KONFIGURÁLHATÓ PORTLET (CONFLET) ALKALMAZÁSA CLUSTERGRID KÖRNYEZETBEN

Pasztuhov Dániel, dani@iit.bme.hu
Dr. Szeberényi Imre, szebi@iit.bme.hu
BME IIT

1 Bevezetés

A korai grid és párhuzamos rendszerek egyik hátránya az, hogy felhasználói felületük gyakran nagyon barátságtalan. Parancssoros kezelői felületük elsősorban a programozók igényeit elégíti ki, az informatikában kevésbé jártas szakemberek nem vagy nem könnyen tudják használni ezeket a rendszereket.

A World Wide Web és ezen belül a portál technológia terjedése azonban új lehetőségeket teremtett a felhasználói felületek kialakításának területén. Tanszékünkön – a GridSphere Portál Keretrendszer elemeit felhasználva – létrejött egy olyan portletcsomag (Condor Portlets [13]), mellyel a felhasználható webes felületről érheti el egy megadott Condor rendszer szolgáltatásait, így már mindössze egy böngésző segítségével lehetett előre megírt feladatokat futtatni. A Condor Portlets hátránya azonban, hogy a Condor rendszer specialitásait nem takarja el, a felhasználó azonban azt szeretné, ha neki csak az alkalmazásának paramétereit kelljen beállítani, a futtató környezet lehetőleg maradjon transzparens. Ha azonban minden alkalmazáshoz külön felületet készítünk, eltakarva a környezet részleteit, akkor a fejlesztőnek nagyon sok ismétlődő feladatot kell újra és újra elvégeznie.

Ezek a megfontolások vezettek a Conflet rendszer alapötletéhez: Készítsünk egyetlen konfigurálható portletet, amely a megjelenítendő oldalak kinézetét és viselkedését egy, vagy több XML leírásból veszi, mely leírások (konfigurációs fájlok) futási időben is cserélhetők. Az első elképzelések alapján elkészült implementáció [14] tapasztalataira építve a rendszert továbbfejlesztettük és segítségével létrehoztuk egy ipari alkalmazás [15] felhasználói felületét (melyet az 5. fejezetben mutatunk be). A bemutatásra kerülő Conflet rendszer alkalmas arra, hogy egy parancssoros program elindítását végző felületet elkészítsünk vele. A rendszer integrálása a ClusterGrid portál felületébe elkészült, így azt a ClusterGrid felhasználói korlátozás nélkül használhatják. A 2. fejezetben megvizsgálunk néhány hasonló rendszert a Conflet rendszerrel szembeni különbségek szempontjából. A 3. fejezetben bemutatjuk a Conflet képességeit, melyet a 4. fejezetben egy egyszerű példán szemléltetünk. Az 5. fejezetben egy bonyolultabb ipari alkalmazás felületének elkészítését írjuk le, a 6. fejezet a továbbfejlesztési lehetőségeket mutatja be.

2 Hasonló Grid rendszerek

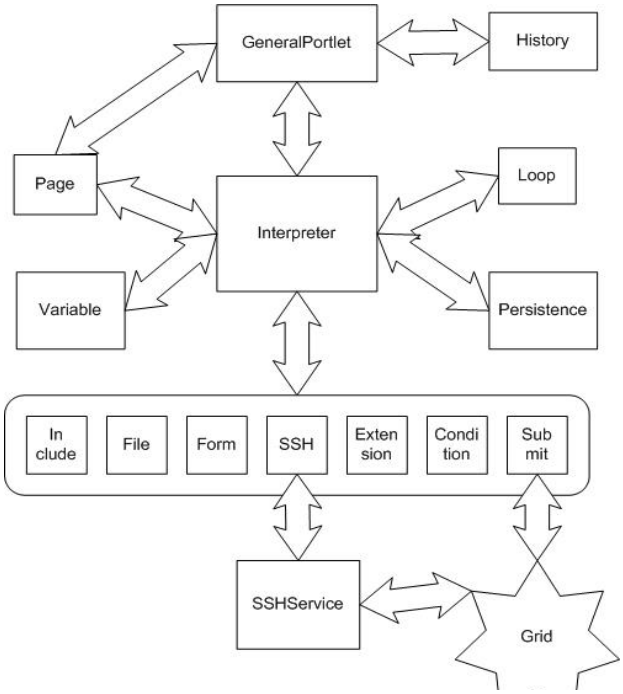
Az általunk ismert rendszerek, amelyek alkalmazás-specifikus portletek létrehozásához nyújtanak segítséget, más megközelítést alkalmaznak. A mi megközelítésünk szerint konfigurációs állományok segítségével paraméterezhető portletre van szükség, míg mások egy programozási keretrendszert valósítanak meg valamely programozási nyelvhez. Ez leggyakrabban egy objektum-orientált keretrendszert jelent, például Java osztályokat. A következőkben összegezzük a legfontosabb keretrendszereket:

- A GridSphere [10][11][12] maga is egy alkalmazásfejlesztő keretrendszer, a Confletet is ebben a környezetben fejlesztettük ki. A keretrendszer a megfelelő felület kialakításához, valamint a Grid rendszer szolgáltatásaihoz való kapcsolódáshoz számos portletet biztosít, ugyanakkor ezek pusztán általános felületet nyújtanak. Az alkalmazás-specifikus felület kialakításához a GridSphere által biztosított portletek, service-ek és más eszközök felhasználása, kiterjesztése szükséges. Egy új felhasználói felület kialakítása Java/J2EE programozást igényel egy fehér-doboz keretrendszer szolgáltatásait igénybe véve.
- A GridPort [16] kezdeményezés csatlakozási felületet nyújt a Globus Toolkiten kívül a Condorhoz és a Grid Portal Information Repositoryhoz (GPIR) is. A projekt célkitűzései közt szerepel ugyan alkalmazás-specifikus portletek fejlesztése, de ez még fejlesztés alatt áll.
- A myGrid [17] egy bioinformatikára fókuszáló e-science projekt az Egyesült Királyságban. A rendszer alkalmazás-specifikus környezete elsősorban a bioinformatika területére koncentrál.

A Conflet rendszerben alkalmazott megközelítés sokkal általánosabb, minden felhasználás és felhasználó számára alkalmas megoldást nyújt feladatok beküldésére vagy egy parancssoros alkalmazás elindítására.

3 Conflet Framework

A Conflet rendszer – melynek neve a CONFigurable portLET (konfigurálható portlet) kifejezés rövidítésével keletkezett – egy könnyen használható keretrendszer alkalmazásfejlesztők számára, akik Grid (speciálisan ClusterGrid) feladataik (vagy parancssoros programjaik) számára szeretnének portál felületet fejleszteni.



1. ábra Architektúra

A rendszer működése leegyszerűsítve a következő: a felhasználónak oldalakat jelenít meg, melyen beállíthatóak az alkalmazás különböző paraméterei, a portlet elvégez néhány egyszerű számítást, létrehoz néhány fájlt, majd végül átadja a feladatot az adott grid rendszer feladatkezelőjének, vagy elindítja a programot. Mivel ezen forgatókönyvek és programok egymáshoz nagyon hasonlók, a Conflet rendszer minél inkább minimalizálni igyekszik az alkalmazásfejlesztő által elvégzett munkát.

A Conflet rendszer a nyílt forráskódú, ingyenes GridSphere Portál Keretrendszeren alapul.A Conflet konfigurációs fájlok feltöltésével a futás közben is testreszabható. A két fő fájltípust *view*-nak és *controller*nek nevezzünk. A view definiálja a portlet kinézetét, azaz a különböző felhasználói felületelemek elhelyezését adja meg, a controller pedig a portlet viselkedését szabja meg, azaz a gombnyomásra vagy link meghívására végrehajtandó akciók sorozatát. A konfigurációs állományokból több példány is lehet a rendszerben, és ezek egyáltalán nincsenek összekötve: egy controller több view-hoz, egy view több controllerhez is tartozhat. A felhasználói beavatkozás hatására a view és a hozzátartozó controller megváltozhat.

3.1 Architektúra

A Conflet architektúrája a 1. ábrán látható. A központi modul, az „Interpreter” csak akkor hívja meg az „Include”, „File”, „Form”, „SSH”, „Extension”, „Condition”, és “Submit” modulokat, ha azokra szükség van. Az alsóbb szintek az „SSH” és „Submit” modulokon keresztül érhetők el. A moduláris felépítés rugalmas és kiterjeszthető rendszerarchitektúrát eredményez. A keretrendszer funkcionalitását könnyen kiterjeszthetjük új modulok hozzáadásával.

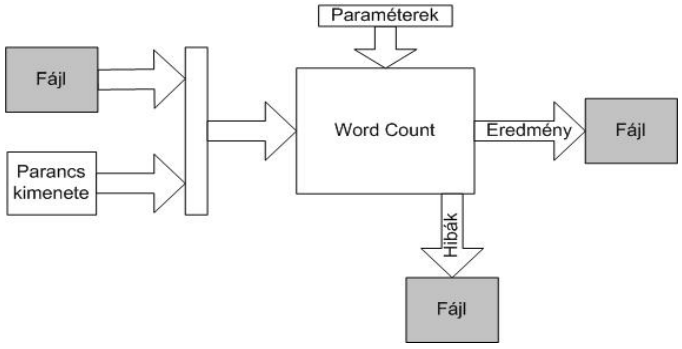
3.2 Controller és view

A controllerek létrehozásához egy egyszerű, XML-alapú parancsnyelvet terveztünk, amelyben karakterlánc változók és egyszerű vezérlési szerkezetek – mint pl. ciklusok, szelekció – használhatók. A távoli gépen tetszőleges parancs meghívható az SSH modul és a nyilvános kulcsú infrastruktúra használatával. A rendszerben több view – melyek mindegyike egy GridSphere UI Tag Library-jával kiterjesztett JSP (Java Server Pages) oldal – is szerepelhet, ezek ún. fülek alá csoportosíthatók. A füleket tartalmazó elem helye a view-n belül egy a Conflet tag library-jába tartozó elemmel könnyedén megadható. A controller az alábbi fő funkciókat képes ellátni:

- Változók definiálhatók, melyek a view-kban és a controllerekben egyaránt használhatók.
- A view-n megjelenő felületelemek értékei a controllerben változókként jelennek meg, a tulajdonságaik a controller egy-egy parancsával állíthatók be. A felületelemek tulajdonságai fájlba menthetők, és visszatölthetők onnan.
- Fájlok hozhatók létre, tölthetők le vagy fel a távoli gépre vagy az alkalmazásszerverre. A fájlok tartalmát reguláris kifejezésekkel változókba írhatjuk. Fájlírás vagy fájlolvasás megkönnyítésére ciklusok definiálhatók.
- Névvel rendelkező változócsoportok hozhatók létre és tárolhatók el XML fájlokban. A nevek listadobozba tölthetők.
- A controller kiterjeszthető speciálisan megírt Java osztályokkal, melyek egyszerű számítások elvégzésére vagy akár képek generálására is használhatók. A kiterjesztések az osztálynév, a bemenet és kimenet megadásával érhetők el.
- A különböző felhasználói beavatkozásokon kívül parancsok rendelhetők az oldalbetöltődés eseményéhez (init), valamint megadhatók olyan utasítások, melyek bármely akcióhoz tartozó parancsok előtt meghívódnak. Az akciók egymásba ágyazhatók.

4 Egyszerű alkalmazás készítése

A Conflet lehetőségeinek és működésének jobb megértése céljából ebben a fejezetben egy egyszerű UNIX-os segédprogram, a *wc* (word count) felületének kialakítását mutatjuk be.



A *wc* képes arra, hogy megszámolja hány sort, szót, betűt tartalmaz egy fájlban lévő szöveg. A gyakoribb funkciókon kívül megkapható a bájtok száma valamint a leghosszabb sor mérete is.

4.1 Előkészítés

A program működési modellje a 2. ábrán látható. A *wc* bemenete lehet egy fájl vagy egy parancs kimenete, paraméterként megadhatjuk, hogy a bemenetből mely tulajdonságokra vagyunk kíváncsiak. Az eredményt és a hibalistát fájlba mentjük.A felülettől azt várjuk, hogy legyünk képesek megadni a forrást, a paramétereket formelemek felhasználásával, valamint a kimenet és hibalista tárolására szolgáló fájlokat. Majd miután ezeket az információkat megadtuk, ellenőrzésképpen megnézhessük, hogy milyen parancsot ad majd ki a portál.

A *wc* program felületét két fülre osztjuk: az egyiken („Paraméterek”) megadjuk a program futtatásához szükséges paramétereket, a másikon („Indítás”) megjelenítjük a futtatandó parancsot, és az indítógombot. A többnyelvűség támogatásával itt nem foglalkozunk.

4.2 Fülek

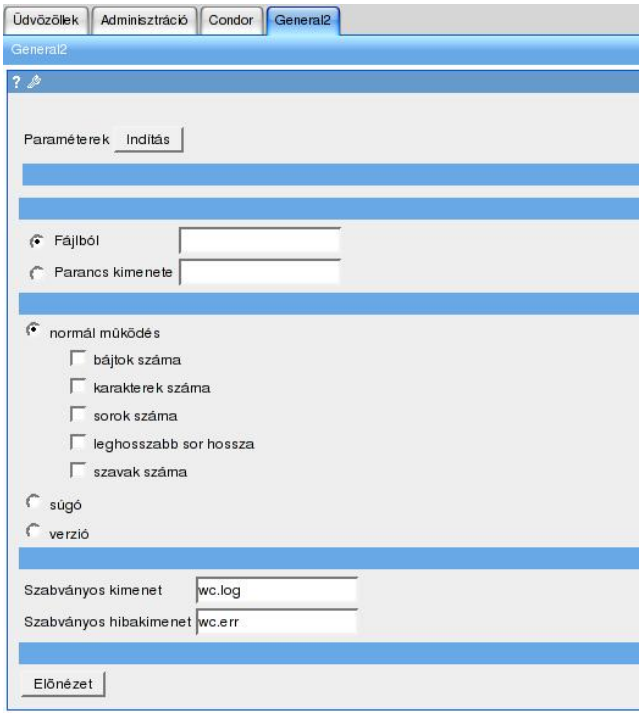
A fülek megadásához elkészítjük a *tabs.xml*-t:

```
<tabs default="params">
      <tab id="params" name="Paraméterek">
            <page view="parameters.jsp" ctl="parameters.xml"/>
      </tab>
      <tab id="start" name="Indítás">
            <page view="start.jsp" ctl="start.xml"/>
      </tab>
</tabs>
```

Mint látható, két fület készítünk el, egyiknek a *params* azonosítót és a „Paraméterek” nevet adjuk, a másiknak a *start* azonosítót és a „Indítás” nevet adjuk. Mindkettő a megadott view-val és controllerrel fog inicializálódni. Az alapértelmezett fül (mely az indítás után először megjelenik, a *params* azonosítójú fül.

4.3 A JSP-oldalak elkészítése

Első lépésként létre kell hozni a megfelelő fájlokat *parameters.jsp*, *parameters.xml*, *start.jsp* és *start.xml*. A JSP-oldalak megírásához használjuk a GridSphere User Interface Tag Libraryt [18]. A fejlesztéshez használhatjuk a Conflet-et is: segítségével webes felületről, az alkalmazásszerver újraindítása nélkül cserélhetjük ki a JSP oldalakat. Az oldalak kifejlesztését nem mutatjuk be részletesen, mindössze az eredményt ismertetjük. Az érdeklődők a teljes felület-konfigurációt megtalálható a http://bscw.ik.bme.hu/pub/bscw.cgi/0/28517 címen.



3. ábra A Paraméterek fül

A 3. ábrán a „Paraméterek” fülön lévő oldalt láthatjuk. (A szövegben való jobb elrendezés miatt az egyes blokkok címsorát elhagytuk). A „Paraméterek” és „Indítás” elemeket tartalmazó rész a Conflet által automatikusan generált rész, melyen keresztül a füleket érhetjük el nagyon könnyen.

Alatta a *wc* program bemenetét definiáló rész következik: a rádiógombokkal választhatunk, hogy milyen típusú legyen a

4. ábra Az Indítás fül

bemenet, majd egy szövegmezőben megadhatjuk azt. A kép közepén található legnagyobb blokk a *wc* segédprogram paraméterezésére szolgál: rádiógombokkal választhatunk, hogy milyen működési módban akarjuk elindítani a programot. A *súgó* és *verzió* módban a program csak a saját paraméterlistáját vagy verziószámát adja vissza, bemenetét nem is nézi. *Normál működés* esetén viszont összeállítható az érdekes információk listája. Az alsó szövegdobozos blokk segítségével a kimenetként és hibalistaként használt fájlok adhatók meg (alapértelmezés szerint rendre *wc.log* és *wc.err*). A gombbal pedig az információkat feldolgozva átválthatunk a másik fülre („Indítás"). A másik fülön – melyet a 4. ábrán láthatunk – szintén megtalálható a Conflet által generált fülnavigációs rész, valamint egy szövegmező és két gomb. A szövegmezőben jelenik meg az összeállított parancs, a *Vissza* gomb segítségével a „Paraméterek" fülre térhetünk vissza, míg az *Indítás* gombbal elindíthatjuk a parancsot.

4.4 A Controllerek elkészítése

A controllerek elkészítésében az első lépés a lapok közötti navigáció biztosítása. Ehhez egy-egy akciót kell létrehozni a két controllerben. (Megjegyzés: Feltesszük, hogy a *parameters.xml*-ben a gombhoz tartozó *action* attribútum értéke *preview*, és a *start.jsp*-ben pedig *back* és *start*. A *parameters.xml*-be az alábbi három sort kell írni, hogy a „Paraméterek" fülről át tudjunk váltani a „Indítás" fülre:

```
<action name="preview">
      <next tabid="start"/>
</action>
```

A *start.xml*-be pedig egy ehhez nagyon hasonló rész kerül. A *back* action feladata, hogy a „Indítás" fülről visszatérjen a „Paraméterek" fülre, míg a *start* feladata a parancs kiadása lesz. Ennél a pontnál kész a felületünk váza, a fülek közt lehet váltogatni az oldalon lévő gombok segítségével. (A füleknek megfelelő gombok kezelése a Confletben automatikus).

4.5 Akciók megírása

Az *parameters.xml preview* akciójának elkészítése a következő lépés. A feladat az, hogy *preview* akcióban összeállítsuk egy változóban azt a parancsot, amelyet az *Indítás* fülön majd kiírunk. Ahogy a felületen, úgy a controllerben is elkülönül a három működési mód. Míg a *súgó* és *verzió* mód parancsa könnyen megadható, a normál működés több munkát igényel: Az első *if* a rádiógomb (*input*) alapján az egyik szövegdoboz tartalmát felhasználva meghatározza a bemenetet, a *for* ciklus pedig vegigiterál az öt jelölőnégyzeten (*param1…param5*), és a bejelölteket hozzáfűzi a parancshoz. A következő részlet a kimenet hozzáfűzését végzi el:

```
<if><defined string="${stdout}"/>
  <then>
    <var name="cmd" value="${cmd}>${stdout} "/>
  </then>
  <else>
    <var name="cmd" value="${cmd}>wc.log "/>
  </else>
</if>
<if><defined string="${stderr}"/>
  <then>
    <var name="cmd" value="${cmd}2>${stderr} "/>
  </then>
  <else>
    <var name="cmd" value="${cmd}2>wc.err "/>
  </else>
</if>
<textarea beanId="command" value="${cmd}"/>
```

A kimenet (*stdout* szövegmező) és hibalista (*stderr* szövegmező) fájljának kijelölése egy-egy elágazást igényel. Amennyiben a felhasználó gépelt be fájlnevet kimenetnek vagy hibakimenetnek, akkor azt használjuk, amennyiben nem, akkor az alapértelmezettet. Végül az összeállított változót beírjuk a szövegdobozba, és ezzel készen is vagyunk a *parameters.xml* controllerrel. A *start.xml*-ben mindössze az űrlapba betöltött parancsot kell SSH segítségével a távoli gépen lefuttatni. A *start* akció a következőkkel bővül:

```
<ssh:exec command="${command}"/>
<end/>
```

Az első parancs elindítja a megfelelő parancsot (ahol *command* a szövegdoboz *azonosító*ja). Az *end* parancs pedig befejezi a portlet értelmező működését, és visszaáll alapállapotba.

5 Bonyolultabb alkalmazás

A Conflet rendszer segítségével felhasználói felületet készítettünk egy építőipari probléma [5][6][7] megoldásához. A program a előfeszített vasbeton gerendák [2][4] és külpontosan nyomott oszlopok térbeli alakváltozását számolja a geometriai és anyagi nemlinearitások figyelembe vételével. A térbeli alakváltozások a görbület rúd tengely mentén történő integrálásával számíthatók. A rúd két végének megfogási viszonyai adják a feladat peremfeltételeit. Az algoritmus magja egy globálisan konvergens rekurzió, amely képes meghatározni a külpontosan nyomott rúdkeresztmetszethez tartozó görbületet. Az iteratív eljárást [1][3] a peremérték-feladatok megoldására szolgáló Párhuzamos Hibrid Algoritmusba [8][9] ágyaztuk be. Habár a javasolt módszer nagyon robusztus, a kielégítően pontos számítás csak párhuzamos környezetben valósítható meg, mivel legalább egymillió rúdalak meghatározása szükséges.

A cél az volt, hogy olyan felhasználói felületet hozzunk létre, mellyel a módszer elérhetővé válik az ipari felhasználók számára is. Egy, a mérnöki életből vett gerenda vagy oszlop leírása 200-1000 paramétert igényel, ezek száma a geometriai komplexitástól függ. A paraméterek többsége a betonkeresztmetszet geometriáját, a vasbetétek és/vagy elfeszítő pászmák helyét és keresztmetszeti területét írja le. A felhasználó által bevitt keresztmetszeti adatok a geometria ellenőrzése céljából grafikusan megjelenítendők. Szintén meg kell adni és meg kell jeleníteni a rúd két végpontjához tartozó, a mérnöki gyakorlatban előforduló és mechanikailag lehetséges megfogáskombinációkat. Az anyagi tulajdonságoknak meg kell felelniük az Eurocode2 szabvány előírásainak. A felhasználó különböző, a szabvány által megadott szilárdságú beton, betonvas és előfeszítő pászma közül választhat. A kiválasztott anyagminőséghez a felhasználói felületnek társítania kell néhány egyéb anyagjellemzőt is, például a rugalmassági modulust. További paraméterek a rúd hosszát, a relatív páratartalmat,

a gerenda életkorát az első megterhelés időpontjában, az esetleges hőkezelés hosszát és hőmérsékletét stb. írják le. További paraméterekkel adhatók meg a rúd terhei, lehetőség van megoszló és koncentrált terhek bevitelére két, egymásra merőleges irányban.Az összes paraméter megadása után a felület elindítja a feladatot, vagy a paramétereket későbbi felhasználás érdekében elmenti.

6 Továbbfejlesztési lehetőségek

Habár a controllerek létrehozása egyáltalán nem bonyolult, a fejlesztők hozzászoktak az integrált fejlesztőkörnyezetekhez (Integrated Development Environment, IDE), ezért a jövőbeni erőfeszítéseink részben egy Conflet konfiguráció létrehozására alkalmas IDE létrehozására fognak irányulni. A választásunk a nyílt forráskódú Eclipse fejlesztőeszközre esett, mivel ez a rendszer könnyedén kiterjeszthető ún. plug-inokkal, így a funkciók jó részének megírásától megkímél bennünket. Más grid interfészek rendszerbe illesztése is a közeljövő feladata, elsősorban a web service felületű rendszerekre koncentrálva. A tapasztalatokat és más rendszerek megoldásait felhasználva egy általános, plug-inokkal bővíthető alrendszert szeretnénk létrehozni a jövőben a middleware-ek kezelésére.

Pillanatnyilag a Conflet csak feladatok vagy programok elindításához nyújt támogatást, de célunk, hogy képes legyen a program vagy feladat teljes életciklusát végigkísérni.

7 Köszönetnyilvánítás

Jelen cikkben bemutatott munka az NKFP OM-00262/2004 és 2/009/04 projekt, az OTKA TO46646, TS49885 projekt, valamint a Pázmány Péter RET-06/2006 program támogatásával készült. Köszönet illeti az EU-INFSO-50883 programot és a BVM Épelem Kft.-t a segítségnyújtásért.

8 Összefoglalás

A Conflet rendszer segítségével az alkalmazásfejlesztő képes nagyon egyszerűen portál felületet létrehozni grid (és ezen belül ClusterGrid) feladatok, valamint parancssoros programok elindítására anélkül, hogy a programokat módosítani kellene. A rendszer ötvözi a legmodernebb technológiákat, a portlet technológiát (a GridSphere Portál Keretrendszeren keresztül), a Java Server Pages, valamint az XML technológiákat (az egyes oldalak és a viselkedés leírására).

A cikkben egy példán keresztül szemléltettük a Conflet rendszer főbb tulajdonságait, működési elveit. Egy konkrét esettanulmány – vasbeton gerendák térbeli alakváltozását számító párhuzamos feladat felületének elkészítése – segítségével igazoltuk, hogy a Conflet bonyolultabb feladatok elvégzésére is alkalmas.

9 Hivatkozások

[1] Domokos, G. & Gáspár, Zs. “A global, direct algorithm for path-following and active static control of elastic bar structures”, *Int. J. Struct. Mech.*, 1995;23, 549-571.

[2] Sipos, A. A., Domokos G. “Asymmetrical, spatial deformations of reinforced concrete columns and prestressed beams”, *fib Symposium “Keep Concrete Attractive”*, 2005, Budapest, Vol. II, pp. 693-698.

[3] Sipos, A. A., Domokos G., Gáspár Zs. “The convergence features of the 2D Pelikan iteration”, *J. of Building Science*, 2005, 33 (1-2), 205-217 (in Hungarian)

[4] Domokos, G. “Global description of elastic bars”, *Zeitschr. Angew. Math. Mech.,* 1994;74, T289-T291.

[5] Brøndum-Nielsen, T. “Stress Analysis of Concrete Sections Under Service Load”, *ACI Journal, Proceedings*, 1979, V. 76., No. 2, 195-211.

[6] Brøndum-Nielsen, T. “Serviceability Limit State Analysis of Cracked, Polygonal Concrete Sections Under Biaxial or Symmetric Bending”, *ACI Journal, Proceedings*, 1986, V. 83., No. 2, 209-218.
[7] Cosenza, E. and Debenardi, P. G. “Calculation of Stresses, Deformations and Deflections of Reinforced and Prestressed Concrete Elements in Service”, *CEB Bulletin* 235, 1997, 105-142.
[8] Gáspár, Zs., Domokos, G., and Szeberényi, I. “A parallel algorithm for the global computation of elastic bar structures”, *Comput. Assist. Mech. Eng. Sci.* 1997;4, 55-68.
[9] Domokos G., Szeberényi I. “A Hybrid Parallel Approach to One-parameter Nonlinear Boundary Value Problems”, *Comput. Assist. Mech. Eng. Sci.* 2004;11,15-34.
[10] M. Russell, J. Novotny, O. Wehrens, “GridSphere: An Advanced Portal Framework”, *GridSphere Project Website (www.gridsphere.org)*
[11] M. Russell, J. Novotny, O. Wehrens, “GridSphere: A Portal Framework for Building Collaborations”, *GridSphere Project Website (www.gridsphere.org)*
[12] M. Russell, J. Novotny, O. Wehrens, “The Grid Portlets Web Application: A Grid Portal Framework”, *GridSphere Project Website (www.gridsphere.org)*
[13] Pasztuhov Dániel, *ClusterGrid Portál*, TDK dolgozat, Budapest, 2003.
[14] Pasztuhov Dániel, *Paraméterezhető portletek fejlesztése ClusterGRID Portál környezetben*, TDK dolgozat, Budapest, 2004.
[15] Pasztuhov Dániel, *Konfigurálható felületű portletek fejlesztése és alkalmazása ipari feladatok megoldásában*, Diplomaterv, Budapest, 2005.
[16] F. Berman, G. Fox and T. Hey, “Building Grid Computing Portals: The NPACI Grid Portal Toolkit”, *Grid Computing: Making the Global Infrastructure a Reality*, Ch 28, eds. John Wiley and Sons, Ltd, Chichester, 2003.
[17] myGrid Project Home Page, http://www.mygrid.co.uk
[18] GridSphere User Interface Tag Library, http://www.gridsphere.org