ILKKA TUOMI

# Internet, innováció és a nyílt forráskód: közreműködők a hálózatban

A szerző a közösség-alapú gyakorlati alkalmazás és tanulás modelljeit kombinálja a közreműködő-hálózat elmélettel, és a bevezetett fogalmak felhasználásával vizsgálja a nyílt forráskódú fejlesztési modell jellemzőit. A cikk elemzi a Linux és fejlesztőközössége növekedését és változását, illetve bemutatja, hogyan alakul át elsősorban közösség-központú gyakorlati eljárások ökológiájával megragadható közösségé. A konkrét elemzés felcsillantja a reményt, hogy a szerző módszertanával közelebb juthatunk a társadalmi-technikai változások dinamikájának megértéséhez.

**Szerzői információ:**

Ilkka Tuomi

Az Európai Bizottság Technológiai Kutatóközpontjának vendégelőadója. 1987 és 2001 között a Nokia Kutatóközpont tudományos főmunkatársa volt az Információs társadalom és tudásmenedzsment osztályon. 1999 júniusától 2000 decemberéig vendégelőadóként a Kaliforniai Egyetem (Berkeley) tanáraként dolgozott. Legújabb könyvében, az Innovációs hálózatok: változás és jelentés az Internet korában (Networks of Innovation: Change and Meaning in the Age of the Internet, Oxford University Press, 2002.), új, társadalomelméleti megközelítésből vizsgálja az innovációs folyamatot, valamint néhány jelentős, internettel kapcsolatos újítás és a nyílt forrású modell kialakulásának történetét elemzi.

**Így hivatkozzon erre a cikkre:**

Tuomi, Ilkka. „Internet, innováció és a nyílt forráskód: közreműködők a hálózatban”. *Információs Társadalom* II, 4. szám (2002): 118–141.

*https://dx.doi.org/10.22503/inftars.II.2002.4.6*

Ilkka Tuomi

# Internet, innováció és a nyílt forráskód: közreműködők a hálózatban

## Bevezetés

Az elmúlt néhány év során a nyílt forráskódú fejlesztési modell újságok címlapjain szerepelt (pl. DiBona, Ockman & Stone, 1999 Wayner, 2000; Leonard, 2000; Raymond, 1998b; Raymond, 1998a; Bezroukov, 1999; Kuwabara, 2000;) és a figyelem középpontjába került. E cikkek azzal érveltek például, hogy a nyílt forráskódú projektek jobb minőségű technológiát eredményeznek, mint a hagyományos vállalati kutatás és fejlesztés (Raymond, 1999). Ennek eredményeképpen sok vállalat jelentős befektetéseket hajtott végre annak érdekében, hogy a nyílt forráskódú fejlesztési modellből a legjobb gyakorlati megoldásokat[1] megpróbálják átvenni.

A hagyományos vállalati szoftverfejlesztés projektjeivel összehasonlítva a nyílt forráskódú projektek egyik megkülönböztető jellemzője a szellemi tulajdonjogok kezelési módja. A nyílt forráskód egyik kulcsfontosságú újítása a GNU Általános Nyilvános Jogosítvány (*General Public License*) (Stallman, 1999), ami lehetővé tette, hogy legálisan továbbfejlesszenek és alkalmazzanak mások által kifejlesztett szoftvereket, miközben a folyamatos továbbfejlesztést is ösztönözte.

A nyílt forráskódú modellt mindazonáltal már a szerzői jogok kidolgozása előtt is alkalmazták a programozás történetében. A CTSS (az MIT-n használt egyik korai időosztásos rendszer) számára készült operációs-rendszer programoknak például mintegy felét a rendszer felhasználói fejlesztették ki (Fano, 1967). A '60-as években az ARPANET projekt beindításának egyik motivációs oka az az elképzelés volt, hogy a különböző számítógépes helyszínek összekapcsolásával a számítógép-programozók közösségei hatékonyabban tudnák megosztani programjaikat és tudásukat (David & Fano, 1965; Abbate, 1999; Naughton, 2000). Valójában az ARPANET két legbefolyásosabb megálmodója, J.C.R. Licklider és Robert Taylor azzal érveltek 1968-ban, hogy az ilyen online közösségek nemcsak a számítógép programozást alakíthatnák át radikálisan, hanem a társadalmat, munkát és az emberi gondolkodást is. Bár a biztonságot és a magántitkot az online közösségek számára fontos kihívásnak tekintették, alapvető feltevésük az volt, hogy – adott hozzáférés-szabályozás mellett – a szoftver szabadon felhasználható és megosztható. (Licklider & Taylor, 1968).

A szerzői jogi megállapodásokat épp ezért úgy kell tekintenünk, mint egy olyan probléma megoldására született mechanizmust, amely nem elsősorban a szellemi tulajdonjoggal kapcsolatban merült fel. Valójában a szerzői jogok a nyílt forráskódú fejlesztési projektekben egy közismert probléma megoldását segítik. Vagyis a forráskódhoz való hozzáférés ösztönzi a tanulást, továbbfejlesztést és a más rendszerekkel való integrálást. A nyílt forráskódú szerzői jogi megállapodások egyik fontos funkciója

tehát, hogy a fejlődésnek ezt az útját nyitva tartják egy olyan gyorsan változó világban, ahol a kutatási és fejlesztési befektetések anyagi költségvonzatát egyre nehezebb előteremteni.

Általánosan azt is mondhatjuk, hogy a szerzői jog egy olyan társadalmi intézmény, mely a társadalom szereplőinek segít interakcióik irányításában és koordinálásában. Ezért feltehetjük a kérdést, hogy a szerzői jogok mellett milyen más mechanizmusokat használnak még a nyílt forráskódú projektek az innováció és fejlesztés ösztönzésére. S így eljuthatunk a társadalmi és a technikai rendszerek együtt-fejlődésének tanulmányozásához.

Jelen cikkünk a nyílt forráskódú modell elméleti magyarázatára tesz kísérletet. Először röviden áttekintjük a tudás és technológia társadalmi és gyakorlattal összefüggő kontextusokban megvalósuló fejlődésének konceptualizálására tett javaslatokat, majd bemutatjuk a közreműködő-hálózat elmélet néhány központi gondolatát. Ezt követően a Linux operációs rendszer evolúciójára vonatkozó adatokon keresztül bemutatjuk a nyílt forráskódú fejlesztési modell néhány fontos jellemzőjét. A tapasztalati eredményeket a bemutatásra kerülő elmélet fényében fogjuk értelmezni, és bizonyos feltételek biztosítását fogunk javasolni a nyílt forráskódú fejlesztési projektek sikere érdekében.

## Gondolat-közösségek: rövid áttekintés

Az elmúlt évtized folyamán egyre növekvő érdeklődés mutatkozott a technológia és a tudás társadalmi alapjainak megértése iránt. Többen azzal érveltek, hogy a tudás csak társadalmi kontextusban lehetséges, s hogy ezt a társadalmi kontextust a társadalmi gyakorlat teremti meg. E nézet szerint a tudás a közösségekben születik meg és bennük reprodukálódik, és a tudásnak csak e közösségeken belül van értelme. Mi több, e nézet visszautasítja azt a gondolatot, hogy a tudás elválasztható lenne a kontextustól, vagy hogy olyasmi lenne, ami bármiféle egyszerű módon egy „külső realitáson" alapulna. Ehelyett, e nézet a tudást a társadalmi folyamat eredményének tekinti. A tudás társadalmi szervezőerő, amennyiben intézményesíti a világ interpretálásának módjait. A tudás a társadalmi gyakorlatba, fogalmi rendszerekbe és a társadalmi gyakorlat során használt anyagi eszközökbe van beágyazva. A technológia, a társadalmi gyakorlat és a tudás kiegészítik egymást, és fejlődésük egyazon folyamat részeiként értelmezhető.

Bár a tudás ilyen jellegű felfogása az elmúlt években került előtérbe, az alapgondolat nem újdonság. Bahtyin már az 1930-as években amellett érvelt, hogy a beszéd és a szöveg csak a műfajok (*genres*) elemzésével érthető meg (Morson & Emerson, 1990). Bahtyin (1987) szerint a műfajok egy történelmi folyamat során jönnek létre, melyben a fogalmak, a használatuk és a gyakorlati kontextusuk koevolúciós folyamatban fejlődnek ki. Egy kompetens felnőtt a műfajok széles repertoárjával rendelkezik, melyeket különféle konkrét szituációkban használ. A családtagokkal beszélgetés, az előadások, a magánlevelek, az akadémikus kéziratok és a hivatalos dokumentumok mind sajátos műfajjal bírnak. A különböző műfajok kifejezéseinek lehet hasonló a formája és a hangalakja, de jelentésük csak az adott műfajon belüli szerepük elemzésével érthető meg. Minden egyes műfajnak továbbá megvan a hozzá kapcsolódó társadalmi közege és gya-

korlata, és a műfaj kifejlődése épp ezért szorosan összefügg a társadalmi gyakorlatok kifejlődésével, így például az e gyakorlatok során használt eszközökkel is. Egy műfaj ezért „mentális" és „materiális" elemekből áll. A kreatív munka – Bahtyin szerint – a műfajt formáló kulturális erőforrások hatékony használatát követeli meg.

Ludwik Fleck – szintén a '30-as években – egy hasonló javaslattal állt elő, ami a szifilisz, mint sajátos betegség-entitás történeti alakulásán végzett kutatásain alapult. Fleck (1979) kimutatta, hogy a tudományos tények hosszú történelmi folyamat során születnek, mely kölcsönösen összefüggő elméleti megfogalmazásokat, diagnosztikai gyakorlatokat és technológiákat hoz létre. Fleck szerint ezeket a megfogalmazásokat, gyakorlatokat, és technikákat a gondolat-közösségek hozzák létre, majd reprodukálják. Minden egyes gondolat-közösségnek megvan a maga gondolkodási stílusa, ami meghatározza, hogy mi lehet jelentéssel bíró az adott közösségben.[2]

Újabban Danoald Schön (1983) bemutatta azokat a tanulási folyamatokat, melyek az értelmiségi szakmák elsajátítását teszik lehetővé. Schön szerint az alkalmazható szakmai gyakorlat úgy is értelmezhető, mint reflektív gyakorlat, s az ilyen gyakorlat a szakmát gyakorlók közösségében tanulható meg és reprodukálható. Egy szakma megtanulása többnyire társadalmi interakciókon, és a megfelelő technikák kellő használatán alapul. Schön például amellett érvel, hogy a tervezőiroda az építészeti tervezés megtanulásának nélkülözhetetlen forrása, mivel sok fontos elsajátítható szakismeret szorosan összefügg a stúdióban hozzáférhető felszereléssel és anyagi eszközökkel. Az építészek közösségének tagjává válni annyi, mint megtanulni egyrészt egy építész szemével látni a világot, másrészt az építészek munkaeszközeit magas színvonalon használni. Mindez csak akkor következhet be, ha egy stúdió keretein belül megfigyeljük a tapasztalt építészeket és interakcióba is lépünk velük.

Yrjö Engeström (1987) válaszul kidolgozta a tevékenységrendszerek és expanzív tanulás elméletét a kulturális-történeti tevékenység-elmélet alapján (Vigotszkij, 1986; Leontyev, 1978; Wertsch 1991; Cole, 1996; Scribner, 1997; Engeström, Miettinen & Punamäki, 1999). A kulturális-történeti tevékenység-elmélet amellett érvel, hogy a társadalmi gyakorlatot eszköz-közvetítette tevékenységként kell felfognunk. A tevékenység maga csak a szociokulturális evolúción keresztül válik jelentéshordozóvá, s ezért minden jelentéshordozó emberi tevékenység inherens módon társadalmi kell hogy legyen. Engeström szerint minden újabb gyakorlat elsajátításához a meglévő tevékenység kiterjesztése szükséges, s ez feszültséget szül egyrészt a különböző, egymással interakcióban lévő tevékenységrendszerek között, másrészt az adott közösségen belüli régi és új tevékenységi formák között. Például mikor egy tevékenységrendszer olyan eszközöket hoz létre, melyek egy másik tevékenységrendszer tevékenységét közvetítik, az egyik tevékenységrendszerben végbemenő változások megkövetelhetnek bizonyos változtatást a másikban is.

Lave és Wenger a tanulás tevékenység-elméleti nézetét kulturális-antropológiai kontextusban alkotta meg, és javasolta, hogy a szociális tanulás alapegységén a gyakorlat-közösséget értsük. Lave és Wenger (1991; Wenger, 1998) szerint a tudás úgy sajátítható el, hogy egy gyakorlat-közösség legitim periferikus résztvevőjévé válunk, s hogy fokozatosan megszerezzük a tudást és a reputációt a társadalmi interakciók folyamatán keresztül. Lave és Wenger azt is állítja, hogy a tanulás alapvetően arról szól, hogy a közösség elfogadott tagjává válunk. A szakértelem, az identitás és a gyakorlat-közösségbeli tagság ezért elválaszthatatlan fogalmak.

A társadalmi-technikai evolúció terén végzett tanulmányaiból kiindulva Edward Constant (1987; 1984; 1980) amellett érvelt, hogy a gyakorlat-közösség a technikai gyakorlat helyszíne. Constant szerint a technológiai gyakorlat közösségei egyénekből vagy szervezetekből állhatnak. A technika általában egy meglévő gyakorlat-közösség keretein belül megvalósuló kiegészítő továbbfejlesztés révén fejlődik, de időnként a közösség olyan problémákkal is szembesül, ami csak radikális újítással oldható meg. A technikai gyakorlati hagyományok és a velük kapcsolatos gyakorlat-közösségek a tesztelhetőség magasabb szintű hagyományaira is támaszkodnak, beleértve az elfogadott eszközöket, munkafolyamatokat és értékeket. A kiegészítő továbbfejlesztés ezeket a peremfeltételeket adottnak veszi, de néha a radikális újítás a mérés és tesztelés teljesen új típusú rendszereit igényli. Ezeken a „magas-szintű" hagyományokon keresztül a specifikus technikai gyakorlat-közösségek az irányadó mérnöki kultúrába kapcsolódnak be. Így – Constant szerint – „a gyakorlat-közösségek tárgyiasítják a gyakorlat hagyományának jelentését önmaguk számára, a kívülállóknak pedig elmagyarázzák és igazolják e hagyományt". [3]

Constant azt is javasolta, hogy a technikát a tudás egy fajtájának tekintsük. Hasonlóképpen, Knorr-Cetina (1999) feltevése szerint a tudományos gyakorlati alkalmazásokat úgy is értelmezhetjük, mint „megismerő kultúrákat" melyek összekötik az eszközöket, a tudást és a tudás megszerzésének sajátos mechanizmusait.

A gyakorlat-közösségek fogalma az utóbbi időben nagy érdeklődést váltott ki a szervezet és innováció elméletek területén (pl. Sawhney & Prandelli, 2000; Brown & Duguid, 2000a; Kuusi, 1999; Tuomi, 1999b). Brown & Duguid (1991; 2000b) felvetette, hogy a szervezeteken belüli tanulás és újítás a gyakorlat-közösségekben s azok között jön létre. Tuomi (1999a) összekapcsolta a gyakorlat-közösségek irodalmát a kulturális-történeti tevékenység elméletével és Nonaka és Konno (1998; Nonaka, Toyama & Konno, 2000) tudás-teremtő „terek" modelljével, és javaslata szerint a szervezeteket úgy értelmezhetjük, mint a kölcsönösen összekapcsolódó tevékenységi rendszerek „fraktál közösségeit".

Jelen cikkünkben mindezek az elméleti javaslatok adják az adatok elemzésének elvi alapjait. Közös jellemzőjük az, hogy az emberi közösségekre mint az újítások helyszínére fókuszálnak, és amellett érvelnek, hogy a tudás, a gyakorlat és a technikai eszközök egy fejlődésben lévő társadalmi rendszer kölcsönösen függő részei. A közösség ilyen fogalma ezért eltér azoktól a megfogalmazásoktól, melyek a közösséget emberek egy csoportjának tekintik. Ehelyett a közösség úgy merül fel bennük, mint egy olyan valami, ami nem jöhet létre pusztán a megfelelő számú egyén összeterelésében. Ellenkezőleg, az egyének csak a különböző közösségekhez fűződő tagságuk révén válnak egyéni identitással bíró személyekké. Az identitás – más szavakkal – nem olyasvalami, amit az egyedi személy „tulajdonságainak" valamiféle listája alapozna meg. Ehelyett a közösségekben van megalapozva, azok sajátos tevékenységrendszereivel és kollektív jelentés értelmezésével.[4]

Ilyen kontextusban a nyílt forráskódú fejlesztési modell nem csupán szoftvert hoz létre. Létrehozza a tudás, a tanulás és a cselekvés interaktív rendszerét, és ezzel megszervezi a közösséget és kapcsolatait más közösségekkel. Valójában, ahogy azt az alábbi empirikus elemzés bemutatja, a nyílt forráskódú fejlesztési modell a közösségek és a technikák egy heterogén hálózata. E modell jellemzője, hogy – megfelelő körülmények között – a technikai fejlődés rendkívül gyors lehet.

## A közreműködő-hálózat elmélet és a komplexitás redukálása

A nyílt forráskódú fejlesztési modell jellemzőinek leírásához célszerű bemutatni a közreműködő-hálózat elmélet néhány kulcsfogalmát. A közreműködő-hálózat elmélet szerint a társadalom emberi és nem-emberi közreműködők (*actors*) hálózataiból tevődik össze (Latour & Woolgar, 1986; Bijker & Law, 1992; Callon, Law, & Rip, 1986; Latour, 1999). Minthogy a közreműködők a hálózatban lehetnek emberek és nem-emberek, a közreműködő-hálózat teoretikusok néha az *aktant* terminussal utalnak az ilyen közreműködőkre. A társadalom, a szervezetek, a szereplők és a gépek mindannyian a közreműködő-hálózat interakcióinak produktumai. Egy személy például nem fogható fel elszigetelt entitásként, hanem úgy, mint aki hozzákapcsolódik az erőforrásoknak és külső tényezőknek a személyt adott vonatkozásban meghatározó heterogén hálózatához.[5] A műszerei, laboratóriuma és társadalmi kapcsolatai nélkül egy tudós például elveszíti identitását, mint tudós.

A közreműködő-hálózat elmélet a tudományos gyakorlat tanulmányozásából fejlődött ki, de a társadalmi jelenségek megértésének általános keretévé vált. Egy tudományos laboratóriumot tekinthetünk olyan hálózatnak, amelyet lombikok, jegyzőkönyvek, tudományos publikációk, költségvetés és kutatók alkotnak, amelyeknek mind megvan a maga „kompetenciája" és „ellenállása". A tudományos ismeret ebben a hálózatban jön létre, és maga is közreműködővé válik a folyóiratokban megjelenő újabb megfogalmazásokon és megfigyeléseken keresztül, vagy például azáltal, hogy beágyazódik tudományos segédeszközökbe és szoftverkódokba. A többi társadalmi intézmény evolúciója is hasonló folyamatban megy végbe: a családok, a szervezetek, a számítástechnikai rendszerek, a gazdaság és a technika fejlődése egyaránt.[6]

A közreműködő-hálózat elmélet egyik kulcsfogalma a „fordítás". A közreműködők egész rendszere a teljes társadalmi hálózatban rendkívül komplikált. E komplexitás redukálása épp ezért a gyakorlati cselekvés szükséges feltétele. A fordítás egy olyan folyamatot jelent, mely során a komplikált alrendszereket *actant*-ok (közreműködők) reprezentálják, s ami eredményeképpen a komplex struktúra gyakorlati szempontból „fekete dobozként" szerepel. Például időnként beszélhetünk a „brit kormányról" anélkül, hogy ismernünk kéne annak konkrét működéseit és azt, hogy kik is tulajdonképpen a tagjai. Hasonlóképpen egy egész szervezetet képviselhet egyetlen személy, és a számlázási eljárások komplex rendszerét reprezentálhatja egy szoftvercsomag.

A fordítás azt jelenti, hogy komplex alhálózatok „pontszerűvé" válnak, és egységes entitásként kezdenek el működni az alhálózattal kapcsolatba kerülő közreműködők szempontjából. Ugyanekkor az ilyen lefordított alhálózatok erőforrásokká válnak. Például egy már létező tudományos műszer anélkül is használható, hogy ismernénk mindazokat a folyamatokat, tudást és az egyéb erőforrásokat, melyek a gyártásához szükségesek. A fordítás ezért azt jelenti, hogy komplex hálózatokat adottnak tekinthetjük. Ugyanekkor azt is jelenti, hogy a fordítás helye a hatalom és kontroll centruma is egyben. A lefordított alhálózatok hatásai erőforrásokká válnak, melyek lokalizálhatóak és kontrollálhatóak. A fordítás e folyamatában a pontszerűvé tett hálózat felfogható úgy, mintha a fordítást végző közreműködő birtokolná azt.

A közreműködő-hálózat elmélet szerint a fordítás éppen zajló folyamatai a társadalmi rend fő forrásai. A fordítás rendteremtő hatásokat eredményez, úgymint szervezeteket, intézményeket, eszközöket és ágenseket. Mindezeknek megvan a maga

ellenállásuk, és a társadalmi változás ezért leginkább arról a küzdelemről szól, melyben az erőforrások és a viszonyok újraszerveződnek a közreműködő-hálózaton belül. Ebben a folyamatban az ellenállások előre láthatóak, és feloldásuk érdekében különböző stratégiák vonultathatóak fel. A fennálló rend felbomlásának folyamatos fenyegetése kísért, s a tény, hogy mégis van rend, arra utal, hogy működnek – legalábbis egyfajta pragmatikus értelemben – a stratégiák és a fordítási folyamatok, és egy viszonylag stabil rendszert hoznak létre. [7]

## A Linux-fejlesztő erőforrások evolúciója

A társadalmi-technikai rendszerek evolúciójának megértéséhez e pillanatban két ellentmondó javaslat áll rendelkezésünkre. A közösség-alapú nézőpont szerint a tudás, a technika, és a tanulás a gyakorlattal kapcsolatos közösségekben jön létre, és a gyakorlat az anyagi és technikai eszközökön alapul. E kontextusban a tanulás mind szocializálja a közösség tagjait – ahogy arra Lave és Wenger mutatott rá –, mind pedig a tevékenység új formáit és új termékeket teremt, ahogy azt Engeström állította. A közreműködő-hálózat elmélet ezzel szemben azt állítja, hogy az emberi és nem-emberi közreműködők szimmetrikusak, és hogy gyakran felcserélhetőek egymással. A fő elképzelés az, hogy az alhálózatok komplexitása redukálható a fordítás folyamatában, ami lehetővé teszi, hogy egyetlen *actant* (közreműködő) képviseljen egy egész alhálózatot.

E két nézőpontot egymáshoz illesztve megmutatkozik, hogy miképpen lehet mindkét megközelítést finomítani és felhasználni arra, hogy leírjuk általuk az olyan társadalmi-technikai rendszerek evolúcióját, mint például a Linux. Azok a segédeszközök, melyeket a társadalmi gyakorlatban használunk, a segédeszközöket létrehozó komplex alhálózatok fordításai, miközben ezzel párhuzamosan önmagukat is létrehozzák, mint a tudás és a vonatkozó gyakorlat hordozóit. Ameddig a technika nem omlik össze, felhasználói alkalmazhatják a technikát, mint eszközt. Egy ilyen tárgy valójában fekete doboz, ami közvetíti a felhasználók tevékenységét, anélkül, hogy rákényszerítené a felhasználót arra, hogy ismerje mindazt a komplex kapcsolatrendszert, ami rejtve marad az eszköz milyenségét meghatározó rendszeren belül. Például, ameddig minden rendben van, a számítógép felhasználója nem kell, hogy tudjon az elektromos és digitális tervezésről, vagy a programok felépítéséről, vagy hogy a gyakorlatban hogyan fejlesztik és gyártják ezeket a dolgokat, mintsem inkább arról, hogy hol találhat egy olyan szakembert, aki tényleg tudja, mi van a dobozban.

Az alhálózatok „fekete-dobozolása" folyamatában, a fordítói folyamatok nem csak az anyagi összetevők komplexitását rejtik el. A fekete-dobozolás a társadalmi hálózatokat és diskurzusokat is elrejti. Mindazonáltal a fordítás több különböző módon is végbemehet. Ha a fekete-dobozt egy konkrét termék képviseli, a fekete-dobozt „eszköznek" tekinthetjük. Ha egy ember képviseli, a fekete-dobozt „szervezetnek" tekinthetjük. Ha a fordítási folyamat mentális terméket hoz létre, a benne foglalt rendszert tekinthetjük „fogalomnak".

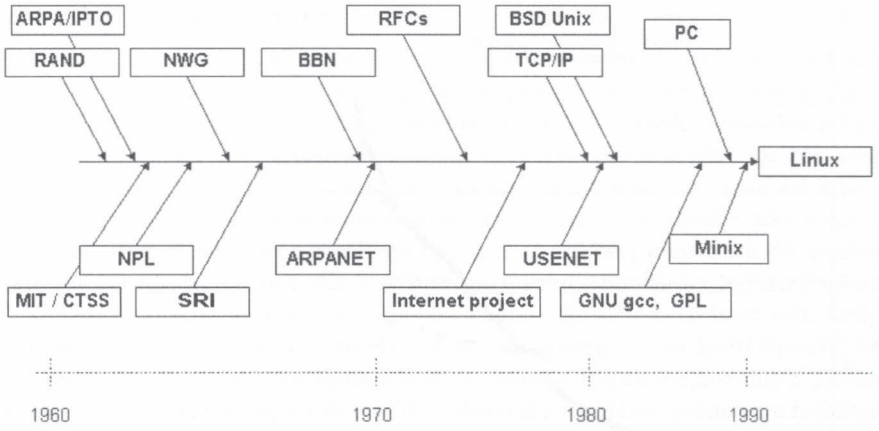
E véleményünket illusztrálhatjuk a Linux operációs rendszer fejlesztése történetének vázlatos bemutatásával.

A Linux történetét többnyire azzal kezdik, hogy a Linux akkor született meg, mikor Linus Torvalds az első verzióját kifejlesztette és megosztotta másokkal, 1991-

ben. Sok szempontból a Linux máig fejlesztés alatt álló befejezetlen mű, és folyamatosan csiszolódik. A Linux megszületésében számos korábbi fejlesztés is fontos szerepet játszott. A Linux fejlesztési folyamata például nagyban támaszkodott a már működő Unix operációs rendszerre, különösen annak BSD és Minix variánsaira, az Interneten működő hírcsoportokra és levelezési lista kiszolgálókra, valamint a GNU c-compilerre és annak programkönyvtáraira, és a GNU Általános Nyilvános Jogosítványára (*General Public License*). Mielőtt a Linux-fejlesztés mint közös együttműködő munka beindult volna, már sok technika- és tudásteremtő közösség volt „lefordított" úgy, hogy a Linux erőforrásaként szolgálhasson.

Az *1. ábra* mutatja a Linux fejlesztő-közösség számára erőforrásként használható néhány fontos közreműködőt. E közreműködők részben olyan közösségek, melyeket „szervezeteknek" vagy „társadalmi hálózatoknak" tekinthetünk, illetve olyan anyagi eszközök és fogalmak melyeket közösségek hoztak létre. Például, a csomagkapcsolt számítógépes kommunikációs protokoll fejlesztésének korai fázisában a viszonylag informális *Network Working Group (NWG)* (Hálózati Munkacsoport) megvitatta a számítógép hálózatok lehetséges alkalmazásait és kifejlesztette a gazdagép-gazdagép (*host-to-host*) típusú protokollok első specifikációit. E viták eredményeit később a *Request for Comments (RFCs)* füzetekben terjesztették, kezdetben nyomtatott formában, később magán az ARPANET-en (Braden, Reynolds, et al., 1999). A *Request for Comments* mechanizmus az Arpanet és az Internet fejlesztésében hasonló szerepet játszott, mint a forráskód a nyílt forráskódú projektekben (Bradner, 1999; Naughton, 2000; Abbate, 1999).

Az *1. ábra* bemutatja, hogy több olyan fontos közösség is működött, amely a Linux-fejlesztés elindítását lehetővé tevő erőforrásokat produkált. Lehetetlen volna e közösségek természetét bemutatni jelen keretek között.[8] Az *1. ábrán* szereplő közreműködők különböző típusainak megkülönböztetésekor mindenesetre feltűnik, hogy némelyek közülük szervezeti közreműködők, mint a ARPA/IPTO, azaz az *ARPA's Information Processing Technologies Office* (az ARPA Információ Feldolgozás Technikai Irodája), ahol a látnoki képességű vezetés és a pénz összpontosult; mások technikai segédeszközök, mint az ARPANET, amelyre az együttműködésen alapuló munka fogalmai és eszközei kifejlesztésében támaszkodtak; megint mások pedig fogalmi segédeszközök, mint a TCP/IP protokoll definíciói, melyeket a *Request for Comments* füzetekben dokumentáltak. Az egyetlen, az ábrán szereplő üzleti szervezet a *Bolt, Beranek and Newman* (BBN), az a cég, amely az ARPA/IPTO megrendelésére kifejlesztette az ARPANET számára az interfész üzenet-processzorokat. Míg néhány egyéb közreműködő szervezet szintén piaci cég, ezek főként passzív struktúraként működtek, melyek a valódi közreműködők hátteréül szolgáltak, mint ahogy a Unix fejlesztőinek az *AT&T Bell Labs.*

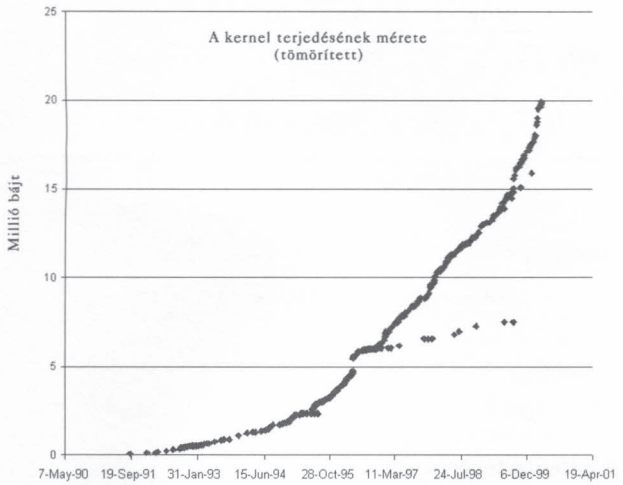



*1. ábra:* a korai Linux fejlesztésben fő erőforrásként közreműködők

Bár az *1. ábra* az okság és a megkerülhetetlen eseménysor látszatát keltheti, a közreműködő-hálózat evolúcióját természetesen nem a Linux későbbi sikerének előrevetítése irányította. Az ilyesféle hálózaton belüli változásokat a különböző közreműködők fokozatos mozgásának kell betudnunk. Egy ilyen rendszerben az evolúció iránya egybeesik a leggyorsabb elmozdulás irányával. Ezért általában több úton is fejlődik egy összetett hálózat. Például az *1. ábrán* szereplő erőforrások nem csak a Linux fejlesztői számára jelentettek erőforrást, hanem más Internet-közeli közösségek számára is.

## A Linux növekedése

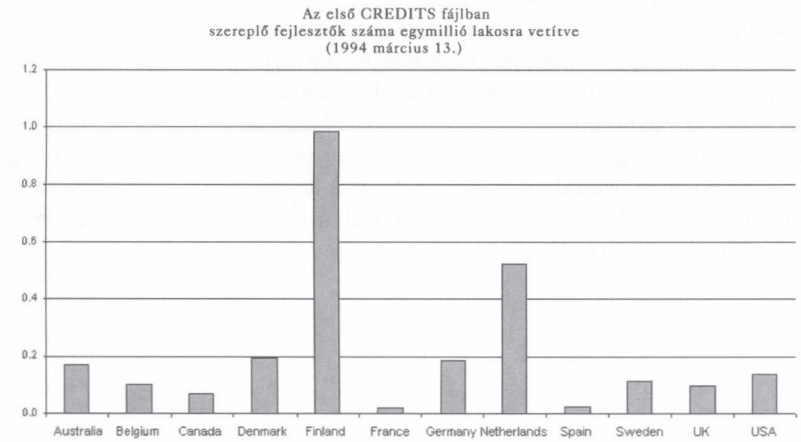
E rövid elméleti áttekintés lehetővé teszi számunkra, hogy a Linux és fejlesztőközössége evolúcióját bemutassuk. Amikor a Linux forráskód fejlesztése 1991-ben elkezdődött, a meglévő erőforrások nagyon gyors növekedést tettek lehetővé. E növekedés még mindig tart, ahogy az a *2. ábrán* is látható. A központi operációs-rendszer, a Linux kernel* szinte exponenciálisan növekedett. Ez azért is jelentős eredmény, mivel egy jó minőségű operációs-rendszer kernel-hez a kódot a sebességhez kell optimalizálni, és mivel az együttműködő fejlesztéses módszer azzal jár, hogy mindenekelőtt a forráskódot annyira egyszerűvé kell tenni, amennyire csak lehetséges. A forráskód növekedése ezért nem azáltal megy végbe, hogy a kernel-hez véletlenszerűen új képességeket adnak hozzá. Ehelyett, ahogy alább látható, a növekedés a kernel rendkívül szervezett kiterjesztésével jön létre.




**2. ábra A Linux *kernel* növekedése**

## Linux fejlesztő-közösség

Már a Linux történetének hajnalától kezdve a fejlesztési folyamat együttműködésen alapult. Ennek az együttműködésnek különleges vívmánya volt, hogy szinte teljesen Internet-alapú eszközökre támaszkodott. Szemben azokkal az érvekkel, hogy a hatékony virtuális együttműködéshez elengedhetetlen a valós szociális interakció, a Linux fejlesztő-közösség csaknem teljesen virtuális volt és maradt. Mi több, eleve virtuális közösségként indult. Ennek eredményeképpen a közösség képes volt olyan tagokkal bővülni, akik földrajzilag nagy távolságban élnek.[9] Ez látható a *3. ábrán.* Az ábra bemutatja a legfőbb Linux fejlesztők országok szerinti megoszlását, millió főre vetítve. Az adatok az első, 1994-es, a fő szerzőket rögzítő CREDITS fájl[10] elemzésén alapulnak.




**3. ábra: A különböző országokban működő korai legfőbb fejlesztők**

## A forráskód „leülepedése"

A Linux fejlesztése első éveiben, a forráskód olyan felület volt, amelyet folyamatosan fejlesztettek. Mikor a Linux már életképes operációs-rendszer lett, elkezdték olyanok is használni, akiket „végfelhasználóknak" hívhatunk. Számukra a Linux nem az interakcióban lévő forráskód modulok és programozási segédeszközök komplex rendszereként jelent meg. Számukra a Linux erőforrássá vált. Továbbá a Linux terjesztők egybecsomagolták az operációs-rendszer kernel-t különféle alkalmazásokkal és segédprogramokkal, és a hatékony terjesztés a szoftver konfigurációk hatékony menedzselését követelte meg. Ez feszültséget teremtett a Linux fejlesztő modellben. Néhány „felhasználó-fejlesztő" számára a Linux egy olyan rendszer volt, melybe gyakran új elemek kerültek be, s amely érdekes lehetőségeket nyújtott újszerű, nagy hatású fejlesztésekre. Az ilyen felhasználók számára a Linux megmaradt a szoftvermodulok és műveletek komplex és fejlődő hálózataként. Mások számára, e hajlékonyság gondot jelentett. A folyamatos változás megzavarta a fordítási folyamatokat és nehézkessé tette azt, hogy a Linuxot erőforrásként használják.

A véghasználók és a fejlesztők közti feszültségnek köszönhetően a Linux fejlődése két ágra szakadt, s e két különböző ösvényen haladt tovább. Az egyik ösvény a „stabil" ág, ahol csak a legalapvetőbb változtatásokat vezetik be. A másik ösvény a „fejlesztői" ág, ahol az újabb, hasznos kódelemeket folyamatosan beépítik a rendszerbe. A forráskód különböző ágait a *2. ábra*, illetve részletesebben a *4. ábra* mutatja be. Ez a folyamat annyiban érdekes, amennyiben megmutatja, hogy miként lehetséges, hogy egyazon termék, miközben lefordítódik erőforrásként, ugyanakkor megmarad folyamatosan evolváló hálózatként is. Az igény, hogy a Linux kódot más közösségek számára erőforrásként legyen lefordítva, megteremti a kód „leülepedett" vagy „fekete dobozos" változatát. A közreműködő-hálózati elmélet terminológiája szerint a fejlődési ösvény elágazása a rendszer fekete-doboz verziójának létrehozása irányába egy olyan fordítási stratégia, ami a végfelhasználók és a fejlesztők közötti feszültséget csökkenti. A stabil ösvényen a fordítási folyamat változatlan maradhat, amíg egy radikálisan új verziót nem fejlesztenek ki, amely közhasználatú erőforrássá válik.

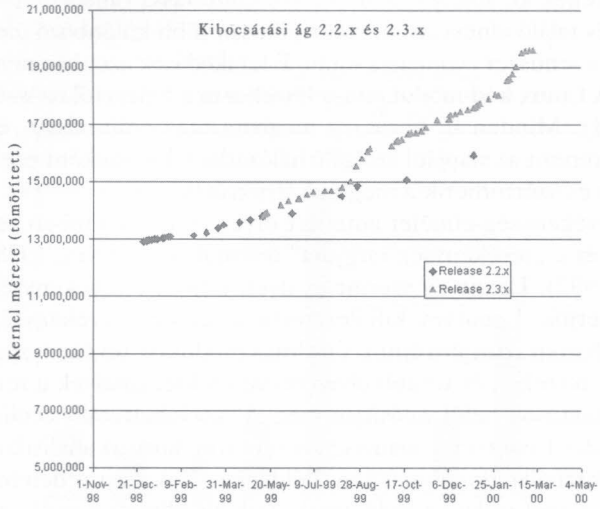
E jelenségre a leülepedés találó elnevezés, mivel tipikusan több különböző üledék réteg is egymásra rakódik a rendszer evolúciója során. E lerakódások azonban nem feltétlen maradnak stabilak. A Linux kód modularitása létrehozza a fejlesztőközösségek ökológiáját (Tuomi, 2001). Minden új közösség meghatározza erőforrásait, és ugyanakkor új feszültségeket teremt az alapjául szolgáló hálózatban. Esetenként ezek a feszültségek deformálhatják és széttörhetik a meglevő struktúrákat.

A kulturális-történeti tevékenység-elmélet amellett érvelt, hogy az emberi tevékenység mindig szándékos és a „tevékenység tárgyára" orientált (Leontyev, 1978; Stetsenko, 1995; Gal'perin, 1992). Leontyev szerint az ilyen tárgyat a tevékenység motivációjaként is értelmezhetjük. Leontyev kifejlesztette az emberi tevékenység egy modelljét, ami az analízis három szintjére épült: értelmes produktív tevékenység, annak felbontása cél-orientált tettekre, és tovább olyan műveletekre, amelyek e tetteket egy meghatározott kontextuson belül valósítják meg. A szociokulturális evolúcióban zajló tevékenység-fejlődés Leontyev-i analízise azt mutatta, hogy az analitikus hierarchia különböző szintjei közt állandó mozgás van. Például a célok könnyedén indítékokká válhatnak. Míg egy csapat vadász a vadászeszközeik előállítását a vadászás tevékenysége kontextusán belüli cél-orientált tettnek értelmezi, amikor a társadalmi

munkamegosztás létrehozza a szerszám-készítők csoportját, e csoport számára a szerszámkészítés a tevékenységük közvetlen tárgyává válik. Az indítékok, a társadalmi szerkezet, a termelési folyamatok és a hozzájuk felhasznált erőforrások ezért kölcsönösen egymástól függnek és dinamikus változásban vannak.

A Linux fejlesztés kontextusában az efféle mozgások tisztán láthatóak. A fejlődés korai fázisaiban a tevékenység tárgya maga a Linux kernel volt. Mikor a Linux eléggé felerősödött ahhoz, hogy az alkalmazások fejlesztéséhez is felhasználható legyen, az alkalmazásokat fejlesztő-közösségek eszközévé vált. Végezetül, mikor a Linuxot már operációs-rendszerként használták az alkalmazások futtatásához, a teljes GNU/Linux disztribúció külön eszközzé vált, amit összekapcsoltak a hardverrel és egy dobozba zártak.

Míg a Linux kernel fejlesztőinek egy viszonylag nyíltan hozzáférhető Linux forráskódra van szükségük, és a felhasználások fejlesztői is jól járnak egy ilyen szabad hozzáféréssel, a végfelhasználók számára főként csak akkor van ennek jelentősége, amikor a fekete doboz elromlik, és megmutatja valódi természetét, ami a hálózatban közreműködők egy komplex rendszere. A nyílt forráskódú modell javasolt „felsőbbrendűsége" ezért nagymértékben utal arra a tényre, hogy a számítógépes rendszerek bizony gyakran elromlanak. A nyílt forráskódú megközelítés sokat veszít az értékéből akkor, ha a végfelhasználók nem rendelkeznek kellő szaktudással a fordítási folyamat megszakadásakor láthatóvá váló alrendszerek diagnosztizálásához. A nyílt forráskódú modell viszonylag jó alkalmazhatósága ezért úgy tűnik, annak köszönhető, hogy ösztönzi a szakismeret kifejlődését. Általánosabban fogalmazva, az adott rendszer átláthatósága lehetővé teszi a végfelhasználó számára, hogy az összes elérhető erőforrását és szaktudását mozgósítsa az adott probléma megoldásához, azokat is beleértve, melyekre korábban senki sem gondolt. Az a különös „stílus", ahogy a nyílt forráskódú rendszerek elromlanak, előmozdítja a problémamegoldás hatékony módjait, s ugyanekkor ösztönzi a szaktudás fejlődését, ami hasonló problémák megoldásához felhasználható a jövőben.




4. ábra: Az erőforrás és a téma fejlődésének irányai

A leülepedés metaforáját kiszélesítve állíthatjuk, hogy a nyílt forráskód azt eredményezi, hogy a leülepedett rétegek képlékenyek maradnak. Ha valami probléma adódik, viszonylag könnyű átásnunk magunkat a modul interfészeken, hogy láthassuk, hol van a probléma, és hogyan oldható meg.

## A Linux-architektúra strukturális evolúciója

Ahogy fent megjegyeztük, a fejlesztői számára egy számítógépes operációs rendszer a modulok komplex hálózata. A rendszer evolúciója folyamán a rendszer működése absztrakttá válik, és a számítógépes kód homogén masszája szétválik viszonylag lazán kapcsolódó elemekre, melyek többé-kevésbé jól meghatározott módon lépnek interakcióba egymással. Valójában, többnyire a modularizációt tekintik a hatékony szoftverfejlesztés kulcsának. A tervezés alatt álló rendszert pár természetes összetevőre bontják le, amiket ezek után a programozók megvalósítanak. Gyakran az egyes modulokhoz más-más programozó csapatot rendelnek, akik a modul kifejlesztéséért és üzemeltetéséért felelnek.

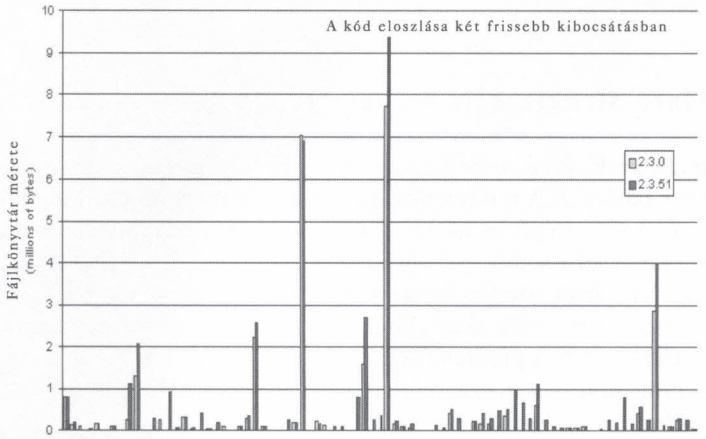
A Linux evolúciójában a szoftver architektúra és az azt fejlesztő közösség szervezete nem a rendszer egy előre adott koncepcióján vagy tervén alapszik. A tisztán funkcionális megfontolások helyett, a Linux-architektúra az erőforrások koordinációjának és mobilizációjának alapvető társadalmi problémáit tükrözi. Bár mind a Unix architektúra meglévő absztrakciói, mind pedig a meglévő mikroproceszszoros hardver architektúrák megszabják a fejlesztői közösség hatékony szervezésének módját, a Linux-architektúra a közösségi együttműködő fejlesztés követelményeit is jelentős mértékben tükrözi.

A Linux fejlesztésben közreműködők belső hálózata vizsgálatának egyik módja az, hogy a Linux forráskód modularizációját tanulmányozzuk. A modularizáció alapvető heurisztikája az, hogy egyetlen „helyre" teszi azt a forráskódot, ami független entitásként fejleszthető. A Linux esetében ez a hely egy fájlkönyvtár (*directory*), mely tárol egy-egy fájlt, vagy szorosan kapcsolódó fájlok egy csoportját. Bár vannak kivételek, és a modulok és könyvtárak között időnként nincs szoros leképezés, első közelítésben a Linux-architektúra evolúcióját lehet úgy is tanulmányozni, mint e könyvtárak evolúcióját.[11]

A különböző Linux kernel könyvtárakban található fájlállományok mérete a két kernel kibocsátás esetében az *5. ábrán* látható. Amint az ábra mutatja, a Linux fejlesztése folyamán új modulokkal gyarapodik, pár régi modulját elhagyja, és a növekedés üteme jelentősen eltér a különböző moduloknál. A Linux fejlesztőinek tudniuk kell, mit tartalmaznak ezek a könyvtárak, és ezek mindegyike fontos a fejlesztők jelenlegi tevékenységéhez.

Egy forráskód modul gyakran pontszerű erőforrásként szerepel. Nem kell ahhoz a pontos megvalósítás részleteit ismernünk, hogy egy adott modullal interakcióba lépő kódot fejlesszünk. A modul meghatározza, hogy milyen interfészt használhatunk a modulhoz való kapcsolódáshoz. Ez az interfész lefordítja a technikai rendszert és az őt fejlesztő és karbantartó közösséget úgy, hogy egy másik közösség erőforrásává válhasson. Egy szabványosított eljárás – amit gyakran, mint „interfészt" iktatnak be – alkalmazható arra, hogy hozzáférjünk az erőforrás által nyújtott szolgáltatáshoz. Amennyiben az erőforrás és a hozzá kapcsolódó szolgáltatás használatához szükséges proto-

koll változatlan marad, az interfész felhasználói nem kell, hogy ismerjék a technikai berendezés belső részleteit vagy az azt előállító hálózat szervezetét.

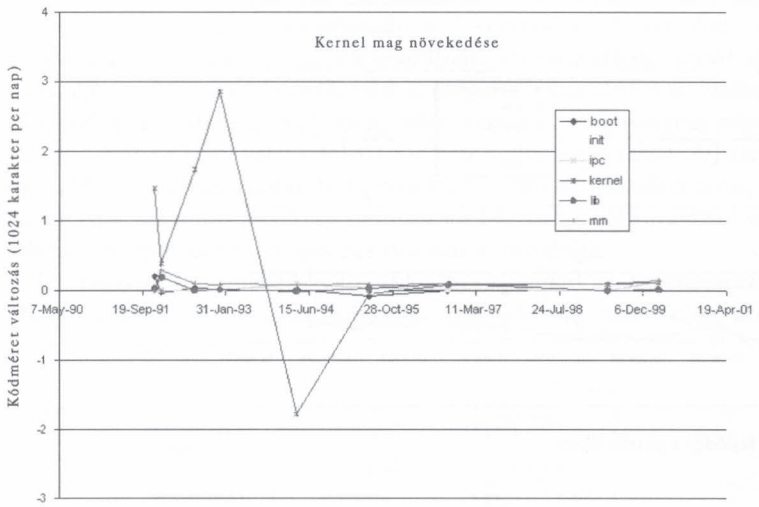



**5. ábra: Kreatív destrukció a Linux kernelben**

Schumpeter szerint (1975) az innováció kreatív destrukciót eredményez. A kapitalista társadalmi-gazdasági rendszer alapvető jellemzője az, hogy az erőforrások a régi használatukból dinamikusan újabbakba helyeződnek át. A kedvező lehetőségek egy vállalkozási folyamatban öltenek testet, és az innováció sebessége attól függ, hogy milyen tempóban sikerül az erőforrásokat átvinni a régi tevékenységekből az újabbakba. Az *5. ábra* megmutatja, hogy ez a folyamat a technikai eszközök szintjén is megjelenik. Bizonyos modulok eltűnnek az evolúció folyamán, és fejlesztőik új területekre összpontosítanak.

E folyamat pontosabb elemzése azonban feltárja, hogy a Linuxnak több, minőségileg különböző „innovációs területe" van. A Linux kernel szintjén megfigyelhető leülepedéshez hasonló folyamat található a kernelen belül is. A végfelhasználó anélkül akarja erőforrásként használni a Linuxot, hogy az *5. ábrán* található különféle moduljainak komplex hálózatára tekintettel lenne. Hasonlóképpen, a Linux kernel fejlesztőinek le kell egyszerűsíteniük a fejlesztői hálózat komplexitását. Pontosabban majd minden modulfejlesztő a rendszer néhány alapvető komponensére támaszkodik. Az ilyen alapvető komponensek fordítási folyamatának az alapul szolgáló alhálózatokat egyidejűleg több különböző közreműködő számára is le kell fordítania. Ez az erőforrás leülepedésével valósulhat meg. Más szavakkal: a változó fordítási folyamatok egy ilyen komplex hálózata fenntartásának lehetséges problémái a fordítási folyamat szabványosításával és a fekete doboz felbontásához vezető fejlesztés megfékezésével oldhatóak meg. Ez látható a *6. ábrán.* Az ábra a kód méretének változásait mutatja a Linux kernel magvának komponenseiben. Ez a „kemény mag" magába foglalja a Linux kernel alap-összetevőit, hogy a Linux-fejlesztők tovább dolgozhassanak a rendszer egyéb elemein.

A tény, hogy e központi mag komponenseinek fejlesztése nagyon hamar lelassult a Linux evolúciója során, rámutat arra, hogy milyen nehéz a sokoldalú fordítási interfészek biztosítása. A *6. ábra* úgy is olvasható, mint ami azt mutatja meg, hogy amikor egyazon alhálózatot több különböző közreműködő a saját különböző szempontjából közelít meg, semmilyen közös absztrakció nem bizonyul elég jónak. Más szavakkal, a változó fekete dobozokhoz nincs eleve hozzárendelt általános csomagoló eljárás. Ehelyett a kódot – végleges technikai eszközként – be kell fagyasztani.



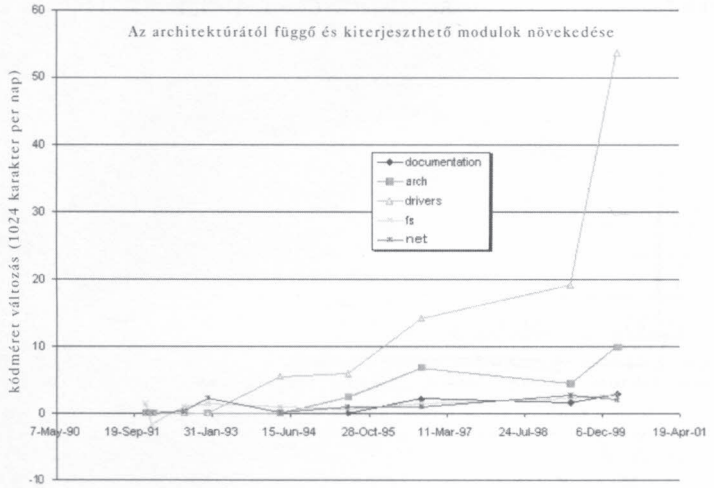

*6. ábra:* **A mag-komponensek gyors stabilizációja**

A Linux **kernel** „magja" a *6. ábrán* úgy definiálható, mint azon komponensek köre, melyek már a fejlesztés korai fázisaiban stabilizálódtak. Ez a megközelítés azt jelenti, hogy nincs a komponenseknek egy előre meghatározott kategorizálása, ami például azon alapulna, hogy elméleti síkon megértjük azt, hogy melyek is egy tipikus Unix operációs rendszer architektúrájának „alapvető" rétegei. Ehelyett az „alapvető" komponenseket úgy definiáljuk, mint az olyan komponenseket, amelyek az alapot biztosítják. A tény, hogy ezek a komponensek ebbe a szerepbe kerülnek, főleg azon múlik, hogy az alapok fordítása különböző közreműködők igényeinek is meg kell, hogy feleljen. Ilyen értelemben a struktúra alapvető komponensei tulajdonképpen annak „intézményesült" összetevői.

A Linux-architektúrában az intézményes innováció ritkának tűnik, és úgy látszik, a forráskód lassú változása a fordítás problémáival függ össze. A Linux-architektúra néhány másik része mindazonáltal nagyon gyorsan növekszik. Ilyen gyorsan növő összetevőket mutat be a *7. ábra.*

A Linux-architektúra leggyorsabban növekvő része az eszközmeghajtók egy csoportja. Mikor valamilyen új hardver bevezetésre kerül, azt a Linux fejlesztők nagyon gyorsan integrálják a Linux operációs rendszerbe. Valójában a Linux fejlesztése nagymértékben olyan „projektekből" áll össze, amelyek különféle hardvertermékeket adaptálnak a Linux közreműködő-hálózatába úgy, hogy egy szoftverrel a rendszerhez

„ragasztják" őket. A Linux ezért tekinthető olyan közreműködőnek, ami az újabb technikai elemeket gyorsan birtokba veszi, és a Linux-felhasználók közössége számára erőforrássá teszi azokat. Ez egyben talán a hagyományos szoftver projektek és a Linux-fejlesztés projektje közötti legfőbb különbség. Világosan látszik, hogy a Linux a társadalmi-technikai fejlődés egy ökológiai rendszere, nem pedig egy olyan projekt, ami egy előre meghatározott tervet ültet át a gyakorlatba.




*7. ábra:* Folyamatos fejlődés a perifériákon

## Az erőforrások és a közreműködők elburjánzása

A Linux evolúciója folyamán sok új fordítási mechanizmust találtak fel. A Linux a fordítási mechanizmusok e fajta elburjánzásának egy kivételesen érdekes példája, mivel a Linux-szoftver fejlesztői képesek a fordítási problémákra technikai megoldásokat létre hozni. Ebben az értelemben a Linux közösség nem csak egy Linux fejlesztő-közösség, hanem egy eszközfejlesztő közösség is egyben. Valójában azt is mondhatjuk, hogy épp ezért lehetett a Linux fejlesztése ilyen gyors. A Linux fejlesztő-tevékenység és a Linux-eszközfejlesztő tevékenység közötti határok nagyobb erőfeszítés nélkül, gyorsan átléphetőek.

A Linux fejlesztési modell ezt az erejét a Unix-kultúrából örökölte. A Unixot azzal az elgondolással fejlesztették, hogy az egy olyan eszközkészlet legyen, amely eszközei könnyen kombinálhatóak és újra felhasználhatóak új eszközök alkotóelemeiként. A Linux fejlesztő-közösséget ezért nemcsak úgy kellene tekintenünk, mint egy olyan közösséget, amelyik a Linux kernelt fejleszti. A fejlesztési modelljének sikere leginkább azokon az erőforrásokon múlik, amelyeket alkalmaz, és a közösség különféle feladataihoz továbbfejleszt.

Az erőforrásnak ez az ökológiája komplex természetű, és ha kompetens Linux fejlesztővé akarunk válni, az egyik legfontosabb kihívás az, hogy meg kell tanulnunk ezeket az erőforrásokat használni. Az erőforrások egy része szervezeti vagy közösségi erőforrásként jellemezhető, mások tekinthetőek technikai eszköznek vagy szerszám-

nak, illetve információforrásnak.

A nyílt forráskód irodalma jelentősen hangsúlyozta a nyílt forráskódú projektek alkalmasságát a megbízható és hibamentes szoftverek írására, és a mellett érvelt, hogy ez a nyílt forráskódú és a hagyományos szoftverfejlesztési projektek közötti legfőbb különbség. Egy pár fontos, a Linux hibajavító folyamatban használt erőforrást mutat be az *1. táblázat*. A táblázat az erőforrásokat olyan kategóriákra bontja, mint információforrások, eszközök, és közösségek. Az információforrások olyan szövegek, amelyekből kiderülnek, hogy mivel is foglalkozik a közösség, melyek a bevett gyakorlatai, és milyen erőforrásokat használnak. Az eszközök olyan erőforrások, amiket a tényleges hibajavítási tevékenység során használnak. A közösségi erőforrásokat a közösség életben tartására és a tevékenységének koordinálására használják. Amint azt a táblázat is mutatja, egy technikai eszköznek több szerepe is lehet ebben az ökológiában. Például a JitterBug rendszer egy web-alapú adatbázis, ami megmutatja, milyen hibákat ismer a közösség, és hogy valaki dolgozik-e már a hiba kijavításán. A JitterBug információforrásként működik azáltal, hogy mindenki számára lehetővé teszi, hogy megtudhassa, melyek az ismert hibák, és ugyanakkor közösségi erőforrásként is, amennyiben a probléma megoldásához szükséges munkát koordinálja.

A hibajavító folyamat fő közreműködői (*aktantjai*) az *1. táblázaton* megtalálhatóak. Ezek a közreműködők erőforrásnak tekinthetők, melyek az alapjukat alkotó alhálózatokat fordítják le.

| feldolgozási fázis | | | információforrás | eszközök | közösségi erőforrások |
|---|---|---|---|---|---|
| detektálás | | | lefordított kód<br>dokumentáció | emberek | LDP |
| hibajavítás | beazonosítás | | forráskód<br>linux kernel levelezőlista GYIK<br>JitterBug<br>oopstracing.txt<br>Kernel Traffic<br>Linux Dokumentációs Projekt<br>Projektspecifikus webhelyek<br>Linux kernel archívum<br>napló fájlok<br>hibabejelentő lap | szövegszerkesztő<br>gcc<br>gyártmány<br>gdb<br>ksymoops<br>IRC<br>számítógép konfiguráció (beállításai) | linux kernel levelezőlista<br>személyes e-levél<br>IRC csatornák<br>kernel newflash<br>LDP<br>projekt-specifikus levelezőlisták |
| | eltávolítás | | forráskód | szövegszerkesztő<br>gcc<br>gyártmány | |
| | | tesztelés | folt<br>MAINTAINERS file | diff<br>gcc<br>gyártmány<br>szövegszerkesztő<br>ftp | személyes e-levél<br>linux kernel levelezőlista |
| terjesztés | | | folt<br>MAINTAINERS file | gzip<br>tar<br>email<br>ftp | linux kernel levelezőlista<br>JitterBug |
| integrálás | | | folt<br>kibocsátás | árhuzamos Verzió Rendszer<br>vger<br>csomagkezelők | MAINTAINERS<br>vger |

**1. táblázat: A Linux hibajavító-folyamat közreműködői**

Az erőforrások és közösségek komplex rendszerének evolúciója során a társadalmi szerveződés és az eszközök együtt fejlődnek. Az új technikai eszközöket olyan embercsoportok hozzák létre, akik a munkájukat az eszköz fejlesztése köré szervezik. Az új eszköz ezért egy új közösséget hoz létre az eredeti közösség környezetében. Ez a folyamat egyre növő differenciálódást eredményez a társadalmi rendszerben. Ugyanakkor, a közösség birtokba vesz olyan erőforrásokat, amik a központi közösségen kívül

jönnek létre. Például, a GNU gcc által kidolgozott C nyelv fordító a Linux fejlesztőközösségen kívül jött létre. A gcc fordító mindazonáltal a Linux közösség alapvető erőforrása (Torvalds, 1999). Ha a fordító nem lenne elérhető, valószínűleg a Linux-fejlesztés is lehetetlenné válna.

A GNU General Public License (Általános Nyilvános Jogosítvány) egyértelműen fontos szerepet játszik itt. Ez garantálja, hogy a GNU gcc fordítót a Linux közösség a fejlesztés központi erőforrásaként birtokba vehesse. A szerzői jog intézményére támaszkodva, a nyílt forráskódú jogosítványok intézményes hátteret biztosítanak a kockázat csökkentéséhez és tudás-alapú érdekszövetség kiépítéséhez (Lewicki & Bunker, 1996). A nyílt forráskódú jogosítvány nélkül nagyon kockázatos lenne egy olyan rendszert kiépíteni, ami ennyire erősen ki van szolgáltatva egy, a közösségen kívül előállított erőforrásnak.

Valójában magukat a nyílt forráskódú jogosítványokat szabványosított fordításoknak tekinthetjük, amelyek párhuzamosan több különböző közreműködő számára biztosítanak sokoldalú interfészeket. Egészen konkrét megfogalmazásban: a nyílt forráskódhoz nem kell a különféle jogosítványokkal kapcsolatban tárgyalásokba bonyolódnunk; a linensz egy egyetemes szabványos interfészt teremt, amely a rendszert a lehetséges fejlesztőkkel és felhasználókkal köti össze. Ez a szabványosított interfész korlátozza a komplexitás növekedését, amikor új közösségek és közreműködők a saját tevékenységeikhez kezdik el használni a lefordított erőforrásokat.

A szellemi tulajdonjogok intézményes alapjait egy sokoldalú fordítási interfész megalkotására használva, a nyílt forráskód – maga is egy nagyon sajátos módon – a rendszert a gazdasági szférával összekötő interfész. A nyílt forráskódú licensz a hagyományos értelemben vett gazdasági értékektől függetlenné teszi a rendszer fejlődését. Bár a pénz korlátokat szabhat a nyílt forráskódú fejlesztésnek, a nyílt forráskódú jogosítványokat gyakran úgy is olvashatjuk, mint annak kinyilvánítását, hogy a valódi értékteremtő tett valahol a gazdaság területén kívül esik, és hogy a pénz irreleváns mérce a nyílt forráskódú projektekben. Más szavakkal, a gazdaság a nyílt forráskód területén kívül esik.

## Összegzés

A Linux operációs rendszer kernel evolúciójának bemutatására a fentiekben az innováció és a társadalmi-technikai változás két különböző megközelítését használtuk. Először amellett érveltünk, hogy a tudás a gyakorlati tevékenységek köré szervezett közösségekben található és ezekben fejlődik. A tudás szorosan kötődik az e gyakorlati tevékenységek során használt technikákhoz, és a közösség által a kommunikációra és a világ értelmezésére használt jelentések rendszeréhez. Ezt a „közösség-centrikus" nézetet korábban a jelentés és a tudás létrehozásának elemzésére használta Bahtyin és Fleck, és mostanában a szociális tanulás magyarázatára Schön és Engeström, és még konkrétabban a már meglévő hagyományokhoz és gyakorlati tevékenységekhez történő szocializáció magyarázatára Lave, Wenger és mások. A közreműködő-hálózat elméletet viszont a társadalmi-technikai rendszerek evolúciójának bemutatására alkalmazták, a hálózatban többnyire a hatalom létrejöttével és alkalmazásával kapcsolatos küzdelmekre és stratégiákra összpontosítva.

Jelen cikk e megközelítéseket a közösségek és egy moduláris technikai architektúra ökológiájának kontextusába helyezi. Pontosabban, megpróbáltuk bemutatni, hogy a technikai architektúrák változása és dinamikája hogyan tükrözi az adott rendszer fejlesztésében megjelenő feszültségeket. A közreműködő-hálózatokat a gyakorlat-közösségeken belül helyeztük el, és röviden bemutattuk, hogy a közösségek hogyan válnak közreműködőkké a közösségek hálózatában.

Ezért megváltoztattuk a közreműködő-hálózatok és a gyakorlat-közösségek szokásos értelmezési módját. A közreműködő-hálózatok elmélete problematikus annyiban, amennyiben az emberi és nem-emberi szereplők túl szimmetrikus értelmezését teszi lehetővé. Mintha a gépek, eszközök és technikák épp úgy saját motivációkkal és akarattal rendelkeznének, mint az emberek. E feltevés persze a motivációk természetének alapos elemzését követelné meg, amibe a tevékenység-elmélet hasznos betekintést nyújthat (Miettinen, 1999). Egy ilyen elemzés mindazonáltal ahhoz a nézethez vezet, amely a motivációkat a társadalmi gyakorlatba, a munkamegosztásba és az eszköz-használó tevékenységekbe ágyazottnak tekinti. A tevékenység helye ekkor az adott specifikus gyakorlati tevékenység köré szervezett közösségben található. A mellett érvelve, hogy a közösségek a közreműködők egy speciális és alapvető típusát adják a közreműködő-hálózatokban, bemutathatjuk, hogy mi teszi lehetővé a közreműködő-hálózatok evolúcióját, s hogyan megy végbe ez az evolúció.

Másrészről a közreműködő-hálózat elméletben bemutatott fordítás, pontosítás és erőforrás fogalmait kiaknázva, jobban megérthetjük a gyakorlati tevékenységek és a közösségek evolúcióját. Ez alapvetően fontos a technikai változás megértéséhez, minthogy az új technikákat mindig az által vesszük birtokba, hogy azok a társadalmi gyakorlatba integrálódnak. Valójában azt is mondhatjuk, hogy az innováció *csak* akkor történik meg, amikor a társadalmi gyakorlat megváltozik. Az ilyen változás gyakran egy olyan új eszköz birtokbavételének eredményeként valósul meg, amely átszervezi a közösség gyakorlati tevékenységét. Ezért az innováció kulcsa azokban a társadalmi kommunikációs és tanulási folyamatokban rejlik, amelyek a társadalmi gyakorlatok változását alapozzák meg. A felhalmozott ismeretek azonban részben leülepednek és beágyazódnak a folyamat során kifejlesztett technikai eszközök architektúrájába is.

A társadalmi gyakorlatok mindenesetre a közösségek ökológiájával is szoros kapcsolatban állnak. Egy adott gyakorlatban alkalmazott erőforrásokat és eszközöket más gyakorlatok során állítják elő. Nem mindig lehetséges a társadalmi gyakorlat megváltoztatása anélkül, hogy ne rombolnánk azokat a fordítási folyamatokat, amelyek egy közösséget más közreműködők számára erőforrássá tesznek. A változás nehézkes, különösen akkor, amikor ugyanazt a fordítási eljárást használja számos különböző közreműködő. Amint azt a Linux evolúciója mutatja, e probléma megoldásának egyik módja az, hogy az erőforrásokat leülepítik, a gyakorlatokat intézményesítik és befagyasztják az innovációt.

A Linux története ugyanakkor azt is mutatja, hogy a hatékony fordítási mechanizmusok gyors fejlődéshez vezethetnek. A modulok közötti interfészek kezelésének problémája az interfészek felépítésének és használatának eléggé szabványosított módjaihoz vezetett. Ez azonban azt jelenti, hogy a rendszerhez könnyen lehet modulokat illeszteni. Mi több, ezek a szabványosított fordítási mechanizmusok azt jelentik, hogy a modulokat viszonylag könnyen használják a különböző közreműködők akkor is, ha azok változnak is.

A Linux ezért sok értelemben nyitott a kombinációs újításokra. A szabványosított interfészek és fordítási folyamatok letisztult modulhatárokat idéznek elő és serkentik az új kombinációk gyors létrejöttét. Maga a forráskód is időnként újrahasználható, de ennél fontosabb, hogy a forráskód által képviselt ismeret más kontextusokban is nagyobb gondok nélkül újra felhasználható. Ennek eredményeképpen a Linux kernel különböző részeit fejlesztő különféle közösségek nagyon mobilissá váltak. Így a fordítás problémájára adott megoldás olyan közösségek ökológiájához vezet, amelyek az erőforrásaikat könnyedén át tudják alakítani.

A Linux fejlesztésben a schumpeteri kreatív destrukció ugyan lerombolja a kód részeit, de a kompetencia és a tapasztalat csekély veszteséggel újraszerveződik. Ebben az értelemben azt is állíthatjuk, hogy a Linux fejlesztési modell és a Szilícium-völgy innovációs modell (Kenney, 2000) hasonló vonásokat mutat. Mivel a motivációk és az értékek bizonyos társadalmi kontextusokon belül keletkeznek és artikulálódnak, azt várhatnánk, hogy a Szilícium-völgy és a Linux-fejlesztő-közösségek kultúrája viszonylag könnyen, nagyobb konfliktusok nélkül integrálható. A fő különbség természetesen, hogy a Szilícium-völgynek a kockázati tőke által irányított vállalkozói kultúrája van, amíg a Linux fejlesztésben a gazdasági szféra viszonylag láthatatlan maradt. E feszültséget aktívan kezeli a Nyílt Forráskód Kezdeményezés (*Open Source Initiative*) (vö. Raymond, 1999). Valójában a Nyílt Forráskód Kezdeményezés úgy is tekinthető, mint egy szervezeti forma vagy közösség, amely a Linux evolúciója során szökkent szárba azért, hogy helyrehozza e két eléggé hasonló kultúra összeütközésbe során keletkező társadalmi károkat. Amint azt a Linux evolúciójának elemzése mutatja, a gyors növekedés megköveteli, hogy a mag intézményesüljön, és hogy egyes fordítási folyamatok biztosítva legyenek. Ebben a modellben az innováció a perifériákon történik. Érdekes, hogy e perifériákat hagyományosan határterületekként ábrázolják. Feltehetnénk azonban a kérdést, hogy vajon – s milyen értelemben – a haladás nem azon múlik-e, hogy kitolják a perifériák határait, vagy ez csupán a magon belüli változást csökkentő stratégia....

*Antos Balázs fordítása*

## JEGYZETEK

* A „kernel" a program alapvető része, tipikusan az operációs rendszer azon része, mely közvetlen kapcsolatban van a hardverrel, és interfészként működik más szoftver szintek és a hardver között. – A szerk.

[1] E cikk az Internet Kutatók Szövetségének (Association of Internet Researchers) a kansasi Lawrence-ben rendezett konferenciáján 2000. szeptember 15-én elhangzott előadás szerkesztett változata.

[2] Maga Fleck a gondolat kollektíva fogalmát használta. Valójában Brown és Duguid (2000b) megjegyzik, hogy a közösségi nézőpont körüli jelenlegi lelkesedés részben a közösség szó varázsának köszönhető. Rámutatnak arra, hogy a lelkesedés kisebb lenne, ha Lave és Wenger (1991), akik a gyakorlat-közösség fogalmát népszerűvé tették, a közösség helyett a káder vagy a kommuna kifejezést használták volna. Fleck a „gondolat-stílus" fogalmát is használta, amit később Mary Douglas (1987; 1996) is átvett.

[3] Constant, 1987, 227. oldal.

[4] Észre kell vennünk, hogy a gyakorlattal kapcsolatos közösségek fent kifejtett különböző megfogalmazásai a jelenség különböző aspektusait világítják meg. A közösségek modelljei többnyire nincsenek részletesen kifejtve a meglévő irodalomban, és gyakran kissé kétértelműen, sőt ellentmondásosan használják a fogalmat. Mármost úgy tűnik, négyféle értelmezés, vagy szemlélet kering az irodalomban. Nevezhetjük ezeket a „termelés közösségeinek", az „interpretáció közösségeinek", az „identifikáció közösségeinek" és a „birtokbavétel közösségeinek". Ezek természetesen szorosan összefüggnek egymással, és nehéz őket elvi alapon, tapasztalati megfigyelés útján, vagy a gyakorlatban megkülönböztetni. Jelen cikkben Fleck-et követve mindezekre utalva a „gondolat-közösség" kifejezést használjuk. Bár a „gondolat-közösséget" könnyen tisztán mentális jelenségnek érthetjük, mint például egy a khuni (1970) értelemben vett tudományos paradigmát, e közösségekről írt magyarázatában Fleck részletesen elemzi a technikák, interpretációk, identitások és a tudástermelés kölcsönös összefüggéseit, és közös történelmi evolúciójukat.

[5] Ebben az értelemben a Nardy, Whittaker és Schwartz (2000) által tanulmányozott intenzív hálózatok is a közreműködő-hálózatok egy példájának tekinthetőek.

[6] Law, 1992, 381. oldal.

[7] Niklas Luhmann (1995) egy ide vonatkozó elképzelésre alapozta a társadalmi rendszerekről alkotott elméletét. Luhmann szerint mind a jelentés, mind a társadalmi rend azért alakul ki, mert a komplexitást le kell egyszerűsíteni. A jelentés és a társadalmi rend ezért „fekete-dobozokból" épül fel, melyek csökkentik az akár szélsőségesen is komplex világban megnyilvánuló esetlegességet. A jelentés például tekinthető rendnek is, ami megjelenik akkor, mikor a kognitív folyamat során a sok lehetséges „látens" interpretáció közül kiválasztódik egyetlen interpretáció. A mögöttes rend, ami a világot „jelentés-teli" világgá teszi, a jelentés-kapcsolatok egy hálózata, ami biztosítja a világ interpretálásának alapját. Hasonlóképpen, a különös rend, ami az alapvetően esetleges kommunikatív interakciót érthetővé teszi, az nem más, mint amit „társadalminak" nevezünk. (vö. Tuomi, 1999a).

[8] Az érdeklődő olvasó a különböző közösségekről további részleteket találhat Abbate (1999) és Naughton (2000) műveiben.

[9] Raymond és mások a mellett érveltek, hogy a nyílt forráskód alapja a poszt-hiánygazdaság és az erőforrások bősége. Mindazonáltal a Linux-fejlesztő-közösség evolúciója mutatja, hogy nem csak erről van szó. Például Torvalds szerint a Linux Interneten történő megosztásának egyik fontos oka a fejlesztői erőforrások hiánya volt a Helsinki Egyetemen (személyes közlés, 2000. szeptember). Ebben az értelemben a Linux-fejlesztés korai stádiuma a World Wide Web (Berners-Lee & Fischetti, 1999) korai stádiumához volt nagyon hasonló. Ahogy

Castells (2001) megjegyezte, az erőforrások hiánya is előmozdítja a nyílt forráskód átvételét azokban az országokban, ahol az erőforrások korlátozottak.

[10] A Linux és fejlesztői közössége történetét részletesen bemutattam egy hamarosan megjelenő könyvemben (Tuomi, 2001).

[11] A következő összegzés egy, a dokumentált Linux és Unix architektúrát összehasonlító tanulmány, az automatikus architektúra kivonatoló által létrehozott konkrét architektúrák és a kernel forráskód fájlok evolúciójának részletes tanulmányozása alapján készült (Tuomi, 2001).

## IRODALOM


**Abbate, J. (1999):** *Inventing the Internet.* Cambridge, Mass., MIT Press.

**Bakhtin, M. (1987):** *Speech Genres and Other Late Essays.* Austin, University of Texas Press.

**Berners, T. & Lee and M. Fischetti, (1999):** *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by its Inventor.* San Francisco, HarperCollins.

**Bezroukov, N. (1999):** „A Second look at the cathedral and the bazaar," *First Monday,* volume 4, number 12 (December), at http://firstmonday.org/issues/issue4_12/bezroukov/

**Bijker, W.E. & Law, J. (1992):** *Shaping Technology/Building Society: Studies in Sociotechnical Change.* Cambridge, Mass., MIT Press.

**Braden, R. & Reynolds, J.K. & Crocker, S. & Cerf, V. & Feinler, J. & Anderson, C. (1999):** „RFC 2555: 30 Years of RFCs," Internet Society, at ftp://ftp.isi.edu/

**Bradner, S. (1999):** „The Internet Engineering Task Force," In: C. DiBona, S. Ockman, and M. Stone (editors). *Open Sources: Voices from the Open Source Revolution.* Sebastopol, Calif., O'Reilly & Associates, pp. 47-52.

**Brown, J.S. & Duguid, P. (1991):** „Organizational learning and communities of practice: toward a unified view of working, learning, and innovation," *Organization Science,* volume 2, pp. 40-57.

**Brown, J.S. & Duguid, P. (2000a):** *The Social Life of Information.* Boston, Harvard Business School Press.

**Brown, J.S. & Duguid, P. (2000b):** „Knowledge and organization: a social-practice perspective," *Organization Science,* in press.

**Callon, M. & Law, J. & Rip, A. (1986):** *Mapping the Dynamics of Science and Technology: Sociology of Science in the Real World.* Houndmills, Basingstoke, Macmillan Press.

**Castells, M. (2001):** *The Internet Galaxy: Reflections on Internet, Business, and Society.* New York, Oxford University Press.

**Cole, M. (1996):** *Cultural Psychology: A Once and Future Discipline.* Cambridge, Mass., Harvard University Press.

**Constant, E.W. (1980):** *The Origins of the Turbojet Revolution.* Baltimore, Johns Hopkins University Press.

**Constant, E.W. (1984):** „Communities and hierarchies: structure in the practice of science and technology," In: R. Laudan (editor). *The Nature of Technological Knowledge: Are Models of Scientific Change Relevant?* Dordrecht, Reidel, pp. 27-46.

**Constant, E.W. (1987):** „The Social locus of technological practice: community, system, or organization?" In: W.E. Bijker, T.P. Hughes, and T.J. Pinch (editors). *The Social Construction of Technological Systems: New Directions in the Sociology and History of Technology.* Cambridge, Mass., MIT Press, pp. 223-242.

**David, E.E. Jr. & Fano, R.M. (1965):** „Some thoughts about the social implications of accessible computing," excerpts reprinted in *IEEE Annals of the History of Computing,* volume 14, number 2, pp. 36-39.

**DiBona, C. & Ockman, S. & Stone, M. (1999):** *Open Sources: Voices from the Open Source Revolution.* Sebastopol, Calif., O'Reilly & Associates.

**Douglas, M. (1987):** *How Institutions Think.* London, Routledge & Kegan Paul.

**Douglas, M. (1996):** *Thought Styles: Critical Essays on Good Taste.* London, SAGE.

**Engeström, Y. (1987):** *Learning by Expanding: An Activity Theoretical Approach to Developmental Work Research.* Helsinki, Orienta Konsultit.

**Engeström, Y. & Miettinen, R.& Punamäki, R.-L. (1999):** *Perspectives in Activity Theory.* Cambridge, Cambridge University Press.

**Fano, R.M. (1967):** „The Computer utility and the community," *IEEE International Convention Record,* pp. 30-34, excerpts reprinted in *IEEE Annals of the History of Computing,* volume 14, number 2, pp. 39-41.

**Fleck, L. (1979):** *Genesis and Development of a Scientific Fact.* Chicago, University of Chicago Press.

**Gal'perin, P.I. (1992):** „The Problem of activity in Soviet psychology," *Journal of Russian and East European Psychology,* volume 30, number 4, pp. 37-59.

**Kenney, M. (2000):** *Understanding Silicon Valley: The Anatomy of an Entrepreneurial Region.* Stanford, Calif., Stanford University Press.

**Knorr Cetina, K. (1999):** *Epistemic Cultures: How the Sciences Make Knowledge.* Cambridge, Mass., Harvard University Press.

**Kuhn, T.S. (1970):** *The Structure of Scientific Revolutions.* Chicago, University of Chicago Press.

**Kuusi, O. (1999):** „Learning communities as sources of innovations and as targets of innovation policy," In: G. Schienstock and O. Kuusi (editors). *Transformation Towards a Learning Economy: Challenges for the Finnish Innovation System.* Helsinki, SITRA.

**Kuwabara, K. (2000):** „Linux: a bazaar at the edge of chaos," *First Monday,* volume 5, number 3 (March), at http://firstmonday.org/issues/issue5_3/kuwabara/

**Latour, B. (1999):** *Pandora's Hope: Essays on the Reality of Science Studies.* Cambridge, Mass., Harvard University Press.

**Latour, B. & Woolgar, S. (1986):** *Laboratory Life: The Construction of Scientific Facts.* Princeton, N.J., Princeton University Press.

**Lave, J. & Wenger, E. (1991):** *Situated Learning: Legitimate Peripheral Participation.* Cambridge, Cambridge University Press.

**Law, J. (1992):** „Note on the theory of the actor-network: ordering, strategy, and heterogeneity," *Systems Practice,* volume 5, number 4, pp. 379-393.

**Leonard, A. (2000):** „Free software project," Salon.com at http://www.salon.com/tech/fsp/index.html

**Leontyev, A.N. (1978):** *Activity, Consciousness, and Personality.* Englewood Cliffs, N.J., Prentice-Hall.

**Lewicki, R.J. & Bunker, B.B. (1996):** „Developing and maintaining trust in work relationships," In: R.M. Kramer and T.R. Tyler (editors). *Trust in Organizations: Frontiers of Theory and Research.* Thousand Oaks, Calif., SAGE, pp. 114-139.

**Licklider, J.C.R. & Taylor, R.W. (1968):** „The Computer as a communication device," *Science and Technology* (April), reprinted in *In Memoriam: J.C.R. Licklider (1915-1990),* Digital Systems Research Center, 7 August 1990, at ftp://ftp.digital.com/pub/DEC/SRC/research-reports/

**Luhmann, N. (1995):** *Social Systems.* Stanford, Calif., Stanford University Press.

**Miettinen, R. (1999):** „The Riddle of things: activity theory and actor-network theory as approaches to studying innovations," *Mind, Culture, and Activity,* volume 6, number 3, pp. 170-195.

**Morson, G.S. & Emerson, C. (1990):** *Mikhail Bakhtin: Creation of Prosaics.* Stanford, Calif., Stanford University Press.

**Nardi, B.A. & Whittaker, S.& Schwartz, H. (2000):** „It's not what you know, it's who you know: work in the information age," *First Monday,* volume 5, number 5 (May), at http://firstmonday.org/issues/issue5_5/nardi/

**Naughton, J. (2000):** *A Brief History of the Future: From Radio Days to Internet Years in a Lifetime.* Woodstock, N.Y., Overlook Press.

**Nonaka, I. & Konno, N. (1998):** „The Concept of „ba": building a foundation for knowledge creation," *California Management Review,* volume 40, number 3, pp. 40-54.

**Nonaka, I. & Toyama, R.& Konno, N. (2000):** „SECI, ba, and leadership: a unified model of dynamic knowledge creation," *Long Range Planning,* volume 33, pp. 5-34.

**Raymond, E.S. (1998a):** „Homesteading the noosphere," *First Monday,* volume 3, number 10 (October), at http://firstmonday.org/issues/issue3_10/raymond/ and at http://www.tuxedo.org/~esr/writings/

**Raymond, E.S. (1998b):** „The Cathedral and the bazaar," *First Monday,* volume 3, number 3 (March), at http://firstmonday.org/issues/issue3_3/raymond/ and at http://www.tuxedo.org/~esr/writings/

**Raymond, E.S. (1999):** *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary.* Sebastopol, Calif., O'Reilly & Associates.

**Sawhney, M. & Prandelli, E. (2000):** „Communities of creation: managing distributed innovation in turbulent markets," *California Management Review,* volume 42, number 2, pp. 24-54.

**Schön, D.A. (1983):** *The Reflective Practitioner.* New York, Basic Books.

**Schumpeter, J.A. (1975):** *Capitalism, Socialism and Democracy.* New York, Harper & Row.

**Scribner, S. (1997):** *Mind and Social Practice: Selected Writings of Sylvia Scribner.* Cambridge, Cambridge University Press.

**Stallman, R. (1999):** „The GNU operating system and the free software movement," In: C. DiBona, S. Ockman, and M. Stone (editors). *Open Sources: Voices from the Open Source Revolution.* Sebastopol, Calif., O'Reilly & Associates, pp. 53-70.

**Stetsenko, A.P. (1995):** „The Role of the principle of object-relatedness in the theory of activity," *Journal of Russian and East European Psychology,* volume 33, number 6, pp. 54-69.

**Torvalds, L. (1999):** „The Linux edge," In: C. DiBona, S. Ockman, and M. Stone (editors). *Open Sources: Voices from the Open Source Revolution.* Sebastopol, Calif., O'Reilly & Associates, pp. 101-111.

**Tuomi, I. (1999a):** *Corporate Knowledge: Theory and Practice of Intelligent Organizations.* Helsinki, Metaxis.

**Tuomi, I. (1999b):** „Inside innovation clusters: collective knowledge creation in networks and communities," In: G. Schienstock and O. Kuusi (editors). *Transformation Towards a Learning Economy: Challenges for the Finnish Innovation System.* Helsinki, SITRA.

**Tuomi, I. (2001):** *Theory of Innovation: Change and Meaning in the Age of Internet* (working title).

**Vygotsky, L. (1986):** *Thought and Language.* Cambridge, Mass., MIT Press.

**Wayner, P. (2000):** *Free for All: How Linux and the Free Software Movement Undercut the High-Tech Titans.* New York, HarperBusiness.

**Wenger, E. (1998):** *Communities of Practice: Learning, Meaning, and Identity.* Cambridge, Cambridge University Press.

**Wertsch, J.V. (1991):** *Voices of the Mind: A Sociocultural Approach to Mediated Action.* Cambridge, Mass., Harvard University Press.