

Benda Klára: Az etnográfiai képzelet és az oktatás virtualizálása

avagy Miért ne higyjünk a mérnököknek?

Bevezetés

Már egy ideje – gyakorlatilag amióta hálózati oktatással foglalkozom – foglalkoztat a gondolat, hogy miért nem voltak még képesek gyakorlatilag sehol a világon olyan digitális technológiákat létrehozni, amelyek az oktatási intézmények társas viszonyrendszerét a hálózati közegben jelenítik meg. Egy ilyen technológiai közeg lehetne a hálózati oktatási intézmények működésének a feltétele, ugyanúgy, ahogy az internetes kereskedelem sikerrel intézményesült az e-business alkalmazásokra támaszkodva a világhálón. És ha a virtuális piactéren valódi emberek valódi pénzért valódi tárgyakat vásárolhatnak egymástól, miközben igencsak valódi és releváns levélváltásokba bonyolódnak, s valódi védelemben részesülnek a csalások ellen, miért is ne lenne lehetséges, hogy a virtuális iskolában ugyanezek a valódi emberek valódi kommunikációs aktusok sorozatában intézményes oktatást valósítsanak meg. Szociológusként nem esik nehezemre a virtuális intézmények elképzelése, hiszen – a tudásszociológia elméleti keretét kölcsönözve – képes vagyok egyénként saját világunkat is a bensővé vált társas képzetekből felépülő virtuális világgént értelmezni. Informatikus mérnökökkel együtt dolgozva, tevékenységüket megfigyelve újra meg újra meg kellett tapasztalnom, hogy *szociológusként* vagyok erre képes, a hálózati oktatási fejlesztéseket és a hálózati oktatásról való beszédmódot azonban inkább a mérnöki képzelet határozza meg.

Először egy 2002-ben született írásban próbáltam meg bemutatni (Benda, 2002), vajon mi a probléma a mérnöki megközelítéssel, de azóta is egyre foglalkoztatott, hogy szociológusi eszközökkel megértsem és bemutassam az okokat, amelyek ezt a megközelítést ilyen mértékben képesek voltak bebetonozni. A megértés folyamata lassan haladt előre – ahogy az általában lenni szokott, amikor a képtelen dolgok szükségszerűségét kell megmutatni; amint azonban a képlet összeállt, megfogalmazódott

az alternatíva lehetősége is. Ebben a tanulmányban azt állítom tehát hogy paradigmaváltásra van szükség az e-learninges fejlesztésekben, a társas szemlélet fokozottabb bevonásával. Állításom minden bizonnyal kiterjeszhető más társadalmi alrendszerekre is, ahol a társas tényező jelentősége, az intézményesülés személyközi jellege hasonlóan erőteljesen jelenik meg – így például a politika és a szervezetek, a munka világának egyes részeire is.

A tanulmányban előbb bemutatom azokat a mérnöki gyakorlatokat és elképzeléseket, amelyek az e-learninges fejlesztéseket meghatározzák, és ezzel párhuzamosan arról is szó lesz, hogy miért és mennyiben alkalmatlanok ezek az eszközök a kitűzött feladatok megoldására. Az első részben először általánosságban a szoftverfejlesztésről szólok, és a második részben térek át az e-learninges fejlesztésekre, a területet meghatározó szabványok elemzése kapcsán. Végül a harmadik részben arról lesz szó, hogy milyen módon érvényesülhetne az a fajta szociológusi belátás, amelynek bevonása nélkül meggyőződésem szerint nem lehet sikeres a társadalmi intézmények virtualizálása, így az oktatás hálózattá válása sem.

I. rész: A szoftverfejlesztésről

Sommerville és Sawyer (1997) a fejlesztési folyamat kapcsán három meghatározó hátráltató tényezőt emel ki:

1. A fejlesztés során a feladat nem egyszerűen a jelenlegi állapot fenntartása (és informatizálása), hanem egy fejlettebb (a projekt jellegétől függően például hatékonyabb, gyorsabb, tartalmasabb) állapot elérése. A problémák kreatív megoldásokat igényelnek.

Ezt a kihívást a szakirodalomban szokás „fogós problémának” is nevezni (wicked problem). A fogalom a tervezéseméletből származik, Horst Rittel vezette be egy 1973-as tanulmányban (Rittel, 1973).

A fogós probléma meghatározásaként általában azt a mozzanatot szokás kiemelni, hogy az ilyen problémákat nem lehet előzetesen átlátni és megérteni, a megértés a megoldással párhuzamosan történik. Frappáns megfogalmazásban a fogós problémákat nem értjük meg igazán, amíg meg nem oldottuk őket. A megértés itt nem csak a probléma logikai dimenzióira vonatkozik, hanem sokkal inkább a különböző szempontokra, amelyeket a probléma megoldása során figyelembe kell vennünk, valamint a szempontok közötti komplex összefüggésekre. A megoldás horizontja nem belátható, amíg valamilyen megoldással nem állunk elő a problémára.

Az elmondottakkal összefüggésben a fogós problémák az alábbi jellegzetességekkel bírnak:

- A fogós problémáknak nincs egyetlen megoldásuk.
- A megoldás soha nem jó vagy rossz, helyes vagy helytelen – csak jobb és rosszabb, többé és kevésbé működőképes megoldásokról beszélhetünk.
- Minden fogós probléma egyedi és teljesen új.
- A megoldások mindig egyszerűek.
- A megoldás során nincs lehetőségünk alternatívák mérlegelésére. (Lásd Crabtree 2003)

2. A fejlesztés során sokféle, különböző háttérű szereplő igényeit kell összehangolni, a megrendelő szakembereitől a megvalósító informatikusokon keresztül a végfelhasználóig.

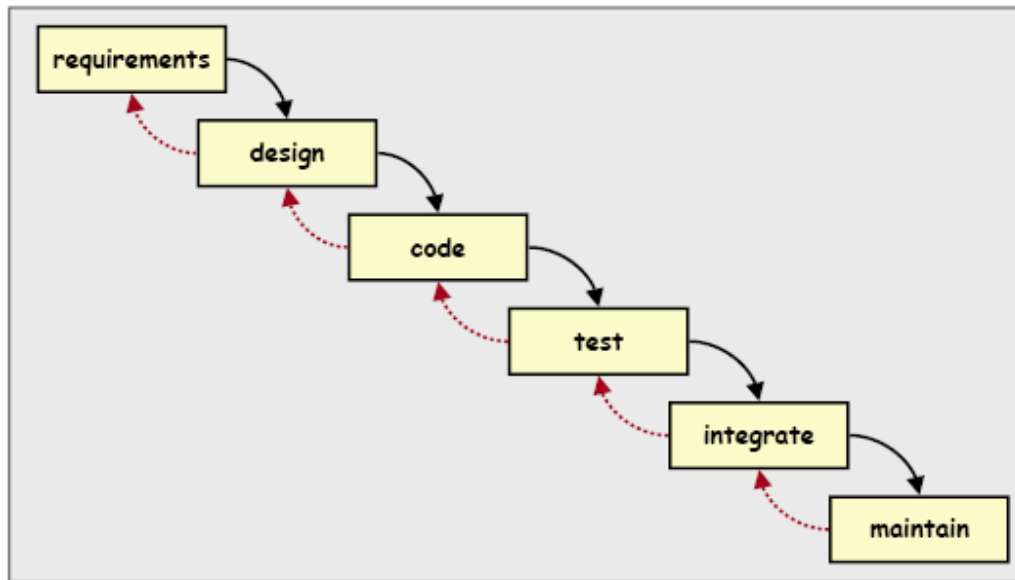
3. A fejlesztésben résztvevő különböző háttérű szereplők igényeinek összehangolása csak sorozatos kompromisszumokon keresztül lehetséges.

A szoftverfejlesztések számának és legfőképpen az egy-egy fejlesztéshez szükséges erőforrások volumenének növekedésével természetes módon jelent meg az igény a fejlesztési folyamat racionalizálására, ami többek között az ütemezési és pénzügyi tervezhetőséget, az eredmények megvalósíthatóságának illetve tényleges megvalósulásának kontrollját, és a biztonsági problémák tervezhetőségét érinti.

A szoftverfejlesztés életciklusában megjelenő feladatok:

1. Követelmények megfogalmazása – funkcionális specifikáció
2. Rendszertervezés (design) - rendszerterv
3. Kódolás, testreszabás és tesztelés
4. Bevezetés

A legrégebbi életciklus modell az ún. Vízesés modell (Waterfall Model) (Royce, 1970), amely a szoftverfejlesztésben felmerülő tevékenységeket egymás után következő, jól elválasztható lépésekben írja elő. A modellt bemutató ábrán látható, hogy miért a vízésésről kapta a nevét: a folyamat az egyes stádiumokon keresztül lépcsőzetesen halad a megvalósulásig.



(Az ábra forrása: Easterbrook, 2001)

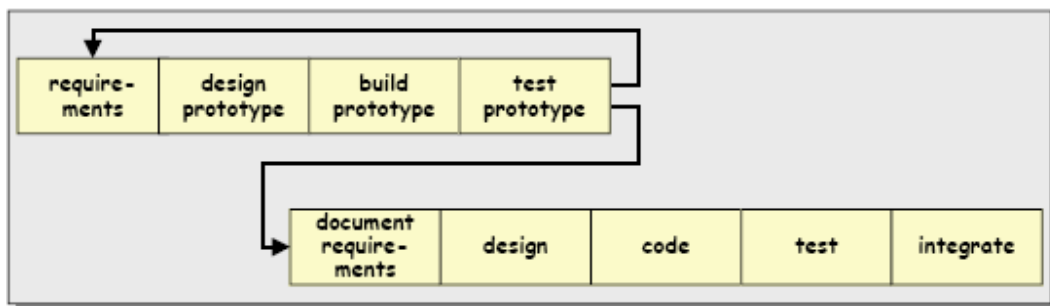
A vízesés modell

Az ábrán szereplő angol kifejezések fordítása rendre: requirements – követelmények; design – tervezés; code – kódolás; test – tesztelés; integrate – bevezetés; maintain – üzemeltetés, karbantartás.

Az ábrán az is látható, hogy a visszacsatolások, kiigazítások az egyes lépések között lehetségesek ugyan, de a helyi iterációktól eltekintve a modell alapján véve lineárisan írja le a fejlesztési folyamatot.

A Vízés modellt ma már meglehetősen elavultnak tekintik. Alkalmatlanságát magyarázza, hogy a fejlesztési feladatok fogós problémák, amelyekben a probléma felmérése a teljes fejlesztési folyamatra kiterjed, így nem választhatók el egymástól a fejlesztés különböző lépései. Ugyanakkor arra sincs lehetőség, hogy a fejlesztésben mindig minden feladatkör és szempontrendszer megjelenjen. A különböző feladatokat különböző szakmai háttérrel, munkamódszerrel dolgozó emberek látják el, ezért annak érdekében, hogy a fejlesztés ne váljon parttalanná, szükség van a folyamat szabályozására.

Az egyik megoldást a problémára a prototípussal dolgozó életciklus modellek jelentették, melyek keretében először a szoftver prototípusát fejlesztik ki, és a prototípus elemzése alapján újrakezdődik a folyamat, és elkészítik a végleges változatot.



(Az ábra forrása: Easterbrook, 2001)

A prototípus-modell

Az ábrán szereplő angol kifejezések fordítása rendre: requirements – követelmények; design prototype – prototípus tervezése; build prototype – prototípus kódolása; test prototype – prototípus tesztelése; document requirements – követelmények; design – tervezés; code – kódolás; test – tesztelés; integrate – bevezetés; maintain – üzemeltetés, karbantartás.

Ez a fejlesztési folyamat a későbbiekben is tovább folytatódhat, és a felmerülő igényeket a meglévő változatba integrálva új változatokat lehet kifejleszteni. Egy ilyen iteratív fejlesztési életciklus a Spirális modell.

A V-modell a Vizesés és Prototípus-alapú modellekkel szemben a fejlesztési feladatok eltérő absztrakciós szintjeit emeli ki, és olyan felépítést ír elő a fejlesztés számára, melyben az első fázis során az egyes stádiumokban létrehozott eredmények rendre meghatározzák a rákövetkező alacsonyabb absztrakciós szinten megvalósuló stádiumokat (a V leszálló ága), a követelmények megfogalmazásától egészen a szoftver kód előállításáig, majd a második fázisban (felszálló ág) megtörténik az elkészült szoftver tesztelése és bevezetése, hivatkozással a korábbi azonos absztrakciós szinten található stádiumokra.

A folyamat egészének racionalizációjával párhuzamosan az életciklusokban található feladatok belső racionalizációja is elindult, ez a folyamat azonban igen erősen technológiaközelit, vagyis erősen mérnöki jellegű maradt. A logikai-matematikai eszköztárral dolgozó fejlesztési modellek a szoftvertervezést és a kódolást fedik le (legismertebbek az adatmodellezés, a folyamatmodellezés, a tényleges megvalósítást előkészítő fizikai modellezés, valamint a modellek és a forráskód validitásának ellenőrzésére szolgáló formális leíró nyelvek és módszerek). Ezek a technológiai megközelítések *kimondott céljuk értelmében* arra szolgálnak, hogy az igényeket tudományos módon átfogalmazzák és így mérnöki eszközökkel uralhatóvá tegyék a fejlesztési feladatot. Azt hivatottak biztosítani, hogy a létrehozandó technológia megbízhatóan és hibamentesen működjön, vagyis a végtermékből a tervezés segítségével eliminálják a leállásokat, hibás működéseket és a biztonsági réseket. Ennek megfelelően erősen formalizáltak, szigorú szabályokkal működő leíró nyelveket használnak és mellőznek minden technológia-idegen szempontot. Ha immáron *külső szemlélőként* tekintünk az ilyen modellekre, azt találjuk, hogy feladatuk a fogós problémáknak a tudomány eszközeivel történő „megszelidítése”, vagyis az, hogy a fogós problémákat egyszerű, megoldható problémák sorozatára vezessék vissza.

Korábban, nagyjából a nyolcvanas évek elejétől fogva az Ember-Számítógép Interakció elnevezésű (Human Computer Interaction, HCI) irányzat keretében a technológiai horizont kiszélesítésére, az emberi tényező tudományos beemelésére is

történt kísérlet. A megközelítés annak a problémának a tervezhetőségére keresett modelleket, hogy miként érhető el, hogy a létrehozott technológia ne csak működőképes, hanem használható is legyen. A megoldást az ember-gép kapcsolat modellezésében kereste, gyakorlatilag ergonómiai célokat tűzött ki, és az atomisztikus egyén technológiahasználatára korlátozódott. Az irányzat formalizmusra hajló tudományos törekvései azonban nem váltották be a hozzájuk kapcsolódó reményeket, így a 90-es évek közepétől erősen háttérbe szorult, széles terepet engedve az ekkortájt felfutó etnográfiai megközelítéseknek.

Az etnográfia megjelenése a vállalati kooperációt támogató kooperatív rendszerek (CSCW, Computer-Supported Cooperative Work) felfutásához kapcsolható, a bebocsátását pedig minden bizonnyal a kvalitatív, etnográfiai megismerő módszerek szervezetkutatásban szerzett jó pozícióinak köszönheti. Az etnográfiai megközelítések azt hangsúlyozzák, hogy más szervezetfejlesztési beavatkozásokhoz hasonlóan a szoftverek alkalmazásánál is a meglévő mindennapi gyakorlatokból kell kiindulni, jó eredményeket pedig csak ezek megismerésével, megértésével és tiszteletben tartásával érhetünk el. Ez a megközelítés a formális, elvárásként jelentkező szervezeti működések helyett az informális, az informális mindennapi munkavégzési rutinok jelentőségét hangsúlyozza.

Az etnográfia hozzájárulása a szoftverfejlesztéshez gyökereinél fogva elsősorban megismerő jellegű. Andy Crabtree az etnográfia és a szoftverfejlesztés kapcsolatát bemutató könyvében a CSCW számára Sacks, Garfinkel és Geertz módszereinek követését ajánlja, a diskurzuselemzést, az etnometodológia módszereit és a sűrű leírást (Crabtree, 2003). Az etnográfia bebocsátása ellenére a tervezés továbbra is mérnöki módszerekkel történik, és a terepkutatások eredményeiből annyi érvényesülhet benne, amennyit a szoftvertervező mérnökök érzékenysége megenged.

A megismerő módszerek hozadéka, hogy a szoftvertervezők figyelmét felhívták a munka világának társas mozzanataira, aminek nem elsősorban a konkrét társas folyamatok megértésében van jelentősége, hanem a tervezést általánosabb szinteken meghatározó szempontok felvetésében (így például arra a problémára, hogy mérlegelés tárgya lehet, hogy a munkafolyamatokból mit automatizálunk, mit szorítunk technológiai korlátok közé, és mennyi mozgásteret hagyunk meg az emberi mérlegelés számára).

A szoftverfejlesztés azonban, mint korábban bemutattam, az esetek többségében fogós problémákkal szembesíti a fejlesztőket, amelyek megoldása a fejlesztési folyamat elején nem belátható. A fejlesztés jellegzetességei kapcsán korábban azt is kiemeltem, hogy olyan fogós problémamegoldási folyamatról van szó, melynek során sokféle szereplő elképzeléseit kell kompromisszumok sorozatán keresztül összehangolni. Ilyen körülmények között fontos lenne egy olyan közös nyelvezet, amely a társas világ szempontrendszerait képes szisztematikusan közvetíteni a megvalósítók felé. A társas fordulat azonban egyelőre csak a fejlesztés pereméig hatolt el, a tényleges problémamegoldást már a technológiai beszéd módok uralják. A technológiai beszéd módok pedig a társas viszonyrendszerek jellemzőit illetően nagy mértékben némák maradnak, vagyis nem teszik lehetővé számtalan kérdés és válasz megfogalmazását. Ez gyakorlatilag azt jelenti, hogy az etnográfiai megközelítés a fejlesztés folyamán csak esetlegesen, a fejlesztők egyéni érzékenységének függvényében érvényesülhet. Az ilyen megközelítés a felhasználói közegben gyakran kudarcra van ítélve; a felhasználók körében való bevezetés során elbukó szoftverek kapcsán a felhasználók felkészületlenségét, ellenállását, a megszokott eljárásokhoz való ragaszkodást – és nem a megalapozatlan fejlesztési megközelítéseket – szokás felhozni mentségül.

A fejlesztés persze nem csak vállalati közegben futhat zátonyra. Dessewffy Tibortól származik a Patyomkin-fejlesztés fogalma, mellyel azokra az állami vagy vállalati központi forrásokból (például kutatás-fejlesztési támogatásokból) finanszírozott, gyakran igen nagy volumenű szoftverfejlesztésekre utal, melyek életciklusa nem jut el a bevezetésig, az elkészült szoftvert nemigen használják semmire. Az ilyen alkalmazások olyanok, mint az elhagyatott kísértetkastély, Dessewffy metaforája azonban arra is utal, hogy nem valós, szándékoltan is látszólagos fejlesztésekről van szó. Meggyőződésem, hogy a Patyomkin-fejlesztésként aposztrofált alkalmazások nagy többségénél is inkább arról van szó, hogy a fejlesztési folyamat technológia-orientált közegben, a felhasználói igényeket és főként a felhasználói közeg társas viszonyrendszerait mellőzve zajlott le.

II. rész: E-learning és szabványok

Mindaz, amit eddig a szoftverfejlesztés életciklusairól és ezek erőteljes mérnöki beágyazottságáról elmondtam, az e-learninges szoftver- illetve rendszerfejlesztések esetén is megállja a helyét. Egy fontos különbségre érdemes azonban rámutatni az e-learning kapcsán. A szoftverfejlesztési folyamatok leírásánál azt a legegyszerűbb esetet vettük alapul, amikor saját célú vállalati vagy intézményi fejlesztés történik. Az ilyen saját célú fejlesztések a hálózati oktatás területén ritkák, az intézmények inkább kész megoldásokat vásárolnak.

Érdemes egy pillanatra a valószínűsíthető okoknál is elidőznünk:

- Az olyan oktatási intézményekben, amelyeknek nem fő profilja a hálózati oktatás, csak ritkán áll rendelkezésre egy teljes rendszer kifejlesztéséhez szükséges forrás.
- Az oktatási intézmények, oktatási feladatok és folyamatok között jelentős hasonlóságok fedezhetők fel, melyek részben a törvényi keretekre is visszavezethetők, így az oktatási tevékenységek vélelmezhetően tipizálhatók.
- Azok a vállalati szereplők, ahol a forrás rendelkezésre állhatna az oktatási fejlesztéshez, az oktatást általában nem tekintik a saját tevékenységi körhöz tartozó feladatnak, így inkább kívülről vásárolják meg.

Ilyen helyzetben a folyamat tervezési és fejlesztési része a testreszabás feladataira egyszerűsödik, melyre a követelmények megfogalmazása nyomán kerülhet sor. Testreszabás alatt nagyjából kétféle tevékenységet szoktak érteni; az egyik a rendszer paraméterezése, amely az e-learning szempontjából kevésbé izgalmas számunkra, a másik pedig a tartalommal való feltöltés. A tartalom alatt felhasználói megnyilvánulásokat és központilag előállított tartalmat is érthetünk, számunkra most az utóbbi terület lesz az érdekes.

Az úgynevezett e-learning keretrendszerekbe kerülő tartalom mindig a rendszer sajátosságaihoz alkalmazkodik. Hogy ez alatt pontosan mit is érthetünk, két konkrét keretrendszer gyors összehasonlításával szemlélhetjük:

A WebCT elnevezésű, USA-ban és Európában is jelentős piaci részesedéssel bíró keretrendszer lineáris vagy lineáris-hierarchikus rendbe szervezett html-oldalak befogadására alkalmas. A html-oldalak természetesen tetszőleges multimédiás elemeket tartalmazhatnak. A hazai fejlesztésű SDT ezzel szemben sokféle elemet tárol és rendszerez (képet, hangot, videót, szöveget, animációt, stb.) mindegyiket a hozzá illő

leíró információkkal látva el. Az elemek a metainformációk alapján egyenként kereshetők, de a tananyagok keretében a lineárisan egymás után fűzött tananyaglapokba ágyazva is lekérhetők, de a rendszer ilyenkor is az egyes elemeket fűzi egymás után.

A tartalmak inkompatibilitása már két rendszer alapján is világosan látható: az SDT elemeit csak feldolgozás után (például html-lapokba illesztve) tudnánk a WebCT-be betuszkolni, a WebCT html-oldalait pedig elemekre bontva helyezhetnénk el az SDT-ben. Ha azonban vennénk két másik keretrendszert, amelyek html-oldalakat illetve elemeket kezelnek, minden bizonnyal újra az eredeti problémába ütköznénk. A keretrendszer igényei szerint kialakított tartalmakon igazítani kellene, hogy új helyükre kerülhessenek.

Ez az alapprobléma hívta életre az e-learning szabványokat. A szabványosítás elsődleges célja, hogy átjárhatóságot biztosítson a tananyagok számára a különböző eLearning rendszerek – például LMS (learning management system, tanulásszervező keretrendszer) és CMS (content management system, tartalomszervező keretrendszer) rendszerek illetve szerzői rendszerek – között, tehát az, hogy mód legyen a tartalmak tetszés szerinti eszközökkel való menedzselésére.

A szabványok alkalmazását támogató érvelés sztenderd elemeket tartalmaz, melyeket széles körben elfogadnak, és ténylegesen is meghatározzák a szabványosítási törekvéseket az e-learning területén. A szempontokat Papp Gyula tanulmánya alapján ismertetem (2005).

Együtműködési képesség (Interoperability)

Az együtműködés szempontja két szinten érvényesülhet:

1. a tananyagok szintjén
2. az alkalmazások szintjén

A tananyagok szintjén azt jelenti, hogy amennyiben a tananyag megfelel egy adott szabványnak, akkor bármely – az adott szabványt támogató keretrendszerben lejátszható.

A keretrendszer szintjén az együtműködés egyrészt azt jelenti, hogy az alkalmazások képesek a megfelelő szabvány szerint kidolgozott tartalmak futtatására, kihasználva a tartalmakban rejlő szabványosan megvalósuló lehetőségek teljes körét.

Másrészt maga a keretrendszer és a benne található alkalmazások is képesek együttműködni más szabványos alkalmazásokkal (például az oktatási adminisztrációt megvalósító szabványos rendszerekkel).

Újrahasznosíthatóság (Reusability)

Az újrahasznosíthatóság elve azt jelenti, hogy a tanulás céljára felhasznált elemeket eredeti kontextustól független új kontextusba ágyazva is fel lehessen használni. Az újrahasznosíthatóságot úgy lehet megvalósítani, ha a kurzusként megvalósuló tananyagcsomagok kisebb, egymástól függetleníthető tananyagelemekből állnak (learning object, LO). Az újrahasznosítható tananyagelemek megnevezése angol eredetű mozaikszóval RLO (reusable learning object).

Testreszabhatóság (Flexibility)

A testreszabhatóság fogalma a tananyagelemek kialakításához kapcsolódik: fontos cél és követelmény, hogy a tananyagokat tetszőleges igények szerint formálhassuk.

Ez az igény az élethosszig tartó tanulás kapcsán a munkaerőpiaci igények diverzifikálódásával függ össze, és a végzettség helyett a kompetencia-alapú szemlélet terjedéséhez kapcsolódik.

Elérhetőség, kereshetőség (Accessibility)

Az újrahasznosítható tananyagelemek újrafelhasználásához, az egyénre szabott kompetenciaalapú tananyagcsomagok összeállításához szükség van a tananyagelemek releváns szempontok szerinti kereshetőségére, amit a tananyagelemeket leíró metaadatok segítségével lehet biztosítani.

Tartósság (Durability)

A hálózati technológiák állandó mozgásban, megújulásban vannak. Ebben a változó technológiai környezetben kellene a szabványoknak biztosítani azt, hogy a tartalmak ne avuljanak el a technológiával.

Költségmegtakarítás (Affordability)

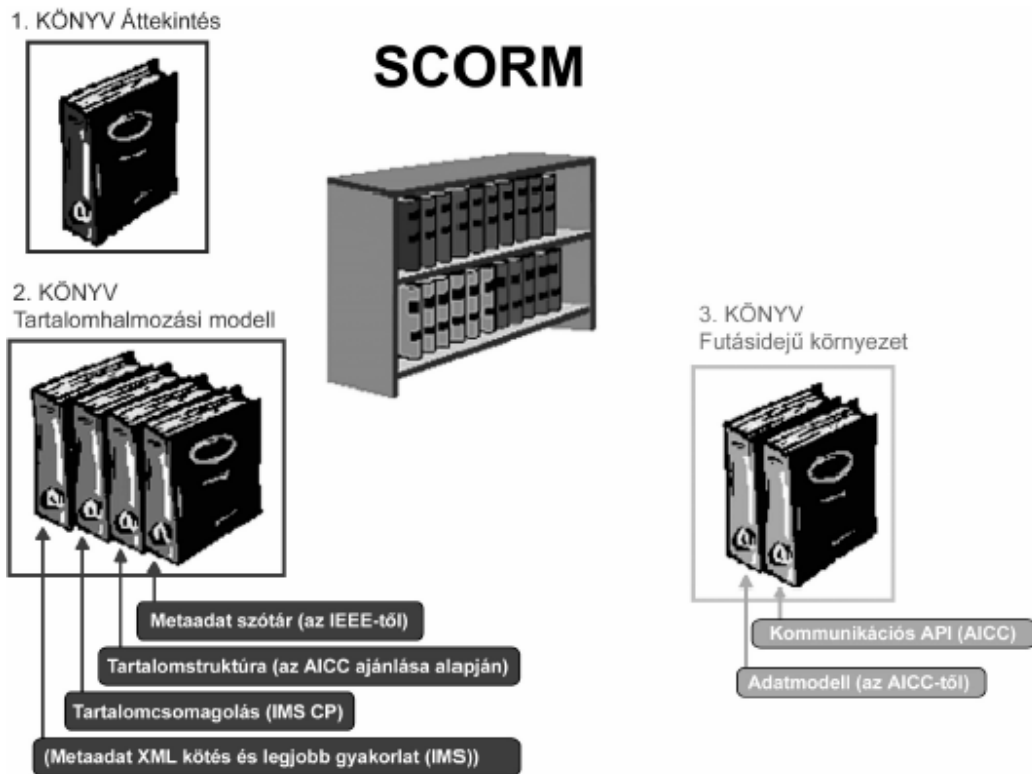
A felsorolt szempontok mögött egy kézenfekvő mögöttes igény húzódik meg, ez pedig a költségek lefaragására vonatkozik.

A tényleges e-szabványokat a tanulmány mellékletében sorolom fel, az elemzéshez számunkra elégséges a széles körben elfogadott szempontok ismerete.

A szempontokból is egyértelműen kiderül, hogy a szabványoknak egyszerre kell vonatkoznia a tartalmakra és az őket kezelő rendszerekre; nem csak szabványos tartalmakról, hanem szabványos rendszerekről is beszélhetünk. Ennek ellenére a szabványok (mindenekelőtt a *de facto* szabványként legszélesebb körben elfogadott SCORM, és az ismertebb európai szabványok, így például az AICC és a LOM) tartalom-centrikus modelleket alakítanak ki (lásd Papp, 2005), és nem jelenik meg bennük például a kommunikációs szemlélet, a változatos kommunikációs elrendezések támogatásának igénye. A szabványosítás hátterét az oktatási tömegtermelés kibontakozása adja, mely szintén az oktatási folyamatok egyszerűsítésének és egységesítésének irányába hat.

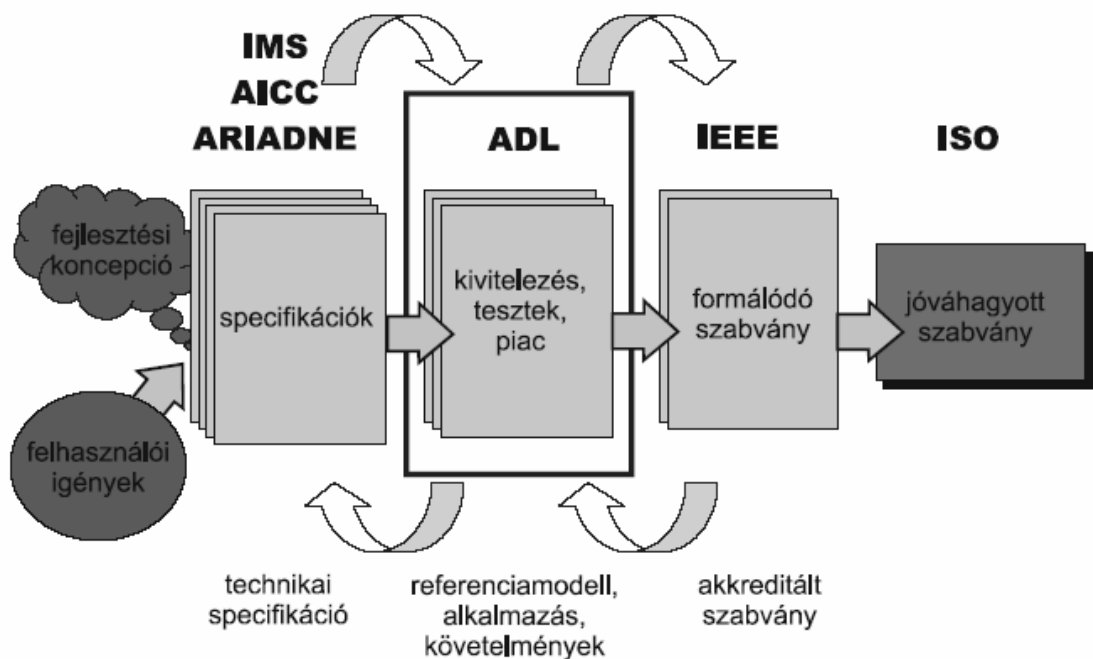
A szabványosítás követelménye mögött jelentős piaci érdekek tapinthatók ki – nem csak a vásárlóknak, hanem a keretrendszerek fejlesztőinek is elemi érdeke, hogy az oktatási célú digitális tartalmak éppen az általuk forgalmazott keretrendszerbe illeszkedjenek, és érdekük olyan szabványok megfogalmazása, amelyek a képviselt rendszerek létező mögöttes logikáját vésik kőbe (Hummel, 2004).

A tényleges szabványok technikai szemléletet tükröznek és gyakorlatiasak. Ez az állítás nehezen lenne teljes körűen bizonyítható anélkül, hogy a számunkra teljes mértékben érdektelen technikai részletekbe belebonyolódnánk, de szemléltetésképpen elég lehet a legáltalánosabban elterjedt SCORM szabvány áttekintése (Papp, 2005):



A SCORM a futásidejű környezetet (értsd: e-learning keretrendszer) adatmodell és API (application program interface azaz alkalmazás-program kapcsolat) segítségével írja le, a tartalmakra vonatkozóan pedig előírja a struktúrát és a metaadatokat, melyeket a metaadat-xml foglal egységbe. A szabványok – más műszaki előírásokhoz hasonlóan – az alkalmazott technológia nyelvén mindenre kiterjedő precizitással adják meg a kialakítandó digitális termékek jellemzőit.

Papp Gyula kitűnő áttekintést ad a szabványok elfogadásának folyamatáról is, mely lényegét tekintve a szoftverfejlesztés életciklusához hasonlít:



Követelmények → Tervezés → Kódolás, tesztelés → Bevezetés

“Egy megoldás szabványként történő elfogadása igen hosszú folyamat, s a potenciális alkalmazók konszenzusán kell, hogy alakuljon. A folyamat a szükségletek felméréséből indul ki, s bizonyos fejlesztési koncepciók mentén vázlatos specifikációk megfogalmazásához vezet. Ezeket nyilvánosságra hozzák, hogy minél szélesebb körben megvitathassák, és újabb szempontokkal bővíthessék, vagy éppen fordítva egyszerűsítsék a megalkotott modellt.

A végleges specifikációk megalkotásával kezdetét veheti a gyakorlatba való átültetés folyamata. E folyamatnak nem csak laboratóriumok, független szervezetek, hanem sok esetben piaci résztvevők is részesei. A szakma vezető vállalatai aktívan részt vesznek a specifikációk formálásában, és élen járnak azok implementálásában.

Sokszor már ebben a szakaszban kiderül, hogy melyek azok az ajánlások, vagy az egyes ajánlásoknak melyek azok a részei, amit piac befogad. Ilyen például az IMS számos specifikációja, illetve az ADL referencia modellje a SCORM.

Ha egy specifikációról bebizonyosodik, hogy a gyakorlatban is jól működik, továbbküldik egy szabványosítási testületnek, hogy kívülállóként tesztelje, véleményezze azt. Az egyeztetések, finomítások folyamata még ilyenkor is – akár évekig is –

elhúzódhat. Végül az akkreditált szabványt egy globális szabványosítási testület elé utalják (pl.: ISO), hogy nemzetközileg jóváhagyott szabvány váljék belőle.”

A szabványosítás elfeledett hetedik szempontja – az oktatás

A mégoly rövid áttekintéséből is kitűnik, hogy a szabványosítás célrendszerében nem jelennek meg az oktatás sajátos szempontjai, vagyis a taníthatóság, az oktatásra (tanulásra, tanításra, stb.) való alkalmasság. Ezek a szempontok szabványosítás előttként jelennek meg, a szabványosítási folyamat is úgy mutatja be őket, melyek a folyamat bemeneténél, a fejlesztési koncepciók és felhasználói igények formájában már adóttak.

A preferált tanulási folyamat azonban a tanulási objektumokhoz kapcsolódó szabványokban is jelen van, de rejtve marad. Az a fajta algoritmizált, programozott oktatás, amely az elterjedt szabványok kialakítását motiválta, csak egy a sokféle tanulási folyamat között. Az elterjedt e-learning szabványokkal szemben igyekszik alternatívát kínálni az IMS Learning Design Specifikációja (IMS 2003), mely a szabványok által megjelenített ún. programozott oktatás mellett öt másik elterjedt pedagógiai módszert sorol fel, melyeknek e-learninges változata is megvalósítható (kooperatív tanulás, kompetencia-alapú tanulás, tevékenységek segítségével történő, gyakorlati, tapasztalati tanulás – learning by doing –, probléma-alapú tanulás, csoportos beszélgetés). Ez a sorozat egyébként minden bizonnyal tovább bővíthető.

A SCORM szabványt leggyakrabban illető kritika, hogy a hálózati tananyagokat egyszerű lapozóskönyvvé silányítja. Hummel és társai (2003) a szabványosításban is nagy hangsúlyt kapó újrafelhasználható objektumok két igen meghatározó szellemi gyökerére is rámutatnak: egyik az objektum-orientált mérnöki és különösen informatikai megközelítés, amely az objektumok új környezetben, új platformokon való szabad újrafelhasználhatóságát jelenti, a másik pedig az a filozófiai irányzat, ami az emberi szellemet a gépek, és főleg a számítógép mintájára képzelel el. A felfogás szerint az emberi elme algoritmikus módon dolgozza fel az információkat, mint a számítógép, a tudáselemek pedig binárisan igazak vagy hamisak. A tanulásra szánt tartalmakat ezért algoritmikusan meghatározott szekvenciákba kell rendezni, és a tanulás eredményességét fekete-fehér, igaz-hamis kérdések formájában lehet ellenőrizni. Ezzel a gépies tanulás- és tudáselmélettel a pedagógiai megközelítések igen kicsiny halmaza azonosulna. A

megidézett tanulásmódszertanok például egyáltalán nem ilyen módon gondolkodnak a tudásról és a tanulásról, a tudást nem tekintik objektíválható, önmagában való entitásnak, hanem inkább az emberi elme részének, ami változatos kontextusokban, a tevékenység folyamán vagy diskurzusok formájában nyilvánul meg.

III. A fejlesztés nyelve

Ma már számtalan egyetemi és fejlesztői körben megfogalmazást nyert az a vélemény, hogy a szabványosítás célrendszerében létezik egy elfeledett hetedik szempont is, ez pedig a mögöttes pedagógiai-filozófia, az oktatásmódszertan, amit a gyakorlatra a learning design (tanulási folyamat) fogalmával szokás lefordítani. Ennek szabványosításba való beemelésére az IMS specifikációban tett kísérletet. Úgy tűnhetne, hogy a szabványok kialakításában egy ilyen hetedik szempont (illetve esetleg a többi területnek megfeleltethető további szempontok) szerepeltetése kiutat jelenthet az eddigi vétkes hallgatásból. Miért tapasztalhatjuk ennek ellenére mégis, hogy a pedagógiai szempontból érzéketlen és korlátozó szabványok hatalma töretlen, az e-learningről való beszédet döntően a mérnöki emberkép uralja, a magukra valamit is adó szereplők egyre kevésbé merik vállalni, hogy nem „szabványosak”, és legfőképpen az IMS specifikációjának megjelenése ellenére sem érzékelhető készülődés az újfajta módszertanok technológiai implementációjában?

A magyarázatot a fejlesztési folyamat korábban elemzett jelegzetességei körül érdemes keresnünk. A fejlesztési folyamat kapcsán az első részben kiemeltem, hogy a szoftverfejlesztés olyan, a megoldás folyamatában belátható „fogós probléma”, amelynek megoldása során különböző háttérű szereplőknek kell konszenzusra jutnia. A szereplők között az informatikus fejlesztő mérnökök mellett jelen lehet többek között a gazdasági-pénzügyi, marketinges, szociológiai-etnográfiai vonulat, a fejlesztési folyamatot ennek ellenére a mérnöki technológiai diskurzusok uralják. Ugyanez mondható el a szabványosítás folyamatáról is, mely az igények megfogalmazása nyomán tisztán technológiai megvalósítási folyamatként tárul elénk – annak ellenére, hogy fogós

problémáról van szó, melynek felmérése a gazdasági, marketinges és etnográfiai dimenziókban is csak a megvalósítás folyamatában bontakozhat ki.

A fogós probléma megoldása során nem lehet elégséges a nem technológiai szempontok bemenetként történő szerepeltetése, hiszen a fejlesztési folyamat kezdeténél a megoldás jellege még eredendően nem ismerhető. Mi sem egyszerűbb, válaszolhatnánk erre: a fejlesztés teljes folyamatába be kell vonni a különböző háttérű szereplőket! Ilyen módon valós lehetőség nyílik, hogy a megoldások a különböző szempontok figyelembe vételével születhessenek meg.

A megoldáskeresés egyik iránya a fentieket a társas helyzetre vonatkozó kihívásként fordította le, és különböző metodikákat javasolnak a többszereplős egyeztetésekhez. Egy példa az ún. dialogue mapping, azaz dialógus-követés (Conklin, 2005), melynek keretében a megnyilvánulásokat előre meghatározott notációk és struktúra szerint valós időben lejegyzik, és megmutatják a résztvevők számára. A struktúra leginkább a döntéselőkészítést támogatja, hierarchikusan elágazó alternatívákat és szempontokat mutat be. Valós idejű bemutatásával visszahat magára a beszélgetés menetére is, és a döntéselőkészítés szempontjai szerint strukturálja azt.

A beszélgetés menetének szabályozása azonban nem képes megoldást nyújtani arra a problémára, hogy a fejlesztésről való beszéd jelenleg teljes mértékben mérnöki. A társadalomtudományok beszédmódja idegen a fejlesztői kultúrától, és gyakran nem érthető a megvalósításban résztvevők számára (Crabtree 2003: 92-95).

Olyan formális módszerek kialakítására is történtek kísérletek, amelyek segítségével a terepmunka eredményeit a mérnökök formalizálni tudják, és így szisztematikusan képesek beemelni a tervezésbe. Az etnográfiai módszerek szoftverfejlesztésbe való integrálásával foglalkozó Crabtree az etnográfia pozícióit felmérve és nem kis mértékben saját kedvezőtlen tapasztalataiból kiindulva a terület eredményeit összegző könyvében felveti egy lingua franca kialakításának szükségességét, amely közös nyelvként szolgálhatna a változatos háttérű szereplők között (2003: 103).

Illusztrációként az ún. mintázatleíró törekvéseket (pattern language) mutatja be. Ezek retrospektív változata arra alkalmas, hogy az etnográfiai megfigyeléseket rendszerezett formában közvetítse a fejlesztők számára, akik azután az eredményeket interiorizálva saját maguk tervezik meg a fejlesztendő szoftvert. A prospektív változatok

már arra tesznek kísérletet, hogy irányokat mutassanak a fejlesztés számára, vagyis a tervezés folyamatai felett is viszonylagos kontrollra törekcszenek. A mintázatileírások a mindennapi élethelyzetekben visszatérő tipikus mintázatokot keresik meg, illetve mutatják be rendszerezett formában, és a fejlesztőktől ezek lehetséges mértékű automatizálását, gépi támogatását várják.

Szintén közös, a mérnöki elme számára érthető nyelvezetnek tekinthetjük a forgatókönyvek specifikálását, amely a Hollandiai Nyitott Egyetem (Open Universiteit Nederland) kutatásaiból vált ismertté. Az egyetemi kutatóknak az Európai Uniós Kaleidoscope hálózat keretében megjelenő ajánlása a kooperatív szkriptek kialakítására (Framework..., 2005) az alábbi elemeket tartalmazza:

- Résztvevők (Participants)
- Csoportok (Groups)
- A csoportok kialakítása (Group formation)
- Szerepek (Roles)
- Források (Resources)
- Tevékenységek jellege (Activities)
- Lépések (Sequencing)

(Az IMS-nek a kooperatív illetve kollaboratív oktatás támogatására elkészített szabványajánlása is ugyanezeket az alapelemeket tartalmazza).

A forgatókönyvekben a lépések segítségével szervezik sorrendbe a különböző szerepeket viselő, csoportokba sorolt résztvevők tevékenységeit és az ezekhez tartozó forrásokat.

A lingua franca, a társas viszonylatokat tekintve kifejező és a fejlesztők számára is értelmezhető nyelvezet kialakításának szükségessége a fejlesztési folyamat természetéből adódóan nyilvánvaló. A nyelvezet kialakítására vonatkozó törekvések azonban meglátásom szerint a kihívást leegyszerűsítve igyekeznek megoldani. Érthetőségüket az adja, hogy megmaradnak a részletközeli leírásoknál, elemi struktúrákat elemeznek, és ilyen értelemben a mérnöki-technológiai beszédmódokhoz hasonulnak. Nem tesznek mást, csak társas terminusokban újrafogalmazzák azt, amit később az informatikusok a gépek nyelvén fognak majd elmondani: a szekvencialitást, az automatizálást, a programozhatóságot. (Kitűnően szemlélteti ezt a forgatókönyvek

eredete is: a forgatókönyv fogalmát tekinthetjük a szociálpszichológiához tartozónak, de nagyon szorosan kapcsolódik a számítógépek megjelenéséhez és ezen belül is a mesterséges intelligenciakutatásokhoz, melyek keretében a gépeket igyekeztek felruházni az emberi gondolkodás jellegzetességeivel (Schank, 1975; Schank – Abelson, 1977).)

Úgy tűnik, hogy a szoftverfejlesztés és így az e-learninges fejlesztések területén a mérnöki-technológiai beszédmódok sikeresen magukhoz idomították az etnográfiai és szociálpszichológiai megközelítéseket. Ez pedig azzal jár, hogy számos probléma kimondására, megfogalmazására a kialakított keretek között nincs lehetőség. Az ilyen kimondhatatlan, nevenincs problémák közül saját tapasztalatom szerint hálózati alkalmazásoknál az alábbiak a legkritikusabbak:

- Miért és mennyiben fogják a felhasználók használni az alkalmazást?
- Hogyan intézményesül és rutinizálódik a használat a mindennapok során?
- Hogyan érvényesülnek az egymástól távol levő felhasználók jogai és kötelességei, mik az ellenőrzés, számonkérés mechanizmusai?
- Mi történik informális szabálykövetés esetén?
- Milyen a felhasználók egymás tevékenységeire és a közösen használt virtuális térre vonatkozó kölcsönös reprezentációja?

Az ilyen jellegű kérdések megválaszolása nem lehetséges analitikus eszközökkel, részletező folyamatleírásokkal, mert új minőségek – új folyamatok, újszerű viszonyrendszerek – elgondolását előlegzik meg (vagyis részben az ilyen kérdések miatt kell fogós problémáról beszélnünk). A válaszok nem is vezethetők le a folyamatok elemzéséből, inkább analógiás gondolkodást igényelnek, a megfigyelt folyamatokból általánosítható mögöttes mozgatók új folyamatokhoz való illesztését.

Túl kell tehát lépni a lingua franca gondolatán, amennyiben ez a mérnöki gondolkodásmódot tükröző nyelvezetet takar, és olyan beszédmód illetve diskurzív gyakorlat kimunkálására van szükség, amely alkalmas a társas viszonyrendszereket támogató hálózati rendszerek jellemzésére és tervezésük hatékony irányítására. A fejlesztésben jelentkező korábban bemutatott kihívásoknak megfelelően ennek a beszédmódnak az alábbi kritériumoknak kell megfelelnie:

1. **Kreativitás:** a beszédmód legyen alkalmas az alkotásra, a létező leírása mellett legyen képes a még nem létező, de lehetséges dolgok elgondolására és bemutatására is.

2. Alkalmas társas viszonyrendszerek és társadalmi intézmények működésének leírására és megértésére, illetve a megértés keretében a mögöttes mozgató elvek feltárására.

3. Nem a részletekre, hanem az egészre irányul, nem az elemek rendszerezésén, hanem az elemek rendszerének megértésén alapul. Nem (csak) analitikus megközelítésre alkalmas, hanem heurisztikus, holisztikus és analógiás megközelítésre is.

4. Nem az igaz/hamis dimenziók mentén láttatja a világot, hanem fokozatokban és összehasonlításban gondolkodik, melyek viszonyítási pontját a társas kontextusból nyeri, így képes az etikai dimenzió beemelésére is.

5. Eredményesen használható a szereplők közötti kommunikációban, de nem feltétlenül olyan módon, hogy mások megértik, hanem inkább úgy, hogy használója képes másokat megérteni, és mindenekelőtt a technológiai beszédmódokat rendszerezve áttekinteni, gyengeségeiket, hibáikat saját szempontjai szerint felmérni és ezekre rámutatni.

6. Fontos felismernünk és elfogadnunk, hogy nem szükséges, inkább csak kívánatos, hogy a megvalósításban résztvevők teljes egészében megértsék ezt a beszédet; sikeréhez vélelmezhetően az is elegendő, ha egy-egy probléma szemléltetésére alkalmas, ezért új fogalomkincs helyett köznapi fogalmakat érdemes használni, így a szemléltetések során támaszkodhatunk a fogalom eredeti, köznapi jelentésére.

7. Ez a beszédmód természetesen csak akkor lehet sikeres, ha képes a mérnöki-technológiai beszéd generalizált jelenlétét háttérbe szorítani, és saját legitimitásra szert tenni. A szemléletesség és közérthetőség ez utóbbihoz is hozzájárulhat.

Az ilyen megközelítés és beszédmód előzményeit a társadalomtudományon belül érdemes keresnünk. A szociológia és a szociálpszichológia lehetséges jelöltként merülhetnek fel, de az előbbi túlságosan tág diskurzív hagyományt képvisel, az utóbbi pedig – bár hétköznapi, közérthető fogalomkészletével hozzájárulhat egy ilyen beszédmód megteremtéséhez – irányultságában inkább magyarázó és nem megértő jellegű.

A bemutatott megközelítés jellemzőit legátfogóbban az etnográfiai hagyományban találhatjuk meg. Az etnográfiában nem szükségszerűen egy irányzaton belül, de jelen van az interpretáció, a rendszerszemlélet és a holisztikus megközelítés, az

összehasonlításra és hétköznapi nyelvezetre való törekvés. Nem arról van szó, hogy valamelyik etnográfiai elmélet megközelítését kellene a hálózati rendszerfejlesztésben alkalmazni, hanem inkább arról, hogy az etnográfiai hagyományban való jártasság olyan kompetencia, amely alkalmassá tehet valakit az irányító és közvetítő szerepre az ilyen projekteknél.

Az etnográfiai megismerő módszerek beágyazódása a fejlesztési módszertanokba megelőlegzi a megközelítés alkalmazásának sikerét, nem elég azonban a résztvevő megfigyelésnél megtorpanni, beérve a mérnöki szemléletmód kiszolgálásával. Az etnográfiai gondolkodást meg kell nyitni a képzelet előtt, és az elemzésben használt analógiás, holisztikus szemléletnek megfelelően a lehetséges virtuális világok elképzelésére is ki kell terjeszteni. Ehhez természetesen szükség van a technológiai lehetőségek mindenkor ismeretére, ami alatt korántsem a technológiai működés részleteit kell értenünk, sokkal inkább azt, hogy milyen elemi formákban képes a digitális technológia az egyéni és társas tevékenységeket támogatni, illetve ezek hordozójává válni. Így érhető el, hogy az etnográfiai képzelet már a technológiai megvalósításban fogalmazódhasson meg, kívánatos működésegyütteseket írjon le, melyekhez már csak az informatikai megvalósítás mikéntjét kell hozzárendelni.

Az etnográfiai képzelet elképzelt, elemzésekkel, részletek megfigyelésével és vázlatokkal tarkított útleírásokhoz hasonlóan mutathatná be a képzeletben megalkotott és bejárt digitális technológiákat, valóban egészen úgy, ahogy azt távoli tájak tényleges megfigyelésénél teszik.

Az ilyen jellegű beszédmódot természetesen nem csak a hálózati oktatás kialakítása során lehetne sikerrel alkalmazni, hanem minden olyan területen, ahol a társas viszonyrendszerek hálózati világba történő beemelése felmerül. Ilyen terület lehet az e-kereskedelem (melyben már idáig is jelentős sikereket értek el), a politika, és a szervezetek világán belül a tudásmenedzsment és a kooperatív munkavégzés. Az intézmények virtualizálásának programja a virtuális intézmények etnográfiájának megteremtésére támaszkodhat.

Melléklet: E-learning szabványok

Az ún. e-learning vagy LT (azaz learning technology) szabványok kialakítása a 90-es évek második felében kezdődött, számtalan kezdeményezést sorolhatunk fel. Az alábbi összegzés alapjául Hummel 2004-ben készült összefoglalása szolgált:

1. Ágazati konzorciumoknál

Legismertebbek ezek közül a nemzetközi tagsággal rendelkező IMS – Instructional Management Systems – konzorcium által kiadott szabványok, így a Learning Resource Meta-data (LRM), a Content Packaging (CP) – tartalomsomag szabvány, a Question and Test Interoperability (QTI), a Learner Information Package (LIP), Simple Sequencing (SS), Enterprise & Enterprise Service, és a Digital Repositories (DR) (lásd: <http://www.imsglobal.org/specifications.html>)

2. Szakértői testületeknél

IEEE LTSC (Learning Technology Standards Committee) – AICC, IMS, LOM

ADL (Advanced Distributed Learning) - SCORM

Prometeus (PRomoting Multimedia access to Education and Training in the EUropean Society, amely egy EU által támogatott szabványosítási kezdeményezés

ISO (International Organisation for Standardisation) – az Oktatási technológiáért az ISO-n belül a JT1- SC36 albizottság felelős (Joint Technical Committees 1 - subcommittee 36)

Ilyen szabvány a LOM (Learning Object Metadata) is, mely a tanulási objektumok metaadataira vonatkozik, IEEE 1484.12.1 számon jelenleg az egyetlen hivatalosan bejegyzett *de jure* nemzetközi eLearning szabvány.

Az USA Hadügyminisztériuma kezdeményezésére szintén szakértői testületnél jött létre a SCORM, azaz Sharable Content Object Reference Model , ami ma az egyik legáltalánosabban elfogadott *de facto* eLearning szabvány, amely mögött komoly piaci érdekek húzódnak meg.

3. Nemzeti és lokális szabványkezdeményezések

Franciaországban és Hollandiában léteznek nemzeti szabványok (melyeket az EU keretében normáknak neveznek), szempontunkból azonban fontosabbak a szabványosítási testületek, az amerikai ANSI, az európai CEN.

Bibliográfia

Benda Klára (2002): Minerva komputerbe költözik. In: Médiakutató, 2002 nyár.

Conklin, Jeff (2005): Wicked Problems and Social Complexity. In: Dialogue Mapping: Building Shared Understanding of Wicked Problems, John Wiley, New York. URL: <http://www.cognexus.org/wpf/wickedproblems.pdf> [2006. 01. 21.]

Conklin, Jeff (2001): The Age of Design. URL: <http://www.cognexus.org/ageofdesign.pdf> [2006. 01. 21.]

Crabtree, Andy (2003): Designing Collaborative Systems. A Practical Guide to Ethnography. Springer Verlag, London.

Hummel, Hans – Jocelyn Manderveld – Colin Tattersall – Rob Koper (2003): Educational Modelling Language and Learning Design: new challenges for instructional re-usability and personalized learning. In: International Journal of Learning Technology, 1, 1, pp. 110-121.

URL: <http://hdl.handle.net/1820/63> [2006. 01. 21.]

Framework for the Specification of Collaboration Scripts. Kaleidoscope Deliverable No. D29.2.1 (Final; 30.11.05)

URL: http://www.iwm-kmrc.de/cossicle/fr_index.html?resources [2006. 01. 21.]

Papp Gyula (2005): eLearning szabványok - Elemző tanulmány. In: eLearning szabványok. Javaslat az IHM által kiadott eLearning szabványajánlással kapcsolatos disszeminációra.

URL:

http://www.matisz.hu/tartalomfejlesztes/csatolmany/2005/03_osszefoglalo_tanulmany.pdf [2006. 01. 21.]

Rittel, Horst and Melvin Webber (1973) "Dilemmas in a General Theory of Planning," Policy Sciences 4, Elsevier Scientific Publishing, Amsterdam, pp. 155-159.

Schank, Roger C. - Robert P. Abelson (1977): Scripts, Plans, Goals and Understanding: an Inquiry into Human Knowledge Structures. L. Erlbaum, Hillsdale, NJ

Schank, Roger C. (1975): The structure of episodes in memory. In: D. G. Bobrow és A. N. Collins (szerk.): *Representation and understanding*. New York: Academic Press, pp. 237-272

Sommerville, I. – Sawyer P. (1997) Requirements Engineering: A Good Practice Guide. John Wiley, New York.

Hálózati hivatkozások:

IMS Learning Design Specification

http://www.imsglobal.org/learningdesign/ldv1p0/imslid_bestv1p0.html [2006. 01. 21.]

Steve Easterbrook prezentációja: Software Lifecycles

URL: <http://www.cs.toronto.edu/~ramona/teaching/csc444/f04/lec/lec4-4up.pdf> [2006. 01. 21.]