket felhasználó tervezési folyamatokban részt vevő tevékenységek két csoportot alkotnak aszerint, hogy a számitógépes rendszer vagy a tervező végzi el. Továbbiakban a felosztás vizsgálatához a számitástechnikából jól ismert alapelvet használjuk fel, amely szerint megkülönböztetünk algoritmizálható és nem algoritmizálható feladatokat. Ezt az elvet a tervezési folyamatokban részt vevő tevékenységekre alkalmazva és kiterjesztve kétféle tevékenység tipust különböztethetünk meg.

1. Algoritmizálhatók.
   A tervezési folyamatban részt vevő olyan tevékenységeket sorolunk ide, amelyek formalizálhatók, igy számitógépre vihetők.
2. Nem algoritmizálhatók.
   Ide soroljuk:
   - azokat a tevékenységeket, amelyeket a tudomány jelenlegi állása szerint ma még nem tudunk algoritmizálni;

- azokat az algoritmizálható tevékenységeket, amelyeknek számitógépre vitele különböző okok miatt nem gazdaságos;
- a tervezési folyamatokban meglevő bizonyos /interaktiv/ döntések végrehajtását;
- a tervezésben jelen levő kreativ tevékenységeket.

Az interaktiv tervező munkában vannak olyan részfeladatok, amelyeket csak interaktiv üzemmódban tudunk hatékonyan megoldani. Ezek általában az ember számára triviális feladatok, mig a számitógépes megoldás roppant időigényes. Példaként a sajtoló szerszámok tervezéséhez kidolgozott optimális sávelrendezési algoritmusokat hozhatjuk fel. Erre a feladatra a Robotron cég által kidolgozott program futási ideje kétsoros elrendezésü sáv esetén fél óra, háromsoros esetben kb. 2 óra /R40-es számitógépen/. Ugyanez a feladat interaktiv megoldásban az ISTER segitségével 5-10 perc időt vesz igénybe ugy, hogy a tervező az optimálist megközelitő geometriai elrendezést a display képernyőjén összeállitja.

Az 1970-es évek közepén egy olyan irányzat indult el, amely szerint a tervező munka automatizálására alkalmas tervező rendszerek létrehozására un. <u>általános célu univerzális rendszereket</u> kell kialakitani [114, 115]. Ezt az irányzatot legjobban képviselte a MIT-ben /Massachusetts Institute of Technology/ kidolgozott ICES /Integrated Civil Engineering System/ rendszer [62]. Az ICES nagy számitógépet igényelő keretrendszer, és olyan részegységeket tartalmaz, amelyek lehetővé teszik, hogy a felhasználók saját feladatuk megoldására célorientált rendszereket hozzanak létre. Már a rendszer kifejlesztését követő alkalmazások megkövetelték, hogy az ICES-nek különböző verzióit hozzák létre. Ilyenek a PSU-ICES, IST, BAL, MIRIAM, REGENT, GERMINAL, ISP-2 [114, 116].

A gyakorlati tapasztalatok azt bizonyitották, hogy az ilyen nagy rendszerek tulságosan rugalmatlanok, a felhasználásuk bonyolult és sok esetben a tervező rendszerek elkészitésének igényeit is nehezen elégitik ki. Annak ellenére, hogy az univerzális keretrendszerek létrehozására nagy erőket fektettek be, jóformán a világ minden táján /Berlinben az IST [63], Párizsban a BAL [117], Angliában a RAINBOW [118], Szovjetunióban az INKON [119], SZIRIUSZ [120], RADUGA [121], a KGST-országokban pedig az AMT monitor rendszerek [122, 123]/ a hozzájuk füzött reményeket nem váltották be.

A komplex és nehezen kezelhető, többnyire nagy számitógépeket igénylő rendszerek helyett az 1970-es évek végén a kis és olcsó számitógépeket alkalmazó tervező rendszerek kezdtek utat törni maguknak. Ezek a kisgépes tervező rendszerek általában egy adott tervezési területet fognak át. Ilyen tipusu rendszerek az un. kulcsátadásos rendszerek, amilyen a kidolgozott ISTER is [4,161,162,163]. A kisgépes tervező rendszerek elterjedése bizonyos fokig háttérbe szoritotta az univerzális problémamegoldó rendszerek irányzatát.

A tervező rendszerek létrehozásának az egyik legfontosabb problémája a modellezés. Modellen itt a tervezés folyamán keletkezett olyan strukturált adathalmazokat értünk, amelyek a tervezendő objektumot egyértelmüen leirják. A tervezési folyamatban a különböző tevékenységek követik egymást. Ezek a tevékenységek többnyire a rendszerhez tartozó modellen keresztül kapcsolódnak egymáshoz. A tervezőrendszer modelljeinek leirása nagyon bonyolult és összetett feladat. Ha azonban az egyes tevékenységekhez külön-külön un. részmodelleket dolgozunk ki, ezeknek a leirása már lényegesen egyszerübb. A kitüzött feladat megoldása csak akkor lesz teljes, ha a részmodelleket valamilyen módon összekapcsoljuk.

A részmodelleket két nagy csoportra oszthatjuk, az általános és az egyedi részmodellekre. Általános egy részmodell akkor, ha a gépészeti tervezés különböző területein szinte változtatás nélkül felhasználható. Például egy 2D geometriai modell a sajtoló szerszámok, sávtervek, szabásminták tervezéséhez egyaránt alkalmas lehet. A második csoportba tartozó részmodellek szorosan az adott feladat megoldására nézve adekvátak, azaz a célorientált rendszer speciális részeit alkotják. Az ilyen egyedi modellek megfogalmazására az irodalom különböző eljárásokat, nyelveket ajánl [124, 60, 61].

A tervező rendszerek általános szintü modellezésére a legjobb példa a gépészeti alkatrészek modellezése. Az egyes kutatóhelyek különböző elvekből kiindulva próbálják a feladatot megoldani és igy nagyon sok irányzat alakult ki. Ezeket az irányzatokat rendszerezi és a legjelentősebb megoldásokat részletesen ismerteti Holló Krisztina és dr. Várady Tamás 1978 őszén megjelent tanulmánya [125].

Nyilvánvaló, hogy a gépészeti konstrukciós tervezésnél a modellezésnek magából a tervezendő objektumból kell kiindulni. Eszerint több szintü hierarchikus modellezésről beszélhetünk, ugy mint:
- tervezési;
- szerkezeti és
- alkatrészmodellezésről.

Az alkatrészmodellek a szerkezeti modellben kerülnek egymással kapcsolatba, amely tagja a rendszer tervezési modelljének. A tervezési modell még a különböző számitási algoritmusok részmodelljeit is tartalmazza.

Több kisérlet született arra, hogy a hierarchikus modellezésre egzakt matematikai reprezentációt találjanak.

Dr. Holnapi Dezső [126] a részhalmazok hierarchiáját /egymásba skatulyázott halmazok/ találja alkalmasnak erre és bevezeti a halmazok fokszámának fogalmát. Dr. Márkusz Zsuzsanna [127] a több fajtáju klasszikus matematikai strukturákat közvetlenül alkalmazza épitészeti konstrukciós feladatok modellezésére.

A hierarchikus modellezéshez is adekvátnak találtam a több fajtáju relációs strukturákat. Eszerint a tervezési modell egy τ pár:

$$\tau = < Ts, R >$$

ahol τ a tervezési modell

Ts a tervezési modell alaphalmaza

R a Ts-en értelmezett több fajtáju relációk halmaza.

Ts egy több fajtáju halmaz

$$Ts = \{A\lambda_i \quad \lambda_i \in S\}$$

ahol $S = \{\lambda_i \quad i < \omega\}$ a fajták halmaza

/Maximálisan ω halmaz lehet/

$A\lambda_i$ a λ_i fajtáju objektumok halmaza

A gépészeti tervezésből vegyünk egy példát.

Legyen

$$S = \{a, c, h, sz, d\}$$

ahol a - az alkatrészek

c - a szerkezetek

h - a hőtechnikai modellezés

sz - a szilárdságtani modellezés

d - a dinamikai modellezés

egységeinek fajtái.

pl.: A_a = {tengely, fogaskerék, csapágy, tengelykapcs. stb.}

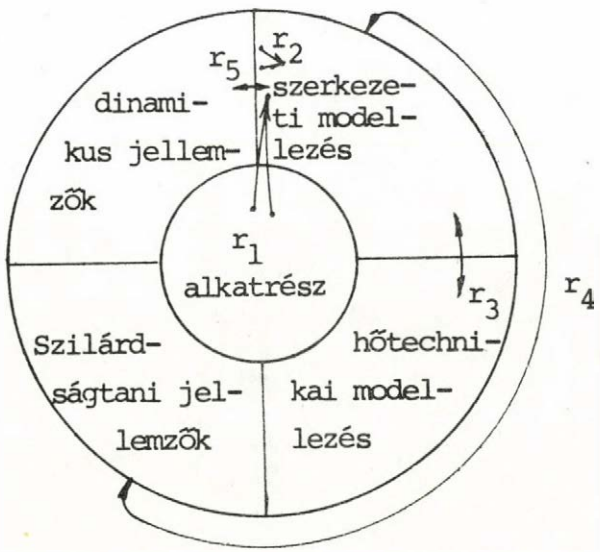
Legyen további

$$R = \{r_1, r_2, r_3, r_4, r_5\}$$

ahol bármelyik r_i $1 \leq i \leq 5$ reláció definicióját az argumentumok fajtáinak sorozatával adjuk meg.

$r_1 = < a, a, c >$
$r_2 = < c, c, c >$
$r_3 = < c, h >$
$r_4 = < c, sz >$
$r_5 = < c, d >$

Az r_1 reláció tulajdonképpen egy függvény, amely két alkatrész fajtáju objektumból egy szerkezet fajtáju objektumot hoz létre. Ehhez hasonló az r_2 reláció, amely két szerkezetből egy uj, összetettebb szerkezetet hoz létre. Az r_3, r_4 és r_5 relációk a szerkezetek és a hőtechnikai, szilárdságtani és dinamikai modellezés egységei közötti kapcsolatokat definiálják. Lásd az 5. ábrán.



5. ábra

3. INTERAKTIV RENDSZEREK

A számitástechnika alkalmazásával különböző célu interaktiv rendszerek születtek. A megvalósitott rendszerek száma olyan nagy, hogy lehetetlen ismertetésüket ezen a helyen megkisérelni. Az összehasonlitás és elemzés helyett hivatkozunk Rouse cikkére [128], amely 175 irodalmi hivatkozás alapján próbálja az áttekintést megadni, valamint a [129, 131] publikációkra.

Ebben a fejezetben a megvalósitott interaktiv rendszerek osztályozásával [132], a bennük levő dialógus kérdésekkel, valamint néhány implementált dialógus rendszer ismertetésével foglalkozunk.

3.1. Az interaktiv rendszerek osztályozása

Az interaktiv rendszerek osztályozására néhány szubjektiv - általam célszerünek itélt - szempontot vezettem be. Ugymint

- az alkalmazások;
- az interaktiv tervezési folyamatban gyakorolt kezdeményező szerep és
- a rendszerszervezés szerinti

csoportositást követem,mint az osztályozás fő rendszerező elveit.

1. Alkalmazások szerinti csoportositás. A különböző felhasználási szempontok különböző tipusu interaktiv rendszereket kivántak meg. Martin [107] a létrehozott rendszereket négy osztályba sorolja aszerint, hogy az információcsere milyen az ember és a számitógép között. Az osztályok a következők:

a/ alfanumerikus;
b/ grafikus;
c/ álló képet kijelző;
d/ mozgó képet megjelenitő
rendszerek.
Az "a" és "b" osztályba tartozó rendszerek perifériái input és output funkciókat töltenek be, mig a "c" és "d" osztályba tartozók csak output perifériaként müködnek.

2. Kezdeményezés szerinti csoportositás. Ezen az értendő, hogy az interaktiv rendszer felhasználása közben melyik fél játssza a kezdeményező szerepet. A kezdeményező lehet
a/ az ember,
b/ a számitógép /a rendszer/,
c/ váltakozva mindkettő [133, 134].
Általában a kisebb rendszerek az "a" megoldást választják, mivel a választási lehetőségek száma kicsi és a kezelést a felhasználó könnyen meg tudja tanulni. Ilyenek általában az operációs rendszerek, ahol a konzol üzenetek kezdeményezése az operátor kezében van. Nagyobb, vagy összetettebb rendszerek esetében a kezdeményező szerepet a számitógép veszi át /"b" eset/. Példák erre az integrált tervező rendszerek, ahol a lehetséges programrészek száma olyan nagy, hogy a tervező nem képes fejben tartani őket. A rendszer menükön keresztül kinálja fel az adott szituációhoz tartozó választási lehetőségeket. Abban az esetben, ha például a gyakran előforduló tervezési feladat elvégzésére szolgáló programrészek közvetlen elérését biztositani akarjuk, akkor "c" tipusu rendszert kapunk. A későbbiekben ezzel a rendszertipussal foglalkozunk.

3. Rendszerszervezés szerinti csoportositás.
   a/ Master vagy executiv programokkal vezérelt interaktiv rendszerek. A rendszerekhez tartozó dialógus részeket, illetve az azokat aktivizáló programszegmenseket a master programon belül, arra a helyre épitik be, ahol aktivizálásuk szükséges. Ezeknek a rendszereknek a felépitése soros és e tekintetben hasonlitanak a hagyományos programokhoz.

   b/ Dialógus rendszer által vezérelt interaktiv rendszerek. Ebben a megoldásban az interaktiv rendszerekhez tartozó egyéb programszegmensek egymáshoz képest párhuzamosan helyezkednek el és aktivizálásukról a dialógus rendszer gondoskodik. A tanulmány ilyen tipusu programrendszerekkel foglalkozik.

3.2. A dialógus rendszer helye és szerepe az interaktiv tervező rendszerekben

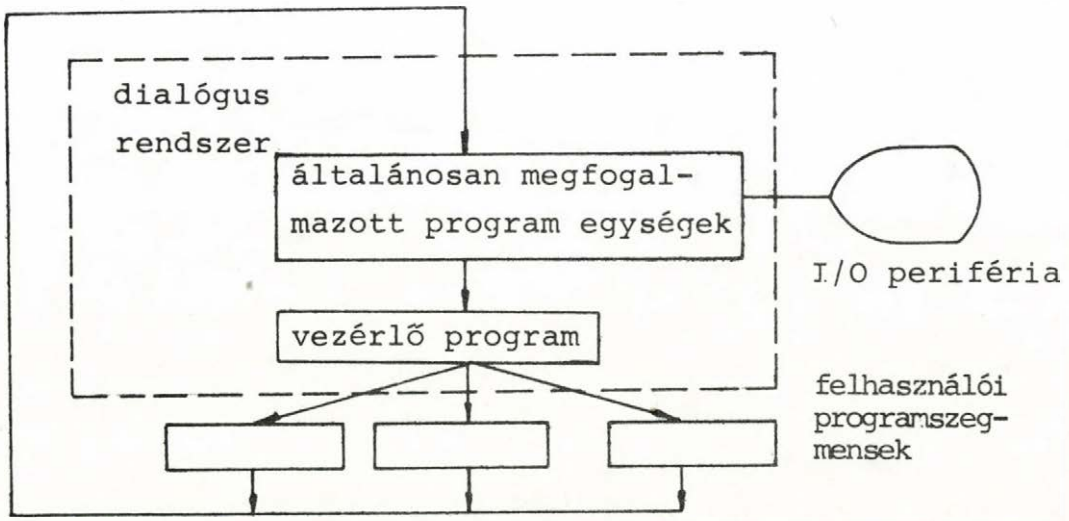
Egy interaktiv tervező rendszeren belül a dialógus rendszer feladata gondoskodni az ember és a számitógép közti kapcsolat megteremtéséről [107, 135, 136, 137, 108, 139, 140, 141, 142, 93, 138].

Ebben a fejezetben a dialógus rendszer által vezérelt interaktiv rendszerekkel foglalkozunk. Vizsgáljuk meg, hogy a vezérlést is tartalmazó dialógus rendszereknek melyek az alapvető feladataik, továbbá milyen követelményeket támasztunk velük szemben.

1. Output adatok kezelése. A rendszer a megfelelő perifériára kiirja a tervezőnek szánt kérdéseket, üzeneteket alfanumerikus vagy grafikus formában.
2. Input adatok kezelése. A tervező által megadott különböző információk beolvasását és értelmezését szintén a rendszer végzi.

3. Paraméterátadás végrehajtása. A dialógus rendszernek kell gondoskodni arról, hogy a beolvasott paraméterek a hozzájuk tartozó programszegmensekhez kerüljenek.
4. Interaktiv rendszer vezérlése. A beolvasott adatok mindig valamelyik felhasználói programszegmenshez tartoznak. A rendszernek kell gondoskodnia arról, hogy a beolvasott adatokhoz tartozó programszegmenst aktivizálja, a beolvasás után.

A dialógus rendszer létrehozásával azt a célt tüztük ki, hogy a felsorolt feladatok megoldására általános módszereket adjunk. Ha a feladatokat sikerül általános szinten megfogalmazni, akkor a megoldásokat reprezentáló programegységek /dialógus rendszer/ az interaktiv rendszeren belül kiemelhetők ugy, hogy a felhasználói programszegmensek párhuzamos kapcsolásával a vezérlést el tudják végezni. A megoldásra a 6. ábrán látható rendszerfelépitést javasoljuk. /Az ISTER ebben a felépitésben készült el./



6. ábra

Az ábrából leolvasható, hogy a rendszer futása zárt ciklusban történik. A futás mindig azon a felhasználói programszegmensen halad át, amelyre a beolvasott paraméterek vonatkoznak.

3.3. Néhány megvalósitott dialógus rendszer

Az 1970-es években nagy számu interaktiv rendszer született /1. 2. fejezetet/. A rendszerek publikációiból nem derült ki, hogy a dialógus problémákat hogyan oldották meg. 1977-ig az irodalmi adatok alapján ugy tünt, hogy ez a probléma nem is létezik, illetve a megoldásuk nem olyan sulyu, hogy publikálható legyen. Ezt az elképzelésünket az 1977 után megjelent ilyen irányu cikkek döntötték meg, amelyekből kitünt, hogy világszerte komolyan foglalkoztak külön a dialógus rendszerek problémáival.

Mielőtt az ismert dialógus rendszereket áttekintenénk, összefoglaljuk, hogy hazánkban 1971 óta e területen milyen eredmények születtek. Krammer Gergely és Forgács Tamás 1971-ben készitették el a DISTAR-B rendszerüket [143, 144], amely jelentős eredménynek tekinthető, és a későbbi munkáinkban sikeresen alkalmaztuk. A rendszer egy dialógus koncepciót ad, a dialógus adatok tárolására táblázatokat és a táblázatok kezelésére szubrutincsomagot ajánl. A DISTAR-B első alkalmazása a jelen munka szerzője által 1973-ban publikált "Interaktiv alkatrész-program-iró rendszerben"-ben [145] valósult meg. A grafikus interaktiv periféria alkalmazására a szerző 1976-ban uj dialógus rendszert hozott létre, amely a DISTAR-B-nél lényegesen fejlettebb volt [3, 146]. Az 1976-ban kifejlesztett dialógus rendszerrel valósult meg az ISTER.

A továbbiakban megemlitünk néhány külföldön kidolgozott dialógus rendszert, amelyről csak vázlatos ismereteink lehetnek a be nem szerezhető információk miatt.

A californiai Sperry Research Centre-ben folyó kutatásokat 1977-ben publikálta Black [147]. Az általuk kidolgozott dialógus rendszer két fő részből áll. Az első a DSL nyelv /Dialógue Specification Language/ a forditó programmal; a második a dialógus processzor. A DSL nyelv az automatizálandó folyamat grafikus ábrájának, azaz az un. "dialógus gráfjának" a leirására szolgál. A DSL nyelv a digalógus gráf leirására használható utasitásokon kivül tartalmazza mindazokat a rekordokat is, amelyeket az algoritmikus nyelvek tartalmaznak. A dialógus rendszert főleg kis számitógépek alkalmazására dolgozták ki.

A Philips cégnél kidolgozott"Dialógus Kezelő Rendszer"-t Bauböck [148] ismertette 1978-ban. A kidolgozott rendszert ezköznek tekinti interaktiv rendszerek létrehozásához. A cikk a rendszer felépitéséről átfogó képet nem ad, de az egyik legnagyobb problémának tekinti a többszintü hierarchikus dialógus struktura leirását.

A bratiszlavai Számitástechnikai Kutató Központban dolgozták ki a DIAGEN /DIAlogue GENerátor/ dialógus rendszert, amelyet 1977-ben [134] publikáltak. A rendszer célorientált interaktiv rendszerek létrehozására szolgál /alfanumerikus interaktiv rendszerekhez/. A részdialógusok leirására a DIAGEN nyelvet hozták létre, amely nyelven irt programot egy transzlátor dolgoz fel. A transzlátor által összeállitott adatok a "dialógus program könyvtár"-ba kerülnek. Az applikációs

programok aktivizálására szolgáló CALL utasitásokat a dialógus programba kell beépiteni.

Hasonló elveken müködik a Rostock-i Egyetemen kidolgozott dialógus rendszer is [149] .

4. AZ EMBER-SZÁMITÓGÉP KAPCSOLAT NÉHÁNY ELMÉLETI MODELLJE

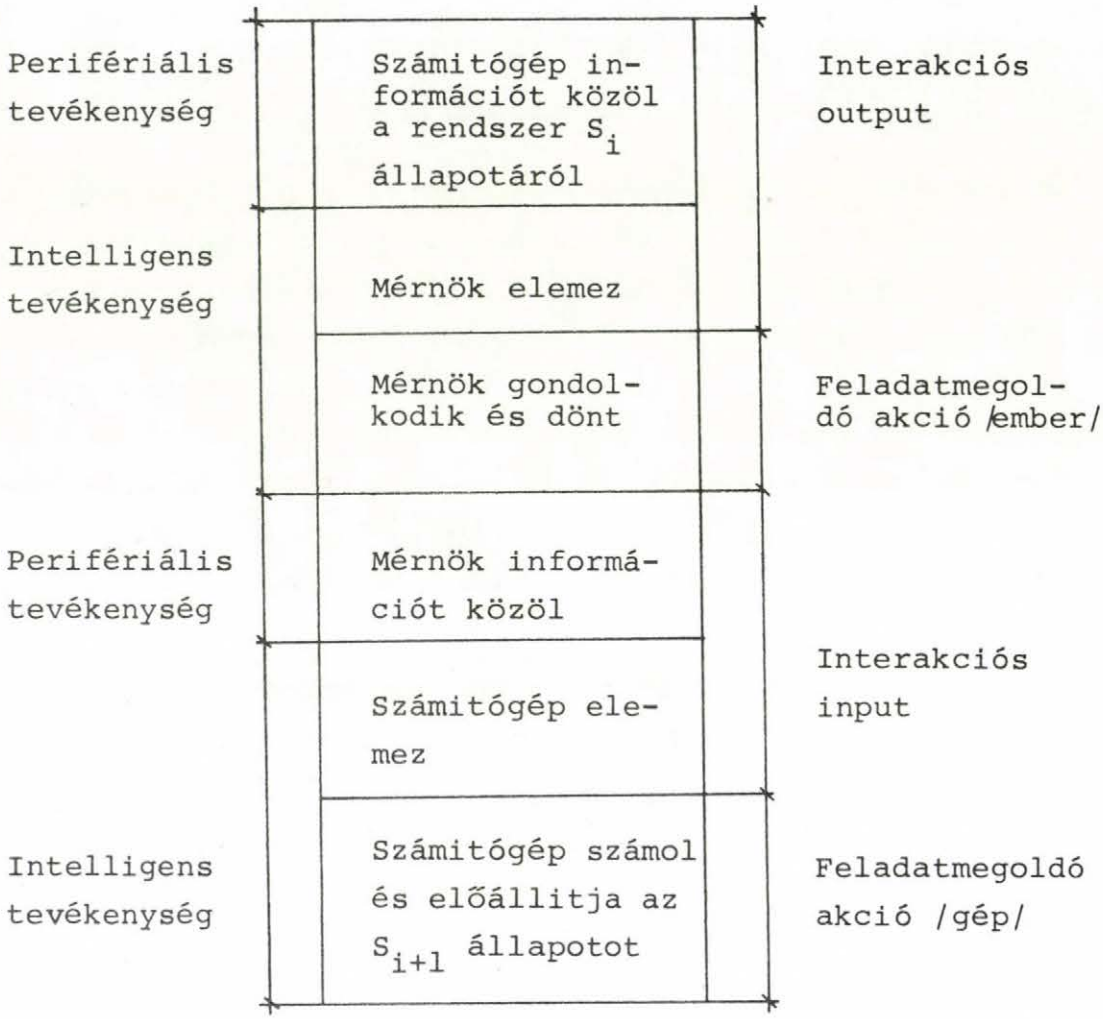
A különböző szerzők több aspektusból ragadták meg az ember-számitógép kommunikáció elméleti és gyakorlati vonatkozásait [150]. Vannak szerzők, akik a kapcsolaton egyszerüen azt a nyelvet értik, amelyen a kommunikáció folyik a két individuum között, de vannak olyanok is, akik a számitógép intelligenciáját növelve intelligens dialógus rendszerek létrehozásával szeretnék a kapcsolatot megteremteni. Ez utóbbi irányzat a mesterséges intelligencia módszerek felhasználását célozza meg.

Ebben a fejelzetben néhány olyan koncepciót ismertetünk, amelyek a dialógus problémák gyakorlati megvalósitásához vezetnek.

Forgács Tamás [151] a dialógus rendszerek problémáját a tervező rendszerek által megkövetelt területekre szükiti le, és elméleti alapot próbál teremteni a feladatok megoldására. Megközelitésében a tervezési folyamatból indul ki. Szerinte a tervezés S_o állapotban lévő adatbázissal indul, amely a tervezés végére S_v állapotba kerül. A tervezési folyamat bizonyos bemenő adatokkal indul /input/ és kimenő adatokkal /output/ fejeződik be. Formálisan ez azt jelenti, hogy

$$(S_o, \text{input}) \; T \; (S_v, \text{output})$$

ahol T a tervezési folyamatot leiró operátor.

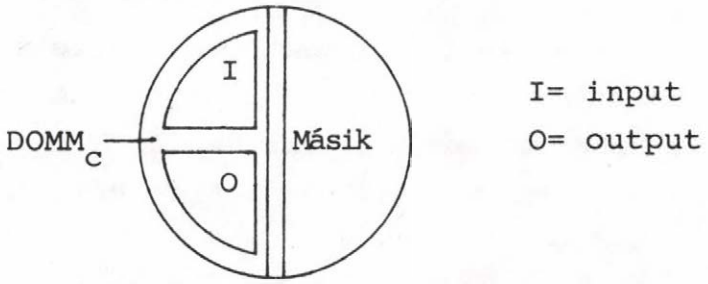


7. ábra

Forgács az interaktiv operátor szerkezetére /T/ a 7. ábrán bemutatott felépitést adta, amely az egyes tevékenységeket nagyon jól szemlélteti. Kochan hasonló eredményekre jutott, amelyet 1977-ben megjelent könyvében [136] publikált.

Krammer Gergely a dialógus problémát sokkal szélesebb perspektivából szemléli [152]. Modelljét Pask alapján

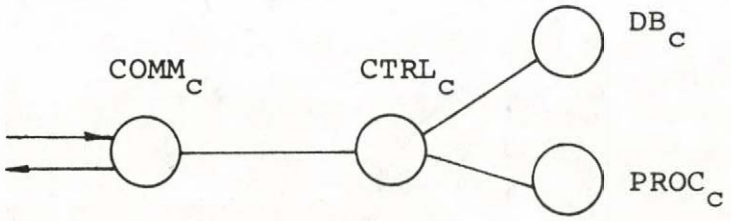
szimmetrikusnak tételezi fel, amelyben a két individuum beszélgetést folytat a világ megismeréséről. A beszélgető individuumokat további két részre bontja, egy kommunikációs részre és egy "másik" részre /8. ábra/. A "másik", azaz a belső rész tartalmazza:



8. ábra

1. a belső szabályozó tevékenységet /CTRL$_C$/;
2. az adatbázist /DB$_C$/;
3. a tudást /PROC$_C$/.

Ezen részek kapcsolatát a 9. ábra szerinti gráf tartalmazza



9. ábra

Krammer a dialógus folyamatot három részre bontja:

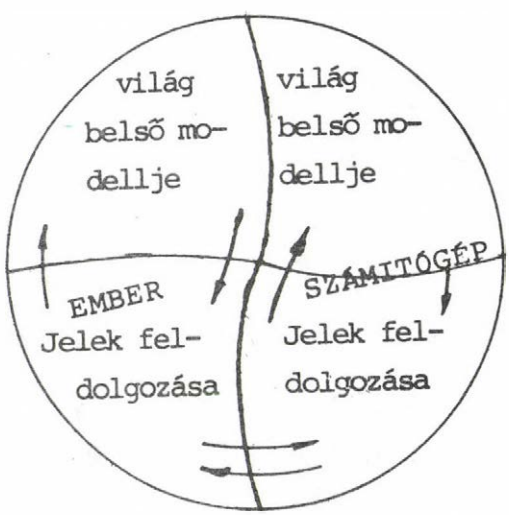
1. megértés;
2. feldolgozás;
3. cselekvés.

Ezt a felosztást - analóg módon - kiterjeszti a számitógépre is:

1. input periféria /I-Device/;
2. C központi rész /számitógép/;
3. output periféria /O-Device/.

A fentiek alapján azt a következtetést vonja le, hogy az input, output és a központi rész fejlettségi foka korlátként lép fel az ember-gép kapcsolat intelligenciájában. Krammer Gergely cikkében az I/O perifériák tulajdonságait vizsgálja, abból a célból, hogy a korlát minél kedvezőbb legyen.

Gaines [153] az ember-számitógép kapcsolatot képességerősitőként fogja fel. Az ember a számitógéppel közösen szeretne megoldani problémákat ugy, hogy a partnert /számitógépet/ intelligensnek tekinti. Gaimes a kommunikációs kapcsolat mellett a kapcsolat általános modelljének felállitására törekszik. A modell létrehozásában két ember kommunikációját tekinti kiinduló pontnak. Feltételezi, hogy a beszélgető partnerek a "világról" /beszélgetés tárgya/ külön-külön belső modellt épitenek és a modellek pillanatnyi állapota szerint cselekszenek. A fenti feltételezést alkalmazva ember-szá-



10. ábra

mitógép kapcsolatra kapja a 10. ábrán feltüntetett kommunikációs ábrát. Az elképzelés legnagyobb problémája a világ /leszükitett világ/ modellezésében van.

A **tanulmány** Pask beszélgetés elméleti eredményeit [154, 155] használja fel a javasolt dialógus rendszer megalapozására. Pask elmélete [156] uttörő jellegü abban, hogy

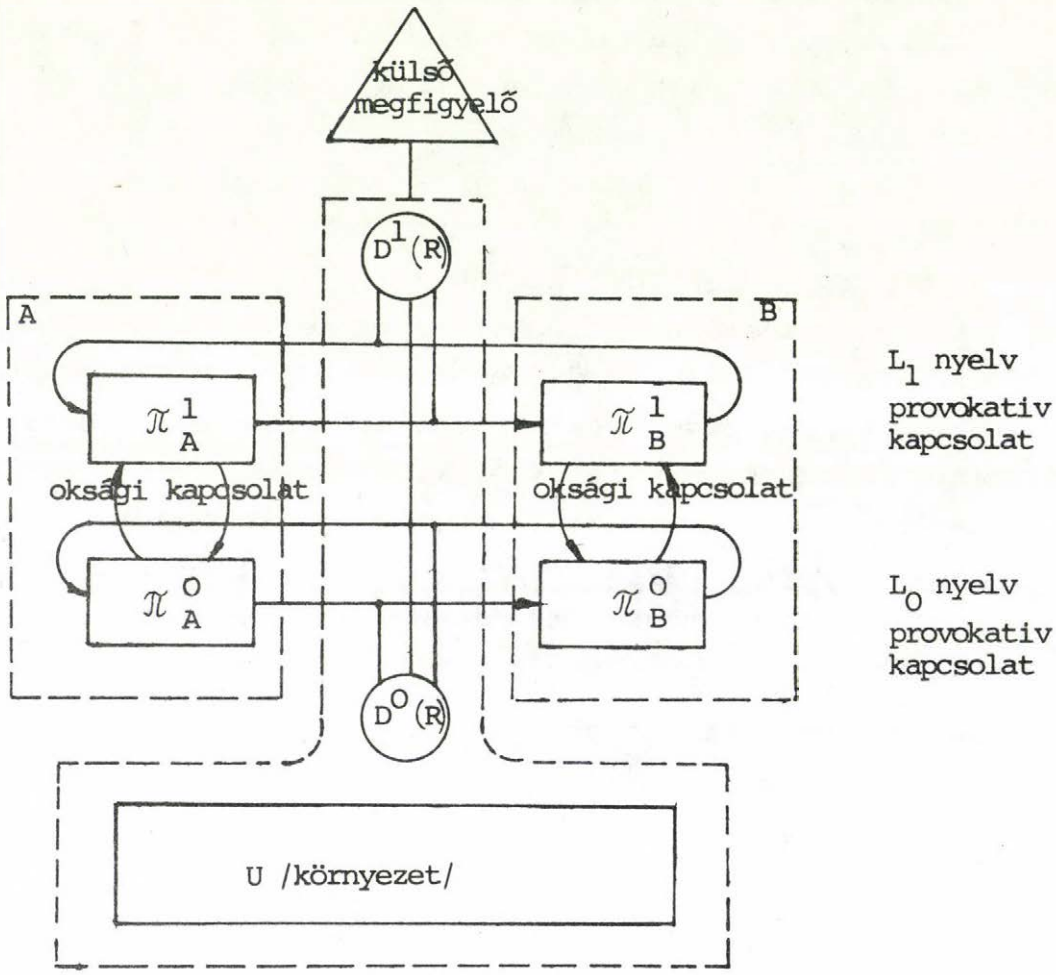
- pontosan megfogalmazza, hogy egy dialógus mikor intelligens,
- sokkal korrektebb definiciót ad a gépi intelligenciára, mint az előzőleg ismertetett szerzők,
- választ ad arra a kérdésre, hogy a gép intelligenciájának foka szükséges feltétele-e az ember-**számitógép** intelligens kommunikációjának.
  A válasz az, hogy nem. Ez azt jelenti, hogy nem kell megvárni azt, hogy intelligens számitógépet tudjunk épiteni ahhoz, hogy intelligens dialógus rendszert készitsünk.

A fejezet további részében röviden áttekintjük Pask beszélgetés elméletét.

Egy intelligens beszélgetés két individuum /A és B/ között folyik /11. ábra/. A beszélgetést egy külső megfigyelő szervezi, irányitja és ellenőrzi. A beszélgetés feltételeire a külső megfigyelő a beszélgetés résztvevőivel szerződést köt L^* nyelven. A szerződési feltételek kiterjednek:

1. az L nyelvre, amelyen a két individuum a beszélgetést folytatja,
2. a beszélgetési témák tartományára /D(R)/ /**a világ** egy része/,
3. a beszélgetési feltételekre, amelyekben a partnerek egy adott időben csak egy témáról kommunikálhatnak és a **beszélgetésnek** megértéssel kell befejeződnie.

4. a beszélgetés folyamán a felek tanulnak /de legalább is az egyik fél tanul/, és a beszélgetés befejeztével tudásuk gyarapszik.



11. ábra

A 11. ábrán feltüntetett sémát Pask a beszélgetés ikonjának nevezi. Az ábra alapján a beszélgetés elemeit a következőkben foglaljuk össze.

A. D(R) a beszélgetési tartomány, amely egy relációs **strukturával** irható le, ahol a struktura alaphalmaza a beszélgetés tárgykörének elemeit foglalja magába.

A beszélgetési tartomány két részre oszlik.

$D(R) = \langle D^1(R), D^0(R) \rangle$

ahol

$D^0(R)$ a témákat

$D^1(R)$ a témák közti összefüggéseket tartalmazza.

B. A felek közti beszélgetés nyelve L, amely két szinten alakul ki.

$L = \langle L^1, L^0 \rangle$

Az L^0 szintü beszélgetés $D^0(R)$ tématartományról folyik és jellemzője a "hogyan" kérdés.
L^1 szinten folyik a beszélgetés $D^1(R)$ tématartományról és jellemzője a "miért" kérdés.

C. Egy individuum két eljárástipust tartalmaz. Ezen belül
a/ Π_i^0, vagy $PROC_i^0$ elmagyarázza, vagy előállitja R_i témát,
b/ Π_i^1 /vagy $PROC_i^1$ elmagyarázza, hogyan kell Π_i^0-t megtanulni vagy előállitani.
A "0" szintü eljárásokat koncepcióknak,az "1" szintü eljárásokat memóriáknak nevezte el Pask.

D. A beszélgetés folyamán két tipusu kapcsolatot különböztetünk meg:
a/ oksági kapcsolat, ami a koncepciók és memóriák között jön létre;
b/ provokativ kapcsolat, ami a két individuum memóriái /L^1/ és koncepcióin /L^0/ keresztül alakul ki.

E. A beszélgetés környezetének nevezzük azt a környezetet /U/, amely mindkét individuummal kapcsolatban van. Ez azt jelenti, hogy oksági kapcsolatban van a Π_A^0 és Π_B^0 eljárásokkal.

Dr. Márkusz Zsuzsanna tanulmányában [156] az eddig létrehozott tervező rendszerekben kialakult kapcsolatok alapján próbálja felrajzolni azok ikonjait.

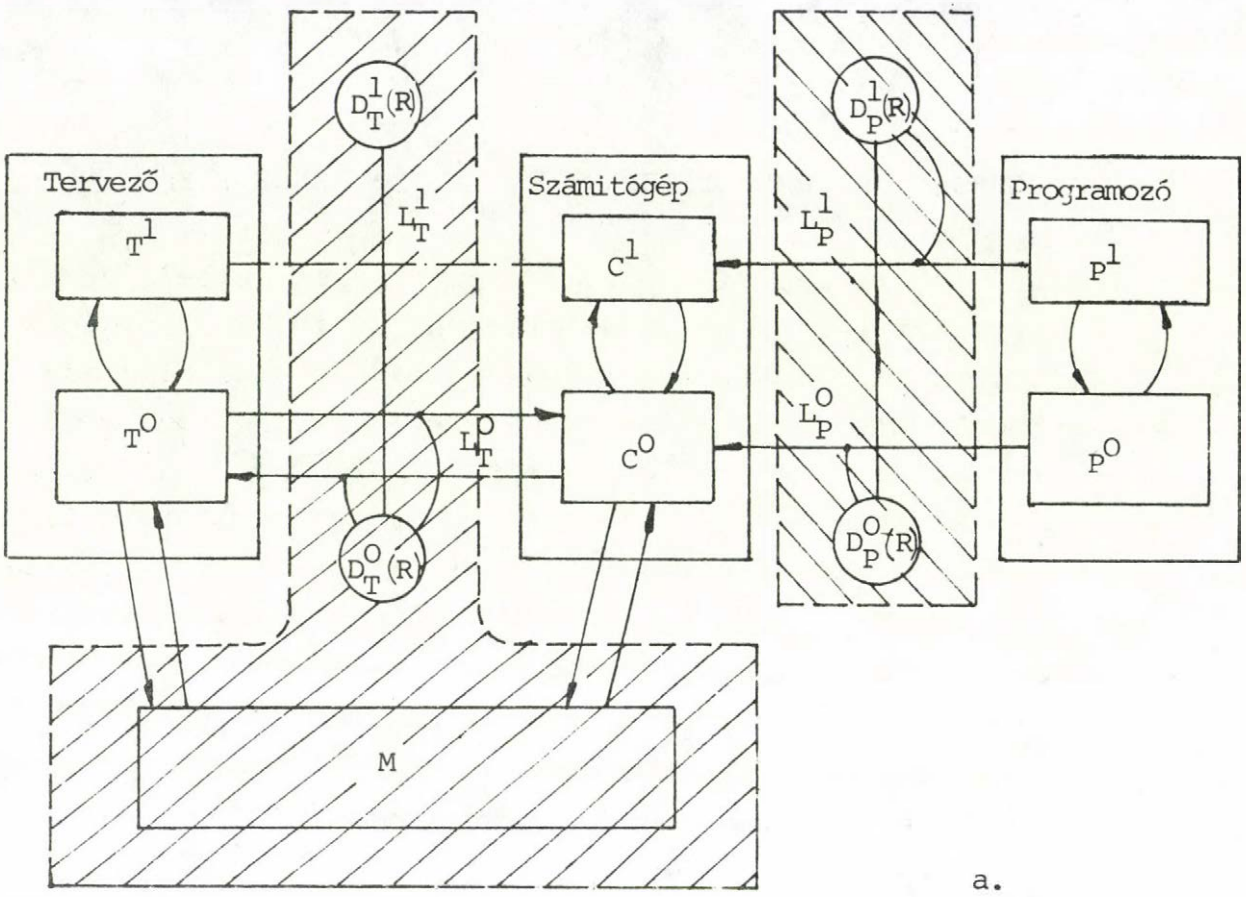
5. DIALÓGUSSAL VEZÉRELT INTERAKTIV TERVEZŐ RENDSZEREK

Ebben a fejezetben Pask beszélgetés elméleti modelljét mint eszközt felhasználva megalkotjuk az interaktiv gépészeti tervező rendszerek létrehozásában és azok alkalmazásában lejátszódó folyamatok elméleti modelljét. A modell felhasználásával szétválasztjuk az általánosan megfogalmazható feladatokat az általánosan nem megfogalmazható feladatoktól. Ezután kapcsolatokat definiálunk a két csoporthoz tartozó, a feladatokat reprezentáló eljárások között.

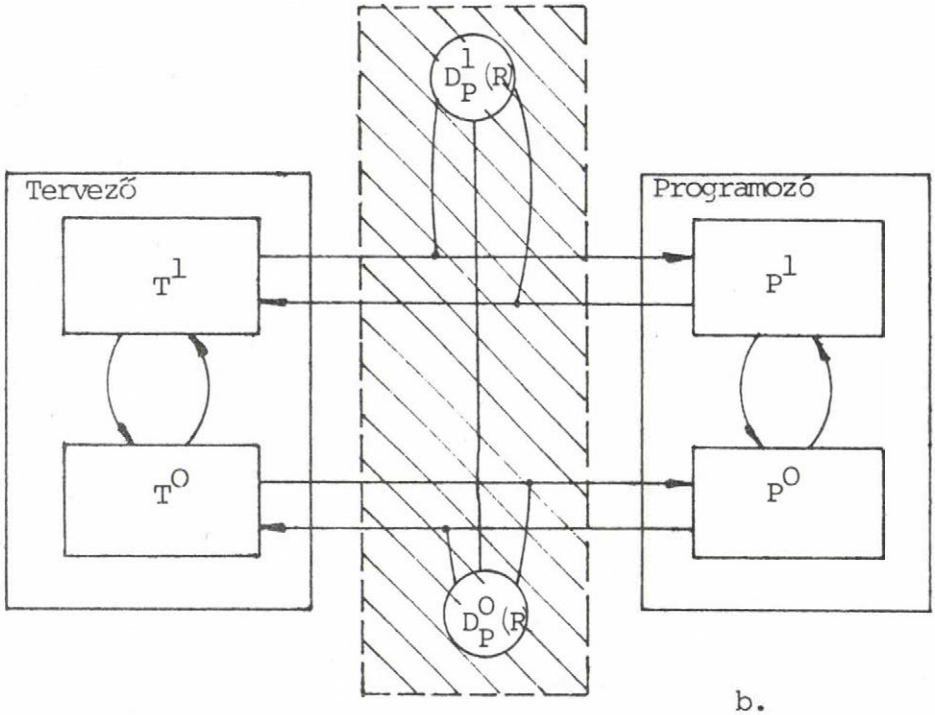
5.1. Az interaktiv tervező rendszerek elméleti modellje

Intelligens interaktiv tervezés megvalósitásának egyik lehetséges módja az, hogy a tervező rendszerbe programgenerátort épitünk be. A programgenerátor képes megvalósitani azt az intelligens kapcsolatra jellemző feltételt, hogy a tervező rendszer alkalmazkodjék a megváltozott külső körülményekhez uj programok vagy programállandók létrehozásával.

A létrehozandó tervezési modellben a programgenerátort arra használjuk fel, hogy az interaktiv tervező rendszert egy induló vagy kezdő állapotra hozza. A kezdeti állapotok előállitása időben megelőzi az interaktiv rendszer felhasználását. Pask szerint ez azt jelenti, hogy időben eltolva két tématartományról folynak beszélgetések. A két tématartományban végbemenő dialógus nem azonos individuumok között folyik. A kezdeti állapot beállitása a programozó /P/ és a számitógép /C/ között, mig a tervezés, azaz a rendszer felhasználása, a tervező /T/ és a számitógép /C/ között folyik. Az időben eltolt és a tématartományban is eltérő beszélgetések között nem elhanyagolható kapcsolatok vannak. Ezeket a kapcsolatokat mutatjuk be a 12. ábrán,


Tervező
Számitógép
Programozó
T^1
T^O
C^1
C^O
P^1
P^O
$D_T^1(R)$
$D_T^O(R)$
$D_P^1(R)$
$D_P^O(R)$
L_T^1
L_T^O
L_P^1
L_P^O
M

a.


Tervező
Programozó
T^1
T^O
P^1
P^O
$D_P^1(R)$
$D_P^O(R)$

b.

12. ábra

amely az interaktiv rendszer ikonjait ábrázolja.

Az ikonok három individuumot tüntetnek fel, a tervezőt /T/, a számitógépet /C/, és a programozót /P/. A T és C individuumok közti tématartomány $D_T(R)$, amely a tervezési területekhez tartozó témák halmaza. A témákat az individuumok felhasználják és a beszélgetés eredményeként kész tervek jönnek létre. A tervek létrejötte azt jelenti, hogy a 12/a ábrán látható M modell a tervezés tárgyától függő adatokkal feltöltődik. A P és C individuumok közti tématartomány a $D_P(R)$ /12/a ábra/, amely a tervező rendszer létrehozásához és a rendszer kezdő állapotának beállitásához a témák halmaza. Ugyanezt a tématartományt találjuk a P és T individuumok közti párbeszédben is /12/b ábra/. Ez a párbeszéd a már létrehozott tervező rendszer korrigálására, javitására, vagy továbbfejlesztésére szoritkozik.

A tervezőben mint individuumban létezik a két eljárásszint /T^0 és T^1/. A T^0 tartalmazza a tervezéshez tartozó összes eljárást, T^1 pedig az eljárások közti kapcsolatokat, vagyis a tervezési modelleket. A tervezőben létrejön a két szint /T^0, T^1/ közti oksági kapcsolat.

A számitógépben a C^0 szint mellett a programgenerátor létrehozásával alakulhat ki a C^1 szint. Mielőtt a C^1 szint feladatkörét meghatároznánk, nézzük meg, hogy a C^0 szintnek, amely dialóguson keresztül tartja a kapcsolatot a tervezővel, milyen főbb részekből /egységekből/ kell állnia, hogy feladatait elvégezhesse:

1. felhasználói, vagy tervezői programok;
2. a dialógust végrehajtó processzor;
3. a felhasználói programokat összeszervező vezérlő program;
4. a dialógus processzort vezérlő adatok.

A dialógus adatok /4. pont/ által vezérelt dialógus processzor /2. pont/ hajtja végre a dialógust a tervező /T/ és a számitógépes rendszer /C/ között. A vezérlő programnak /3. pont/ kell gondoskodnia arról, hogy a program futása arra a tervezői programra /1. pont/ kerüljön, amelyre a dialógus processzor által beolvasott adatok vonatkoznak. A dialógus processzor függetlenithető a létrehozandó tervező rendszer tipusától /célorientált rendszertől/, azaz teljesen általánosnak tekinthető . A felhasználói programok teljesen függnek a tervezés tárgyának tipusától /célorientáltak/. A vezérlő program, valamint a dialógus adatok a tervezési folyamatnak és a felhasználói programnak a függvénye.Ha megtaláljuk azokat a relációkat, amelyek képesek bizonyos külső adatokból /az előkészitő szakaszban/ létrehozni a dialógus adatokat és a vezérlő programot, akkor a C^1 szinten elhelyezkedő, a relációkat reprezentáló generátort elő tudjuk állitani. A C^0 és C^1 szintek közötti teljes oksági kapcsolat csak ugy biztositható, hogy a C^1 szinten elhelyezkedő generátor számára szükséges külső adatokat a C^0 szinten levő tervezési programok, vagy azoknak egy része is képes előállitani./a tervezési szakaszban/. Ennek hatására a generátor /C^1 szint/ uj dialógus adatokat /állapotokat/ hoz létre C^0 szinten. Tervező rendszerekben erre a kapcsolatformára nagy szükség van és meg is valósitható. Ezt nevezzük dinamikus dialógusnak.

A P-C kapcsolat nem is nevezhető teljes értékü beszélgetésnek, mivel a dialógus egyirányu. A programozónak mint individuumnak ebben a kapcsolatban az a feladata, hogy a számitógépet megtanitsa a szükséges feladatok elvégzésére. E munka megvalósitása két szinten történik. Először el kell késziteni azokat a programszegmenseket, amelyek a tervezési feladatokat fogják elvégezni, azaz a C^0 szinten levő eljárások bizonyos részét. E feladat elvégzését a programozó P^0 szintjére ruházzuk.

Másodszor a C^1 szinten levő programgenerátor számára szükséges adatokat kell előállitani. Ezt a P^1 szintü eljárással végeztetjük el. Ezen a szinten folyik a rendszer generálásához szükséges kapcsolatok kialakitása. A P^1 eljárás a P^0 szinten levő programszegmensek dialógus kapcsolatait teremti meg, de a P^0 természetesen befolyást gyakorol P^1 szintre is. Ez azt jelenti, hogy a két szint közti oksági kapcsolat itt is megtalálható. Természetes, hogy oksági kapcsolatokat találunk még a T és M /azaz a modell/, valamint C^0 és M között is.

Az intelligens beszélgetés egyik leglényegesebb feltétele a beszélgető individuumok közti provokativ kapcsolatok megléte, szinvonala, illetve iránya. Vizsgáljuk meg a 12. ábra alapján az interaktiv rendszer ikonjában levő provokativ kapcsolatokat.

A tervező /T/ és a számitógép /C/ "O"-ás szintjei között létezik provokativ kapcsolat. A kapcsolat L_T^0 nyelven valósul meg. Az L_T^0 nyelv szinvonalát a grafikus display által nyujtott input/output lehetőségek szabják meg [152]. A 12/a ábrán eredményvonallal tüntettük fel az "1" szintek közti provokativ kapcsolatot /L_T^1/. Ez a kapcsolat modellünkben nem szerepel. Felmerül a kérdés, hogy a realizált tervező rendszerek igénylik-e ezt a kapcsolatot, azaz szükség van-e rá. Megjegyezzük, hogy ezt a kapcsolatot mindkét irányban megvalósitani még senkinek sem sikerült, tanulórendszerekben egyirányban azonban igen. A 2.2. fejezetben foglalkoztunk azzal, hogy az integrált rendszerek megjelenésével bizonyos specializálódás jött létre. A specializálódott munkamegosztásban a tervező kész rendszert kap, amely különböző szolgáltatásokat nyujt a munkák elvégzéséhez. A tervezőnek közvetlenül nincs lehetősége arra, hogy a rendszert módositsa, változtassa. A módositás, változtatás egy más

szakterülethez tartozik, ábránkban a P individuumhoz /programozóhoz/. Ha megengednénk, hogy a tervező /T/ a változtatásokat elvégezhesse - ez az L_T^1 nyelv létét feltételezné - akkor meg kellene követelni, hogy a rendszertechnikában is járatos legyen. A T individuumnak megengedjük, hogy a rendszer módositásában részt vegyen, de csak közvetett módon. Ezt a módot mutatja a 12/b ábrán feltüntetett ikon. Az ikon a tervező és a programozó mint individuumok között folyó beszélgetést ábrázolja. A beszélgetés tématartománya /$D_p(R)$/ éppen a változtatások témáinak halmaza. Nem ennyire egyszerü a P és C individuumok közötti provokativ kapcsolatok kérdése. A P individuum a kettős feladatát /az emlitettek szerint/ két szinten fogalmazza meg, külön-külön nyelveken irja le és közli a számitógéppel. Az individuum P^0 szinten általános számitógépes algoritmikus nyelven /pl. GESAL* vagy C**/ irja le a felhasználói programszegmenseket, mig P^1 szinten egy célorientált nyelven fogalmazza meg a tervező és a számitógép közti dialógushoz szükséges információkat, valamint a dialógus és programszegmensek közti kapcsolatokat.

---

* GESAL az MTA SZTAKI-ban kidolgozott strukturált programozásra alkalmas általános célu, magas szintü számitógépes nyelv [157].

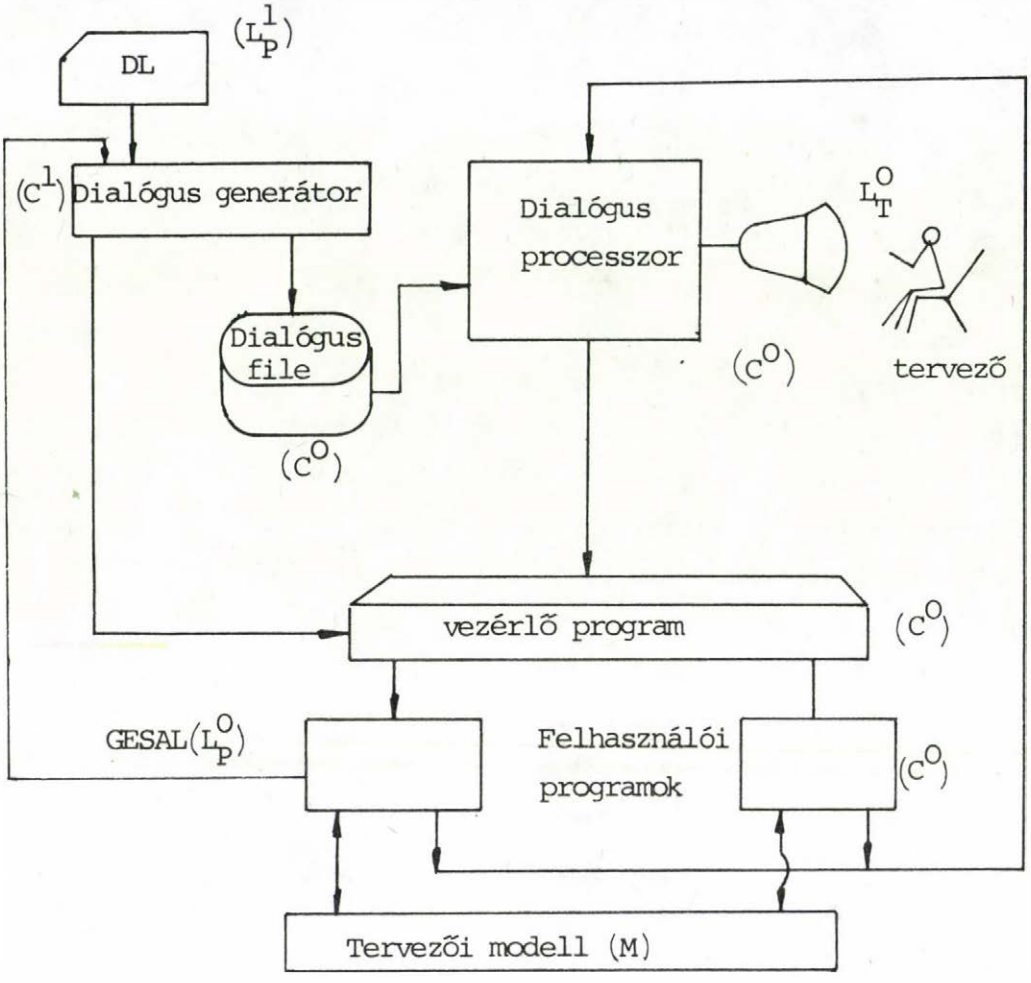
** A Bell Laboratóriumban kidolgozott strukturált programozásra alkalmas nyelv [158].

A 12/a ábrán L_P^O reprezentálja a számitógépes algoritmikus nyelvet /GESAL, C/ és L_P^1 a célorientált dialógus nyelvet /pl. DL. 1. a 6.2. fejezetet/. Mindkét kapcsolat csak egyirányu, igy nem teljesitik a provokativ kapcsolat feltételeit /1. 12. ábrát/. Megjegyezzük, hogy a nem teljes provokativ kapcsolatot teljessé lehetne tenni azzal, hogy az ezeket reprezentáló nyelveken irt programok elkészitéséhez interaktiv rendszereket dolgoznánk ki. Ez a megoldás azonban nem célszerü, és jelenleg nem is volna gazdaságos.

5.2. Interaktiv tervező rendszerek felépitése

Az interaktiv rendszer felépitését /13. ábra/ az előző fejezetben ismertetett modell alapján valósitottuk meg. Az elméleti modell és a blokkvázlat közti kapcsolat könnyen felismerhető. A tervező a display képernyője előtt ül és L_T^O nyelven folytat dialógust a számitógéppel. A programozó P^1 szinten irja a rendszergeneráló programját DL /L_P^1/ nyelven és P^O szinten a felhasználói programokat algoritmikus /GESAL,/L_P^O// nyelven. A számitógép C^1 szintjén van a dialógus generátor, C^O szinten pedig a rendszerhez tartozó dialógus processzor, amely az ember és a gép közti kapcsolatot valósitja meg, a generált vezérlő program, a dialógus adatok, valamint a programozó által készitett felhasználói programszegmensek.

A rendszer teljes müködése két szakaszra bomlik. Az első szakaszban történik a célorientált tervező rendszer megtervezése, előállitása, azaz a programok összeszerkesztése. Erről a szakaszról a későbbi fejezetekben részletesen lesz szó, itt csak összefoglaljuk,



13. ábra

hogy milyen tevékenységek tartoznak ide:

1. A rendszer létrehozásához szükséges dialógus program elkészitése DL nyelven /vezérlő program előállitásához és a dialógus adat-file feltöltéséhez/.
2. A felhasználói programszegmensek elkészitése.
3. A felhasználói programszegemnsek közötti kapcsolatokat megteremtő tervezési modell strukturájának elkészitése.

A tervező rendszer müködésének második szakaszában történik a tervező munka. A tervezési folyamat, adott fázisban, a dialógus file-ban elhelyezett adatok alapján a processzor

kéri be a tervezőtől származó információkat, paramétereket. A vezérlő program gondoskodik arról, hogy arra a felhasználói programra kerüljön a rendszer futása, amelynek adatait a dialógus processzor bekérte. A kiválasztott felhasználói program egyrészt a tervezői modellt, másrészt a tervezőtől származó adatokat felhasználva ujabb adatokat állit elő és elhelyezi azokat a tervezői modell megfelelő részébe. A felhasználói program lefutása után a vezérlés ujra a dialógus processzorra kerül és a ciklus ismétlődik addig, amig a tervező által elképzelt terv elkészül.

6. CÉLORIENTÁLT INTERAKTIV TERVEZŐ RENDSZEREK ÉS ELEMEIK

A bevezetőben emlitettük, hogy a tanulmány célja olyan módszerek és eszközök létrehozása, amelyek segitségével célorientált interaktiv tervező rendszereket lehet létrehozni. Az ISTER-ben megvalósitott rendszer-felépitésének koncepciója, valamint a megvalósitással szerzett tapasztalatok tették lehetővé, hogy az interaktiv tervező rendszerek létrehozásának és használatának elméleti megalapozását elvégezzük /5.1. fejezet/. Az elméleti modell alapján az 5.2. fejezetben az interaktiv tervező rendszerek rendszertechnikai felépitésére javaslatot tettünk. Ez a javaslat koncepciójában azonos az ISTER-ben alkalmazott felépitéssel, de annál jóval fejlettebb.

Ez a fejezet a javasolt rendszer /13. ábra/ elemeit definiálja és az egyes elemek kapcsolatait fogalmazza meg.

Ahhoz, hogy a javasolt általános rendszert, illetve épitőelemeit használni tudjuk, az elérendő célt, azaz a célorientált tervező rendszerrel megvalósitandó tervezési folyamatot meg kell tudnunk egzakt módon fogalmazni. A 6.1.1. fejezetben a gráfelmélet fogalmainak felhasználásával olyan uj eljárást ismertetünk, amely a tervezési folyamatok ábrázolására, illetve leirására szolgál. A meghatározott szabályokkal rendelkező dialógus gráf tartalmazza a folyamatban részt vevő nem algoritmizálható /dialógus/ és az algoritmizálható /programszegmensek/ tevékenységeket és azok kapcsolatait. Az ember és a számitógépes rendszer kapcsolatának megfogalmazásához egyrészt ismernünk kell az interaktiv periféria által nyujtott és az azt kibővitő grafikus lehetőségeket, valamint a dialógus fogalmakat. A grafikus lehetőségeket a 6.1.2. fejezetben foglaljuk össze és a dolgozatban definiált dialógus alapfogalmakat a 6.1.3. fejezetben irjuk le. A dialógus gráf

és az alapfogalmak segitségével megfogalmazott tervezési folyamat leirása a 6.2. fejezetben ismertetett dialógus nyelvet /DL nyelv/ dolgoztuk ki. A DL nyelven leirt program információtartalmából előállitható a célorientált tervező rendszer müködéséhez szükséges dialógus adat-file /1. a 13. ábrát/ tartalma, valamint a rendszer vezérlésére szolgáló vezérlő program /13. ábra/. Az implementált dialógus generátort, amely a DL nyelv fordító programja és egyben előállitja a dialógus adatokat, valamint a vezérlő programot, a 6.3. fejezetben ismertetjük. A DL nyelv alkalmazására, a dialógus adatok és a vezérlő program automatikus előállitására, azaz az implementált dialógus generátor müködésére, egy példát mutat a [163] és a [164] irodalom. A dialógus processzortól /1. a 13. ábrát/ megkivánt követelményeket a 6.4. fejezetben foglaltuk össze. A processzor által megvalósitott funkciók részletes leirását a [146] irodalom mutatja be. A [146] irodalom az ISTER-ben megvalósitott dialógus processzort tartalmazza, de ez csak kis mértékben tér el a 6.4. fejezetben összefoglalt követelményektől /az eltérések az interface felületen jelentkeznek/. A 6.5. fejezetben azokat a követelményeket foglaltuk össze, amelyeket a felhasználói programok /1. a 13. ábrát/ elkészitésénél be kell tartani, hogy a célorientált tervező rendszerekbe beépithetők legyenek.

6.1. Alapfogalmak

6.1.1. A dialógus gráf elemei és tulajdonságai

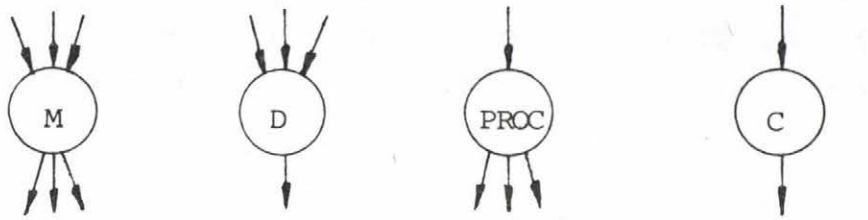
A dialógus gráf a célorientált tervező rendszerekben végbemenő folyamatok interakcióit és a felhasznált programszegmenseket, valamint köztük levő kapcsolatokat tartalmazza.

Egy interaktiv tervező rendszerben, ahol az ember és a számitógépes rendszer közvetlen kapcsolata megvalósul, a tervezési folyamatban szereplő algoritmizálható és nem algoritmizálható tevékenységek /l. 2.3.2. fejezetet/ felváltva kapcsolódnak egymáshoz. A nem algoritmizálható tevékenységeket a tervező dialógus formában oldja meg, mig az algoritmizálható tevékenységeket programszegmensek /PROC-ok/ hajtják végre. A dialógus részdialógusokból tevődik össze, amelyeknek két tipusát különböztetjük meg, attól függően, hogy milyen szerepet töltenek be a tervezési folyamatban. Ennek megfelelően a tervező a menü tipusu részdialógusokkal a tervezési folyamatban elágazásokat tud megvalósitani és az adatbekérő részdialógusokkal adatokat képes megadni a soron következő futó programszegmens /PROC/ számára. A rendszer futása közben szüksége lehet különböző üzeneteket /commenteket/ közölni a tervezővel. Ezek az üzenetek lehetnek tervezési információk, hibajelzések stb., amelyek a programszegmensek futásának eredményeképpen jönnek létre.

A felsorolt tulajdonságokkal rendelkező tervezési folyamat elemeit a gráfelmélet felhasználásával csomópontoknak nevezzük és irányitott gráfban meghatározott szabályoknak eleget tevő kapcsolatokat definiálunk köztük.

Összefoglaljuk a tervezési folyamatot ábrázoló un. dialógus gráf elemeit és tulajdonságait /l. a 14. ábrát/.

1. Menü tipusu részdialógus vagy csomópont /14/a ábra/. A tervezési folyamatban a tervező ezeken a részdialógusokon keresztül képes befolyásolni a tervezés menetét. A csomópont tetszőleges számu bemenettel és kimenettel /élekkel/ rendelkezhet.



menü	adatbekérés /Dialógus/	Program szegmens /PROC/	Üzenet /COMMENT/
a.	b.	c.	d.

14. ábra

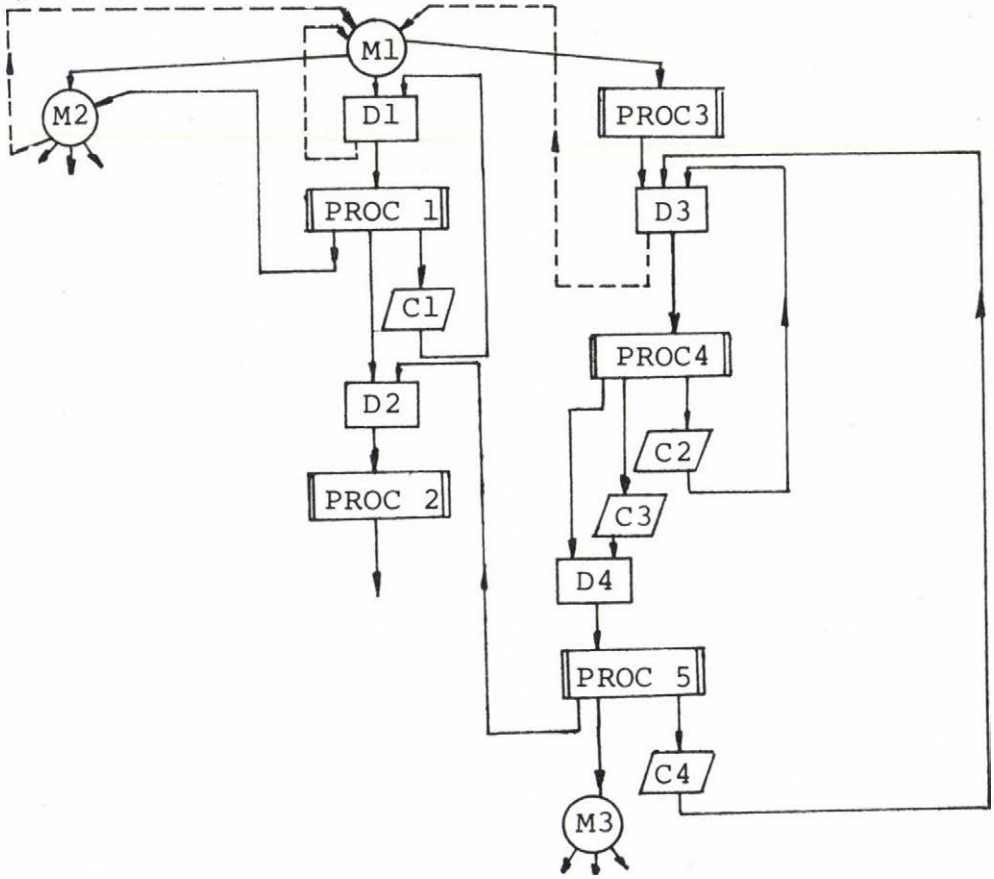
2. Adatbekérő tipusu részdialógus vagy csomópont /14/b ábra/. A tervezőtől származó adatok bekérésére és megadására szolgál. Több bemenettel, de csak egy kimenettel rendelkezhet.
3. Algoritmizálható tevékenységeket /PROC/ reprezentáló csomópont /14/c ábra/. Egy bemenettel, de tetszőleges számu kimenettel rendelkezhet.
4. Tervezőnek szánt üzenet /COMMENT/ csomópont /14/d ábra/. E csomópont csak egy bemenettel és egy kimenettel rendelkezik.

A csomópontok közötti kapcsolatokra szabályokat definiáltunk.

a/ Egy menü tipusu csomópont /14/a ábra/ kimenő élei bármely tipusu csomóponthoz csatlakozhat, kivéve az üzenet tipusut /14/d ábra/.
b/ Egy adatbekérő tipusu csomópont /14/b ábra/ egyetlen kimenő éle csak egy programszegmens /PROC/ tipusu /14/c ábra/ csomópontjához csatlakozhat.
c/ Egy programszegmens /PROC/ tipusu csomópont /14/c ábra/ bármely csomóponthoz csatlakozhat.
d/ Egy üzenet tipusu csomópont /14/d ábra/ vagy egy menü /14/a ábra/, vagy egy adatbekérő /14/b ábra/ tipusu csomóponthoz csatlakozhat.

A 15. ábrán egy példát mutatunk be a dialógus gráf kezelésére. Az ábra alapján néhány olyan további tulajdonságot foglalunk össze, amelyek a tervezési folyamatok tervezésében megkönnyitik a gráf alkalmazását.

1. A menü tipusu csomópont a tervezőtől függő tervezési folyamat elágaztatására szolgál. Lehetőséget ad arra, hogy a tervező választással különböző tervezési egységeket aktivizáljon.
2. Egy adatbekérő részdialógus mindig egy programszegmens előtt áll.
3. Egy üzenet vagy COMMENT mindig egy programszegmens /PROC/ kimeneteként jöhet létre. Az üzenet kiirás után a tervező rendszer valamilyen ujabb beavatkozást vár, ezért valamelyik részdialógushoz kell hogy csatlakozzon.



15. ábra

4. A programszegmens /PROC/ annak bemenetét tekintve kétféle lehet.
   a/ Nem kiván a tervezőtől adatokat. Ekkor egyszerü menüből való kiválasztással aktivizálható /pl. 15. ábrában PROC3/.
   b/ A PROC futásához a tervezőtől adatok szükségesek. Ebben az esetben az adatbekérő részdialógus előzi meg /pl. 15. ábrában D3, PROC4/.

A programszegmens kimenetei két csoportot alkotnak.

a/ üzenet nélküli kimenetek /részdialógusokhoz csatlakoznak/;

b/ üzenettel rendelkező kimenetek /üzenet csomópontokhoz csatlakoznak/.

Mindkét tipushoz tartozó kimenetek a program belsejében lévő feltételektől függően több is lehet, de mindkét tipusu kimenetek közül a szegmens lefutása után csak egy aktivizálódhat.

5. A 15. ábrán szaggatott vonallal jelöltük az interaktiv rendszereknél feltétlenül szükséges, un. VISSZA funkciókat. Ez a rendszernek az a szolgáltatása, amely lehetővé teszi a tervezőnek, hogy a gráfon ne csak előre haladhasson, hanem visszafelé is. Hibás vagy téves menüelem kiválasztását, vagy adatmegadást a tervező ezzel a rendszerszolgáltatással tudja korrigálni. A visszalépés funkció paramétereit abban a részdialógusban definiáljuk, ahonnan a visszaugrás történik.
6. Szemben a 15. ábrán felvázolt dialógus gráfrésszel a tervezési folyamatot ábrázoló teljes gráfnak mindig zártnak kell lenni. Ez azt jelenti, hogy a gráf elemeit mindig valamilyen más elemhez csatlakoztatni kell. /A dialógusgráf a folyamatábra fogalmának egy magasabb szintü általánositásaként is felfogható; a folyamatábrák zártságára vonatkozó követelmény pedig triviális./

A felsorolt elemek és tulajdonságaik alapját képezték a kialakitott DL nyelvnek.

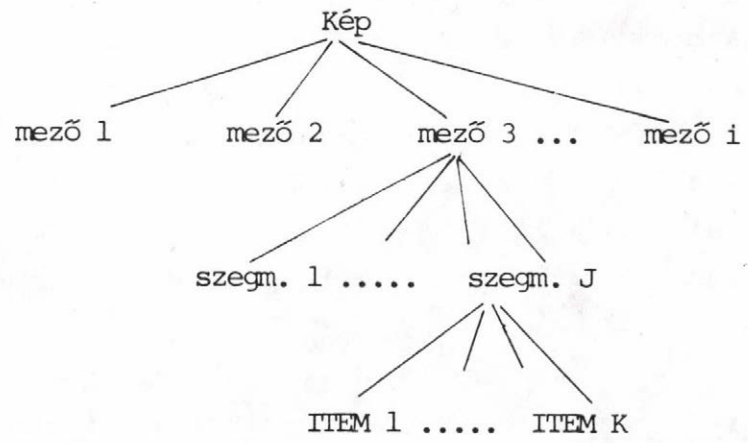
6.1.2. Grafikus alapfogalmak

Egy interaktiv tervező rendszerben az alkalmazott grafika attól is függ, hogy a rendszer milyen interaktiv perifériát használ. A ma használható legmagasabb szintü periféria a grafikus display. Rendszereinket az MTA SZTAKI-ban kidolgozott GD'71 és GD'80 grafikus display család tagjaira [159], valamint az azokhoz kifejlesztett GTU [160] és GSS'80 grafikus alapsoftware-jeire [56] alapozzuk. Ebben a fejezetben csak azokat a grafikus fogalmakat foglaljuk össze, amelyeket a tanulmány a tervező rendszerek által is igényelt követelmények kielégitésére javasol. Az uj fogalmak az általános grafikus rendszerek /GSS'80/ fogalmaira épülnek és kibővitik azokat.

A GSS'80 alapsoftware lehetővé teszi, hogy a GD'80 szolgáltatásait általános szinten, azaz felhasználástól függetlenül, eljárások hivásával, programok, programrendszerek készitésénél felhasználhassuk. Az interaktiv tervező rendszerek követelményei, amelyek már a GSS'80 szintjéhez viszonyitva speciálisaknak számitanak, megkivánják, hogy az alapsoftware-re épülve magasabb szintü, általános rendszert alakitsunk ki. A továbbiakban a tanulmány által javasolt általánosabb érvényü grafikus rendszerhez tartozó fogalmakat foglaljuk össze.

A display képernyőn megjelenithető grafikus információkat különböző szintü csoportokba rendezve ábrázoljuk. A csoportok közti kapcsolatokat több szintü fastrukturával irjuk le. A fastruktura négy hierarchiaszintet

tartalmaz, mégpedig: kép, mező, szegmens, item /lásd a 16. ábrát/. /A szegmens és item a GSS'80-hoz tartozó fogalmak./ Az egyes szinteken levő fogalmak a következők.



16. ábra

1. Kép. A display teljes területét feloszthatjuk több részre. Ezek a részek egymáshoz képest eltolva, vagy egymást átfedve helyezkedhetnek el. Egy meghatározott felosztás valamilyen elrendezést reprezentál, és ezt képnek nevezzük. Egy rendszeren belül több kép definiálására adunk lehetőséget és mindegyikhez külön nevet rendelünk. Munka közben bármely képet a nevén keresztül aktivizálhatunk. Ez azt jelenti, hogy a régi képet elengedjük és az uj képet jelenitjük meg a display képernyőn.

2. Mező. Az 1. pontban tárgyaltak szerint egy "kép" több részből állhat. Egy-egy ilyen részt nevezünk "mező"-nek. Egy tetszőleges mező felhasználás szempontjából önálló display perifériaként viselkedik. A mezők tulajdonságait az alábbiakban foglaljuk össze.

   a/ A mezőket névvel látjuk el. Munka közben mindig az a mező lesz "élő" a képernyőn, amelyet név szerinti hivással utolsóként aktivizáltunk. Az egyes mezők felfoghatók ugy, mintha önálló perifériák lennének.

   b/ Két tipusu mezőt különböztetünk meg. Ezek:

   - Alfanumerikus mező, amely ugy viselkedik, mint egy önálló alfanumerikus display. A mező definiálására a négyszögkeret bal alsó pontjának koordinátáival /display koordináta-rendszerben/, valamint a keretben lehelyezhető sorok és oszlopok számával történik.
   - Grafikus mező, amely téglalap alaku grafikus terület. Definiálása két sarokpontjának /bal alsó, jobb felső/ koordinátáival /display koordináta-rendszerben/ történik. Ezt a területet a GSS'80 "view-port -nak nevezi. Szükséges továbbá minden mezőhöz un. "window"-t /ablakot/ rendelni. A "window" a leképzendő rajzterületen az a keret, amelyet a mezőben meg akarunk jeleniteni. Paraméterei a keret bal alsó és jobb felső sarokpontjainak koordinátái a rajz koordináta-rendszerében.

3. Szegmensek. Egy adott mezőben levő összetett grafikus információk képi szegmensekben helyezkednek el. Minden létrehozott szegmenshez nevet kell rendelni. A szegmensekre való bontás nemcsak a grafikus információk logikus szétválasztása miatt szükséges, hanem

bizonyos GSS'80 alap-software szolgáltatások csak megnyitott szegmensekre müködnek. /Villogtatás, fényceruza-élesités, képi információk törlése stb./

4. Item. A szegmensben elhelyezett grafikus információk szétválasztására lehetőséget nyujtó GSS'80 szolgáltatás. Az ITEM segitségével grafikus utasitások közé tetszőleges információ épithető be. Fényceruza azonositással ezt az információt kapjuk vissza a szegmens névvel együtt.

6.1.3. Dialógus alapfogalmak

A dialógus rendszer egzakt megfogalmazásához a tanulmány a következőkben tárgyalt dialógus alapfogalmakat vezeti be.

Az interaktiv rendszerben dialógus alakul ki az ember és a számitógép között. A tervező beavatkozása részdialógusokon keresztül történik. A részdialógusok akciókból épülnek fe. Az akciók többfélék lehetnek, attól függően, hogy milyen szerepet töltenek be. A részdialógus funkciója egy menüág kiválasztása, vagy egy felhasználói programszegmenshez /PROC/ szükséges adatok megadása. Betöltött szerepüktől függően a részdialógusoknak két tipusát különböztetjük meg.

1. Menü, azaz "J" /JUMP/ tipusu részdialógus. Ez a részdialógus szolgál arra, hogy a gráfban levő csomópontokat /elágazásokat/ segitségével leirjuk. A benne szereplő akciók a J tipusu akciók, amelyek a menü elágazásaihoz szükséges információkat tartalmazzák. Egy "J" tipusu részdialógusban tetszőleges véges számu "J" tipusu akció szerepelhet.

2. Adatbekérő részdialógus. Programszegmensekhez paraméterek megadására szolgál. Az adatok megadása kérdések /QUESTION/ és válaszok /ANSWER/ egymásutáni sorozatából áll. Minden kérdéshez egy válasz tartozik, tehát ezek összetartozó akciók, ezért az adatbekérő részdialógust QA részdialógusnak is nevezzük. A kérdéseknek és válaszoknak több tipusa van /l. a 6.2.3. fejezetet/.

6.2. DL: Dialógus nyelv

Ebben a fejezetben az elméleti modellben /12. ábra/ szereplő L_p^1, vagy másképpen a 13. ábrán szereplő DL nyelvet ismertetjük.

A DL /Dialogue Language/ nyelv és az e nyelven irt program kettős szerepet tölt be. Az egyik az, hogy az előre megtervezett részdialógusok információit megadhassuk, valamint a gráfban szereplő részdialógus és a különféle PROC-ok közti kapcsolatokat leirhassuk. A másik pedig az, hogy deklaráljuk azokat az adatokat /grafika stb./, amelyek a célorientált rendszer összeállitásához szükségesek.

A DL nyelv egy célorientált nyelv, amely szabad formátummal, de kötött szintaxissal rendelkezik.

6.2.1. A DL nyelven irt program elemei és felépitése

A DL nyelven irt program utasitásokból áll. A használható utasitások kétfélék lehetnek:

1. azonositókkal rendelkezők, azaz

   Azonositó = FŐSZÓ/.....

alaku utasitások. Ezekre az utasitásokra más utasitásokban azonositójukon keresztül hivatkozhatunk;

2. azonositó nélküliek, azaz

   FŐSZÓ/....

   alaku utasitások. Hivatkozni rájuk más utasitásokban nem lehet.

Az utasitások /rekordok/ a következő egységeket tartalmazhatják:

1. Főszavak. Minden rekordhoz tartozik egy főszó, amely az utasitás tipusát adja meg /l. 6.2.2. fejezetben/. A főszavak kötött szavak.
2. Módositók. Kötött szó, amely a főszóhoz tartozó rekord értelmét teszi egyértelmüvé /l. 6.2.2. fejezetet/.
3. Azonositó. A programkészitő által adott az angol ábécé maximum 6 karakterből álló string. Utasitások megnevezésére szolgál.
4. Konstansok. Három tipusát különböztetjük meg: integer, real, karakter jellegü string.
5. Speciális karakterek /elválasztó jelek, végjel/.
   a/ "=" az azonositóval rendelkező rekordban az azonositót választja el a főszótól. Pl.:

   ALMA = TEXT/....

   b/ "/" a rekordban a főszót választja el az utasitásban szereplő egyéb információktól.
   c/ "," az egyes információkat elválasztó jel.
   d/ ";" a rekord végét jelző irásjel.

A DL nyelvben levő utasitásokat négy csoportba soroljuk. Ezek a következők:

1. Beállitó /inicializáló/ utasitások. Ebbe a csoportba tartozó utasitásokkal a grafika beállitásához szükséges információkat adjuk meg /pl. kép, mező, szegmens stb./
2. Deklarációs utasitások. E csoportba tartozó utasitásokkal adjuk meg a létrehozandó célorientált rendszerhez tartozó összes felhasználói programokat /PROC/, üzenetek /COMMENT/ információit /l. 6.1.1. fejezet/ és a dialógus programban gyakran előforduló azonos szövegeket. Ezekre a szövegekre a dialógus programban szereplő rekordokban csak hivatkozni kell.
3. Részdialógusok utasitásai. A részdialógusok definiálása, valamint a részdialógusokhoz tartozó akciók megadása e csoporthoz tartozó utasitásokkal történik.
4. Programot befejező utasitás.

6.2.2. A DL nyelv leirásában használt jelölések összefoglalása, értelmezése

1. Főszavak

a/ Azonositóval rendelekező utasitások főszavai

AAREA	alfanumerikus terület
DPROC	felhasználói program definiálása
ERR	hiba /comment/
GAREA	grafikus terület
PART	részdialógus
PICTUR	kép
TEXT	szöveg

b/ Azonositóval nem rendelkező főszavak

ANSW	válasz
CALL	procedure hivás
FBOTT	funkcionális tasztatura
FINI	program vége

GRMACR	grafikus menüelem makrója
JUMP	menüelem
QUEST	kérdés
SEGMCR	szegmens definiálása
TRAP	részdialógus vége

2. Módositók

AREA	alfanumerikus területet jelöl
ATEXT	szöveges válasz
BALL	pozicionáló gömb
CHTYP	karaktertipust jelöl
DIAL	részdialógust jelöl
ERROR	hibakiirást jelöl
FI	IF-FI zárójelek közül végső zárójel
FROM	alfanumerikus terület bal alsó koordinátát jelöl
GPIC	grafikus menüelem
GREA	grafikus terület
IF	IF-FI zárójelek közül a kezdő zárójel
ITEM	grafikus információ
JFUNC	JUMP funkció a funkcionális tasztaturán
LOCAT	potenciométeres lokátor
NRFUNC	No Return funkció a funkcionális tasztatura utasitásban
OFF	kikapcsolás
OFUNC	OUT funkció a funkcionális tasztaturán
ON	bekapcsolás
POZ	pozicionálás
PROC	procedure megjelölés
PTEXT	szövegtábla mutató
RFUNC	return funkció a funkcionális tasztaturán
SEGM	szegmensmegnevezés
SFUNC	switch funkció a funkcionális tasztaturán
SIZE	méret
VIEW	view-port } grafika
WIND	window } grafika

3. Konstansok és változók

$azon_a$	alfanumerikus terület azonositó
$azon_e$	hibaüzenet azonositó
$azon_g$	grafikus terület azonositó
$azon_K$	kép azonositó
$azon_p$	részdialógus /part/ azonositó
$azon_{pr}$	procedure azonositó
$azon_T$	szövegtáblában szöveg azonositó
comf	comment /hiba/ feltétel
item	item értéke
lsz	lokátor potenciométerének sorszáma
macrn	grafikus menü makrójának neve
n_i	funkcionális tasztatura gomb száma
O	oszlopszám
Osz	oszlopok száma
procf	procedure kimenet feltétel
procn	külső procedure neve
s	sorszám /alfanumerikus területen/
segmn	szegmensnév
ssz	sorok száma
szöveg	szöveg
x_i, y_i	raszter koordináták /display/ $0 \leq x \leq 1$ $0 \leq y \leq 1$
U_i, v_i	koordináták a felhasználói sikon

4. Jelölések a szintaxis leirásához.

a/ "|" a rekordban előforduló alternativákat választja el. Nem zárja ki azt a lehetőséget, hogy egy rekordban mindkét alternativa szerepeljen, ha azok egyébként nem zárják ki egymást.

b/ "< >" olyan egységeket zárnak közre, amelyek tetszőleges számban fordulhatnak elő egy rekordban.

c/ "( )" összetartozó egységeket zárnak közre.

d/ "[ ]" Olyan utasitásokat zárnak közre, amelyek megadása lehetséges, de nem kötelező.

6.2.3. Az utasitások szintaxisának leirása

1. Beállitó utasitások

a/ Mezők

alfanumerikus terület /mező/

$azon_a$ = AAREA/FROM, x, y, SIZE, ssz, osz
[, CHTYP, a, b, c, d];

- grafikus terület /mező/

$azon_g$ = GAREA/VIEW, x_1, y_1, x_2, y_2
[, WIND, u_1, v_1, u_2, v_2];

Ha WIND nincs programozva a rendszer kezdőértékként WIND-nek a VIEW értékeit adja. Egy programon belül mindkét tipusu mezőből több is szerepelhet.

b/ Szegmensek felsorolása

Csak azokat a szegmenseket kell felsorolni, amelyek a fényceruzával történő válaszadásnál különös jelentőséggel rendelkeznek.

SEGMCR/GREA, $azon_g$, segmn <, segmn>
[<(,GREA, $azon_g$, segmn <, semgn>)>];

c/ Grafikus menüelemek definiálása

GRMACR/macrn <, macrn>;

d/ Kép definiálása
több mező összerendelése

$azon_k$ = PICTUR/$azon_g$|$azon_a$ <, $azon_g$|$azon_a$>;

e/ Funcionális tasztatura gombjainak beállitása.

A nyomógombokhoz rendelhető funkciókat tulajdonságaik szerint három csoportba sorolhatjuk.

- SFUN /Switch Function/. Bizonyos funkciókhoz két nyomógombot rendelünk. Az egyik gomb aktivizál /ON/, a másik letiltja annak hatását /OFF/. /Pl. képkikapcsolás, bekapcsolás./
- JFUNC /Jump Function/. A dialógus gráf kiemelt pontjaira, vagy a gráfon nem szereplő programszegmensre /OFUNC/ való ugrás ugy, hogy vissza lehessen térni arra a pontra, ahonnan a kiugrás történt. Minden kiugráshoz külön-külön egy-egy nyomógombot rendelünk. Bármely helyről történő visszaugrás egy közös gomb aktivizálásával érhető el, azaz a RFUNC-al /Return Function/,
- NRFUNC /No Return Function/. Vissza nem állitható funckiók aktivizálása. /Pl. visszalépés a gráfban, programmentés stb./ Minden funkcióhoz egy-egy nyomógomb tartozik.

```
FBOTT/[SFUNC,ON,n_i,OFF,n_j<,ON,n_i,OFF,n_j>,]
      [(JFUNC,n_k,azon_p<,n_k,azon_p>,)|
      (OFUNC,n_L,procn<,n_L,procn>,)RFUNC,n_m,]
       NRFUNC,n_n,proc_n<,n_n,proc_n>;
```

2. Deklarációs utasitások

a/ Közös szövegek definiálása

```
azon_T=TEXT/szöveg;
```

b/ Comment szövegek kiiratása

```
azon_e=ERR/AREA, azon_a ,<POZ,S,O,
     (ATEXT,szöveg)|(PTEXT,azon_T)>,
     DIAL, azon_p;
```

Csak alfanumerikus területre irható comment, vagy hibaüzenet.

c/ Procedure /felhasználói program/ definiálása

```
azon_pr=DPROC/DIAL(,azon_p)|(,IF,procf,azon_p<;
     prof,azon_p>,FI)
     [,ERROR(,azon_e)|(,IF,comf,azon_e<,
     comf,azon_e>,FI)];
```

Hiba nélküli kimenet esetén ha egy kimenete van a definiált procedure-nak, a DIAL módositót használjuk; ha több kimenettel rendelkezik, az IF-FI zárójelek közé irt párokat adjuk meg.

Hiba kimenet esetén, a fentihez hasonlóan - értelemszerüen - kell megadni az ERROR vagy az IF-FI zárójelekkel megadható információkat.

3. A részdialógus utasitásai

A részdialógushoz tartozó utasitásokat két csoportra osztjuk. Az első csoportba tartoznak a részdialógusokat megnyitó és lezáró utasitások, a második cso-

portba a részdialógus belsejében előforduló akciók rekordjai.

a/ Részdialógus megnyitása

```
azon_p=PART/azon_p [,szöveg];
```

Az utasitás belsejében szereplő "azon$_p$" jelenti annak a dialógusnak az azonositóját, ahová a visszalépés történhet /ld. a 15. ábrán lévő gráfot/

b/ Részdialógus lezárása

```
TRAP;
```

c/ Menü tipusu utasitás

```
JUMP/(AREA,azon_a,<POZ,S,O,(ATEXT,szöveg,)|(PTEXT,
     azon_T,)>)|(GREA,azon_g,<POZ,U,V,GPIC,macrn,>)
     (DIAL,azon_p)|(PROC,azon_pr);
```

Megjegyzés:
egy részdialóguson belül, vagy alfanumerikus, vagy grafikus menüt használhatunk. Egyszerre a kettő programozása hibakiirást eredményez.

d. Kérdés akció utasitása

```
QVEST/(AREA,azon_a<,POZ,S,O,(ATEXT,szöveg)|
      (PTEXT,azon_T) [,ITEM,item] >)|
      (GREA,azon_G<,POZ,U,V,GPIC,macrn
      [,ITEM,item] >);
```

Megjegyzés:

1. JUMP-nál mondottak itt is érvényesek.
2. Ha a kérdésre adandó válasz módja fényceruza, ITEM információt kell rendelni a kérdéselemhez.

e/ Válasz akció utasitása

A válaszadás módjai a következők lehetnek:

1. Alfanumerkus válasz. Két paraméter tartozik hozzá
   - alfanumerikus terület neve
   - a válasz kezdő karakterének helye /S,O/
2. Fényceruza rámutatás. Két lehetőséget engedünk meg:
   - grafikus területhez tartozó összes szegmenst fényceruzára érzékennyé tesszük
   - csak meghatározott szegmenseket teszünk fényceruzára érzékennyé.
3. Lokátor válaszolási módhoz meg kell adni, hogy mely potenciométerek beállitását várjuk.
4. Pozicionáló gömbbel meghatározott koordináták beolvasása az adott grafikus területen

```
ANSW/('AREA,azon_a,POZ,S,O,)|
     (<GREA,azon_g,[<SEGM,segmn>]>)|
     (LOCAT<,lsz>)|(BALL,GREA,azon_g);
```

f/ Procedure /program szegmens/ hivása

```
CALL/azon_pr
```

4. Dialógus program vége

```
FINI;
```

5. A DL nyelven irt programban a következő szabályokat kell betartani:

   a/ Két vagy több egyforma azonositó egy dialógus programban nem lehet,

b/ a rekordokban csak olyan azonositókra hivatkozhatunk, amelyek a programon belül definiáltak /utólag definiált azonositás megengedett/,

c/ egy részdialógus leirása mindig PART utasitással kezdődik és TRAP utasitással zárul,

d/ menü tipusu részdialógusokban csak JUMP utasitások fordulhatnak elő.

e/ adatbekérő részdialógusokban a QUEST-ANSW utasitáspár többször is előfordulhat. A CALL utasitás egyszer és csak is egyszer szerepelhet. Ebben a tipusu részdialógusban JUMP utasitás nem fordulhat elő.

6.3. Dialógus generátor

A dialógus generátor /ld. 13. ábrát/ a DL /Dialogue Language/ nyelven irt dialógus program rekordjait szekvenciálisan dolgozza fel. A rekordokban lévő információkból egyrészt a dialógus adatbázist tölti fel adatokkal, másrészt a tervező rendszer vezérlő programját generálja.

A dialógus adatbázis /dialógus file/ megfogalmazásához az MTA SZTAKI-ban kidolgozott adatbázis kezelő rendszer adatleiró nyelvét /DDL, Date Description Language/ [61] használtuk fel. A DDL nyelv segitségével adattipusokat, az adatok összerendelésével strukturált rekordtipusokat és a rekordtipusok összerendelésével halmaztipusokat definiálhatunk. Az ilymódon definiált adatbázisnak adatokkal való feltöltése azt jelenti, hogy a rekord- és a halmaz tipusokra előfordulásokat hozunk létre.

A program-generátor a megtervezett és definiált strukturált adattipusokra előfordulásokat állit elő. A dialógus file rekord-és halmaztipusait a [163] irodalom mutatja be /DDL nyelven/. Ezeket a strukturákat a célorientált rendszer tervezőjének nem kell ismernie, hiszen ez a dialógus rendszer belső ügye. A generált adatok a dialógus processzor vezérlésére szolgálnak és az összeszerkesztendő célorientált rendszer elinditásához a kezdeti állapotok beállitásához szükségesek.

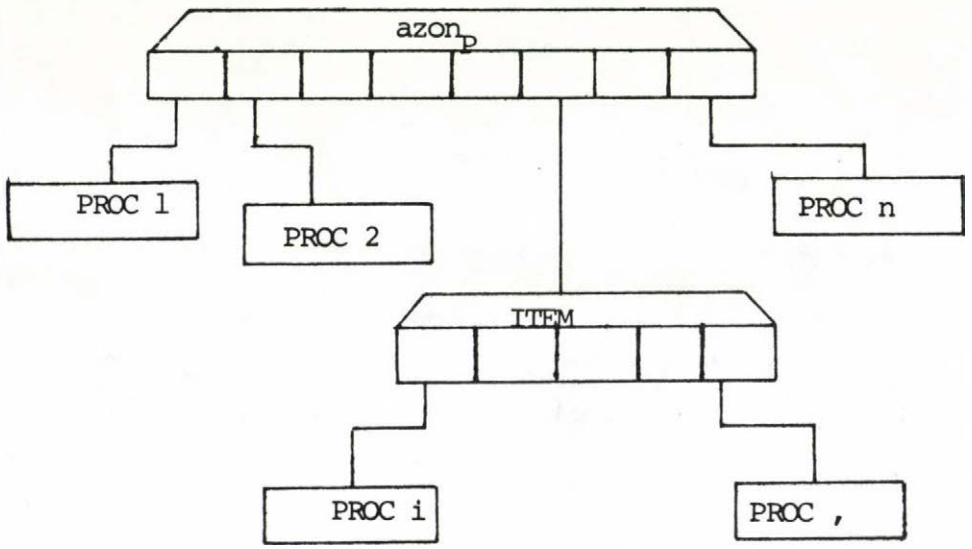
A dialógus generátor előállitja még az interaktiv rendszer vezérlő programját, algoritmikus számitógépes nyelven /pl. GESAL, C./. A vezérlőprogram előállitásához a következő szabályokat használtuk fel:

1. A dialógus processzorral rendelkező interaktiv rendszerekben a felhasználói programok /PROC/ egymáshoz képest párhuzamosan helyezkednek el /ld. a 13. ábrát/.
2. Q-A tipusu részdialógushoz mindig csak egy felhasználói program /PROC/ tartozik.
3. JUMP tipusu részdialógushoz több /maximálisan annyi, ahány eleme van a menünek/ felhasználói program /PROC/ rendelhető. A menüben minden menüágat, amelyhez PROC tartozik, automatikus módon, egy közbenső információval jelöljük meg. Ezzel válnak azonosithatóvá a menü tipusu részdialógus egyes ágaihoz rendelt felhasználói programok.

Mivel minden részdialógus egy névvel /azonositóval/ rendelkezik, egy QA részdialógus azonositója meghatározza a hozzátartozó felhasználói programot. A JUMP tipusu részdialógus esetén a részdialógus azonositó, valamint a menüághoz rendelt közbenső információ /ITEM/ határozza meg a PROC-ot. A 17. ábrán szemléltetjük a felhasználói

programok elérési módjait. Az ábrán feltüntetett elágazások például a GESAL nyelv CASE utasitásaival leirhatók.

A generátor által létrehozott vezérlő program két részből áll: [164]



17. ábra

1. a DL nyelven irt dialógus-programban szereplő PROC-ak felsorolása EXT utasitásokban;
2. a vezérlő program, amely a CASE utasitások sorozatából áll.

A dialógus generátor első változatát TPA'70 számitógépen GESAL nyelven implementáltuk. Az implementált generátor futási eredményei a [164] irodalomban találhatók meg.

6.4. A dialógus processzor

Egy interaktiv tervező rendszerben a dialógus processzor helyét a 13. ábra mutatja. A dialógus processzor az interaktiv rendszerek müködésében általános célu feladatokat old meg, igy általánosan megfogalmazható programszegmensnek tekinthető, azaz tetszőlegesen összeállitott célorientált tervező rendszerekben használható.

A dialógus file-ban lévő adatok vezérlésével a dialógus processzor hajtja végre az ember és a számitógépes rendszer közti dialógust, valamint a rendszer elinditásához szükséges kezdő állapotok beállitását. Részletesebben kifejtve az alábbi legfontosabb feladatokat kell elvégeznie.

1. A grafikus rendszert alapállapotba kell helyezni, azaz a DL nyelvü dialógus programban definiált képeket, grafikus mezőket létre kell hoznia. Ez lehetővé teszi, hogy a dialógus lefolytatásához a grafika müködőképes legyen, valamint a felhasználói programokban lévő grafikus információk kapcsolódhassanak a rendszerhez.
2. A DL nyelvü dialógus programban előirt **kérdés/válasz** /QUESTION/ANSWER/ funkciók végrehajtása. A display képernyő megfelelő helyére ki kell irnia a kér-

déseket, üzeneteket /output funkció/, a programozott válaszokat be kell olvasnia /input funkció/ és azokat értékelnie kell.

3. A beolvasott adatokat értékelésük után megfelelő struktura változókon keresztül át kell adnia a felhasználói programoknak.
4. A dialógus file-ban elhelyezett adatok alapján gondoskodnia kell a vezérlés végrehajtásához szükséges paraméterek előállitásáról.
5. A lefutott felhasználói programtól vissza kapott információkból el kell döntenie, hogy milyen uj részdialógust kell aktivizálnia.

A felsorolt feladatok megoldásait nem ismertetjük, csupán néhány fontosabb részt emelünk ki. Fontosnak tartjuk, más programszegemesekkel való kapcsolatok /interface felületek / definiálását, mivel azok kihatnak a felhasználói programok /ld. 13. ábrát/ elkészitésére.

A dialógus processzor részdialógusokat dolgoz fel. Egyszerre csak egy részdialógussal foglalkozik ugy, hogy a részdialógusban lévő akciókat sequenciálisan aktivizálja. A processzor a program futását a vezérlő programra csak akkor engedi át, ha a részdialógushoz tartozó akciókat feldolgozta. A vezérlésátadás függ még a végrehajtandó részdialógus tipusától is. Mégpedig:

1. JUMP /Menu/ tipusu részdialógus feldolgozása után, hogy hol folytatódik a program futása az attól függ, hogy milyen tipusu menüelemet választott ki a rendszer felhasználója.
   Két eset lehetséges:
   - Ha a kiválasztott menüelemhez egy ujabb részdialógus volt hozzárendelve, a program futása a procesz-

szorban marad és az uj kiválasztott részdialógus feldolgozását kezdi el.

- Ha a kiválasztott menüelemhez egy felhasználói programszegmens volt hozzárendelve, a program futása a vezérlő programra kerül.

2. QA tipusu részdialógus feldolgozása esetén a processzor a tervezőtől az akciókon /kérdés-válasz/ keresztül bekéri az adatokat és elhelyezi azokat egy strukturaváltozóba. A program futása csak akkor kerül a vezérlő programra, ha a részdialógushoz tartozó összes adat beolvasása megtörtént.

A fentiekből következik, hogy a felhasználói programszegmensek a tervezőtől származó adatokat a dialógus processzor által feltöltött strukturaváltozon keresztül kapják meg. Röviden a dialógus processzor kimenete és a felhasználói programszegmensek bemenetei közti kapcsolatot a strukturaváltozó teremti meg.

Mielőtt a strukturaváltozót definiálnánk, összefoglaljuk a beolvasott válaszok fajtáit és azok tipusait. A beolvasott válaszok fajtáit a grafikus display perifériái határozzák meg.

1. Alfanumerikus tasztauráról beolvasott válaszok a következő tipusuak lehetnek:
   a/ valós szám;
   b/ egész szám;
   c/ karakter sorozat.
2. Fényceruza válasz esetén a következő információkat kapjuk meg:
   a/ egész szám /grafikus szegmens neve/;
   b/ egész szám /a grafikus elem/ek/-hez rendelt ITEM értéke/.

3. Pozicionáló gömbbel mozgatható szálkereszt válasz esetén a beolvasott információk:
   a/ valós szám /a szálkereszt Ø pontjának x koordinátája/
   b/ valós szám /a szálkereszt Ø pontjának y koordinátája/.
4. Lokátor /állitható potenciométer/ válasz esetén: valós szám /Ø és 1 közötti érték/

Mint a 6.2.3. fejezetből kitünik a rendszer felhasználójának megadjuk azt a lehetőséget, hogy egy kérdésre több módon válaszolhasson. Például egy pont koordinátáit megadhatja alfanumerikus tasztaturáról, szálkereszttel, vagy fényceruza azonositással. Az alfanumerikus tasztaturáról bekért adat tipusát sem irjuk elő, igy beolvasáskor kell eldönteni, hogy az valós, egész vagy karakter sorozat. A beolvasott információkat tipusuknak megfelelően kell tárolnunk, másszóval olyan **strukturaváltozót** kell kialakitanunk, amely a beolvasott információn kivül annak tipusát is megadja. A **strukturaváltozót** a GESAL nyelv deklarációs utasitások segitségével fogalmazzuk meg.

A **strukturaváltozó** definiálása előtt a benne előforduló változók tipusait határozzuk meg. Enumeration /TYPEC/ tipussal választjuk szét azokat a perifériákat, ahonnét a válaszok beérkezhetnek, és az alfanumerikus tasztaturáról beolvasható információk tipusait.
A periféria tipusa

```
TYPEC PERTYP = (TAST,      /x alfanumerikus tasztatura x/
               LIGHTP,     /x fényceruza x/
               POZG,       /x pozicionáló gömb x/
               LOCAT);     /x lokátor x/
```

A tasztaturáról beolvasott információ tipusa

```
TYPEC ATTYP = (EGESZ,        /x egész szám x/
               VALOS,        /x valós szám x/
               KARAKT);      /x karakter sorozat x/
```

UNION tipussal foglalunk le olyan közös memória területet, ahol az egész, a valós és a karakter tipusu információk elhelyezhetők. Ezzel a bemenő /input/ információk tipusa:

```
UNION ININF = ( 10 CHAR I1OCHA;  /x karakter string x/
                2 INT  I2INT;    /x 2 db egész szám x/
                2 REAL T2REAL); /x 2 db valós szám x/
```

A fenti tipusok segitségével a **strukturaváltozó** tipusát megfogalmazhatjuk.

```
STRUCT INPARM =(PERTYP PERF;     /x periféria tipusa x/
                ATTYP TYPUS;     /x beolvasott információ
                                    tipusa x/
                ININF INFORM);   /x beolvasott információ x/
```

A struktura tipus segitségével a **strukturaváltozó** a következő

```
[N] INPARM DPARM
```

A **strukturaváltozót** tömbnek kell kialakitani, mert a részdialógushoz tartozó minden akcióhoz rendelt és beolvasott információ tárolásához egy **tömbelemet rendelünk.** "N" egy egész szám, amelyet az egyes részdialógusokhoz tartozó maximális akciók száma határozza meg.

A fent definiált strukturaváltozó kapcsolatot teremt a dialógus processzor /mint output/ és a felhasználói programok /mint input/ között. Definiálnunk kell még egy másik változót is, amely kapcsolatot teremt a felhaszná-

lói programok /mint output/ és a dialógus processzor /mint input/ között. Az utóbbi változón keresztül kap a dialógus processzor információt arról, hogy a felhasználói programok milyen feltételekkel fejezték be a futásukat. A feltételeket két részre osztottuk /ld. 6.1.1. fejezetet/, mégpedig üzenet nélküliekre és üzenettel rendelkezőkre. Mind a két részhez egy-egy valós tipusu változót rendeltünk, amelyeket egy struktura tipussal fogunk össze, azaz

```
STRUCT SPRCON = (INT PRCOND,     /x PROC állapot x/
                    ERCOND);     /x üzenet állapot x/
```

A felhasználói program lefutásának állapotát mutató strukturaváltozó:

SPRCON	PROUT

A PROUT.PRCOND változó megadja az üzenet nélkül lefutott felhasználói program kimenetére jellemző számot, mig a PROUT.ERCOND a hibás futás esetén a hiba üzenetre jellemző számot. A DPARM változó a dialógus processzorban kap értékeket és a felhasználói programok használják fel, a PROUT változót a felhasználói programok töltik fel és a dialógusprocesszor dolgozza fel.

6.5. A felhasználói programszegmensek

A felhasználói programszegmensek a tervezési folyamathoz tartozó algoritmizálható feladatokat oldják meg. A célorientált rendszerben a helyüket a 13. ábra mutatja.

A felhasználói programszegmensek egyedi feladatokat oldanak meg, igy mindegyiket egyedileg kell elkésziteni.

Specifikációjukat a tervező rendszer tervezésekor /dialógus gráf elkészitése közben /ld. 6.1.1. fejezetet//, programozásukat pedig azon a nyelven kell elkészíteni, amely nyelven az egész rendszert létrehozzuk /GESAL, C stb./. A szegmensek elkészitésénél az alábbi szempontokat kell betartani ahhoz, hogy a teljes rendszerbe beilleszthetők legyenek.

1. A szegmensek egymással közvetlen nem lehetnek kapcsolatban. Ez alatt azt értjük, hogy a program futása az egyik programszegmensről nem kerülhet át a másik programszegmensre.
2. Az egyik programszegmens a másik programszegmens futási eredményeit csak a tervezési modellen /ld. 13. ábrát/ keresztül kaphatja meg.
3. A programszegmensek által előállitott uj adatokat a tervezési modellbe kell elhelyezni.
4. A szegmensek a tervezőtől származó adatokat a dialógus processzor által feltöltött struktura változón /DPARM/ /ld. a 6.4. fejezetet/ keresztül kapják meg.
5. Minden programszegmensnek gondoskodnia kell arról, hogy a futási eredményeitől függő PROUT strukturaváltozó /ld. a 6.4. fejezetet/ értékét beállitsa.

Sok esetben szükség van arra, hogy a felhasználói programon belül a kiszámitott értékeket, vagy a szegmens futásától függő szövegeket kell kiirni a display képernyőre. Ezek az információk a rendszer tervezésekor előre nem határohatók meg, mivel a programszegmensek hozzák létre, vagy generálják azokat. A kiiratás elvégzésére a felhasználói programszegmensekbe beépithető olyan programszegmenst hoztunk létre, amelynek meghivása előtt annak for-

mális paramétereit kell beállitani. A programszegmens a következő:

```
EXT PROC KIIRAS = (REF LISTA A)
```

Ahol a LISTA A strukturaváltozó a következő elemeket tartalmazza:

```
STRUCT LISTA =([6] CHAR ATER;      /x alfanumerikus terü-
                                      let neve x/
               [6] CHAR GRTER;     /x grafikus terület
                                      neve x/
                   INT SOR,        /x pozicionálás sora x/
                       OSZLOP;     /x pozicionálás oszlopa x/
                   ATTYP OUTTYP;   /x ld. 6.4. fejezetben x/
                   INT DSZ;        /x karakterek száma x/
                   ININF ÜZENET);  /x kiirandó üzenet ld. 6.4.
                                      fejezet x/
```

A LISTA-ban szereplő SOR, OSZLOP alfanumerikus terület esetén a kezdő karakter sor, illetve oszlop számot jelent. Grafikus terület esetén pedig koordinátákat. A DSZ változónak karakter és real szám /számjegyek száma/ kiiratása esetén van jelentősége.

7. INTERAKTIV TERVEZŐ RENDSZER LÉTREHOZÁSA A TANULMÁNYBAN LÉVŐ JAVASLATOK FELHASZNÁLÁSÁVAL

A tanulmányban tárgyalt módszerek, eljárások és programszegmensek felhasználásával összefoglaljuk, hogy egy interaktiv célorientált tervező rendszer létrehozásához milyen feladatokat kell megoldani, illetve munkákat kell elvégezni. A megoldandó feladatokat és az elvégzendő munkákat munkafázisokba csoportositottuk. A felsorolt munkafázisok között vannak olyanok, amelyek elvégzésére, vagy megoldására módszereket, vagy eljárásokat tudunk adni, de vannak olyanok is amelyeket javaslatok hiányában egyedileg kell elvégezni. A munkafázisok a következők:

1. Pontosan specifikálni kell azt a tervezési területet, amelyre célorientált tervező rendszert akarunk létrehozni. A specifikációnak tartalmaznia kell azt a gyártmány tipust vagy tipusokat, azon belül annak, vagy azoknak főbb jellemzőit, és a tervezés végeredményét tartalmazó dokumentációk formáit.
2. Előzetes tervet kell késziteni arról, hogy a tervezési folyamat milyen tevékenységeket tartalmaz. A tevékenységeket ketté kell választani olyan szempontból, hogy azok algoritmizálhatók-e vagy sem. /ld. a 2.3.2. fejezetet/
3. A folyamatban résztvevő tevékenységek ismeretében a következőket kell elvégezni:
   a/ az algoritmizálható tevékenységeket reprezentáló felhasználói programszegmensek pontos specifikációját;

b/ a nem algoritmizálható tevékenységekre az interaktiv megoldás megfogalmazását. Ezek a tevékenységek feloszthatók két részre. Az egyik a tervezőtől származó beavatkozás, a másik a beavatkozás programszintü feldolgozása. Ennek megfelelően tervet lehet készitени a tervezői beavatkozásról és a programszegmensek megfogalmazásáról.

4. A rendszer adatkezelésének és forgalmának megtervezése. A programszegmensek ismeretében meg lehet határozni, hogy az egyes felhasználói programszegmensek milyen tipusu adatokat igényelnek és hoznak létre /ld. 2.3.2. fejezetek/. Az adatforgalomban három tipusu adat vesz részt. Ezek:
   - adatbázis adatok
   - más programszegmensek által előállitott adatok
   - a tervezőtől származó adatok.

   Az adatok ismeretében meg lehet tervezni a rendszer adatbázisának és modelljének strukturáit, valamint a tervezőtől származó adatok által meghatározott részdialógusokat /ld. 6.1.3. fejezetet/. A strukturák megtervezése rugalmas adatbázis kezelő rendszer felhasználását igényli.
5. A rendszer dialógus gráfjának a felrajzolása /ld. 6.1.1. fejezetet/. Ebben a tervezési fázisban határozzuk meg a rendszer müködésének folyamatát. A tervezési folyamat kialakitásánál figyelembe kell venni, hogy az egész tervezési folyamatnak a tervező is részese. Az ember és számitógép kapcsolatban a tervező mint irányitó vesz részt, azaz a rendszer segitségével tervezett konkrét feladat megoldásakor, a tervező szabadon választhassa ki azt

a tervezési sorrendet, amely számára az adott körülmények között a legmegfelelőbb. E feltétel ugy valósitható meg, hogy a rendszerben nem alakitunk ki merev tervezési folyamatot. A követelményt menü technikával realizáljuk, ami biztosithatja azt, hogy az egyes tervezési szakaszok párhuzamosan is előállithatók. A menü technika alkalmazása sokszor a nem algoritmizálható feladatoknak a tervező által történő megoldását is lehetővé teszi. A nem algoritmizálható feladatok nagyrésze emberi kreativitást igényel. A kreativitást igénylő feladatok is csak ugy oldhatók meg, ha a megoldásukhoz tetszőlegesen kiválasztható eszközöket adunk. A fentiek figyelembevételével a gráfban rögziteni kell a tervezési folyamatban lévő részdialógusok, felhasználói programszegmensek és comment üzenetek helyeit és kapcsolatait.

6. Grafika megtervezése. Meg kell tervezni a munka egyes fázisaihoz rendelt képeket és ezeken belül a mezőket. A felhasználói programszegmensek ismeretében könnyen meghatározhatók a mezőkhöz tartozó, azon képi szegmensek, amelyeket valamilyen célból kiemelten szeretnénk kezelni. /ld. 6.1.2. fejezetet/
7. A gráfban előforduló részdialógusok és comment kiirások megtervezése. A tervezés folyamán el kell dönteni a részdialógusokhoz tartozó kérdéseket /alfanumerikus, grafikus/ és a hozzájuk tartozó válaszokat, azok tipusait, milyen mezőben és a mezőn belül hol helyezkednek el. Ugyanez vonatkozik a comment kiirásokra is.

8. A specifikált felhasználói programszegmensek elkészitése, betartva a dialógus rendszer által előirt követelményeket /ld. 6.5. fejezetet/.
9. A grafika, a gráf szerkezete, a részdialógusok, a commentek és a procedure-k ismeretében el lehet készíteni a dialógus programot DL nyelven. /ld. 6.2. fejezetet/
10. A DL nyelvü dialógus program feldolgozása a dialógus generátoron. A generátor előállitja a tervező rendszer vezérlő programját és feltölti adatokkal a dialógus adat file-t /ld. 6.3. fejezetet/.
11. A rendszer összeszerkesztése. Össze kell szerkeszteni a megirt felhasználói programszegmenseket a generált vezérlő programot, valamint a dialógus processzort egy rendszerré.

A fentiekből látható, hogy egy interaktiv tervező rendszer létrehozásához a tanulmányban javasolt eljárások, módszerek és a programszegmensek használata komoly segitséget nyujt. A legnagyobb segitség két területen jelentkezik. Az egyik az ember és a számitógép kapcsolat megfogalmazása és implementálása, a másik a teljes rendszer összeszervezése területén. A tanulmány nem oldja meg és javaslatot sem tartalmaz a modellezési problémák általános megoldására, csak az e területen jelentkező feladatokat próbálja összefoglalni.

8. ÖSSZEFOGLALÁS

A tanulmány témája az 1012/1972. sz. Kormányhatározattal kiemelt K-g "A Gépgyártástechnológia Kutatása, Fejlesztése" c. országos program 9. támacsoportjához kapcsolódik /Automatizálás/. A tanulmány eredményei az OMFB által támogatott kulcsra kész CAD/CAM rendszerek létrehozásához nyujtanak segitséget. A javasolt elvek módszert adnak ahhoz, hogy interaktiv AMT /Automatizált Müszaki Tervezés/ rendszer tervezője megfogalmazhassa a tervezési folyamatok modellezését, az ember és számitógép közötti dialógust, a rendszerhez tartozó programszegmenseket és ezek kapcsolatát. A tanulmányban leirt javaslatok nagyban megkönnyitik a rendszer tervezőjének munkáját. A kutatási eredmények alkalmazásával olyan eszközöket használhat, amelyek segitségével a dialógus megszervezése és a rendszert vezérlő programjának elkészitése már megoldott. A javaslatok alkalmazása lehetővé teszi, hogy a rendszer tervezője valóban a tervezési folyamat felépitésének strukturájával foglalkozzon és elképzeléseit a részletek kimunkálása nélkül realizálhassa. Az ilyen módon tervezett rendszerek nemcsak rövidebb idő alatt implementálhatók, hanem egyszerüen módosithatók is. Ez egyrészt a tervezendő rendszer vezérlő programjának automatikus előállitásának, másrészt a moduláris és strukturált rendszerfelépitésnek tulajdonitható.

A tanulmányban tárgyalt elvek részbeni felhasználásával készült el az Egyesült Izzó és Villamossági Gyár RT számára az ISTER. Az ISTER ipari bevezetése 1978-ban történt. A sikeres alkalmazását bizonyitja, hogy

a gyár vezetői jelenleg azon munkálkodnak, hogy a rendszert továbbfejlesztve megujitsák és kibővitsék.

Az ISTER ipari bevezetése után, annak használatában szerzett tapasztalatokat is figyelembe véve, egy kiértékelést végeztünk. Az értékelésről, amely kiterjed a rendszer rendszertechnikai, felhasználói és tiszta gyakorlati /pragmatikus/ területekre, rövid áttekintést adunk.

1. Rendszertechnikai szempontból helyesnek bizonyult az az elképzelés, hogy a tervezési folyamatba tartozó tevékenységeket úgy alakitottuk ki, hogy azok önállóan is végrehajthatók legyenek és a felhasználó azokat a tervezési folyamatban tetszőleges helyen akár többször is aktivizálhassa. Ez az elképzelés ellentétes a batch üzemmódban használt rendszer alkalmazásától, ahol a rendszer merev felépitése miatt a konstruktőr egy kötött tervezési sorrendet kénytelen betartani. Az elképzelés megvalósitásához egy sor uj rendszertechnikai feladatot oldottunk meg. A tervezési folyamat bizonyos egységeinek általánositásával olyan programszegmenseket kellett létrehozni, amelyeket ismételt aktivizálással különböző célokra lehetett felhasználni. A programegységek közül a dialógus rendszert [146] emeljük ki, amellyel sikerült megoldani:
   - az ember és a számitógép dialógus kapcsolatának megfogalmazását,
   - a kapcsolatokat megvalósitó programrészek implementálását.

A dialógus rendszer nagy előnye az, hogy a teljes tervező rendszer vezérlését rá lehet bizni. Ezt azért tehettük meg, mert a tervező rendszerhez tartozó egyéb programszegmenseket egymáshoz képest párhuzamosan füztük fel. A párhuzamosan felfüzött programszegmensek egymás közti kapcsolatát egy adatbázis jellegü modellel valósitottuk meg. Ez azt jelenti, hogy a programszegmensek által létrehozott adatokat a rendszer a modellbe helyezi el és ezen adatokat kiolvasással más egységek felhasználhatják. A fenti elvnek az alkalmazásával választottuk le a teljes rendszerről a geometriai jellegü feladatokat megoldó ugynevezett "Geometriai Rendszert" [162].
Ez a 2D. rendszer nemcsak különböző tipusu rajzok készitésére használható, hanem a modellben lévő információkat sikeresen használják az ilyen adatokat igénylő egyéb programszegmensek is.

A geometriai rendszer interaktiv alkalmazásánál bizonyos nehézségeket jelentett a sok tervezői beavatkozás. Ennek az áthidalására dolgoztuk ki a GEIN /Geometriai Input/ nyelvet, amely lehetővé tette, hogy a tervezői munkához szükséges kezdeti geometriai paramétereket off-line módon is bevihessük a rendszerbe. Az adatbázis jellegü modell alkalmazásával az archiválási problémákat is sikeresen meg lehetett oldani.

Rendszertechnikai szempontból az ISTER-ben még megoldatlan volt a dialógus és a programszegmensek közti kapcsolat algoritmusa. Ezért ezt a feladatot

manuálisan oldottuk meg. Ez a rendszer összeszerkesztésénél és módositásánál okozott némi többletmunkát.

2. Felhasználói szempontokat tekintve a rendszer szintén sikeresnek tekinthető. A már emlitett tervezési tevékenységek önállóan történő alkalmazhatósága lehetővé tette, hogy a tervező kreativ képességeit ki tudja fejteni. Ez a lehetőség ergonómiailag is növelte a rendszer hatékonyságát. A menütechnika alkalmazása, valamint a funkcionális tasztatura gombjaihoz rendelt rendszerfunkciók a rendszer alkalmazását rugalmassá tették. Az önállóan elérhető programszegmensek alkalmazása azt is biztositotta, hogy a rendszer necsak a sajtolószerszámok tervezésére legyen használható, hanem egyéb célokra is. Például:
   - különböző rajzok készitése, módositása /vázlatok, mühelyrajzok, pantográfrajzok stb./;
   - vezérlő lyukszalagok előállitása szikraforgácsoló NC szerszámgépekhez;
   - rajzok beméretezése;
   - különböző célu sávtervek /elrendezési rajzok/ készitése stb.

   Kiemelnénk a geometriai rendszerhez kifejlesztett nagyon rugalmasan használható manipulációs lehető ségeket. Ezeknek a lehetőségeknek az alkalmazásával a készülő rajzok kényelmesen továbbfejleszthetők, módosithatók stb.
3. Pragmatikus, gyakorlati szempontokból is érdemes volt a végzett munkát analizálni, mivel az itt szer-

zett tapasztalatok nagyon fontosak az uj rendszerek implementálásánál. Nagyon segitette a munkánkat, hogy a tervezési folyamat tevékenységeit ketté választottuk aszerint, hogy azok algoritmizálhatók-e vagy sem. A vizsgálatok egyértelmüen bizonyitották, hogy e feladat megoldása messze nem olyan triviális, mint amilyennek első pillanatban tünik. Az interaktiv számitógépes rendszerrel segitett tervezésben az egyes tevékenységek másképpen jelentkeznek, mint a hagyományos tervezésben. A tevékenységek egy részét a számitógép veszi át, egy másik részét hasonlóan a hagyományos módszerekhez a tervező oldja meg. Vannak viszont olyan tervezési tevékenységek is, amelyeknek automatizálása uj módszereket igényelnek, amelyek eltérnek a hagyományos eljárásoktól és a batch-ben futtatott programok alkalmazásánál kialakult módszerektől is.

Az ISTER létrehozásának sikeres használatán tul óriási szemlélet módositó hatása is van. A benne megvalósitott tervezési módszerek és eljárások egy uj és fejlett tervezési irányzatot képviselnek. Ezt bizonyitja az a nagy érdeklődés, amely az ISTER iránt megnyilvánul mind hazánkban, mind külföldön /Magyarországon: BHG; BRG, GAMMA, DANUVIA, IGV, VIDEOTON stb.; Külföldön: Csehszlovákia, Német Demokratikus Köztársaság, Szovjetunió /MINSZK//.

A tanulmány az ISTER-ben alkalmazott elveken tul uj módszereket és eljárásokat is tartalmaz. Ezek teszik lehetővé, hogy az ISTER-hez hasonló felépitésü célorientált rendszereket egyszerübb módon hozzunk létre.

Egy a Müvelődésügyi Minisztérium által támogatott kutató és fejlesztő munka folyik e módszerek felhasználásával. Célja tengelyek konstrukciós, előgyártmány és gyártástechnológia tervezésére célorientált rendszer létrehozása. Ezt a kutató munkát a BME Gépészmérnöki Kar néhány intézete és az MTA SZTAKI együttmüködésével végezzük.

A Müszaki Egyetemen folyó elméleti és gyakorlati oktatásban, a mérnök-továbbképzésben a tanulmányban foglaltak közvetett hasznositására nyilik majd lehetőség. Itt jegyezzük meg, hogy a BME Gépészmérnöki Kar Gépgyártástechnológiai Tanszékén folyó oktatási anyagba az ISTER ismertetését már beépitették.

Irodalomjegyzék

1. Hatvany J.: Ember és gép információcseréjének uj eszközei
Ipari Mérés és Szabályozás Szimpózium, Balatonszéplak, 1967.

2. Pikler Gy.: Mini-számitógépes interaktiv alkatrészprogramiró rendszer NC szerszámgépek automatikus programozásához.
MTA SzTAKI Tanulmányok, 18/1974.

3. Pikler,Gy., Simon,V.: A General Dialogue System for Interactive Graphic Programming of NC and CAD System.
Prolamata'76 Conference 1976.

4. Pikler Gy.: Egy általános interaktiv gépészeti tervező rendszer.
Mérés és Automatika, 7. 1978.

5. Ferenczy J.: Sajtolószerszámok és sajtolási technológia számitógépes interaktiv tervezése.
MTA SzTAKI Közlemények, 8/1969.

6. Archer, L.: Systematic Method for Designers.
London: Council of Industrial Design, 1965.

7. Booker, P.J.: Written Contribution Appended to.
Conf. on the Teaching of Engineering Design, 1964.

8. Farr, M.: Design Management.
London: Hutchinson, 1966.

9. Jones, J.C.: Design Methods Reviewed.
London: Butterworths, 1966.

10. Page, J.K.: Contribution to Building for People.
London: Ministry of Public Building and Works, 1966.

11. Allexander, C.: The Determination of Components for an Indian Village.
Conf. on Design Method /1963/.

12. Jones, J.C.: Design Method Seeds of Human Futures.
Lond: Wiley-Interscience Publ. Co., 1970.

13. Hill, P.H.: The Science of Engineering Design.
Holt, Rinehart and Winston Inc. /1970/.

14. Isber, M.: Three Dimensional Beam Synthesis Applications.
Computer and Structures, Vol. 7 /1977/.

15. Ellis, J.: Jonson's Method of Optimum Design Applied to a Problem with Simple Functional Relationship.
Computer Aided Design, Vol. 8 /1976/.

16. Astrop, A.: Parts Coding is Alive and Well.
Machinery and Production Engineering, 14 June,1978

17. Gausemeier, J.: Der Computer Plant und Konstrukwert
VDI Nachrichten Nr. 35/2, September, 1977.

18. Wilcox, L.J.: Finite-Element Analysis Pinpoints Geor-Tooth Stresses.
Machine-Design, February 23 /1978/.

19. Gerwin, R.: Computer konstruieren sich selbst.
Datenträger ZFD 4.1. 1966.

20. APT Part Programming
McGraw-Hill Book Co., New York, 1967.

21. Mittman, B.: Computers and Graphical Processing.
Metalworking Production, 4. Jan. 1967.

22. Siders, R.A.: Computer Graphics. A Revolution in Design.
American Management Association Inc., New York, 1966.

23. Sutherland, J.E.: Sketchpad: A Man-Machine Graphical Communication System.
Proc. Spring Joint Comp. Conf., 1963.

24. Chasen, S.H.: The Introduction of Man-Computer Graphics into the Aerospace Industry.
Proc. Fall. Joint. Comp. Conf., 1965.

25. Siders, R.A.: Computer Aided Design.
IEEE Spectrum, Nov. 1967.

26. Miller, V.: Maximizing Engineering Effectiveness.
Machine Design, Dec. 11, 1975.

27. Neumann J.: Válogatott elõadások és tanulmányok.
Közgazd. és Jogi Könyvkiadó, Budapest, 1965.

28. Ross, D.T.: A Generalized Technique for Symbol Manipulation and Numerical Calculation.
Commun. of the ACM 4.3. 1961, March.

29. Coons, S.A.: The Use of Computers in Technology.
Scientific American 215.3., March, 1966.

30. Coons, S.A.: An Outline of the Requirement for a Computer--Aided Design System.
Proc. Spring. Joint. Comp. Conf., 1963.

31. Coons, S.A.: Computer Graphics and Innovative Engineering Design.
Datamotion, May, 1966.

32. Licklider, J.C.: Man-Computer Symbiosis.
IRE Trans. Human Factors in Electronics, HTE-1, March, 1960.

33. Sutherland, J.E.: Computer Graphics.
Datamation, May, 1966.

34. Chasen, S.H.: On-line Systems and Man-Computer Graphics.
Computers and Automation, Nov. 1967.

35. Christiansen, D.: Computer-Aided Design: Part 1. The Man-Machine Manager.
Electronics 19. Sept. 1966.

36. Prince, M.D.: Man-Computer Graphics for Computer-Aided Design.
Proc. of the IEE 24., 12. Dec. 1966.

37. Taylor, R.W.: Man-Computer Input-Output Techniques.
IEEE Trans. on Human Factor in Electronics, 8.1.
March, 1967.

38. Computer Aided Design.
NEL Report 242, 1966.

39. New Computer-Aided Design Centre at Cambridge.
Instrument Practice, Sept. 1967.

40. Gott, B.: A National Centre for Computer-Aided Design.
Paper of Joint Colloquium on Computer-Aided Design,
Budapest, Apr., 1973.

41. An Application of the Ferranti Drawing Measuring Machine.
Machinery and Production Engineering, 15, March, 1967.

42. Opitz, H., Simon, W.: The Programming of Numerically
Controlled Machines with EXAPT.
Machinery and Production Engineering, Aug. 1967.

43. Simon, R.: Rechnerunterstütztes Konstruieren.
Industrie Anzeiger, 90, Okt. 1968.

44. Goranszkij, G.K.: A gépek tervezésének automatizálása.
GTE Szeminárium anyaga; MTESZ 68/5782.

45. Hatvany J.: Néhány uj irányzat a gépgyártás automatizálásában.
III. Automatizálási Kollokvium, Budapest, 1966.

46. Pikler Gy.: EXAPT1 Számitógépes programozási nyelv honositási feladatai és tapasztalatai.
Számitógép Alkalmazása a Gépiparban Konferencia, Miskolc,
1969.

47. Hatvany J., Kovács M., Pikler Gy.: Számjegyes vezérlésü
szerszámgépek számitógépes programozása.
MTA SzTAKI Közlemények 2/1969.

48. Horváth, M., Molnár, B.E., Nagy, S.: An Attempt in High
Level Automation of Programming of NC Lathes.
The FORTAP System.
Prolamat'73, Budapest, 1973.

49. Ferenczy J., Kovács M.: Számitógépes interaktiv tervezés /Computer Aided Design/.
MTA AKI Közlemények, 5/1969.

50. Elliot, W.S.: Interactive Graphics in Computer-Aided Mechanical Engineering Design.
Proc. of the Symp. on Computer-Aided Design in Mechanical Engineering, Milano, Oct. 1976.

51. Computers Aircraft Companies Use Them Well.
The Economist, June 1967.

52. Joyce, J.D.: Reactive Display: Improving Man-Machine Graphical Communication.
Proc-AFIPS. Fall. Joint. Comp. Conf., 1967.

53. Frank, W.L.: Software for Termianl-Oriented Systems.
Datamation, June, 1968.

54. Lawson, H.W., Herold, W.: The Formal Definition of Human/Machine Communication.
Software-Practice and Experience, Vol. 8, 51-58 /1978/.

55. Lukács, G., János, J.: Define-IT-Yourself Languages.
CAD in Medium Sized and Small Industries.
North Holland Publ. Co., Micado., 1981.

56. GSS'80 Grafikus szubrutinok.
MTA SzTAKI Belső Tanulmány, 1979.

57. Sorgie, C.D.: Algol W. Reference Manual.
Brown University, Providence Rhode Island, July 4, 1976.

58. Bevezetés a GESAL nyelv használatába.
MTA SzTAKI Belső Tanulmány /1980/.

59. Ross, D.T.: Structured Analysis /SA/: A Language for Communicating Ideas.
IEEE Transaction of Software Engineering, Vol. SE-3., No. 1, Jan. 1977.

60. Nijssen, G.M.: Modelling in Data Base Management Systems.
IFIP Technical Committee, 2, 1976.
North Holland Publ. Co.

61. Andor L.: Kisgépes adatbázis kezelő rendszer.
MTA SzTAKI Tanulmányok, 92/1980.

62. General ICES Information.
Civilg Engineering Systems Laboratory, MIT Cambridge,
USA, 1973.

63. Pohl, P.J.: Structure and Functions of the Information
System Technology /IST/.
Proc. of the Workshop, Dec., 1974.

64. Prichardt, M.A.: SU-ICES.
Colloque international sur les systèmes integrés en
genie civil.
30-31/8 et 1/9/1972.
CEPOS Université de Liège, Belgique, avril 1974.

65. Ross, D.T.: ICES System Design.
MIT Press Massachusetts USA, 1967.

66. Beilschmidt, L.: Data and File Management in the In-
formation System Technology /IST/.
Workshop on General Purpose CAD Systems CERT, Toulouse,
Dec. 1974. Proceedings of Workshop Dec. 1974.

67. Rhodes, T.R.: The Computer-Aided Design Environment
Project/COMRADE/
AFIPS. National Computer Conference 1973.

68. Garrock, G.A., Hurley, M.J.: The IPAD System: A Future
Management /Engineering/ Design Environment.
Computer Apr. 1975.

69. Eversheim, Wiewelhowe: Design and Automatic Set-up of
Drawings and Work Plans.
16th International Machine Tools Design and Research
Conference, Manchester, 1975.

7O. Weck, M.: Programmbibliothek zur Berechnung von Maschinenbauteilen mit Tischrechnern.
Industrie Anzeiger Nr. 46. 1977.

71. Opitz, H.: Rechnereinsatz in der Konstruktion.
Z. Ind. Fertig. 67. /1977/ No. 3.

72. Pikler Gy.: Gyártmánytervezés számitógépes interaktiv módszerekkel.
Automatizálás 1O/1978.

73. Holló K.: Interaktiv gépészeti tervező rendszer geometriája.
Automatizálás 1O/1978.

74. Turai I.: Adatkezelő alrendszer számitógépes tervezéshez.
Automatizálás 1O/1978.

75. Warman, E.A.: Man-Oriented of Computer Aided Design.
Software World, Vol 6. No. 6. 1975.

76. Black, P.H., Adams, O.E.: Machine Design.
McGraw Hill Book Company, New York, 1955.

77. Hendley, E.J., Williams, R.A.: Graph Theory in Modern Engineering.
Academic Press, New York, 1973.

78. Spillers, W.R.: Basic Questions of Design Theory.
North Holland Publ. Co., 1974.

79. Cvetkov,, V.D.: Avtomatizacija szinteza szvednüh konsztruktorszkih i tehnologicseszkih szpecifikacij szlozsnüh objektov.
Vücsiszlitelnaja tehnika v masinosztroenii i jun 197O, Minszk.

8O. Foster, J.M.: List Processing.
MacDonald and Co. Ltd., 1969.

81. Weizenbaum, J.: Symmetric List Processor.
Commun. Ass. Comput. March 6, 1963.

82. "COMIT Programmer's Reference Manual".
MIT Press, Cambridge, Massachusetts, 1963.

83. Sacerdoti, E.D.: A Structure for Plans and Behaviour.
Elsevier Computer Science Library, New York, 1977.

84. Hansen F.: Módszertani géptervezés.
Müszaki Könyvkiadó, Budapest, 1969.

85. Hatvany, J.: The Engineer's Creative Activity in a CAD Environment in Computer Aided Design.
Proc. IFIP Working Conference on CAD, 1973.

86. Shackel, B.: The Ergonomics of the Man/Computer Interface.
MAN/Computer Communication, Infotech State of the Art Report, Vol. 2. 1980.

87. Newman, A.D.: Patterns. In the Design Methods.
London: Butterworths, 1966.

88. Claussen, U.: Computer Aided Design of Mechanical Elements and Mechanisms in a Small Design Office.
Proc. of the Symposium on Computer Aided Design in Mech. Eng., Milano, Oct. 1976.

89. Trauboth, H.: An Engineering View of System Specification and Design Tools.
Infotech State of Art Report, Structured Software Development, 1980.

90. Freeman, P.: A Perspective on Requirement Analysis and Specification.
Infotech State of Art Report Structured Software Development, 1980.

91. Gilb, T.: Structured Design Methods for Maintainability.
Infotech State of Art Report Structured Software Development, 1980.

92. Jones, J.C.: Design as a Creative Activity.
Infotech State of Art Report Structured Software Development, 1980.

93. Yeh: Software Requirement Engineering. A Perspective. Infotech State of Art Report Structured Software Development, 1980.

94. Kleim, H.F.: Finite Element Programs Based on General Programming Systems.
Computer and Structures, Vol. 8, 1978.

95. Bencsik Zs.: Eljárások előirt térgörbére illeszkedő felület által burkolt forgásfelület számitására.
NME Közleményei IV. sorozat, 1976.

96. Drahos J., Bencsik Zs.: Forgácsoló szerszámok geometriájának számitógéppel segitett tervezése.
COMPCONTROL'79.Sopron, 1979.

97. Gould, I.H., Inghman, P.: Computer Aided Analysis of Kinematic Mechanisms.
Internat. Conf. on Interactive Techniques in Computer Aided Design, Bologna, 1978.

98. Farkas J., Szabó L.: Hegesztett I-tartók optimalizálása "backtrack" programozással.
COMPCONTROL'79, Sopron, 1979.

99. APT Part Programming. The APT Long Range Program Staff. IIT Research Institute, 1967. McGraw-Hill Book Comp.

1oo. Cser L., Turi Z.: Grafikus és optimalizációs rendszer gépjármü szélvédők optikai tervezéséhez.
COMPCONTROL'79, Sopron, 1979.

1o1. Gribocsebszkij: Komplekszsznaja szisztema avtomatizirovannovo proektirovanaja i izgotovlenija razgyelitelnüh stampov dlja privoposztroenija "AVTOSTAMP".
Triborü i szisztupravlenija, No.4. 1975.

1o2. Muranski: Computer-assisted Design of Press-tools.
Machinery and Production, Jan. 12, 1977.

1o3. Risor, M.L.: A Vote for In-House Programming.
Manufacturing Engineering and Management, Nov. 1970.

1O4. Leslie, W.: Feature of Computer Programs for Numerical Control.
Proc. Full Join Comp. Conf., 1969.

1O5. Newman, W.: Principles of Interactive Computer Graphics.
McGraw-Hill Book Co., 1973.

1O6. Sami, J., Banna, A.: Parametric Design and Interactive Computer Graphics.
Special Conference on Computer Graphics as Related to Engineering Design, July 1973. Vol. 2. Columbia Univ.

1O7. Martin, J.: Design of Man-Computer Dialogue.
Prentice-Hall International Inc., London, 1973.

1O8. Jones, P.F.: Four Principles of Man-Computer Dialogue.
Computer Aided Design. Vol. 1O, No. 3. May, 1978.

1O9. Armit, A.P., Elliot, S.W.: The Design of Interactive Systems for CAD.
CAD'76 Conference on Computers in Engineering and Building Design, 1976.

11O. Rouse, W.B.: Design of Man-Computer Interface for On-line Interactive Systems.
Proc. of the IEEE, Vol. 63, No. 6, July 1975.

111. Warman, E.A.: Man-Oriented Aspects of Computer Aided Design.
Software World, Vol. 6, No. 6. 1980.

112. Allum, R.F.: Techniques for Improving the Role of Graphics in CAD.
CAD'74, 1974.

113. Miller, L.A.: Behavioral Issues in the Use of Interactive Systems.
Int. J. Man-Machine Studies, 1977.

114. Freeman, C.C.: Resource for CAD Systems.
Proc. of Workshop, Dec. 1974.

115. Sabin, M.: Workshop on General Purpose Computer Aided Design Systems.
CAD, Vol. 7. No. 2. April, 1975.

116. Joequart, R.: Current Trends in the Development of Integrated General Purpose CAD Systems.
Proc. of the 12th Design Automation Conference, Massachusetts, June 23-25, 1975.

117. Blain, B., Labarthe, A.: BAL: An Aid to Scientific Application Programming.
Proc. of the Workshop, Dec. 1974.

118. Hilles, J.O.: RAINBOW. An Early Integrated System for CAD.
Proc. of the Workshop, Dec. 1974.

119. Oreszkin, I.T.: O szredsztvah realizacii problemno-orientirovannïh jazükov.
Programipovanue No. 2. 1977.

120. Elin, V.S.: Szisztema dlja razrabotki bolsih programmnüh komplekszov.
Programipovanie No. 2. 1977.

121. Szuhjamun, V.A.: Mateszisztema dlja posztroenija nproblemno--orientirivannïh jazükovüh szisztem i paketov prikladnih problem.
Programirovanie No. 2. 1976.

122. Tamm, B.G.: Integrirovannija szisztema programmirovanija INEUM;
Trudi Insztituta, 43. Moszkva, 1974.

123. Andor L., Lukács G.: Kisgépes AMT monitor.
1. Orsz.Autom.Müsz.Terv.Konf., 1982.

124. ORACLE Relational Data Base Management System.
Relational Software Incorporated, California, 1979.

125. Holló K., Váradi T.: Számitógépes alkatrészmodellezés.
MTA SzTAKI Belső Tanulmány, 1978.

126. Holnapi D.: Az automatizált müszaki tervezés matematikai alapjai.
Építéstudományi Intézet Tanulmány, 1979.

127. Márkusz,Zs.: Knowledge Representation of Design in Many-Sorted Logic.
Proc. of IJCAI-81, Vancouver BC. Canada, 1981.

128. Rouse, W.B.: Design of Man-Computer Interface for On-line Interactive Systems.
Proc. of the IEEE, Vol. 63, No. 6, June, 1975.

129. Cugini, U.: A Survey on Computer Aided Design in the Field of Mechanics in Italy.
Proc. of the Symposium on CAD in Mechanical Engineering, Milano, Oct. 1976.

130. Hitch, H.: Computer Aided Design in Mechanical Engineering in the U.K. An Overview.
Proc. of the Symposium on CAD in Mechanical Engineering, Milano, Oct. 1976.

131. Rankers, H.: Computer Aided Design in Mechanical Engineering in The Netherlands in 1976.
Proc. of the Symposium on CAD in Mechanical Engineering, Milano, 1976.

132. Taylor, R.W.: On the Relation of Interactive Computing to Computer Science.
Proc. of the IEEE, Vol. 63, No. 6, July, 1975.

133. Gaines, B.R.: Man-Computer Communication. What Next?
Internat. J. of Man-Machine Studies, Vol. 10, May, 1978.

134. Kaiser, P., Štetina, I.: DIAGEN-Dialogue Generator.
Computer Research Centre, Bratislava, 26. Sept. 1977.
CES/SEM 9/6.

135. Nagel, R., Schnupp, P.: Expected and Unexpected Acceptance Problems with an Interactive Probleming Tool.
Pragmatic Programming and Sensible, Software 4577, 1977.

136. Kochan, D.: Fertigungsprozessgestaltung und Informationsverarbeitung.
VEB Verlag Technik, Berlin, 1977.

137. Armitage, B.S., Hall, P.A.: Conceptual Schema for CAD Computer Aided Design, Vol. 9, No. 13, July, 1977.

138. Kestner, W.: A Dialogue System for Creating and Manipulating Graphical Symbols and Structures.
CAD, Vol. 8, No. 2, April 1976.

139. Hebditch, D.: Design of Dialogues for Interactive Commercial Application.
Infotech State of the Art Report, MAN/Computer Communication, Vol. 2. 1980.

140. Hebditch, D.: The Programming Implications of Good Dialogue.
Pragmatic Programming and Sensible Software /1978/.

141. Jakobsen, K., Vangen, G.: Interactive Graphics in Parametric Machine Design.
Internat. Conf. on Interactive Techniques in CAD, Bologna, 1978.

142. Gaines, B.R.: Programming Interactive Dialogue.
Pragmatic Programming and Sensible Software /1977/.

143. Forgács T., Gerhard G., Kocsis J., Krammer G.:
DISTAR-B általános dialógus rendszer.
MTA SzTAKI "Felhasználói dokumentáció", 1971. október.

144. Forgács T., Krammer G.: A dialógus generátor
MTA SzTAKI Tanulmányok, 1973.

145. Pikler, Gy.: A Minicomputer-Based, Conversational Program Writing System.
Prolamat'73 Conference, 1973.

146. Pikler Gy.: Dialógus rendszer interaktiv rendszerekhez.
MTA SzTAKI Belső Tanulmány, 1976.

147. Black, J.L.: A General Purpose Dialogue Processor.
National Computer Conference, 1977.

148. Bauböck, E.: Dialogue-handling System for Graphics and CAD Applications.
Internat. Conf. on Interactive Techniques in Computer Aided Design, Bologna, 1978.

149. Kutschke, K.: Ein mathematisches Modell für die Steuerung von Dialogsystemen.
Rechentechnik Datenverarbeitung, 4/1979.

150. Ledgard, H., Singer, A., Whiteside, J.: Directions in Human Factors for Interactive Systems.
Lecture Notes in Computer Science, 103, 1981.

151. Forgács, T.: Interactivity in CAD, why? How and What?
Proc. of Design III., Research Education, Practice, Tom 1, Wroclaw, 1978.

152. Krammer, G.: In Search on Overall Model for Man--Computer Problem Solving Dialogue".
The IFIP WG. 5.2. Workshop on the "Methodology of Man-Machine Interaction" Seillac, 1979.

153. Gaines, B.R.: Man/Computer Communication. An Overview.
Infotech State of the Art Report, Vol. 2. 1980.

154. Pask, G.: Aspects of Machine Intelligence Graphical Conversation Theory.
Massachusetts Institute of Technology Press, 1977.

155. Pask, G.: Organisational Closure of Potentionality Conscious System.
Internat. Conference on Applied General Systems Research, 1977 August at Bingnamton New York.

156. Márkusz Zs.: Intelligens interaktiv rendszerek elvi problémái.
MTA SzTAKI Belső Tanulmány, 1978.

157. Hermann G.: Bevezetés a GEZA nyelv használatába.
MTA SzTAKI Belső Tanulmány, 1981.

158. Kernighan B.W., Ritchie D.M.: The C Programing Language.
Bell Laboratories, Murray Hill, New Jersey, 1978.

159. Hatvany J., Verebély P.: "Distributed Intelligence in the GD'80 Display System".
SID Symposium, Chicago, 1979.

160. Krammer G., Darvas P., Hosszu P., Sági J.: A GTU grafikus run-time rutinok.
MTA SzTAKI BK.GK, 1975.

161. Pikler Gy.: Interaktiv gépészeti tervező rendszerek elméleti és gyakorlati kérdései 1. Függelék: Interaktiv sajtolószerszám tervező rendszer /ISTER/ általános leirása.
Kandidátusi értekezés, 1981.

162. Pikler Gy.: Interaktiv gépészeti tervező rendszerek elméleti és gyakorlati kérdései 3. Függelék: Az ISTER-ben használt geometriai rendszer leirása.
Kandidátusi értekezés, 1981.

163. Pikler Gy.: Interaktiv gépészeti tervező rendszerek elméleti és gyakorlati kérdései 4. Függelék: A dialógus file strukturája DDL nyelven.
Kandidátusi értekezés, 1981.

164. Pikler Gy.: Interaktiv gépészeti tervező rendszerek elméleti és gyakorlati kérdései 5. Függelék: Példa a rendszergenerálásra.
Kandidátusi értekezés, 1981.

A TANULMÁNYSOROZATBAN 1982-BEN MEGJELENTEK

130/1982 Barabás Miklós - Tőkés Szabolcs: A lézer printer képalkotás hibái és optikai korrekciójuk

131/1982 RG-II/KNVVT "Szisztemü upravlenija bazani dannüh i informacionnüe szisztemü" Szbornik naucsno-iszszledovatel'szkih rabot rabocsej gruppü RG-II KNVVT, Bp. 1979. T o m I.

132/1982 RG-II/KNVVT T o m II.

133/1982 RG-II KNVVT T o m III.

134/1982 Knuth Előd - Rónyai Lajos: Az SDLA/SET adatbázis lekérdező nyelv alapjai /orosz nyelven/

135/1982 Néhány feladat a tervezés-automatizálás területéről. Őrmény-magyar közös cikkgyüjtemény

136/1982 Somló János: Forgácsoló megmunkálások folyamatainak optimálási és irányitási problémái

137/1982 KGST I-15.1. Szakbizottság 1979. és 80. évi előadásai

138/1982 Kovács László: Számitógép-hálózati protokollok formális specifikálása és verifikálása

139/1982 Operációs rendszerek elmélete 7. visegrádi téli iskola

A TANULMÁNYSOROZATBAN 1983-BAN EDDIG MEGJELENTEK:

140/1983 Operation Research Software Descriptions (Vol.1.) Szerkesztette: Prékopa András és Kéri Gerzson

141/1983 Ngo The Khanh: Prefix-mentes nyelvek és egyszerü determinisztikus gépek