SISTEM SOFTWARE DE MONITORIZARE ÎN TIMP REAL, REALIZAT PRIN INTEGRARE QNX, CORBA, WINDOWS

Conf. dr. ing. Daniela Saru
Adrian Petcu

Universitatea "Politehnica" Bucureşti, Facultatea Automatică şi Calculatoare
saru@aii.pub.ro, pady@ss.pub.ro

Rezumat: Lucrarea prezintă o parte dintre rezultatele activităţii de cercetare, desfăşurate în cadrul unui proiect ce a avut drept scop studierea şi implementarea mecanismelor de funcţionare a unei întreprinderi virtuale, folosind ca instrument de integrare tehnologia middleware orientată obiect CORBA. Sistemul software, proiectat şi implementat ca rezultat al cercetării, permite monitorizarea în timp real a unei celule flexibile de fabricaţie, de la distanţă, prin intermediul oricărei reţele de comunicaţie ce utilizează protocol TCP/IP. Acest sistem înglobează module ce se execută pe platforme QNX (de timp real), ataşate celulei flexibile de fabricaţie şi module ce lucrează sub sistem de operare Windows, care permit utilizatorilor aflaţi la distanţă să monitorizeze activitatea celulei prin intermediul unei interfeţe utilizator ergonomice şi cu facilităţi multiple.

Cuvinte cheie: întreprindere virtuală, celulă flexibilă de fabricaţie, arhitectură client/server, sisteme distribuite eterogene, timp real, QNX, middleware, CORBA, ORBACUS/E, monitorizare, comandă la distanţă.

1. Introducere

Crearea unor entităţi de tip întreprindere virtuală şi monitorizarea eficientă a activităţii în cadrul acestor entităţi constituie, pentru majoritatea domeniilor de activitate, una dintre cele mai moderne abordări. Infrastructura informatică, necesară pentru buna funcţionare a întregului ansamblu, se caracterizează printr-o arhitectură complexă, întreprinderea virtuală fiind, în esenţă, un sistem distribuit eterogen. Din acest punct de vedere, este esenţială folosirea unor soluţii standardizate, de integrare a echipamentelor şi a aplicaţiilor care să ofere caracteristici de scalabilitate, interoperabilitate, flexibilitate. O opţiune profesională, care asigură toate aceste deziderate, este utilizarea unei tehnologii de integrare de tip middleware orientat obiect, aşa cum este CORBA [1][7].

Deoarece problemele legate de interacţiunea diverselor componente ale unei întreprinderi virtuale sunt multiple, în cadrul activităţii noastre de cercetare am abordat doar unul dintre aspectele posibile: integrarea unei celule flexibile de fabricaţie astfel încât ea să poată fi supravegheată şi monitorizată din orice punct al întreprinderii, indiferent că acesta este situat în aceeaşi încăpere sau într-o altă localitate. Deoarece la momentul iniţierii proiectului dispuneam de o celulă de fabricaţie monitorizată local, printr-un sistem software ce se executa sub sistem de operare QNX [6], am proiectat un nou sistem software, de data aceasta eterogen, care să permită coexistenţa şi colaborarea unor module sub sistem QNX şi Windows, integrate prin tehnologie CORBA. Noul sistem de monitorizare are o arhitectură client/server bine organizată [5], păstrează toate caracteristicile impuse de funcţionarea în timp real a celulei, oferă utilizatorului uman o interfaţă grafică deosebit de bogată în facilităţi, cu performanţe ergonomice şi de vizualizare deosebite, şi permite monitorizarea eficientă a funcţionării celulei flexibile de fabricaţie din orice punct al unei reţele bazată pe protocol TCP/IP. Datorită acestei particularităţi, sistemul proiectat poate fi folosit şi în cadrul activităţii educaţionale de tip clasic sau la distanţă atât pentru elevi, studenţi, cât şi în cadrul unor cursuri de perfecţionare a personalului unei întreprinderi sau companii.

2. Obiective propuse şi descrierea succintă a instalaţiei folosite pentru implementare

Celula flexibilă de fabricaţie, utilizată în cadrul proiectului, este o celulă de tip educaţional, DEGEM 2000 [2]. Activitatea desfăşurată a folosit drept bază pachetele software, elaborate în cadrul unui proiect anterior, în care se reuşise trecerea de la conducerea celulei prin automat programabil, la conducerea acesteia cu ajutorul calculatorului.

Structura iniţială a sistemului cuprindea trei module prin care se realiza conducerea posturilor de lucru, ca în figura 1.

Figura 1. Structura iniţială a sistemului de monitorizare

Primul modul din partea dreaptă a figurii reprezintă *diagrama ladder din automatul programabil*. Diagrama ladder este alcătuită din mişcările elementare, care se realizează în cadrul unui post de lucru. Fiecare mişcare elementară este iniţiată de o stare care poate fi setată în automatul programabil prin informaţii transmise de către calculator. Prin aceste stări setate cu ajutorul calculatorului, se realizează legătura dintre automatul programabil şi calculator.

Al doilea modul este *driver-ul de comunicaţie* cu automatul programabil. Este necesar ca realizarea comunicaţiei cu calculatorul să se facă printr-un proces separat deoarece, în cazul în care se va lucra pe un alt tip de automate, singura modificare ce va trebui făcută este scrierea unui nou driver care să realizeze comunicaţia cu noul automat. Driver-ul de comunicaţie are rolul de a transmite mesaje care conţin drept informaţii acţiunile pe care utilizatorul doreşte să le realizeze asupra instalaţiei, specificate într-un mod pe care automatul să-l poată înţelege. Pentru aceasta, driver-ul de comunicaţie trebuie să creeze mesaje conform unor reguli de format, specifice automatului programat ce urmează a fi folosit.

Al treilea modul este cel de *interfaţă grafică cu utilizatorul*, modul care permite transmiterea comenzilor de la utilizator către automatul programabil prin intermediul driver-ului de comunicaţie şi care rulează pe aceeaşi maşină cu acesta.

Scopul activităţii noastre de cercetare a fost de a realiza un sistem software, care să permită comanda celulei flexibile de fabricaţie de la distanţă, prin intermediul oricărui tip de reţea bazată pe protocol de comunicaţie TCP/IP.

Noua aplicaţie conţine o componentă client (modulul de comandă), care poate fi instalată pe staţii hardware cu sistem de operare Windows™ şi un modul software special, care mediază comunicaţia între componenta client şi driverul asociat automatului programabil. Acest modul se execută sub sistem de operare QNX, pe platforma hardware ataşată procesului (celulei flexibile de fabricaţie), şi nu interacţionează direct cu utilizatorul, având rol de server pentru modulul de comandă. Pentru comunicarea între client şi server am folosit tehnologia CORBA, un standard relativ recent, creat pentru a facilita interoperabilitatea aplicaţiilor software, distribuite eterogene[9][10].

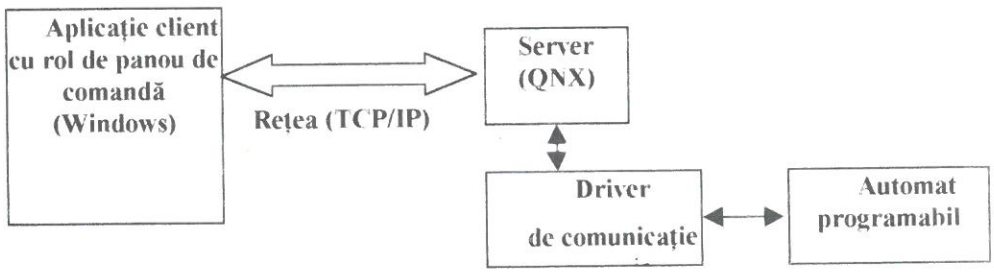

Figura 2. Noua structură a sistemului de monitorizare

Acest articol descrie doar prima etapă de dezvoltare a modulelor software, în care s-a urmărit proiectarea, implementarea şi testarea modulului software client (modulul de comandă). În acest scop, s-a folosit, în rol de server, un modul software prototip, care să simuleze comportarea serverului real prin execuţie sub sistem de operare Windows. Serverul provizoriu a înlocuit, în aceasta etapă, prezenţa fizică a celulei flexibile de fabricaţie şi a uşurat munca de testare a modulului client deoarece ambele componente ale aplicaţiei se executau sub acelaşi sistem de operare. Cea de-a doua etapă dedicată realizării modulului software server definitiv, gândit pentru a lucra sub sistem de operare QNX şi a comunica prin intermediul unei reţele cu modulul client, va fi descrisă într-un articol separat.

3. De ce am ales ca soluţie de integrare tehnologia CORBA?

În ultimii zece ani, specificaţiile CORBA propuse de către OMG (Object Management Group) [3], au devenit un standard de facto în realizarea aplicaţiilor distribuite eterogene, asigurând un înalt nivel de integrare, colaborare şi interoperabilitate a aplicaţiilor în sistemele informatizate [9][1]. De la introducerea, în 1991, a standardului CORBA 1.1, un mare număr de dezvoltatori software au implementat produse proprii bazate pe aceste specificaţii, destinate unui număr impresionant de platforme, sisteme de operare şi limbaje de programare. Concurenţa dintre dezvoltatori oferă utilizatorilor libertatea de alegere a produsului dorit în funcţie de caracteristicile problemei ce urmează a fi soluţionate, iar prezenţa unor specificaţii standardizate asigură compatibilitate şi portabilitate [3][10].

Alegerea unui produs ia în calcul, în special, considerente de performanță, de preț sau licență de utilizare. În cazul proiectului pe care l-am realizat aveam nevoie de un produs care să permită integrarea unor componente software cu execuție sub sistem de operare de uz general (Windows), dar și de timp real (QNX). De asemenea, era necesar ca produsul să poată fi utilizat gratuit, în scopuri noncomerciale (deoarece proiectul nu avea în vedere realizarea comercializării imediate a soluției implementate), dar, în același timp, se dorea utilizarea unui produs performant pentru ca soluția să fie viabila și, eventual, să fie comercializată prin achiziționarea unei licențe corespunzătoare. În acest context, am ales ca implementare CORBA produsul ORBACUS/E, dezvoltat de Object Oriented Concepts (OOC), care poate fi utilizat gratuit în scopuri noncomerciale, fiind disponibil pentru Microsoft Windows ™ și pentru o gamă largă de implementari Unix/Linux și fiind special proiectat pentru a rula pe sisteme încapsulate (*embedded systems*) și de timp real. Limbajele de programare ce pot fi folosite pentru implementarea aplicațiilor sunt C++ și Java, produsul fiind distribuit *open source* [4][9].

4. Cerințe specifice aplicației. Interfața IDL proiectată

Principalele funcții ale modulului de comandă (modulul client) sunt cele de vizualizare și de acționare, prin intermediul unor comenzi, a/asupra evoluției celulei flexibile de fabricație. Ar părea, deci, suficientă includerea a doar două metode apelabile în cadrul interfeței ce descrie facilitățile de colaborare între componentele client și server: una de citire a stărilor serverului și alta pentru transmiterea codului specific comenzilor.

Deoarece componentele aplicației software urmau să colaboreze conform standardului CORBA, aplicația fiind de tip distribuit eterogen [7][9], interfața a fost scrisă cu ajutorul OMG IDL (Interface Definition Language) [10][8]. Înainte, însă, de a fi proiectată, au fost analizate mai în detaliu particularitățile problemei ce urma a fi rezolvată, pentru a fi identificate și alte funcții necesare pentru o cât mai completă și eficientă monitorizare a procesului avut în vedere. În continuare, sunt prezentate cateva dintre aceste particularități.

În primul rând, s-a constatat că apelurile trebuie să se desfașoare rapid, fiind tratate într-un timp scurt de către server care, în varianta finală, rulează sub QNX, un sistem de operare în timp real [6].

De asemenea, deoarece se execută interogări periodice ale serverului pentru aflarea stării acestuia, iar frecvența interogărilor poate fi crescută de către operatorul uman în funcție de necesități, se dorește ca volumul de informații vehiculate prin rețea, în timpul apelurilor de metode, să fie cât mai redus. Această cerință este motivată și de faptul că, datorită folosirii protocolului TCP/IP ca suport de comunicație CORBA, nu este obligatoriu ca serverul și clientul să se afle în aceeași rețea locală (LAN), ci oriunde în rețeaua Internet. În acest caz, pot apărea dificultăti de comunicație datorită limitărilor de bandă sau a variațiilor de trafic pe parcursul rețelei. Informația transmisă trebuie să fie, deci, foarte succintă.

În afară de modulul client, cu rol de panou de comandă pentru celula flexibilă de fabricație, se pot proiecta și alte tipuri de aplicații client, care să folosească aceeași interfață (și, deci, același modul server) pentru a obține date în plus față de minimumul necesar pentru vizualizare și comandă, de exemplu informații legate de timp. O astfel de aplicație poate stoca informațiile colectate într-o bază de date, folosită ulterior pentru a se analiza funcționarea celulei flexibile de fabricație pe durata de timp în care s-a realizat supravegherea.

Toate aceste particularități, identificate în etapa de analiză a problemei, au determinat proiectarea unei interfețe care să conțină mai multe variante de metode de apel, bazate pe cele două tipuri de metode descrise inițial. În continuare, se poate observa structura fișierului "Cellcom.idl", cu metodele comentate corespunzător:

```
// timpul întors de server
struct inftime {
        short sec;
        short min;
        short hour;
        short day;
        short month;
        short year;
};
interface cellcom {
        long display ();
//întoarce un bitmask cu stările activate ale celulei
        long display_t (out inftime time);
//întoarce informații de stare și de timp
        void command (în short cmd);
```

```
//trimite o comandă celulei
        long display_cmd (în short cmd);
//trimite o comandă şi întoarce informaţii de stare
        long display_cmd_t (în short cmd, out inftime time);
//trimite o comandă şi întoarce informaţii de stare şi timp
};
```

În aplicaţia client, cu rol de panou de comandă pentru celula flexibilă de fabricaţie, se folosesc doar metodele *display()* şi *command()* ale interfeţei.

5. Structura şi funcţionalitatea aplicaţiei *Remote Cell Control*

Aplicaţia software cu rol de panou de comandă pentru celula flexibilă de fabricaţie – *Remote Cell Control* - este destinată execuţiei pe platforme Microsoft Windows™. În cadrul proiectării acestei aplicaţii, s-a acordat o deosebită atenţie interfeţei cu utilizatorul, tipului de informaţii prezentate şi formei prezentării. Bazat pe o interfaţă robustă şi intuitivă, *Remote Cell Control* implementează, pe lângă utilitatea sa de bază, facilităţi sporite de vizualizare grafică şi statistică a evoluţiei în timp real a celulei de fabricaţie căruia îi este destinat.

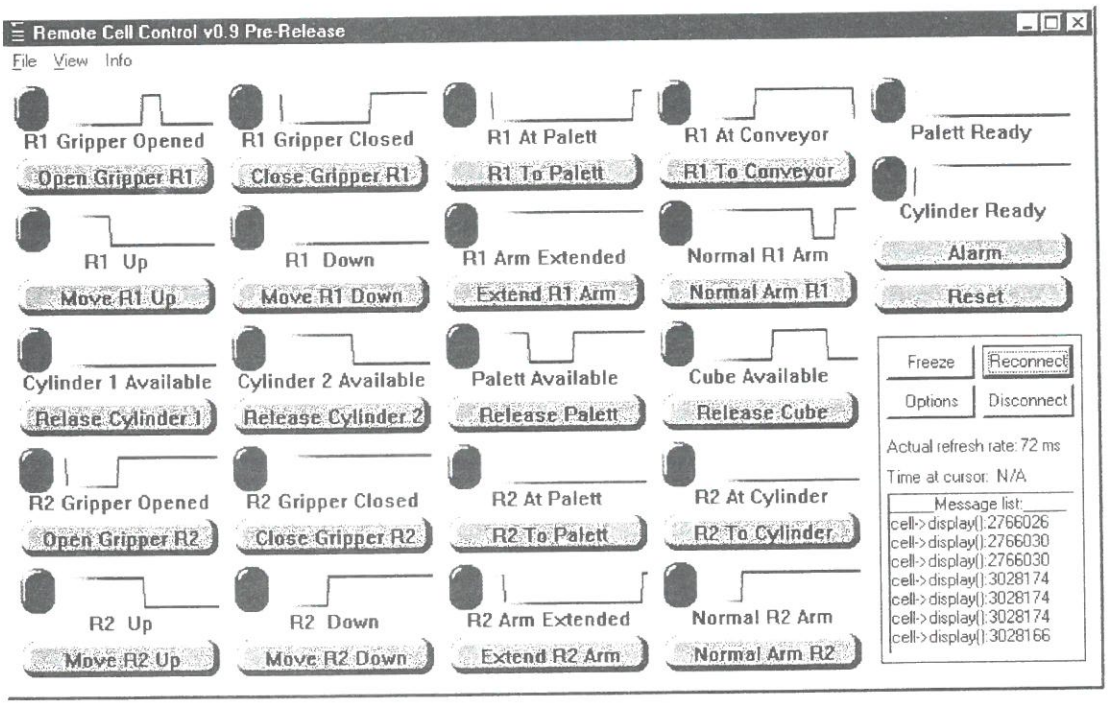

Figura 3. Interfaţa grafică a aplicaţiei cu rol de panou de comandă, *Remote Cell Control*

Valorificând informaţiile furnizate de server care arată evoluţia în timp a stărilor obiectului monitorizat, *Remote Cell Control* construieşte grafice animate pentru toate cele 22 de variabile care descriu starea acestui obiect. Animaţia se realizează prin deplasarea fiecărui grafic spre partea stângă a miniferestrei care îl găzduieste, unde sunt vizualizate evenimentele cele mai "vechi" în timp faţă de momentul curent. Pe măsură ce graficul "înaintează", în partea dreaptă apar noi informaţii de stare.

Programul permite controlul direct asupra celor mai importante variabile de stare ale celulei flexibile de fabricaţie prin acţiunea asupra butoanelor inscripţionate sugestiv, plasate sub graficele de evoluţie. Operatorul poate sesiza producerea tranziţiilor de stare la nivelul diferitelor componente ale celulei şi datorită unor elemente vizuale cu rol de led-uri ON/OFF, plasate în partea stângă a fiecărui grafic. La apăsarea unui buton din setul celor 22, se trimite prin reţea o anumită comandă adresată serverului, folosind mecanismele de comunicaţie CORBA. Comanda va fi livrată apoi celulei de fabricaţie, prin intermediul driver-ului, de către server. Până la confirmarea vizuală în fereastra aplicaţiei *Remote Control Cell* a tranziţiei comandate, operatorul uman va percepe o scurtă întârziere, de obicei neglijabilă. Această întarziere însumează timpul de propagare a informaţiei în reţea şi timpul de reacţie a celulei de fabricaţie, dar poate fi afectată în mod considerabil de o eventuală opţiune făcută de către operator pentru a micşora frecvenţa de împrospătare a informaţiilor de pe ecran.

Minipanoul afişat în partea dreaptă a ecranului permite controlul conectării cu serverul, accesul la opţiuni, şi "îngheţarea" animaţiei graficelor prin folosirea butonului *"Freeze"*.

Opţiunea *"Freeze"* facilitează observarea în detaliu a tranziţiilor de stare, produse de la un moment de tip anterior până la iniţierea comenzii de "îngheţare" a graficelor. Ordinea în care s-au produs evenimentele pe unul dintre graficele afişate se poate determina usor, prin simpla poziţionare a cursorului mouse-lui într-un punct ales pe grafic şi citirea timpului, etichetat pe minipanou cu *"Time at cursor"*. Această informaţie reprezintă timpul scurs de la producerea evenimentului indicat de cursorul mouse-lui până la producerea evenimentului figurat în extrema dreaptă a graficului (apărut în timp exact la momentul apăsării butonului *"Freeze"*). În conditiile existenţei unei incertitudini referitoare la ordinea de succesiune a evenimentelor de pe mai multe grafice, metoda descrisă oferă posibilitatea realizării unor comparaţii cu grad mare de acurateţe, care nu pot fi furnizate de către metoda mai simplă, a comparaţiei vizuale.

După apăsare, butonul *"Freeze"* îşi schimbă eticheta în *"Continue"*. O nouă apăsare va determina revenirea la modul de afişare animată a graficului de la momentul curent, spre partea stângă a miniferestriei apărând evoluţii ale graficului ce au avut loc în timpul stării *"freeze"*, dar care nu au putut fi vizualizate la momentul respectiv. De menţionat că singurele elemente "îngheţate" sunt graficele, led-urile continuând să semnaleze operatorului stările curente ale serverului.

Butoanele *"Reconnect"* şi *"Disconnect"* controlează conectarea panoului de comandă la server. Reconectarea se realizează, în general, pentru a activa noi caracteristici de funcţionare, precizate prin modificarea anumitor valori cu ajutorul facilităţii *"Options"*, dar se poate folosi şi după închiderea sau pierderea conexiunii panou de comandă – server.

Opţiunile (*"Options"*) permit alegerea numelui serverului la care se va încerca realizarea conectării, adresa IP a acestuia, precum şi portul pe care se va realiza comunicaţia. Este obligatorie introducerea a cel puţin unuia dintre primele două câmpuri, celălalt fiind opţional. Portul pe care serverul va aştepta conexiuni are valoarea implicită 2002, şi nu trebuie schimbat decât în situaii speciale (dacă pe calculatorul ce găzduieşte serverul există o altă aplicaţie ce foloseşte acest port). Tot ca opţiune se poate specifica, în milisecunde, rata de *refresh* cu care se face vizualizarea informaţiilor pe ecran. Inversul acestei valori reprezintă frecvenţa de interogare a serverului pentru obţinerea informaţiilor de stare. Dacă reţeaua sau serverul influenţează timpul de răspuns în sensul creşterii valorii, această frecvenţă va fi mai mică. De aceea, în minipanoul din partea dreaptă a ferestrei aplicaţiei se dau şi informaţii cu privire la rata reală de *refresh*, sub forma unei latenţe (timp scurs între două apeluri consecutive ale metodei *display()* a serverului). Pentru a furniza o viziune generală asupra comportării conexiunii, se calculează şi se oferă ca informaţie media ultimelor 10 valori ale latenţei.

În cadrul minipanoului din partea dreapta-jos a ecranului, sunt afişate şi toate mesajele (numele apelurilor de metode) adresate serverului. Cele mai frecvente apeluri aparţin metodei *cell->display()*, alături de care se specifică răspunsul întors de server sub forma unei valori de tip *long*, conţinând codificarea stărilor.

6. Particularităţi ale etapei de testare a aplicaţiei

Deoarece dezvoltarea aplicaţiei client - *Remote Cell Control* – s-a realizat pe baza unor specificaţii bine stabilite, a fost posibilă şi, în acelaşi timp, mai convenabilă folosirea unui server de simulare provizoriu (prototip), proiectat pentru a fi executat sub sistem de operare Windows. Modulul software utilizat a înlocuit, în aceasta etapă, prezenţa fizică a celulei flexibile de fabricaţie şi a uşurat munca de testare a modulului client deoarece ambele componente ale aplicaţiei se executau sub acelaşi sistem de operare.

Serverul provizoriu trebuia să înglobeze, în mare, funcţionalitatea unui server real, a driver-ului şi automatului programabil comandat. Pentru simularea stărilor din automatul programabil, serverul provizoriu foloseşte un vector cu 22 de stări bivalente. (Pseudo)aleator, dar conform unei probabilităţi precizate de operator, serverul generează o serie de tranziţii în cadrul celor 22 de stări, complementându-le. În momentul primirii de la aplicaţia client a unei comenzi de modificare a stării, serverul memorează modificarea cerută. Astfel, modificarea stărilor simulate în server se realizează fie în mod automat, fie la cererea clientului, fără o coordonare specială, dar suficient de realist, pentru efectuarea testării sau realizarea unei demonstraţii de funcţionare a aplicaţiei panou de comandă.

Înlocuirea serverului provizoriu cu serverul real, care se execută sub sistem de operare QNX, se face foarte uşor, printr-o simplă substituţie, deoarece comunicaţia se bazează pe folosirea aceleiaşi interfeţe IDL. După cum am precizat deja, problemele legate de dezvoltarea aplicaţiei server QNX capabile să colaboreze cu aplicaţia client prin mecanisme specifice tehnologiei CORBA vor fi prezentate în cadrul unui articol separat.

7. Concluzii

Activitatea de cercetare, descrisă în cadrul acestui articol, a avut drept scop realizarea modulelor software necesare pentru supervizarea şi controlul de la distanţă a celulei flexibile de fabricaţie DEGEM 2000, celulă de tip educaţional [2]. S-a avut în vedere construirea unui sistem de supervizare/control distribuit eterogen, care să permită utilizarea sistemelor de operare de tip Windows şi QNX [6] pentru modulele client şi, respectiv server, ca instrument de integrare fiind folosită tehnologia CORBA [8]. Pentru testarea aplicaţiei client cu rol de panou de comandă a celulei flexibile de fabricaţie, s-a utilizat un modul server provizoriu, special conceput, care să înlocuiască prezenţa fizică a celulei flexibile de fabricaţie şi să uşureze munca de testare a modulului client, executându-se sub acelaşi sistem de operare ca şi clientul. Serverul provizoriu poate fi uşor înlocuit cu modulul server real, care lucrează sub sistem de operare QNX şi care realizează efectiv transmiterea comenzilor către driver-ul celulei flexibile de fabricaţie deoarece, în implementarea ambelor servere, s-a folosit aceeaşi interfaţă OMG IDL [10][8]. Detalii legate de proiectarea aplicaţiei server QNX vor fi prezentate în cadrul unui articol separat.

În cadrul proiectului, a fost conceput şi implementat un modul cu rol de panou de comandă, care permite vizualizarea stărilor celulei flexibile de fabricaţie şi controlarea acestor stări prin intermediul unor comenzi. S-au avut în vedere atât aspectele legate de operarea în timp real, cât şi cele legate de funcţionalitatea interfeţei cu utilizatorul, concepută ca panou de comandă complex, ergonomic, intuitiv şi multifuncţional. Prin modalitatea de proiectare aleasă şi prin utilizarea protocolului TCP/IP ca suport de comunicaţie CORBA s-a asigurat funcţionarea corectă şi performanţă a aplicaţiei atât în situaţia în care serverul şi clientul se află în aceeaşi reţea locală (LAN), cât şi în situaţia în care aceste componente sunt situate oriunde în reţeaua Internet.

Deoarece funcţionalitatea celulei flexibile de fabricaţie poate presupune că, în afară de aplicaţia cu rol de panou de comandă, pot exista şi alte tipuri de aplicaţii client, ne propunem ca, în activitatea de cercetare viitoare, să creăm module software suplimentare pentru îmbogatirea opţiunilor oferite utilizatorilor. De exemplu, avem în vedere construirea unei baze de date, care să stocheze în timp real şi să actualizeze informaţii despre funcţionarea diverselor componente ale celulei flexibile de fabricaţie, despre stările în care se află aceste componente, despre operaţiile realizate pe anumite perioade de timp, despre materialele consumate şi existente în magazie, despre utilizatorii celulei etc. Aceasta bază de date va fi folosită prin intermediul unui server dedicat, care să poată interacţiona cu modulul server ce supervizează şi comandă funcţionarea celulei flexibile. Pentru implementare, ne-am propus să folosim MySQL şi PHP, luând în considerare atât criterii de performanţă, cât şi pe cele legate de posibilitatea utilizării gratuite a acestor instrumente software.

În ansamblu, aplicaţia proiectată constituie un instrument util în studierea mecanismelor ce stau la baza funcţionării unei întreprinderi virtuale, poate fi utilizată cu succes ca suport pentru implementarea unor soluţii practice, viabile în acest domeniu şi constituie un produs software ce poate fi folosit în cadrul activităţii educaţionale de specialitate clasice sau de tip învăţământ la distanţă.

Bibliografie

1. ***: CORBA documentation, http://www.corba.org , documentaţie Internet.
2. ***: Documentaţia pentru celula flexibilă de fabricaţie DEGEM 2000.
3. ***: Object Management Group, http://www.omg.org , documentaţie Internet.
4. ***: Object Oriented Concepts Inc., http://www.ooc.com , documentaţie Internet .
5. **ORFALI, R., D. HARKEY, J. EDWARDS**: Client/Server Survival Guide, Wiley Computer Publishing Group, New York, USA, 1999.
6. ***: QNX – Watcom C – Library Reference Manual, documentaţie de firma.
7. **SARU, D.**: Contribuţii la studiul tehnologiilor actuale de realizare a aplicaţiilor software în sisteme distribuite eterogene, Teză de doctorat, Universitatea "Politehnica" Bucureşti, 1998.
8. **SARU, D., A.D. IONIŢĂ**: Sisteme de programe orientate pe obiecte, Editura ALL Educaţional, Bucureşti, Romania, 2000.
9. **SERAIN, D.**: Enterprise Application Integration, L'architecture des solutions e-business, Dunod, Paris, 2001.
10. **SIEGEL, J.**: CORBA 3 Fundamentals and Programming, Second Edition, Wiley Computer Publishing Group, New York, USA, 2000.