SISTEM PENTRU MODELAREA PROCESELOR DE BUSINESS INTER INSTITUȚII – SIBPA

Victor Popa
vpopa@ici.ro

Liliana Constantinescu
lconst@ici.ro

Institutul Național de Cercetare-Dezvoltare în Informatică, ICI, București

Rezumat: Lucrarea prezintă un sistem flexibil pentru modelarea, execuția și monitorizarea proceselor de business din diverse domenii de activitate (business, administrație, industrie, financiar, sănătate etc.) bazat pe servicii Web. Sistemul pune la dispoziția utilizatorilor (proiectanți de procese, manageri de procese, executanți în cadrul proceselor) instrumente adecvate pentru eficientizarea muncii lor. Astfel, proiectanții de proces utilizează editorul de procese, pentru a proiecta în mod grafic diagrama procesului, care este un graf orientat de task-uri ce conține ca noduri task-uri manuale, task-uri automate, servicii Web. Managerii de procese utilizează instrumentele de monitorizare pentru superviza execuția proceselor, iar utilizatorii executanți în cadrul proceselor utilizează instrumentele pentru execuția task - urilor manuale.

Cuvinte cheie: procese de business, servicii Web, rețele Petri, interoperabilitate intre servicii, modelare procese.

SIBPA modelează procesele de business interorganizații ca servicii *Web compuse din subprocese locale (module)* [1]. Variabilele globale, utilizate în interiorul modulelor, sunt de tipul: document, binary, string, datetime, boolean, decimal și intră ca operanzi în diverse acțiuni ale task-urilor componente ale modulelor.

Variabilele de tip document stochează documente XML, care suportă operații de creare, actualizare, vizualizare, ştergere, transformare. Descrierea structurii unei variabile de tip document este realizată prin schema XML ataşată acesteia. Operațiile pe o variabilă de tip document utilizează frecvent schema XML ataşată, astfel că, la crearea documentului, macheta documentului și regulile de validare sunt generate automat, conform schemei ataşate. Operațiile de actualizare sau transformare utilizează schema XML pentru selectarea părților din document, care vor fi actualizate sau transformate.

Variabilele de tip binary sunt utilizate pentru a memora documete în format binar sau text. Ele suportă operații de creare, ştergere, căutare.

Variabilele de tip string memorează șiruri de caractere alfanumerice, variabilele de tip datetime memorează informații de tip oră sau dată calendaristică, variabilele de tip boolean memorează valorile „true" sau „false", iar variabilele de tip decimal memorează date numerice.

SIBPA include în arhitectura sa instrumente ca:

- editorul de dezvoltare a subproceselor de business (modulelor) [1], [3], [4], [5], [6], [7];
- serverul de aplicație pentru execuția modulelor proceselor de business [2];
- consola pentru monitorizarea execuției proceselor de business [2];
- editorul de scheme XML [1], [3], [4], [5], [6], [7];
- editorul de dezvoltare scripturi XSLT [1], [3], [4], [5], [6], [7];
- instrumente de vizualizare a stărilor modulelor.

Editorul este un mediu grafic intuitiv care permite dezvoltarea modulelor componente ale proceselor de business, utilizând obiecte grafice de tip: Task, Web Service, Decizie, Timer, Join, Event.

Modulele dezvoltate în editor sunt codificate în format XML și sunt salvate în fișiere cu extensia XML.

Serverul de aplicație este dezvoltat ca un serviciu Web, care se instalează pe mașini Windows și execută instanțele modulelor proceselor de business.

Serverul de aplicație execută modulele instalate pe server conform logicii codificată în interiorul modelelor.

Consola de monitorizare se instalează pe mașini Windows, Pocket PC etc. și permite interacțiunea utilizatorului cu sistemul.

Consola permite utilizatorilor să execute activități cum sunt: conectarea la un server de aplicație, instalarea modulelor pe serverul de aplicație, lansarea în execuție a modulelor de tip manual instalate pe un server de aplicație, vizualizarea agendei de lucru proprie fiecărui utilizator, selectarea task-urilor de pe

agendă în vederea execuției sau abortării acestora, crearea şi actualizarea documentelor, actualizarea drepturilor de acces (roluri, parole etc.).

Editorul de schemă XML permite construirea schemelor XML în mod grafic, înlăturând astfel munca dificilă de scriere a tagurilor.

Editorul de dezvoltare scripturi XSLT permite generarea scheletului scriptului de transformare a unui document XML plecând de la schema XML ataşată documentului.

Instrumentele de vizualizare a stării modulelor aflate în execuție permit vizualizarea drumului parcurs în execuție de modul până la momentul cererii vizualizării.

1. Structura aplicațiilor

O aplicație SIBPA este compusă din module distribuite pe diverse servere de aplicație, care interacționează între ele conform logicii de business.

Tipurile de module permise sunt *module manuale* (care se lansează în execuție manual de către utilizator utilizând consola), *module callable* care sunt lansate în execuție de către alte module prin acțiunea CALL şi *module de tip event* care sunt lansate în execuție de către alte module utilizând acțiunea Send.

Comunicarea între module se face în mod asincron, utilizând acțiuni ca Send, Reply, Receive, cât şi în mod sincron utilizând acțiunea CALL.

Instanțele modulelor create prin execuția acțiunii Send sunt identificate prin chei logice unice, astfel în schimbul utilizării ID-urilor fizice de instanțe în procesul de comunicare se utilizează chei logice.

O aplicație SIBPA poate de asemenea chema clase externe sau servicii Web pentru a integra componentele externe deja existente cu SIBPA.

Un caz particular de aplicație îl reprezintă acela în care toate modulele sale sunt instalate pe un singur server de aplicatie, aşa cum sunt dezvoltate aplicațiile client-server.

Paşii necesari dezvoltării unei aplicații SIBPA sunt următorii:

- stabilirea modulelor componente şi a tipurilor acestora,
- dezvoltarea modulelor, utilizând editorul (se specifică variabilele, blocurile modulului etc.) şi salvarea modulelor în fişiere de format XML,
- instalarea modulelor pe serverele de execuție aplicații,
- lansarea în execuție a modulelor manuale şi monitorizarea execuției.

2. Structura modulelor

Un modul este construit cu ajutorul editorului SIBPA şi conține un singur bloc de tip Start, o mulțime arbitrară de blocuri de tip Decision, Task, Web Service, Join, Event, Timer şi cel puțin un bloc de tip Stop.

Serverul de aplicație execută fiecare modul începând cu blocul Start al acestuia şi continuând cu execuția în paralel a blocurilor succesoare lui (blocurile sunt conectate prin săgeți de nodul Start). Execuția unui modul se temină când este executat un bloc de tip Stop.

Blocurile de tip Task sunt cele mai importante într-un modul, deoarece conțin liste de acțiuni ce vor fi executate secvențial. Dacă una din acțiuni iese pe excepție, atunci întregul task iese pe excepție şi utilizatorul trebuie să monitorizeze felul cum va continua execuția modulului.

Execuția unui Task de către serverul de aplicație este făcută în următorii paşi:

1) trecerea task-ului în starea „activ" şi înregistrarea task-ului în agenda de taskuri;
2) selectarea task-ului din agendă, de către utilizator;
3) execuția secvențială a acțtiunilor din interiorul task-ului, de către serverul de aplicație.

Acțiunile din lista unui task pot fi de diferite categorii (tipuri) şi anume:

- acțiuni pentru manipularea variabilelor de diverse tipuri (decimal, string, binary, document, boolean);
- acțiuni de gestiune a bazelor de date (SaveDB, LoadDB etc.);
- acțiuni de integrare cu aplicațiile deja existente (Call Class, Call Web service);
- acțiuni de comunicare între module (Send, Reply, Receive, Call);

- alte acțiuni (trimitere e-mail-uri, reasignare task-uri etc.).

Controlul în interiorul unui modul este realizat atât implicit, prin succesiunea blocurilor din modul, cât şi explicit, cu ajutorul blocurilor de decizie, blocuri ce conțin liste de condiții ce sunt evaluate în timpul execuției şi, în funcție de valoarea de adevăr calculată, se merge pe o ramură sau pe alta.

Pentru sincronizarea diferitelor ramuri din modul, se folosesc blocuri de tip „Join" (And şi Or). Astfel, dacă într-un modul este nevoie ca mai multe ramuri executate în paralel să se aştepte una pe alta până se termină toate ca apoi să se unească într-o singură ramură, se foloseste un bloc de tip „Join And". Blocurile de tipul „Join Or" sunt utilizate în situațiile când mai multe ramuri au nevoie să se unească în una singură, în momentul când cel puțin una din acestea s-a executat.

Blocurile de tip „Event" simulează acțiunea Receive astfel: când un bloc de tip Event este executat, se caută în coadă de mesaje mesajul care verifică condițiile specificate. Dacă un astfel de mesaj există, acesta este memorat în variabila specificată de acțiunea Receive şi controlul trece la execuția următoarei acțiuni. Dacă un astfel de mesaj nu există, blocul Event se blochează până la sosire, în coada de mesaje a unui mesaj care îndeplineşte condițiile.

Alt tip de blocuri întâlnite într-un modul sunt blocurile de tip „Timer". Un bloc de tip „Timer" este adăugat în interiorul unui modul când este necesar ca o ramură a modulului să fie suspendată pentru o perioadă de timp sau până la o anumită dată, în funcție de desfăşurarea logică a procesului modelat. După expirarea acestei perioade, ramura este deblocată şi se pot executa în continuare blocurile din modul.

Terminarea execuției unui modul se face la întâlnirea unui bloc de tip „Stop". Dacă modulul a fost lansat în execuție de către alt modul, terminarea modulului apelat duce la transferul controlului, împreună cu o listă de valori de ieşire, către modulul apelant.

3. Studiu de caz

Aplicația va fi constituită din trei module care vor interacționa între ele prin accesul la baze de date comune şi acțiuni de tip Call. Aceste module vor fi:

- introduceSolicitare
- rezolvăSolicitare
- urmărireSolicitare

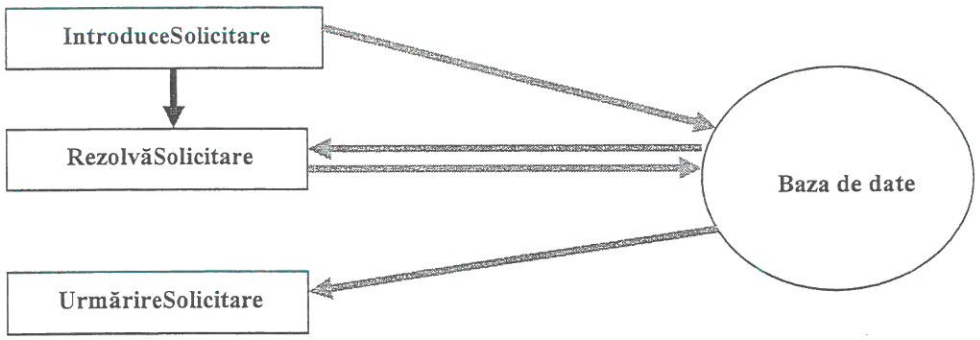

Figura 3. Structura aplicației

În primul modul, este nevoie ca utilizatorul să transmită solicitarea către registratură, iar funcționarul din acest departament să îi acorde un număr de înregistrare, să completeze registrul, să îi comunice

cetățeanului numărul de înregistrare şi să trimită mai departe solicitarea către secretarul genereal. Acesta va trebui să fie lansat în execuție de către cetățean şi va avea două task-uri care trebuie executate de el, cel în care cetățeanul transmite solicitarea şi cel în care acesta îşi vizualizează numărul de înregistrare. Între cele două task-uri va mai fi un task în care vor fi cuprinse acțiunile compartimentului de înregistrare.

Al doilea modul va fi de tip Event, acesta fiind lansat în execuție la trimiterea de către registratură a solicitării către secretarul general. Primul task va fi asignat secretarului general. Task-ul va cuprinde acțiuni în care secretarul va vizualiza solicitarea şi o va repartiza spre rezolvare unui compartiment.

Se poate observa că următorul task, cel care cuprinde acțiunile compartimentului de rezolvarea va trebui să fie asignat în mod dinamic. În acest ultimul task, compartimentul de rezolvare va trebui să vizualizeze solicitarea, să o rezolve şi să completeze rubrica Ieşire din înregistrarea ataşată solicitării.

Al treilea modul este cel cu ajutorul căruia cetățeanul vizualizează stadiul în care se află solicitarea sa. Acest modul va trebui să fie lansat în execuție de către cetățean şi va cuprinde un singur task în care cetățeanul va introduce numărul de înregistrare al solicitării şi va vizualiza în ce stadiu se află aceasta.

În continuare, sunt prezentate cele trei module, prin precizarea structurii şi proprietățile acestora:

Primul modul (figura 4):

- Nume: IntroduceSolicitare
- Tip: Manual
- Role: Utilizator

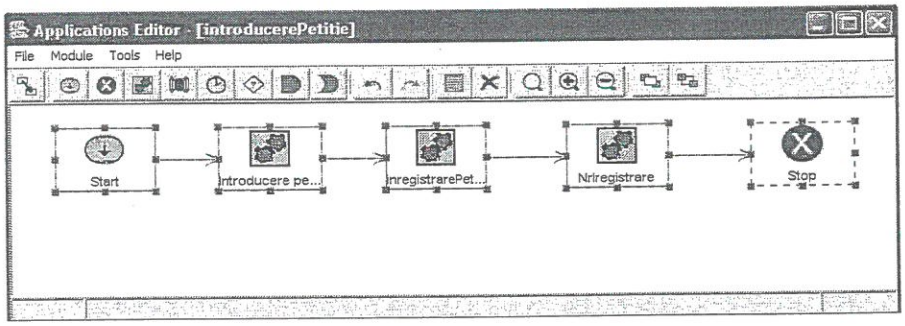

Figura 4. Modulul IntroducereSolicitare

- Primul block task:
  - Eticheta: Incarcă Solicitare
  - Role: Utilizator (taskul va fi rulat doar de cetățean)
  - Acțiuni:
    1) Create Binary - se crează şi se inițializează variablila binară Solicitare
- Al doilea block task:
  - Eticheta: Înregistreză Solicitare
  - Role: Registru (taskul va fi rulat doar de registru)
  - Acțiuni:
    1. View Binary --- se salvează pe calculatorul clientului variabila binară solicitare în scopul vizualizării acesteia
    2. Update Document --- se completează câmpurile variabilei document Registru (de către registratură)
    3. SaveDB --- se salvează în baza de date înregistrarea, în stadiul actual
    4. SaveBinary --- se salvează în baza de data solicitarea
    5. SendDocument (toLocalServer) --- se trimite documentul Registru în coada de aşteptare a serverului cu scopul de a porni, printr-un eveniment, modulul asignat secretariatului
- Al treilea block task:
  - Eticheta: NrInregistrare
  - Role: Utilizator
  - Acțiuni:
    1) View document --- se vizualizează documentul Registru, în scopul comunicării cetățeanului a numărul de înregistrare

Cel de-al doilea modul (figura 5):

- Nume: RezolvăSolicitare
- Tip: Event
- Role: Orice (nu contează deoarece modulul va fi lansat în execuție în mod automat)

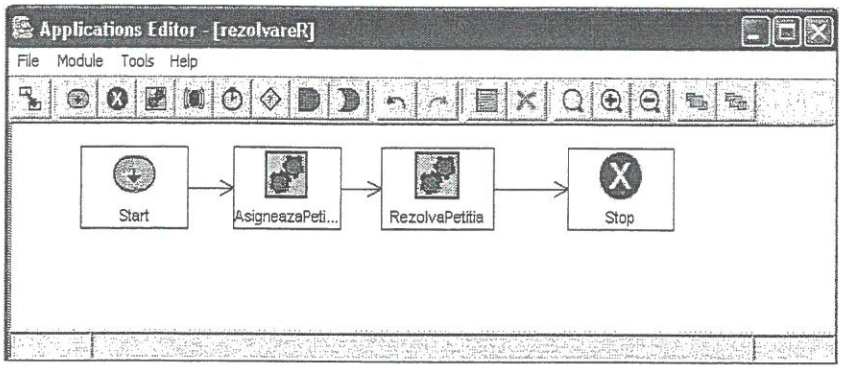

Figura 5. Modulul RezolvaSolicitare

- Block Start
- Primul block task:
  - Eticheta: Incarcă Solicitare
  - Role: Secretar
  - Acțiuni:
    1. LoadBinary --- se încarcă din baza de date solicitarea
    2. LoadDB --- se încarcă din baza de date documentul Registru
    3. ViewBinary --- se vizualizează solicitarea
    4. UpdateDocument --- secretarul completează câmpurile din documentul Registru
    5. SetVariabile --- se atribuie variabilei Rezolvare valoarea cuprinsă în câmpul Rezolvare din documentul Registru
    6. ReassignTask --- se asignează task-ul rolului din valuarea variabilei Rezolvare
    7. SaveDB --- se salvează câmpurile completate de secretar în baza de date
- Al doilea block task:
  - Eticheta: RezolvăSolicitare
  - Role: Orice (nu contează deoarece a fost reasignat în cadrul task-ului anterior)
  - Acțiuni:
    1. View Binary --- se salvează pe calculator variabila binară Solicitare în scopul vizualizării ei
    2. Update Document --- se completează câmpurile variabilei document Registru (necesare de completat de către compartimentul de rezolvare)
    3. SaveDB --- se salvează în baza de date documentul Registru; în acest moment el fiind completat în totalitate
- Block Stop

Cel de-al treilea modul de urmărire Solicitare va avea următoarea structură şi proprietăți (figura 6):

- Nume: UrmărireSolicitare
- Tip: Manual
- Role: Utilizator

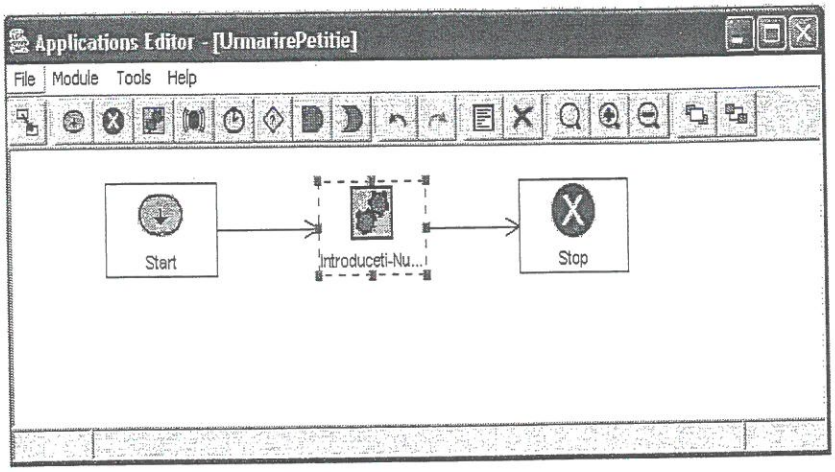

Figura 6. Modulul UrmarireSolicitare

- Block Start
- Block task:
  - Eticheta: UrmăreşteSolicitare
  - Role: Utilizator
  - Acţiuni:
    1. CreateDecimal --- se crează o variabilă de tip decimal în care utilizatorul va introduce numele de înregistrare
    2. LoadDB --- se încarcă din baza de date documentul Registru cu numărul de înregistrare specificat de utilizator
    3. ViewDocument --- se vizualizează documentul Registru
- Block Stop

4. Concluzii

Sistemul SIBPA permite proiectanţilor (dezvoltatorilor) de procese de business să-şi definească procesele de business, să execute procesele şi să monitorizeze execuţia acestora. Utilizarea sistemului SIBPA în domeniile în care legislaţia se schimbă frecvent duce la creşterea productivităţii întreţinerii programelor software.

Bibliografie

1. **BANERJI, A., C. BARTOLINI, D. BERINGER, V. CHOPELLA, K. GOVINDARAJAN, A. KARP, H. KUNO, M. LEMON, G. POGOSSIANTS, S. SHARMA, S. WILLIAMS**: Web Services Conversation Language (WSCL) 1.0. World Wide Web Consortium, 2002. W3C Note.
2. **CURBERA, F., Y. GOLAND, J. KLEIN, F. LEYMANN, D. ROLLER, S. THATTE, S. WEERAWARANA**: Business Process Execution Language for Web Services, Version 1.0. BEA, IBM, Microsoft, 2002.
3. **POPA V., STĂNESCU E., CONESCU D.** (2004): Sistem pentru Cooperarea Între Institutii, in Proceedings of Networking in Education and Research RoEduNet International Conference, 5-6 iunie Timişoara.
4. **STĂNESCU E., POPA V.** (2005): A System Architecture for using Contextual Information for the Mobile User in 4th International Conference RoEduNet, 20-22 mai Târgu Mureş.
5. **STĂNESCU E., POPA V.** (2005): Integrarea europeană bazată pe operabilitatea serviciilor de e_Goverment, al 10-lea simpozion cu participare internaţională de comunicări ştiinţifice ale cadrelor didactice, Facultatea Româno-Americană.
6. **MOISE M., POPA V., ILIESCU R.** (2005): WEB_Business, Mediu pentru dezvoltarea proceselor de busniess, al X-a simpozion al cadrelor didactice Universitatea Româno-Americană.
7. **MOISE M., POPA V., PALADE P.** (2000): Introducere în Programarea Logică, Editura Intercontinental.