A Grid Underground (GUG) projekt - azaz web szolgáltatás alapú grid rendszerek ClusterGridtől desktopokig

Szalai Ferenc, szferi@niif.hu
NIIFI

1 Bevezető

Napjaink grid rendszerei szembeszökő fejlődésen mennek keresztül. Az évezred elejét jellemző általános felbuzdulás és ebből fakadó szerteágazó kutatás, fejlesztési irányok mára konszolidálódni és fokuszálódni látszanak. Az ipari és akadémiai szereplők között a párbeszéd lassan kialakul elsősorban ipari nyomásra keletkeznek, még ha hihetetlenül lassan is, szabvány szerű elképzelések. Mindezeket a törekvéseket a Global Grid Forum (GGF) [1] és annak Open Grid Service Architecture (OGSA) [2] ajánlása fogja össze esernyő szerűen. A tendenciák egyre inkább abba az irányba mutatnak, hogy a grid és web technológiák konvergálnak és néhány (< 10) grid infrastruktúra alakul ki a világ különböző területein, melyek között a átjárást a szabványos interfészek és adatmodellek biztosítják. Az alábbi előadásban igyekszem összefoglalni a jelenlegi helyzet és azt, hogy az új generációs ClusterGrid [3] rendszer hogyan követi és talán diktálja is új köztes réteg rendszerével (Grid Underground - GUG projekttel [4]) a trendeket hazánkban.

2 Mi az a grid?

Az ún. grid technológia körül nagy a felhajtás az utóbbi időben. Számos infrastruktúra és szoftver projekt fut melyek a különféle grid elképzeléseket igyekszenek megvalósítani. Miközben arról látszólagos egyetértés alakult ki, hogy a grid rendszerek feladata elosztott erőforrások használatának koordinálása biztonságos egységes módon függetlenül az erőforrások konkrét megvalósításától és felügyeletének jellegétől, a gridek egyedi karakterisztikáiról és minimálisan szükséges funkcionális követelményekről valamint programozási modelljéről továbbá azokról az absztrakciókról amivel a gridet modellezhetjük már kevésbé egységesek az elképzelések. Más szóval a "grid" korunk számítástudományának divatszava amit mindennel kapcsolatban használnak következésképpen jelentés nem jól definiált. A grid különböző dolgot jelent mindenkinek és úgy tűnik mindenkinek meg is van a maga definíciója.

Habár a jelenlegi grid rendszerek leginkább a számítási erőforrásokra fokuszálnak ami jól megegyezik az Internet által összekapcsolt gépek képével, a grid rendszerek magában kellene foglalniuk tágabb értelemben vett erőforrásokat is mint az adattárolási, hálózati, szoftver és általános adathalmaz erőforrásokat és számos nem tipikus erőforrást is mint a grafikus és audio i/o egységeket, manipulátorokat, szenzorokat stb. Ennek ellenére az összes ilyen erőforrásnak a megosztása az Internet segítségével feltételezi, hogy az erőforrás valahogy reprezentálva van egy gépen futó folyamattal (ez alól az egyetlen kivétel lehet a hálózat maga ami így egy problémás grid erőforrás is egyben). Ezek az erőforrásokat reprezentáló folyamatok újabban hálózaton keresztül elérhető grid szolgáltatások.

A szolgáltatás definíciója az OGSA szerint:

> A *szolgáltatás* egy hálózati eléréssel ellátott entitás ami képességeket tesz elérhetővé. […] Az OGSA leírásban *szolgáltatásokra* fokuszálunk: számítási erőforrásokat, tárolási erőforrásokat, hálózatot, programokat, adatbázisokat és hasonlókat mind

szolgáltatásokkal reprezentálunk.

A szolgáltatás orientált megközelítés szerint - ami mára általánosnak mondható - a grid rendszerre tekinthetünk úgy, mint földrajzilag elosztott kommunikáló szolgáltatások halmazára. Olyan szolgáltatásokéra amik absztrakciókon keresztül reprezentálnak erőforrásokat illetve felhasználókat. Ezek az absztrakciók általában az adat reprezentáció szintjén jelennek meg.

A legtöbb ma gridnek nevezett rendszer (kezdve klaszter rendszerektől az önkéntes felajánlason alapuló desktop rendszerekig) nem rendelkezik számos olyan attribútummal ami miatt őt elvárásainknak megfelelően gridnek tekinthetnénk. Mindez csak a kép összezavarására szolgál illetve jól mutatja, hogy a grid jelenleg egy jól bevált divatszó.

A szolgáltatás orientált megközelítés előnye, hogy a szolgáltatásokat azok interfésze és az általuk használt adatmodell definiálja ami teljesen független azok konkrét implementációjától. Az alkalmazásoknak csak azzal kell törődniük, hogy a szolgáltatások által meghatározott interfészeket és a kommunikációhoz szükséges hálózati protokollokat beszélni tudják.

Az interfész és annak implementációja a szolgáltatások készítői számára is előnyös mivel módjukban áll a teljes szolgáltatás implementációt lecserélni. Mindaddig amíg az új implementáció ugyanazt az interfészt használja, mint a korábbi aggodalomra nincs okunk.

Az interfész és az implementáció szétválasztása természetesen nem új keletű ötlet. Meglehetősen régóta alkalmazzák az elosztott objektum rendszerben (OPENSTEP/GNUStep DO [5], Java RMI [6], CORBA [7]). Az elosztott objektum rendszerek ugyanakkor ezt az elvet az objektumok szintjén alkalmazzák ami jól működhet teljesen új alapokon létrehozott alkalmazások esetén de túl alacsony szintű megoldás már létező szoftver komponensek összekapcsolása esetén. Az elosztott objektum rendszerek számos objektum orientált tulajdonságot mint például az interfész és az implementáció öröklődést is támogatják. Ráadásul mivel az interfész definiálása gyakran objektum orientált programozási nyelven történik a legtöbb elosztott objektum rendszer nyelv specifikus.

Magasabb szinten működő paradigma ami ugyanezeket az alapelveket követi a komponens orientált programozás. A komponens orientált programozásban az interfész szeparációja és az implementációnak az interfészhez történő kései kötése a két alapelv és hanyagolja a objektum orientált nyelvek öröklési tulajdonságát. Az interfészek csak egy magasabb szinten vannak definiálva amiket a komponensek tetszőleges nyelven implementálhatnak (nem feltétlenül objektum orientált módon).

Így jól láthatóan a grid manapság elosztott komponens alapú rendszernek is tekinthető. Következés képen az grid alap infrastruktúrájának támogatnia kell a komponensek közötti kommunikációt, az interfészek definiálásának képességével együtt. Továbbá ennek az alacsony szintnek lehetővé kell tennie, hogy egy kiválasztott interfészhez megfelelő implementációt találjunk. Ezen az alapon olyan szolgáltatások építhetőek amik megvalósítják az erőforrások és felhasználók absztrakcióját.

3 Kapcsolat grid és web között

Az előző fejezetben bemutattuk, hogy ami elképzelésünk szerint a grid szolgáltatások erőforrás és felhasználó absztrakciót valósítanak meg és jól definiált interfészekkel kell rendelkezniük amik elrejtik az megvalósításuk részleteit. Mióta a web a legtöbb ember fejében egyet jelent az Internettel az IT ipar szinte mindent a web alapú technológiákkal próbál megoldani. Ugyanakkor mindenki az IT iparban tisztában van a web technológia hátrányaival amik alapvetően a a HTTP-hez kapcsolódó protokollok szinkron, állapot független jellegéből adódnak.

Az utóbbi öt és egynéhány évben számos új web alapú szabvány látott napvilágot amik

igyekszenek áthidalni a rendszer fejlesztők igényei és a reális lehetőségek között tátongó óriási szakadékot. Az eredeti egyszerű webtechnológia mára számos a tradicionális elosztott objektum rendszerekből kölcsönzött technikákkal egészült ki. Ilyen például az egyre népszerűbb RPC mechanizmus SOAP segítségével [8]. Ezek az új szabványok (WSDL, SOAP, XML stb.) egy új divatszó alá lettek összegyűjtve: "web szolgáltatások".

A grid technológia nem tud a web alapú technikáktól függetlenül létezni az ipari szereplők nyomása miatt. Így úgy tűnik manapság senki sem tudja máshogy elképzelni jövő grid szolgáltatásait, mint olyan web szolgáltatásokat melyek interfészét WSDL-ben írják le és SOAP üzenetekkel kommunikálnak egymással. Mindképpen elismerésre méltó a választás abból a szempontból, hogy a web szolgáltatások alapvetően Internet léptékre lettek tervezve szemben mondjuk egyes multicast/broadcast alapú RPC protokollokat beszélő rendszerekkel, melyek lokális hálózatokra vannak optimalizálva (pl.: Java JINI)

A korai grid keretrendszerek, mint a Globus, web szolgáltatás alapú keretrendszerekké formálódtak [9]. Általában tényleg csak keretrendszert készítenek konkrét grid szolgáltatások nélkül így ez az átalakulás nagy áttörést nem hozott a grid technológiában. A grid és a web technológia megtette az első lépéseket egymás irányában de az alapvető kérdések mint az jogosultság kezelés, erőforrás leírás stb. továbbra is megoldatlanok.

4 OGSA

A GGF által babusgatott OGSA ajánlása valójában egy keretet határoz meg csak. Leginkább mint szabvány tervezetek gyűjteményét lehet meghatározni. Ez a gyűjtemény az alábbi fontos csoportokat foglalja magában [10]:

- *Web Service Resource Framework (WSRF)*: maga is egy szabvány gyűjtemény, és lényegében arról szól, hogy mi az a szabványos interfész amin keresztül meg lehet tudni, hogy egy szolgáltatás milyen erőforrásokat valósít meg. Az általa definiált WS-Resource valójában egy reláció szolgáltatás és erőforrások között. Ehhez kapcsolódnak még a nevezéktannal kapcsolatos problémák megoldása amiket a WS-Addressing és WS-Naming ajánlások foglalnak össze. Ehhez kapcsolódik az OGSA Resource Name Service ajánlása ami a nevek kiosztására mindegy DNS szerűen felügyelő protokollt határoz meg web szolgáltatásokra.
- *Feladat (job) kezelés*: ez egy nagy kalap amiben a feladatok kezelésével kapcsolatos szabványokat gyűjtik. Ezek közül talán a három legfontosabb a jobok leírását meghatározó Job Submittion Definition Language (JSDL), ami egy XML sémát határoz meg, a Basic Executation Service (BES) ami a feladat futtatást végző szolgáltatás interfészét definiálja elsősorban valamint a WS-Agreement ami a futtatást kérő szolgáltatás és végrehajtó szolgáltatás közötti erőforrás foglaláshoz szükséges XML alapú kommunikáció módját határozza meg.
- *Biztonság*: Az OGSA Security Profile tartalmazza az összes szolgáltatás által meghatározott biztonsági követelményeket valamint a OGSA Attributes ajánlás tartalmazza az jogosultság kezeléshez és gridhez kapcsolódó AAI rendszerkben alkalmazható felhasználói és szolgáltatás attribútumok definícióját.
- *Információs modell*: elsősorban az ipar által már használt Common Information Model (CIM) grides alkalmazhatóságát vizsgálják (kezdve az erőforrások leírásával és a futási környezettel egészen az automatizált hálózati erőforrás kezelésig), illetve a szükséges absztrakciók elkészítése folyik.
- *Adatkezelés*: két kiemelkedő kezdeményezés folyik ezzel kapcsolatban, az egyik a ByteIO ami web szolgáltatás alapú blokk szintű állomány hozzáférés kereteit határozza meg,

valamint a Data Access Interface (DAI), ami állomány szintű hozzáférés interfésztét definiálja.

Természetesen van néhány szabványosítási próbálkozás mely nem kötődik olyan szorosan a GGF-hez és az OGSA-hoz mint a fentiek. Ilyen például a Stroage Resource Management (SRM) [11] interfész specifikáció ami az adattároló erőforrások hozzáférését kívánja egységesíteni.

5 Meghatározó grid rendszerek

Az alábbiakban benyomásainkat adjuk közre az általunk meghatározónak gondolt grid rendszerekről.

5.1 Globus GT4

Habár a grid technológia korai időszakában egyértelműen a Globus 2.x változatokra alapozott grid rendszerek voltak a meghatározóak, köszönhetően az interfészek és komponensek kidolgozatlanságának valamint az implementáció és architektúra számos alapvető problémájának, már a világ java része elmozdult a holtpontról és vagy jelentősen módosította az eredeti Globus 2.x rendszert (pl.: LCG, Nordugrid) vagy pedig teljesen új irányba mozdult (pl.: ClusterGrid, Unicore). A képet az is megzavarta, hogy maga a Globus csapat sem hitte komolyan a 2.x-es változat sikerének lévén új (GT3) majd még újabb (GT4) [9] változatokkal rukkolt elő, melyek mára sem értek el stabilitás szinten a 2.x-es változat minőségét, ami maga sosem erről tulajdonságáról volt híres. A Globus GT4 némileg elsiette a fejlesztéseit és nem haladt a párhuzamosan kialakuló szabványokkal, így állhatott elő már az a szerencsétlen helyzet, hogy a Globus csapat által meghatározónak gondolt és erőltetett WSRF ajánlást a legkevésbé a GT4 valósítja meg. Jellemző a Globus fejlesztési modelljére, hogy továbbra is konkrét szolgáltatások implementációja helyett keretrendszert nyújt amiben a szolgáltatások saját igényeinknek megfelelően implementálhatóak. A GT4 Java nyelven íródott és az erőforrás oldalon komoly igényei vannak az alkalmazott technológiából kifolyólag, desktop rendszereken lényegében használhatatlanul lassú és erőforrás igényes.

5.2 GLite

A EGEE projekt kezdeményezése többek között a korábbi LCG [12], (ami egy Globus 2.x alapú központosított a CERN igényeihez igazított, ezáltal az általánosságát régen elvesztő) köztes réteg web szolgáltatás alapokra történő áthelyezését tűzte ki célul. Sajnálatos, hogy a tervezés során nem vették figyelembe a kialakuló szabványosítási törekvéseket, így számos esetben párhuzamos megoldások születtek kialakuló OGSA megoldásokkal, melyek természetesen nem kompatibilisek egymással. A GLite fejlesztésénél elkövették azt a hibát, hogy egy általános keretrendszer fejlesztettek majd abban próbálták a szolgáltatásokat megvalósítani így a legtöbb jelenleg elérhető szolgáltatás funkcionalitása messze elmarad a korábbi LCG szoftverétől, így magán az EGEE projekten belüli bevezetés is várat magára. Ezzel együtt az inkompatibilis interfészek miatt lassan elveszíti Európában meghatározó szerepét. A rendszer kerete Java nyelven íródott de dicséretes módon az API-kat JAVA, C, C++, Python nyelveken is megvalósították. Nagy erőforrás igénye van az erőforrás oldalon, desktop rendszereken nem használható, telepítése nehézkes és nem kiforrott.

5.3 UNICORE

A Unicore [16] projekt elsősorban a Németországi szuperszámítógép központok

összekapcsolására létrejött kezdeményezés. Felépítésében nagyon hasonlít a ClusterGrid köztes rétegre és hasonló komponensek találhatóak benne. Minden erőforrás halmazt (usite) egy gateway komponensen keresztül lehet elérni. Minden ilyen usite több vsiteból állhat melyek mindegyike tartalmaz egy olyan komponenst mely a helyi ütemezővel való kapcsolatot végzi (Network Job Superviser - NJS). Végül az NJS a kiválasztott erőforráson a Target System Interface (TSI) segítségével elvégzi a tényleges futtatást. A rendszerkomponensek között a kommunikációt X509 tanúsítványok segítségével hitelesítik így a komponensek között a ClusterGrid-hez hasonlóan host alapú azonosítás történik és bizonyos komponensekhez a kliensek nem férnek hozzá közvetlenül ezáltal nincs szükség problémás proxy tanúsítványokra sem. A feladatok leírására Abstract Job Object-et (AJO) használják. A feladat végrehajtás mellett a UNICORE adatkezeléssel, felhasználói jogosultság kezeléssel, workflow kezeléssel kapcsolatos egyéb szolgáltatásokat is tartalmaz. A szolgáltatás interfészek és adatmodellek számos tekintetben nagyon közel állnak a OGSA legtöbb ajánlásához. A UNICORE Java nyelven íródott alapvetően nem volt cél a desktop felhasználás viszont a kitűzött célt, miszerint a Németországi nagy teljesítményű számítási és adattárolási erőforrásokat transzparensen össze tudja kötni, messze túlteljesítette. Jól láthatóan átgondolt tervezés eredménye és nem ijedtek meg a fejlesztők, hogy olyan rendszert hozzanak létre ami leszámol a Globus rendszerekből érkező sztereotípiákkal. Az UNICORE rendszer OGSA kompatibilis továbbfejlesztése az Unigrids [17] projekt.

5.4 ARC

A ClusterGriddel kb. egy időben kezdődött meg a NorduGrid [13] együttműködés kialakulása is (2001-2002), melyben elsősorban Svéd, Norvég, Finn és Dán intézmények vettek és vesznek részt ma is. Azóta különösen a sikeres (Advanced Resource Connector) ARC köztes réteg szoftvernek köszönhetően a fejlesztői csapat nemzetközi szintre fejlődött. Az együttműködés 2003 közepétől elsősorban az ARC szoftver fejlesztésére koncentrál. A tervezésnél a szokásos skandináv módszereket vetették be: kicsin, egyszerű letisztult formák, moduláris szerkezet, ne okozzon különösebb problémát az erőforrás oldalon, legyen intelligens és hatékony a kliens oldalon, ne legyen egyszeres hibapont (SPF).

A fejlesztést a Globus 2.x sorozat API-jaira alapozták de a szolgáltatásokat teljes egészében újra írták mivel azok nem voltak megfelelőek és az új szolgáltatások némelyiket már SOAP interfésszel is ellátták. Különös és hatékony megoldást eredményezett, hogy az erőforrás oldali interfész és a gridftp szerver szoros kapcsolatba került így a job műveletek egyszerű állomány műveletekre redukálódtak. Az információs rendszer egy jól optimalizált LDAP rendszer alkotja amiben csak a legszükségesebb statikus információk terjednek. Az ütemezési döntéseket a kliens oldalon hozzák meg, így az feltételezi, hogy a kliensek közvetlenül kapcsolatba tudnak lépni az erőforrásokkal. A jelenlegi rendszer legnagyobb problémája a Globus örökségből fakad amitől a fejlesztők folyamatosan igyekszenek megszabadulni.

5.5 Mindenki más

Számos más alkalmazás vagy tudományterület specifikus grid rendszer (pl.: myGrid [15]) létezik a világban melyek többé kevésbé a fenti alaprendszerek valami fajta mutációi (Japán, Kínai és Indiai grid rendszerek) vagy nem törekszenek általános megoldásra. Architektúrális szinten talán a NextGrid [14] kezdeményezés még említésre méltó, mely az OGSA-ra alapozva egy konkrétabb architektúrális ajánlást kíván megfogalmazni grid rendszerek fejlesztőinek, a projekt azonban a tervezési szakaszban tart jelenleg.

6 Grid Underground projekt

A ClusterGrid projekt kezdeti szakaszában, 2002-ben, nem találtunk megbízhatóan üzemeltethető grid köztes réteg rendszert ezért akkor az NIIF úgy döntött a saját céljaihoz illeszkedő saját rendszer fejlesztésébe kezd. Ennek első változata a ClusterGrid bróker bizonyította, hogy jó döntést hoztunk. Figyelembe véve azonban a grid technológia változását a ClusterGrid bróker rendszert teljesen újraterveztük és szabványos alapokra helyeztük. Ez a fejlesztés a új generációs ClusterGrid köztes réteg rendszer a Grid Underground azaz a GUG projekt. Már a ClusterGrid bróker tervezésekor is fontos szempont volt, hogy nem használjuk ki a ClusterGrid rendszer speciális hálózati, operációs rendszer szintű felépítését de ez az igény tovább erősödött a GUG rendszer tervezésekor. A célunk egy teljesen általános szuperszámítógépektől desktopokig használható egyszerű grid keretrendszer valamint kapcsolódó konkrét szolgáltatások létrehozása volt. A projekt 2005 novemberében indult és mára már eredményeket hozott és megtörtént az első lépés a ClusterGrid bróker lecserélésére is 2006 márciusban. A GUG köztes réteg két fő részből áll. Egy keretrendszerből mely általános nem csak grid szolgáltatások gyors fejlesztését teszi lehetővé. Ez a keretrendszer gondoskodik a szolgáltatások külvilággal történő kommunikációjáról ill. tartalmaz néhány speciális szolgáltatást (Manager, Grid Információs rendszer - GIS), mely kezeli és vezérli a rendszerben található szolgáltatások életciklusát és lehetővé teszi a szolgáltatás leírások terítését és keresését a rendszerben. A keretrendszer maga a szolgáltatásokra, mint dinamikusan betölthető modulokra tekint. A jelenlegi implementációban a keretrendszer a szolgáltatások között HTTP(S)/SOAP üzenetek továbbítását támogatja.

A rendszer másik komponense magunk a keretrendszerben megvalósított szolgáltatások. Jelenleg a rendszer az alábbi szolgáltatásokat implementálja:

- *Manger*: a speciális szolgáltatás a többi szolgáltatás életciklusát kezeli (leállítás, elindítás, konfigurálás, status). Mivel maga is web szolgáltatás így ezen keresztül dinamikus web alapú management képzelhető el az egész GUG alapokon felépített grid rendszerben.
- *Grid Információs Rendszer - GIS*: speciális szolgáltatás a szolgáltatások által magukról készített leírok peer-to-peer rendszerben történő terjesztését és keresését végzi. A GIS és a Manager szolgáltatás minden grid csomóponton fut, és minden szolgáltatás alapértelmezésben a helyileg futó információs rendszer komponenstől szerzi be, hogy a rendszerben milyen más lokálisan vagy távol futó komponensek vannak. Maga az információs rendszerben hirdetett adat rögzíti továbbá azt is, hogy maga a szolgáltatás milyen absztrakt erőforrásokat valósít meg. Az információs rendszer elkülöníti a adatokat a meta adatoktól a meta adatok a hirdetésekről és azok terjesztésének módjáról tartalmaznak információkat az adatok pedig maguk a hirdetések melyet az információs rendszer semmilyen formában nem értelmez így annak formátuma tetszőleges. Jelenleg implementált grid szolgáltatások XML leírásokat hirdetnek magukról.
- *Futtató (Exec) szolgáltatás*: egy SMP gépen egyszerű platform specifikus feladat végrehajtást és felügyeletet tesz lehetővé. A legnépszerűbb platform osztályokra készítettük le ezt a szolgáltatást (Linux, FreeBSD, Solaris, Windows). Ez a szolgáltatás teszi lehetővé, hogy a grid feladatokat akár egy otthoni felhasználó asztali számítógépén is futtathassunk.
- *Job Controller (JC)*: Megvalósítja az OGSA Basic Executation Service (BES) interfész ajánlását és ezzel egységes felületet nyújt tetszőleges feladat végrehajtó rendszerekhez. Ilyen feladat végrehajtó lehet maga az Exec szolgáltatás egy desktop rendszeren vagy a valamilyen klaszter szintű ütemező (Condor, SGE, PBS stb), szuperszámítógépeken vagy klasztereken. Maga a JC teljesen moduláris mind a feladat végrehajtó rendszerek, mind a futtatási környezet kialakításához készítettünk és készíthetőek modulok vagy más néven

backendek. A JC komponens végzi a feladatok futtatási környezetének (be- és kimeneti állományok, biztonsági jogosultságok) létrehozását is. A JC nem végez ütemezést, a hozzá ékező feladatokat gondolkodás nélkül végrehajtja.

- *SuperScheduler (SS)*: alapvetően grid szintű feladat ütemezést tesz lehetővé. Ugyanazt az interfészt beszéli mint a JC. Az ütemezési algoritmusok és erőforrás interfészek tekintetében moduláris így akár klaszter szinten, grid szinten vagy gridek összekötésére is használható. A feladat leírásokat a JC-hez hasonlóan az OGSA JSDL szabványa szerint várja.
- *Elosztott katalógus (DM)*: elosztott katalógus szolgáltatást nyújt. Többféle katalógust használhatunk egyszerre. Minden katalógusban kulcs érték párokat tárolhatunk, sem a kulcs sem az érték formátumára megkötést nem tartalmaz, az adatokat elosztottan hibatűrően tárolja a katalógus csomópontokon. Keresni benne kulcs alapján lehet. Az elosztott tároló rendszer például ebben a katalógusban tárolja UNIX inode rendszerhez hasonló adatstruktúrát az állományokról és könyvtárakról.
- *Storage Controller (StC)*: feladata, hogy állományokat eltároljon és visszaadjon. Megvalósítja a StorageElement erőforrás absztrakciót. Minden szabad adattárolási kapacitással rendelkező grid csomóponton telepíthető. Az SRM ajánláshoz közeli interfésszel rendelkezik. Az adatok mozgatására független transzport szolgáltatásokat használ (HTTP, FTP, GridFTP).
- *Storage Manager (SM)*: POSIX szerű interfészt (ls, rm, mkdir, cp, mv stb.) nyújt minden egyéb szolgáltatásnak az elosztott tároló rendszerhez. Maga állapot független így akár több független SM is lehet a rendszerben terhelés kiegyenlítés és redundancia végett.
- *Virtual Organization Manager (VO)*: a grid rendszerben lévő felhasználók és szolgáltatások virtuális szervezetekbe tömörödhetnek. A Virtuális szervezeteken belül a különböző szolgáltatásokhoz és egyéb entitásokhoz (pl.: állományok, feladatok stb.) való hozzáférés szabályozásáról és az ehhez szükséges adatok tárolásáról gondoskodik ez a szolgáltatás. A GUG rendszerben minden entitást X509 tanúsítvánnyal azonosítunk és minden entitás tagsági igazolványokkal rendelkezik arra vonatkozóan, hogy mely virtuális szervezetek tagja. Minden művelet esetén a VO szolgáltatás hivatott megválaszolni azt a kérdést, hogy ki milyen műveletet min hajthat végre.
- *Fordító szolgáltatás (Compiler)*: GUG rendszer heterogén erőforrás halmazon is lehet használni. Ebben az esetben a futtatandó alkalmazásoknak vagy architektúra függetlennek kell lennie vagy rendelkezésre álló architektúrák egy részhalmazára futtatható formában rendelkezésre kell állnia. A fordító szolgáltatás segíti a felhasználókat abban, hogy a gridet fordítási feladatokra is fel tudják használni.

Hangsúlyoznánk, hogy bár a fenti szolgáltatások mindegyike a nagyteljesítményű feladatvégrehajtással és adattárolással kapcsolatos maga a GUG keretrendszerben elkészíthető szolgáltatások nem korlátozódnak erre a területre abban tetszőleges web szolgáltatás megvalósítható.

A fenti szolgáltatásokat jelenleg egy moduláris parancssori interfészen keresztül érhetik el a felhasználók ill. számos szolgáltatáshoz (pl. elosztott tároló rendszer, biztonsági rendszer) készítettünk a használatuk megkönnyítő API-kat is.

A rendszer eddigi működés során jó tapasztalatunk volt a teljesítményét és erőforrás (cpu, memória, merevlemez) igényét illetően. Az elkészült kód mérete kicsiny gyorsan könnyen áttekinthető akár desktop gépeken is. Különös figyelmet fordítottunk arra, hogy az elkészült szolgáltatások és a keretrendszer maga könnyen telepíthető és konfigurálható legyen.

7 Összefoglalás

Bemutattuk hol tartunk és merre megyünk. Az NIIF által támogatott ClusterGrid rendszer új generációs web szolgáltatás alapú köztes rétege a GUG minden tekintetben a nemzetközi grid kezdeményezések élvonalába tartozik, szabványimplementációkat tartalmaz. Az NIIF a NorduGriddel közösen beadott és elnyert EU 6. keretprogramja által támogatott KnowARC projekt keretében a továbbfejlesztése funkcióinak bővítése tovább folyik a projektben elkészülő rendszer a GUG és ARC rendszerek legjobb tulajdonságait fogja ötvözni ezzel megnyitva az utat a két grid rendszer közötti akadálytalan átjárás felé is.

A GUG rendszer egyik legnagyobb erénye, hogy feloldja a szakadékot a hagyományos grid rendszerek és az asztali gépeken is használható rendszerek között. Keretrendszert nyújt nem csak girdes web szolgáltatások megvalósítására hanem általánosan bármilyen szolgáltatásra, de konkrét megoldásokat is nyújt a nagyteljesítményű adattárolás és feladatkezelés terén amik segítségével intézmények vagy vállalatok saját grid rendszereiket felépíthetik és összekapcsolhatják.

Az új generációs ClusterGrid rendszer és a mögötte álló tapasztalatok ipari támogatását és bevezetését az NIIF által 2006-ban létrehozott Avaxio Informatikai Kft. (www.avaxio.hu) végzi.

Hivatkozások

[1] http://www.ggf.org

[2] http://www.ggf.org/documents/GFD.30.pdf

[3] http://www.clustergrid.hu

[4] http://gug.grid.niif.hu

[5] http://www.gnustep.org

[6] http://java.sun.com/products/jdk/rmi/

[7] http://www.corba.org/

[8] http://www.w3.org/TR/soap/

[9] http://www.globus.org/

[10] http://www.ggf.org/documents/GFD.53.pdf

[11] http://sdm.lbl.gov/srm-wg/

[12] http://lcg.web.cern.ch/LCG/

[13] http://www.nordugrid.org

[14] http://www.nextgrid.org

[15] http://www.mygrid.org.uk/

[16] http://www.unicore.org/

[17] http://www.unigrids.org/