Program PARTENERIATE - Proiecte colaborative de cercetare aplicativă - Competiţie 2013

RST - Raport ştiinţific şi tehnic

CALCULOS - Arhitectură cloud pentru o bibliotecă deschisă de blocuri funcţionale logice reutilizabile pentru sisteme optimizate

Codul proiectului: PN-II-PT-PCCA-2013-4-2123
Contract Nr.: 257/2014

Etapa I – Cerinţele utilizatorului, analiza acceptabilităţii şi platforma colaborativă

Cuprins

1. Introducere

Obiectivul principal al proiectului este proiectarea unei **platforme cloud şi a serviciilor associate**, platformă care va furniza resursele de prelucrare pentru accesarea şi rularea algoritmilor de control avansat şi optimizare a instalaţiilor industriale la scară mare. Aceste servicii vor permite utilizatorului să efectueze analize de risc online şi prevenirea pericolelor folosind algoritmi generici de control, optimizare, defectoscopie, diagnoză, prevenire a avariilor şi analiza defecţiunilor.

Obiectivele specifice ale proiectului sunt:

- Proiectarea unei platforme care să folosească o interfaţă prietenoasă de programare adresată mai multor tipuri de utilizatori: ingineri software; ingineri specializaţi în identificare parametrică şi modelare matematică; ingineri specializaţi în controlul proceselor (automatişti), care lucrează cu algoritmi complecşi, inclusiv algoritmi genetici şi reţele neuronale;
- Crearea unor maşini virtuale care să poată găzdui module/aplicaţii, algoritmi de simulare şi optimizare;
- Reducerea costurilor de mentenanţă pentru instalaţiile industriale;
- Îmbunătăţirea relaţiei între mediul academic şi cel industrial;
- Îmbunătaţirea proceselor şi instalaţiilor industriale româneşti prin metode şi servicii accesibile.

Proiectului își propune să abordeze integrat aspectele de optimizare şi securitate/siguranţă a funcţionării proceselor, aspecte tratate în mod separat mai înainte. De asemenea, pentru a veni în sprijinul inginerilor automatiști din industrie, se dorește utilizarea unor funcţii bloc standardizate pentru încapsularea de algoritmi şi strategii de control. Schemele actuale de control de proces sunt implementate prin echipamente hardware şi software dedicate, care adesea fac foarte dificilă sau chiar imposibilă implementarea îmbunătăţirilor sau adaptărilor. Mai mult, utilizarea puterii de calcul cloud pentru dezvoltarea, testarea şi validarea unor noi algoritmi şi strategii de control reprezintă o provocare majoră în domeniul proiectării şi analizei sistemelor de control, şi bătătoreşte calea pentru îmbunătăţirea continuă a calităţii, flexibilităţii, fiabilităţii şi eficienţei proiectelor şi implementărilor.

CALCULOS propune o platformă bazată pe cloud pentru servicii de proiectare şi validare ce vor asigura suport pentru o bază de date deschisă şi acces la soluţii algoritmice standardizate. Baza de date va consta în algoritmi, secvenţe şi strategii de control optimizate pentru instalaţii şi procese specifice, permitând o creştere substanţială a eficienţei şi fiabilităţii, reducând riscul de apariţie a funcţionării defectuoase şi facilitând evaluarea experimentală a diverselor scenarii. **Totuşi, identificarea unor soluţii potrivite reprezintă o altă provocare majoră**. În proiect va fi dezvoltată o soluţie, bazată pe căutări semantice şi potrivirea algoritmilor cu proprietăţile cerute, conducând la modificarea sistemului de control. Verificarea la nivelul sistemului va fi făcută în cloud de către modulul simulare-în-buclă, accesat de către dezvoltatori folosind interfaţa web. Proiectul tratează unele din cele mai importante provocări în domeniul sistemelor distribuite de control la scară largă şi se foloseşte de cele mai avansate tehnici din domeniile **ingineriei procesării în cloud, serviciilor bazate pe internet, funcţiilor bloc refolosibile standardizate**, **optimizării sistemelor complexe**, **controlului tolerant la erori şi analizei de pericole,** pentru a crea algoritmi noi, eficienţi şi fiabili care vor răspunde cerinţelor actuale de control din industrie. Este nevoie de elaborarea unui cadru capabil să ofere nu numai instrumente şi algoritmi, dar şi procedurile şi capabilităţile de calcul pentru a rula în timp real strategii complexe şi algoritmi sofisticaţi, inclusiv simulări, evaluări de risc şi optimizări. Proiectul va dezvolta o arhitectură capabilă să satisfacă aceste obiective.

Obiectivul etapei de execuţie. Conform planului de realizare a proiectului, în cadrul acestei prime etape de execuţie a proiectului, *Etapa I – Cerinţele utilizatorului, analiza acceptabilităţii şi platforma colaborativă*, au fost efectuate de către parteneri următoarele activităţi principale:

- Activitate I.1 (A1.1): identificarea cerinţelor utilizatorului şi analiza acceptabilităţii,
- Activitate I.2 (A1.2): investigarea fundamentelor teoretice şi a posibilităţilor de implementare,
- Activitate I.3 (A1.3): elaborarea unui cadru arhitectural standardizat comun,
- Activitate I.4 (A1.4): analiza posibilităţilor de interfaţare,

având ca rezultate un studiu privind stadiul industriei, cerinţele şi aşteptările sale, un raport de analiză ştiinţifică şi tehnică cu privire la posibilităţile de implementare a sistemului, elaborarea unei arhitecturi de control a proceselor şi, respectiv, analiza posibilităţilor de interfaţare cu utilizatorii şi cu procesul.

2. Cerinţele utilizatorului şi analiza acceptabilităţii

Situaţia pe plan mondial. La nivel mondial, evoluţia sistemelor de conducere este marcată de câteva tendinţe semnificative, dintre care menţionăm:

- Utilizarea unor standarde internaţionale atât pentru certificarea echipamentelor, a comunicaţiei sau a pachetelor de programe, cât şi pentru sisteme sau ansambluri precum bucle de reglare. Un exemplu este referitor la Sisteme cu Siguranţă Mărită (SIS − Safety Instrumented Systems).
- Diversificarea configuraţiei sistemelor, la sistemele DCS (Distributed Control Systems) şi PLC (Programmable Logic Controller) adăugându-se cele total distribuite (compatibile standardului Fieldbus Foundation), sau cele compacte PAC (Programmable Automation Controller).
- Creşterea gradului de integrare atât prin utilizarea de standarde de interfaţare, cât şi prin dezvoltarea de sisteme de conducere şi siguranţă mărită, bazate pe platforme de comunicaţie unice (magistrală internă).
- Extinderea utilizării protocoalelor de comunicaţie standardizate pentru dezvoltarea de noi produse (traductoare, elemente de execuţie, regulatoare), cât şi de noi sisteme (DCS, PLC, RTU, SCADA, MES, PI). Menţionăm standardul IEC 61850, a cărui utilizare a creat premizele trecerii de la reţele de utilităţi monitorizate de sisteme informaţionale la Smart Grid-uri.
- Consolidarea utilizării de standarde pentru algoritmi sau expresii matematice atât pentru automate programabile, cât şi pentru alte regulatoare sau programe de monitorizare şi control.
- Cresterea ponderii iniţiativelor şi produselor „open source” atât în sistemele informatice de gestiune, cât şi în cele de timp real. Un exemplu elocvent este iniţiativa OPC.

În ultimii 10 ani a fost introdusă în industria de proces o largă varietate de tehnologii şi echipamente de magistrale de câmp (fieldbus). A fost acceptat gradual faptul că sunt necesare variate tehnologii de comunicaţie pentru a răspunde la cerinţele diverselor aplicaţii. Totuşi, inginerii au nevoie de o interfaţă şi de funcţionalitate comune pentru a opera sistemele de control, independent de tehnologia utilizată sau de producătorul acestora. O evoluţie importantă o reprezintă standardul IEC 61804, care oferă utilizatorului final specificaţiile necesare pentru satisfacerea cerinţelor sistemelor de control distribuit bazate pe blocuri funcţionale. Specificaţiile definesc cerinţele pentru ca blocurile funcţionale să controleze şi să faciliteze operaţiile de mentenanţă şi de management tehnic, ca aplicaţii care interacţionează cu elemente de execuţie şi cu echipamente de măsură. Standardul include: modelul care defineşte componentele unui echipament compatibil cu IEC 61804; specificaţiile conceptuale ale blocurilor funcţionale de măsurare, acţionare şi prelucrare; tehnologia EDD (Electronic Device Description), care permite integrarea de detalii ale componentelor sistemului de automatizare prin utilizarea unei semantici similare instrumentelor de gestionare a ciclului de viaţă al unui sistem de control.

Inginerii automatişti din industrie preferă aplicaţii de control construite din componente bazate pe blocuri funcţionale (sau funcţii bloc). Blocurile funcţionale sunt încapsulări de variabile, parametri şi algoritmi de calcul, necesari în proiectarea procesului şi a sistemului său de control. Aplicaţia poate fi distribuită pe mai multe echipamente, conectate printr-o reţea sau o ierarhie de reţele de comunicaţie. În Fig. 1 sunt prezentate diverse blocuri funcţionale care se regăsesc în schema de control a unui proces industrial. Blocul tehnologic reprezintă procesul specific ataşat unui echipament. Blocul este compus din componente de achiziţie şi transformare prin care se efectuează măsurătorile sau se execută comenzile de acţionare. Blocul operaţional este specific aplicaţiei şi execută operaţii de prelucrare de semnal, cum ar fi: scalare, detectare alarme, control şi calcul. Blocul de funcţii elementare execută funcţii logice şi matematice. Blocul echipament reprezintă resursa care conţine informaţii despre echipament, despre sistemul de operare al acestuia şi despre unităţile hardware asociate.

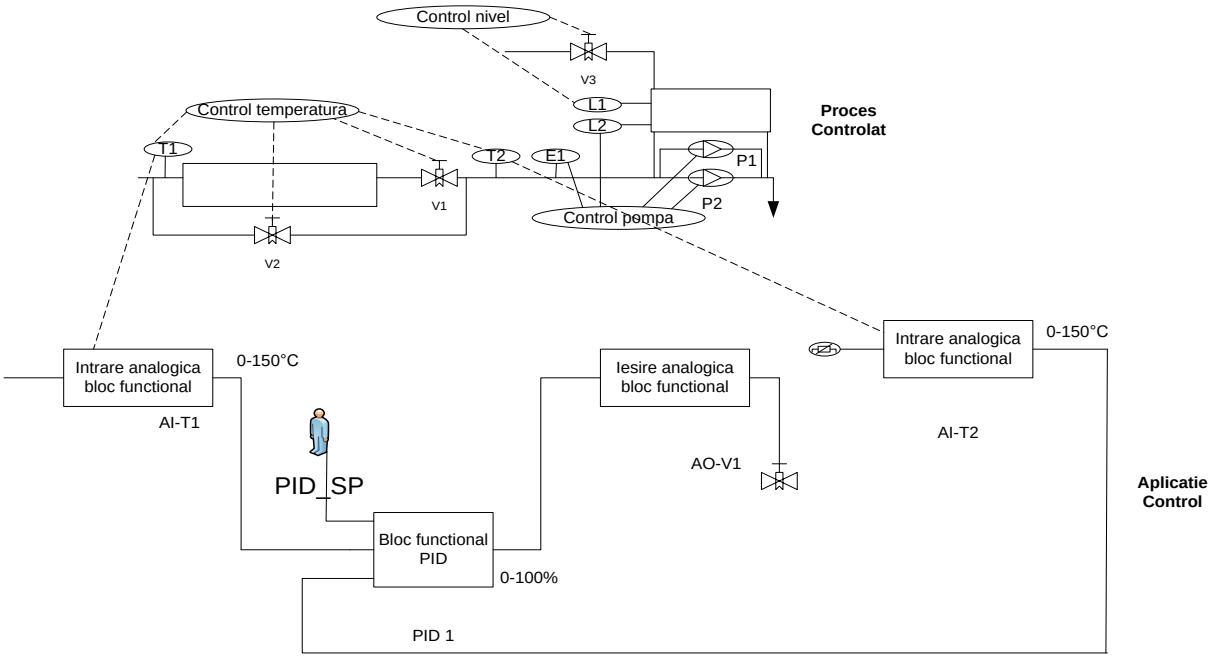

Fig. 1. Tipuri de blocuri funcţionale utilizate în conducerea proceselor

Stadiul industriei în ţară. Industria sistemelor de automatizare este structurată pe categorii de firme:

- Firme producătoare de aparatură şi echipamente. Firmele producătoare de aparatură de automatizare sunt în general firme de dimensiuni mici, desprinse din personalul fabricilor cu tradiţie, precum FEA, FEPA, ITRD, IAMC, AMPLO, şi care realizează în general senzori şi traductoare de complexitate medie. Dintre întreprinderile cu tradiţie, FEPA este singura care mai supravieţuieşte la un nivel scăzut de tehnologie şi cu o paletă redusă de produse. Firmele producătoare de echipamente au fost majoritatea relocate, reduse sau înlocuite cu firme private care au preluat personalul de calitate sau protofoliul de produse.

Marile firme internaţionale stabilite în ţară au ca principal obiectiv vânzarea produselor proprii în detrimentul producţiei interne. Se profilează o schimbare în acest sens în perioada următoare prin investiţii în mici unităţi de producţie.

- Integratori de sisteme. Integratorii de sisteme sunt singura categorie care a evoluat continuu acoperind în acest moment atât realizarea de aplicaţii „la cheie", cât şi realizarea de actualizări, modernizări sau mentenanţă. Cu toate că dispun de o putere limitată din punct de vedere financiar şi numeric, aceasta le permite să preia o mare categorie de aplicaţii şi sisteme mici şi chiar medii în detrimentul marilor corporaţii (precum Siemens, Honeywell, Yokogawa, Emerson, Schneider).

Cerinţele şi aşteptările clienţilor industriali. Sistemele de control bazate pe utilizarea Internetului, IBCS (Internet-Based Control System), devin din ce în ce mai populare datorită flexibilităţii şi accesibilităţii pe care le oferă. Noua generaţie de aplicaţii trebuie să se conformeze cerinţelor industriale de a susţine integrarea la nivel de companie, controlul la distanţă şi chiar implementarea de sisteme bazate pe Internet, care să permită execuţia distribuită şi cooperarea între diferite echipamente [1,2]. O altă direcţie de cercetare din domeniul sistemelor de control la distanţă de tip supervizor o reprezintă conectarea instalaţiilor la un centru dedicat de analiză şi diagnoză a riscului, capabil să ofere soluţii de gestionare a unor situaţii critice [3]. Aceasta necesită capabilităţi de execuţie la distanţă a unor algoritmi de analiză în scopul identificării situaţiilor critice, precum şi a unor algoritmi de management al riscului. Execuţia la distanţă permite unui controller dintr-o reţea distribuită să aibă acces la o putere de calcul şi o bază de cunoştinţe mai mari decât cele disponibile prin propriile resurse. În acest mod, algoritmii ce necesită un efort de calcul ridicat (din domenii precum modelarea proceselor, optimizare, control avansat bazat pe tehnici de inteligenţă artificială,

analiză a riscului, prelucrare de imagini etc.) pot fi stocaţi şi executaţi pe un echipament aflat la distanţă, folosind o conexiune de reţea pentru a accesa parametrii din proces şi a trimite rezultatele.

Se acordă o atenţie tot mai mare dezvoltării de servicii cloud dedicate, care să preia atribuţiile actualelor sisteme de conducere SCADA (Supervisory Control and Data Aquisition) şi să pună la dispoziţie funcţionalitatea acestora pentru a construi arhitecturi dinamice ce acţionează şi pe planul vertical, intregrând datele din proces cu sisteme precum MES (Manufacturing Execution System) şi/sau ERP (Enterprise Resource Planning) [4]. Aceste servicii permit relocarea resurselor de prelucrare în afara organizaţiei, oferind putere de calcul şi spaţiu de stocare superioare celor disponibile într-o instalaţie, capabile să asigure atât nivele mari de „uptime", precum şi redundanţa, siguranţa şi integritatea datelor **Error! Reference source not found.**. Toate acestea se aliniază direcţiei internaţionale de dezvoltare şi interconectare a tot mai multor echipamente, cunoscută şi sub numele de Internet of Things – IoT [6,7]. Această tendinţă are ca obiectiv adăugarea de module inteligente fiecărui echipament care să îi permită accesul la o gamă mai mare de informaţii şi servicii (management al activelor, planificarea resurselor, execuţie la distanţă, control autonom şi/sau distribuit etc.), precum şi distribuirea datelor către alte dispozitive din reţea. Principalele elemente ale IoT sunt reprezentate de senzorii încorporaţi (embedded), tehnologiile de recunoaştere a imaginilor, cu aplicaţii atât industriale cât şi pentru consumatori, şi extinderea tehnologiei de comunicaţie NFC (Near Field Communication) pentru aplicaţii de eficientizare a serviciilor, precum efectuarea plăţilor [8].

Din punctul de vedere al sistemelor de automatizare şi control, principalele tendinţe identificate pentru viitor sunt [9]:

- implementarea unor sisteme de control care să uniformizeze fluxul de operaţii şi informaţii la nivel de întreprindere;
- simplificarea arhitecturilor de control prin contopirea mai multor nivele, astfel ca informaţia să poată ajunge, de exemplu, direct de la nivelul 0 al echipamentelor de proces la nivele superioare precum MES sau ERP;
- creşterea performanţelor echipamentelor de proces;
- adoptarea unor standarde de comunicaţie bazate pe servicii web, precum OPC UA;
- definirea unor instrumente pentru analiza unor seturi mari de date;
- dezvoltarea de module software sau echipamente care să permită adăugarea de noi funcţionalităţi instalaţiilor existente fără a renunţa la vechile componente;
- extinderea capabilităţilor de monitorizare la distanţă prin dezvoltarea de aplicaţii mobile;
- dezvoltarea de soluţii avansate pentru securitatea informaţiei, de exemplu prin migrarea la IPv6;
- definirea unor strategii de proiectare integrate care să includă instrumente de modelare şi optimizare pentru obţinerea unor procese mai eficiente.

Standarde utilizate (IEC 61499, IEC 61131, IEC 61804, OPC etc.) Principalele piedici în reutilizarea unor secţiuni ale logicii de control dintr-o aplicaţie sunt lipsa unei standardizări a funcţiilor bloc şi/sau interdependenţa între echipamentul hardware şi modulele de program. Standardul IEC 61131 [10] a fost dezvoltat pentru rezolvarea acestei probleme, însă lipsa unor specificaţii referitoare la ordinea de execuţie a blocurilor şi diferenţele de reprezentare la nivel de limbaj maşină au făcut ca reutilizabilitatea să nu fie posibilă.

Standardul IEC 61499 [11] a fost creat pornind de la caracteristicile lui IEC 61131, pentru a susţine cerinţele de reutilizabilitate, reconfigurabilitate şi adaptabilitate utilizând o metodă de programare bazată pe funcţii bloc. Acest standard oferă o nouă abordare din punctul de vedere al reutilizabilităţii. O aplicaţie se construieşte ca un tot, implementarea fiind bazată pe funcţionalitate şi nu pe echipamente. Aplicaţia se poate diviza apoi în subaplicaţii, fiecare putând fi asociată unui anumit echipament, corespunzător acelei interfeţe. Adoptarea acestui standard la nivel industrial a început cu procesele de fabricaţie [12], însă există şi studii care au analizat posibilitatea de utilizare pentru aplicaţii de control al proceselor de tip batch [13-15] şi chiar pentru control în buclă închisă [16,17]. Pe măsură ce standardul câştigă tot mai mulţi adepţi, se doreşte lărgirea domeniului de aplicaţie prin dezvoltarea unor sisteme care să permită integrarea web a acestuia. Reglementarea din standard prin intermediul unor profiluri de conformitate duce la obţinerea unor instrumente software deschise din punctul de vedere al portabilităţii, configurabilităţii şi interoperabilitaţii,

asigurând astfel un suport ridicat pentru reutilizabilitatea funcţiilor bloc dintr-o aplicaţie [18]. De asemenea, trecerea de la controlul PLC centralizat la cel distribuit permite constituirea unui sistem capabil să se adapteze şi să fie reconfigurat în funcţie de instalaţie, care să poată fi utilizat în instalaţii distincte cu o gamă largă de regulatoare. Din punctul de vedere al arhitecturii [19], sisteme de control pot fi: sisteme cu control centralizat (ca majoritatea automatelor de tip PLC existente, ce utilizează standardul IEC 61131), sisteme cu control ierarhizat (unde apar modelele de intrări-ieşiri distribuite, întâlnite atât la DCS-uri, cât şi la PLC-uri) şi sisteme cu control distribuit (unde funcţiile de control sunt distribuite între echipamentele care formează aplicaţia şi care lucrează împreună pentru îndeplinirea cerinţelor). Cel mai performant sistem distribuit se bazează pe reglementările tehnologiei Fieldbus Foundation şi este promotorul standardului IEC 61104. Standardul IEC 61499 aduce noutatea distribuirii unei aplicaţii software între mai multe echipamente care pot acţiona independent pentru a-şi îndeplini funcţiile [11]. Se pot implementa astfel arhitecturi bazate pe agenţi inteligenţi, capabili să lucreze împreună în sisteme distribuite sau să se reconfigureze automat pentru asigurarea unor arhitecturi deschise, modulare, scalabile şi tolerante la defecte. O analiză asupra diferenţelor de abordare şi a performanţelor obţinute în proiectarea unor sisteme distribuite bazate pe IEC 61131 şi respectiv pe IEC 61499 a fost făcută în [20], unde se arată că pentru majoritatea criteriilor se obţin rezultate mai bune prin utilizarea standardului IEC 61499, avantajele lui IEC 61131 bazându-se în special pe gama mai mare de echipamente disponibile şi pe experienţa inginerilor de proces, care nu necesită astfel instruire suplimentară.

OPC UA este cea mai recentă specificaţie OLE pentru Controlul Proceselor (OPC) de la OPC Foundation şi diferă semnificativ de predecesorii săi prin faptul că include capabilităţi ridicate de asigurare a securităţii şi fiabilităţii datelor, permite integrarea modelelor de date, precum date istorice, alarme sau programe şi nu mai este dependentă de utilizarea DCOM din Windows, ceea ce oferă o mai mare flexibilitate. Open Platform Communications Unified Architecture sau OPC UA este un standard de comunicaţie ce conţine un set de documente care oferă reguli şi informaţii despre modul cum aplicaţiile software şi echipamentele pot trimite şi primi diferite tipuri de date. Scopul standardului este de a oferi o cale prin care aplicaţiile software să comunice cu diferite tipuri de automate programabile şi echipamente de proces fără a fi necesară o implementare a unui sistem software dedicat, comercial, prin utilizarea unei platforme independente şi sigure. Arhitectura OPC este o arhitectură tipică client-server. Utilizarea stivelor client şi server, dar şi a interfeţei API asigură flexibilitatea şi posibilitatea de a unifica diverse tipuri de servere, permiţând integrarea sistemelor de control cu alte aplicaţii şi făcând posibilă migrarea de la aplicaţiile curente la cele bazate pe OPC UA.

3. Fundamentele teoretice şi posibilităţile de implementare

Automatica este o ştiinţă formalizată bazată pe diferite domenii ale matematicii. Datorită limitărilor de spaţiu, nu este posibilă includerea în acest capitol decât a unei scurte treceri în revistă a problematicii algoritmice relevante pentru proiectul CALCULOS. O categorie de algoritmi o constituie cei de identificare experimentală [21,22 şi referinţele citate]. O altă categorie o formează algoritmii de optimizare, în special, cei de optimizare liniar-pătratică [23-25]. Evoluţiile recente includ exploatarea structurii problemelor de calcul optimale (implicând, de pildă, matrice Hamiltoniene sau simplectice, sau fascicole de matrice anti-Hamiltoniene/Hamiltoniene) [26]. De mare interes sunt sistemele tolerante la defecte şi sistemele optimale reconfigurabile [27-29], pentru care sunt adecvaţi algoritmi iterativi, de pildă, de tip Newton [30,31]. Multe detalii se găsesc în referinţele citate. In continuare, ne vom referi la abordarea de implementare a soluţiilor folosind sisteme de calcul moderne, de mare performanţă, bazate pe Cloud Computing.

Pentru anii următori, se prevede o creştere semnificativă a capacităţii de calcul în cloud [8]. Există mai multe tipuri de servicii disponibile pentru utilizare în cloud: Network as a service, Storage as a service, Data as a service, Database as a service, Test environment as a service, Desktop virtualization, API as a service, Backend as a service, servicii Comune etc., dar cele mai frecvent folosite sunt serviciile SPI [32]:

- Software as a service (SaaS)
- Platform as a service (PaaS)
- Infrastructure as a service (IaaS)

IaaS reprezintă modelul fundamental în care o companie îşi pune la dispoziţie partea de hardware ce asigură suport pentru stocare, comunicaţie şi capacitatea de calcul, pentru a-şi integra produsele în arhitecturile cloud. PaaS este permis de către modelul IaaS şi oferă o arhitectură de tip cloud ce asigură şi sistemele de operare şi platformele software pe care rulează aplicaţiile utilizatorilor. Modelul SaaS oferă un nivel mai ridicat de abstractizare pe PaaS şi se bazează pe furnizarea unor servicii software specifice. Unul dintre avantajele procesării în cloud este posibilitatea de a înlocui cheltuielile mari cu infrastructura cu costuri reduse şi variabile, ce se dimensionează după tipul afacerii. Un alt avantaj important îl reprezintă mobilitatea lor, limitată numai de reţeaua de comunicaţie. S-au format şi dezvoltat câteva companii care profită de pe urma acestor oportunităţi, concurând la implementarea arhitecturilor de tip cloud şi a serviciilor SPI: Google, Amazon, VMWare, Citrix Systems, Microsoft, Rackspace, Salesforce, Verizon. Proiectul va include platforma colaborativă pentru a utiliza procesarea cloud a algoritmilor.

Soluţii comerciale şi open-source pentru utilizarea serviciilor cloud de tip Platform-as-a-Service. Soluţiile cloud SaaS sunt utilizate în mod obişnuit, deoarece oferă servicii bine stabilite, cum ar fi e-mail, mesagerie instant, calendar comun şi gestiunea documentelor (incluse, de exemplu, în Google Apps şi Microsoft Office 365). Modelul SaaS este excelent pentru astfel de servicii, dar nu este suficient de flexibil şi uşor de personalizat. Modelul IaaS oferă accesul de la distanţă (prin Internet) la infrastructuri de calcul, de obicei, sub formă de sisteme virtuale (maşini virtuale sau VM-uri). Utilizatorii IaaS trebuie să administreze întreaga stivă software pentru a rula propriile aplicaţii. Astfel, modelul IaaS este potrivit pentru acele proiecte care necesită o configurare specială sau pentru a construi infrastructuri cloud de tip PaaS utilizând suportul IaaS. Soluţiile IaaS necesită, în plus, efectuarea unor operaţiuni regulate de întreţinere, cum ar fi aplicarea actualizărilor de securitate pentru sistemul de operare de bază şi mediul de execuţie a aplicaţiilor. Aplicaţia utilizator poate deveni vulnerabilă atunci când această întreţinere regulată nu este realizată corect. Modelul PaaS reprezintă un compromis avantajos între nivelurile IaaS şi SaaS, oferind platforma de execuţie şi serviciile necesare (de exemplu, bază de date sau server de aplicaţii) pentru aplicaţii existente, menţinând totodată sistemul de operare şi mediul de execuţie. Soluţiile PaaS sprijină capacitatea de scalare verticală prin creşterea sau scăderea resurselor (RAM, CPU, etc), care sunt acordate către o singură instanţă. Scalarea verticală poate fi asigurată şi de nivelul IaaS de bază. Mai multe platforme PaaS comerciale oferă capacităţi de scalare orizontale prin adăugarea sau eliminarea de instanţe suplimentare pentru o aplicaţie distribuită.

În cadrul proiectului au fost investigate cele mai importante soluţii pentru servicii PaaS, anume platformele Google App Engine, OpenShift, Cloud Foundry şi Heroku, fiind analizate caracteristicile şi costurile de utilizare ale acestora, inclusiv unele probleme de portare a aplicaţiilor, restricţii, politici de stabilire a preţurilor şi procesul de instalare locală.

Google App Engine. Lansată în 2008, Google App Engine (GAE) oferă un mediu de rulare pentru aplicaţii web în centrele de date administrate de Google. GAE acceptă limbajele Java, Python (ambele cu unele restricţii), PHP (suportat nativ) şi Go (experimental). GAE rulează în prezent aplicaţii Java folosind containerul web Jetty. Sunt prevăzute mai multe metode de stocare a datelor. Cloud Datastore este un depozit de date NoSQL complet scalabil, bazat pe tehnologia Bigtable. Google Cloud Storage este un sistem de stocare on-line de obiecte, de obicei fişiere, de tip RESTful. Aceste obiecte sunt organizate în containere şi permisiunile pot fi definite în listele de control al accesului (ACL-uri). Cloud SQL este o bază de date relaţională, similară MySQL, cu suport de replicare a datelor. Există restricţii semnificative impuse de mediul de execuţie GAE, rezonabile în mediul cloud, dar care pot complica portarea aplicaţiilor existente la GAE. Este disponibil numai un subset din clasele standard JRE, aplicaţiile care folosesc alte clase trebuind rescrise şi recompilate. Aplicaţiile nu pot utiliza sistemul de fişiere local, ci doar tehnologii de stocare specifice GAE. Google are o listă de preţuri cu o granulaţie foarte fină în care resursele specifice sunt facturate per gigabyte, pe oră, pe gigabytes pe lună, etc., în funcţie de natura resurselor (trafic în reţea, performanţă maşină virtuală, stocare date, etc.). Există şi o cotă maximă de resurse alocate gratuit.

Platforma open-source AppScale, creată la Universitatea din California, permite dezvoltatorilor să implementeze şi să ruleze aplicaţii construite pentru GAE oriunde în afara ecosistemului Google, de pildă,

pe propriile servere (virtuale), pe maşini virtuale oferite de furnizori terţi, cum ar fi Amazon, sau pe maşini virtuale gestionate de tehnologii deschise, cum ar fi OpenStack sau Eucalyptus. AppScale foloseşte unele părţi ale GAE disponibile în Google App Engine Software Development Kit (SDK), oferit de Google, care permite dezvoltatorilor să testeze şi să ruleze aplicaţiile GAE la nivel local, fără a fi necesară implementarea lor efectivă în cloud. De asemenea, AppScale utilizează şi alte proiecte open-source existente, de exemplu Cassandra (bază de date NoSQL), MySQL (bază de date relaţională), MongoDB, Hbase şi altele.

OpenShift. OpenShift este un sistem PaaS dezvoltat de Red Hat, lansat în 2011, sub forma a trei ediţii: OpenShift Origin − varianta open-source, compatibilă cu celelalte ediţii, OpenShift Online − serviciul cloud public comercial şi OpenShift Enterprise − implementat în centrul de date al companiei sau într-un cloud privat. OpenShift acceptă mai multe limbaje de programare, baze de date, servicii şi aplicaţii, care sunt capabile să ruleze pe sistemul Red Hat Linux. Suportul pentru o anumită tehnologie este întotdeauna încapsulat sub forma unei componente denumite cartuş (cartridge). Mai multe cartuşe sunt disponibile pentru Java Runtime (JBoss, Tomcat), mediul de execuţie PHP (Apache cu mod_php), baze de date relaţionale (MySQL, PostgreSQL), baze de date NoSQL (MongoDB), servicii (Cron, Jenkins), etc. Aplicaţiile existente pot fi, de asemenea, ambalate în cartuşe pregătite să permită unor non-dezvoltatori să ruleze aplicaţiile favorite în cadrul platformei (sunt necesari doar câţiva paşi de configurare).

Se pot crea, de asemenea, propriile cartuşe personalizate, deşi multe cartuşe suplimentare au fost deja implementate de către comunitatea de utilizatori OpenShift. Spre deosebire de platformele strâns cuplate cu anumite tehnologii (cum ar fi, de pildă, Google App Engine), OpenShift este deschis către orice tehnologie. Pentru a satisface aceste facilităţi de permisivitate este strict necesară aplicarea unei politici de securitate foarte puternice. De aceea, cartuşele sunt implementate în aşa-numita componentă Gear, un mediu izolat, construit pe baza mai multor directoare, ce utilizează tehnologia cgroups Linux (de punere în aplicare a limitării utilizării resurselor) şi politicile SELinux (consolidarea securităţii şi izolare). Deoarece OpenShift Gear este de fapt un sistem GNU/Linux containerizat, restricţiile impuse sunt minimale. Pe de altă parte, din cauza acestei arhitecturi simple, nu există soluţii „avansate" pentru scalarea orizontală a sistemului de stocare de fişiere şi/sau a bazelor de date relaţionale. Chiar dacă este permis să scrie în sistemul local de fişiere, se garantează persistenţa modificărilor în decursul ciclului de viaţă al unei aplicaţii doar pentru directorul OPENSHIFT_DATA_DIR, fapt ce poate fi considerat drept o limitare. Codul sursă al aplicaţiei este întotdeauna obţinut dintr-un repository al sistemului de gestiune al codului sursă Git. Spre deosebire de Google App Engine, OpenShift nu are un suport direct pentru scalarea/replicarea MySQL. Aplicaţiile scalabile trebuie să utilizeze o bază de date comună, implementată într-un alt container, suficient de puternic pentru a evita o limitare a performanţei.

Cloud Foundry este o platformă cloud open-source dezvoltată de VMware şi lansată în anul 2011. La început, Cloud Foundry a oferit suport pentru limbajele Java, Scala, Ruby şi Node.js. În ultima versiune a platformei a fost introdus un nou element care oferă suport pentru execuţia oricărei aplicaţii. Această caracteristică se bazează pe tehnologia Heroku Buildpacks care automatizează împachetarea unei aplicaţii împreună cu mediul său de execuţie. Există două tipuri de servicii. Serviciile pentru utilizator permit unui dezvoltator de aplicaţii să furnizeze un serviciu extern existent (de exemplu, o bază de date Oracle) către o aplicaţie. Serviciile de administrare sunt servicii gestionate şi implementate de platforma Cloud Foundry. Sunt suportate implicit unele baze de date relaţionale (MySQL, PostgreSQL), baze de date NoSQL (MongoDB) sau alte servicii (Memcached). Serviciul MySQL cu disponibilitate ridicată este oferit de un serviciu terţ numit ClearDB. Cloud Foundry este o platformă deschisă care permite execuţia aplicaţiilor fără constrângeri speciale, ceea ce face ca politicile de securitate puse în aplicare de către platformă să fie foarte stricte. Se foloseşte o componentă specială denumită Warden, respectiv un nivel de abstractizare care gestionează resursele disponibile în containere izolate şi limitează utilizarea acestor resurse de aplicaţiile care sunt executate în cadrul platformei. Deşi este similar cu tehnologia Linux Containers (LXC), sistemul Warden este proiectat special pentru a fi multi-platformă. Cu toate acestea, implementarea de referinţă suportă numai sistemele GNU / Linux care utilizează tehnologia cgroups din kernelul Linux.

Interoperabilitatea infrastructurilor cloud. Tehnologia cloud computing reprezintă o nouă paradigmă de calcul prin care se oferă posibilitatea de a accesa la cerere anumite resurse (spre exemplu, CPU, spaţiu de stocare, reţea, baze de date, aplicaţii) şi pentru care se plăteşte doar costul utilizării efective. Prin

provizionarea rapidă şi eliberarea resurselor cu un minim de efort din partea utilizatorului şi de interacţiune cu furnizorul de servicii cloud, această tehnologie facilitează implementarea unor soluţii de calcul scalabile şi eficiente. Modelul cloud computing are următoarele caracteristici esenţiale:

- disponibil la cerere;
- bazat pe auto-servirea de către utilizator;
- uşor accesibil prin intermediul Internet-ului;
- partajarea resurselor de calcul;
- elasticitate şi contabilizarea utilizării serviciilor.

Tehnologia cloud este extrem de utilă în diverse aplicaţii, inclusiv cele care sunt mari consumatoare de resurse de calcul sau de spaţiu de stocare, ori cele care necesită o lărgime de bandă foarte mare [33]. Furnizorii de servicii cloud au proliferat rapid în ultimii 6-7 ani, cei mai importanţi fiind Amazon, Google, Microsoft şi Salesforce, fiecare promovându-şi propria infrastructură cloud, precum şi standarde şi formate incompatibile pentru accesarea resurselor cloud. Totuşi, clienţii cloud sunt de obicei reticenţi în a fi legaţi de serviciile cloud oferite de un singur furnizor, care nu ar fi în măsură să satisfacă toate cerinţele lor, existând riscul potenţial de creştere exagerată a costurilor ca urmare a acelei dependenţe. Utilizatorii doresc infrastructuri cloud interoperabile, în care să poată avea un control deplin asupra modului de instalare a aplicaţiilor şi să poată să migreze cu uşurinţă atunci când este nevoie, fără investiţii de dezvoltare suplimentare. Au apărut iniţiative pentru stabilirea unor standarde pentru federalizarea infrastructurilor cloud deţinute de diferiţi furnizori, în special susţinute de către furnizorii de servicii cloud relativ mici (de exemplu, Rackspace, GoGrid), dar şi de către cei nou intraţi pe piaţă (de exemplu, Red Hat, Dell, Oracle, etc.). Interoperabilitatea asigură mai bine îndeplinirea scopului final al paradigmei cloud computing, acela de furnizare la scară globală, „nelimitată”, cu acces prin interfeţe unificate, a resurselor de calcul. Industria încearcă să abordeze problemele de interoperabilitate cloud prin standardizare. O altă soluţie pe termen scurt a fost dezvoltarea de soluţii tehnologice care să permită interoperarea între anumite infrastructuri cloud, eficiente atât pentru furnizorul de servicii, cât şi pentru utilizator.

Utilizatorii cloud pot alege între diferitele tipuri de servicii cloud, în funcţie de propriile nevoi de portabilitate şi automatizare. De exemplu, în cadrul PaaS se oferă posibilitatea de configurare mai rapidă a aplicaţiilor decât în cadrul IaaS, iar utilizatorii pot exploata oportunităţile de găzduire gratuită oferite de unii furnizori de soluţii PaaS. Serviciile de tip PaaS şi SaaS oferite de către un furnizor cloud sunt de obicei implementate pe baza unei infrastructuri de tip IaaS, ceea ce face ca interoperabilitatea nivelului IaaS să fie mult mai importantă decât a celorlaltor două niveluri. Interoperabilitatea PaaS depinde în mare parte de mediile de dezvoltare (de exemplu, Django, Ruby pe Rail, PHP, etc.), ce pot fi mai bine susţinute la nevoie de către comunităţile proprii de dezvoltatori. Nivelul SaaS oferă un cadru complet pentru un serviciu utilizator, astfel că principala problemă de interoperabilitate este în mare parte legată de accesul la date.

Implementările reprezentative, la nivel de referinţă, pentru sistemele de management cloud de tip IaaS sunt OpenStack, OpenNebula şi Eucalyptus. Aproape toate astfel de implementări încearcă să asigure interoperabilitatea cu platforma dominantă, respectiv Amazon EC2.

OpenStack, proiect open-source administrat de către fundaţia omonimă, este o platformă de management cloud de tip IaaS bazată pe o suită de servicii [33]:

- Controlerul sistemului de calcul, numit Nova, oferă servere virtuale la cerere. Arhitectura Nova poate fi scalată orizontal utilizând echipamente standard, fără cerinţe hardware sau software proprietare, şi are capacitatea de a se integra cu sistemele existente, precum şi cu alte sisteme de tip IaaS. Nova poate să utilizeze hipervizoarele Xen, KVM, LXC şi chiar Microsoft Hyper-V şi suportă diferite arhitecturi (de exemplu, x86, amd64 şi ARM), asigurând interoperabilitatea.
- Sistemele de stocare, Swift şi Cinder, care asigură stocarea la nivel de obiect şi respectiv dispozitiv de tip bloc, sunt concepute pentru a fi interoperabile cu cele similare de la Amazon S3 şi EBS, în special Swift, care implementează un subset al API-ului S3 cu ajutorul middleware-ului WSGI.
- Serviciul de stocare imagini, Glance, oferă un catalog de imagini ale discurilor virtuale şi include multe caracteristici ale serviciului Amazon AMI (Amazon Machine Imagine). Aceste imagini sunt utilizate în mod obişnuit de către modulul OpenStack Compute. Deşi AMI-urile de la Amazon nu

pot fi încă utilizate direct de către serviciul Glance, elementele din API comune permit unor clienţi terţi să automatizeze managementul imaginilor în cadrul OpenStack.

Sistemul OpenStack pentru reţelistică, Quantum, gestionează configurarea reţelelor şi a adresele IP în cloud. Serviciul Quantum facilitează capacitatea de reţea definită de software în cadrul OpenStack, ce reprezintă o soluţie eficientă pentru implementarea nivelului de reţea la nivel WAN.

Eucalyptus, creat de Eucalyptus Systems, Inc, este o platformă software care creează infrastructuri cloud private şi hibride scalabile în cadrul unei infrastructuri IT existente. Acesta oferă un API care reproduce cu mare fidelitate API-ului Amazon AWS (Amazon Web Services), cu scopul de a interopera cu cloud-ul AWS. Principalele componente ale platformei Eucalyptus sunt [33]:

- Cloud Controller, care oferă o funcţionalitate compatibilă cu Amazon EC2;
- Walrus, un sistem de stocare care asigură o funcţionalitate compatibilă Amazon S3;
- Cluster Controler, care gestionează un cluster în cloud;
- Storage Controler, care asigură o funcţionalitate compatibilă Amazon EBS;
- Node Controller, care controlează instanţele maşinilor virtuale.

Platforma oferă flexibilitate în alegerea formatului maşinii virtuale utilizate (se pot executa atât imagini cu sisteme Windows cât şi Linux) şi se poate construi o bibliotecă de imagini ale maşinilor virtuale compatibilă cu formatul AMI. De asemenea, imaginile VMware şi vApps pot fi uşor modificate pentru a rula pe Eucalyptus. Utilizatorii cloud au libertatea de a folosi diverse hipervizoare, vSphere, ESX, KVM şi Xen. Este posibil managementul integrat al infrastructurilor cloud hibride Eucalytus şi AWS.

OpenNebula este o platformă open-source ce oferă soluţii flexibile pentru managementul complet al centrelor de date virtualizate, şi permite crearea de infrastructuri cloud private. Diferitele interfeţe oferite pot fi folosite pentru a interacţiona şi gestiona resursele fizice şi virtuale. Principalele moduri de interacţionare cu OpenNebula sunt [33]:

- Interfeţe cloud, ce includ API-uri ca OCCI, EC2 şi interfaţa EBS, şi un portal self-service;
- Interfeţe de administrare, utilizând linia de comandă şi o interfaţă grafică flexibilă, Sunstone;
- Un API extensibil pentru diferite limbaje de programare precum Ruby, Java;
- Un catalog de aplicaţii care sunt gata de utilizat în cadrul unei instanţe OpenNebula.

În ceea ce priveşte interoperabilitatea cu alte infrastructuri cloud, OpenNebula implementează API-ul EC2 Query, care permite integrarea cu infrastructura cloud Amazon EC2, dar şi o interfaţă bazată pe standardul OCCI, care este deschis şi public. Utilizarea acestora înlesneşte implementarea de infrastructuri cloud de mari dimensiuni, deschise către un public larg. Administrarea unei astfel de infrastructuri cloud reprezintă o provocare semnificativă din punct de vedere al interoperabilităţii între resurse. Din acest motiv, OpenNebula implementează două tehnologii specifice, respectiv oZones şi VDCs (centre de date virtuale).

Virtualizarea sistemului de operare prin intermediul containere-lor Docker. Docker este un instrument open-source care automatizează lansarea aplicaţiilor în cadrul containerelor software prin introducerea unui nivel de abstractizare la nivelul virtualizării sistemului de operare. Docker utilizează facilităţi de izolare a resurselor ce sunt prezente în cadrul nucleului (kernel) Linux, precum cgroups şi namespaces, ce permit unor containere independente să fie executate în cadrul unei singure instanţe a unui sistem de operare Linux. Aceasta reduce consumul de resurse şi creşte performanţa în comparaţie cu utilizarea maşinilor virtuale. Componenta oferă posibilitatea ca fiecare aplicaţie să aibă propria imagine asupra mediului în cadrul sistemului de operare, inclusiv a listei de procese, a id-urilor utilizatorilor, a stivei de protocoale TCP/IP sau a sistemelor de fişiere montate, în timp ce componenta cgroups oferă izolarea resurselor computaţionale precum CPU, memorie, subsistem I/O ce include accesul la disc şi reţea. Docker

include biblioteca libcontainer ca implementare de referinţă pentru utilizarea containerelor. Aceasta este construită pe baza componentelor libvirt, LXC şi systemd-nspawn ce furnizează interfeţe pentru facilităţile oferite de către nucleul Linux [34]. Prin utilizarea containerelor, resursele sunt izolate, serviciile pot fi restricţionate, iar procesele pot fi create cu o vedere proprie asupra sistemului de operare. Acesta include propriul spaţiu de procese, propria structură de fişiere sau de interfeţe de reţea. Mai multe containere pot să acceseze acelaşi kernel, dar fiecare container poate fi izolat şi constrâns să utilizeze o cotă de resurse.

Prin utilizarea Docker pentru crearea şi administrarea containerelor, activitatea de implementare a sistemelor distribuite devine mult mai simplu de realizat. Astfel, este posibil ca mai multe aplicaţii şi procese care au sarcina de a executa anumite activităţi să fie rulate în mod independent în cadrul unei maşini fizice sau a unui grup de maşini virtuale. Aceasta permite lansarea în execuţie a sistemelor în funcţie de disponibilitatea resurselor sau a necesarului de putere de calcul. Prin acest model de execuţie se pot implementa infrastructuri de calcul cloud de tip PaaS şi se pot scala aplicaţii precum Apache Cassandra, MongoDB sau Riak. De asemenea, se simplifică crearea şi operarea cozilor de execuţie pentru job-uri sau a altor sisteme distribuite. Docker poate fi integrat în cadrul mai multor instrumente de administrare a infrastructurilor de calcul, inclusiv în cadrul unor platforme cloud publice, de exemplu, Amazon Web Services (AWS), Microsoft Azure, OpenStack Nova, Vagrant, Puppet, Chef, CFEngine, Salt, Ansible, Jenkins, ş.a [Wikipedia, Docker].

LXC reprezintă o facilitate de virtualizare a sistemului de operare prin care se pot crea medii de execuţie izolate şi independente fără a fi necesară utilizarea maşinilor virtuale, ce implică un anumit grad de penalizare în ceea ce priveşte performanţa [35]. Docker extinde tehnologia LXC, făcând să devină mult mai accesibilă pentru utilizatori. Costul utilizării Docker în termeni de consum al resurselor este mult mai mic decât în cazul utilizării maşinilor virtuale, deoarece nu emulează sistemul de operare complet, ci doar bibliotecile şi fişierele binare ale aplicaţiilor virtualizate. Un container Docker se execută în cadrul unui mediu complet virtualizat, astfel încât un algoritm poate fi implementat în orice limbaj care este suportat de sistemul Linux (R, Python, MATLAB, C, etc.) şi poate fi fi compilat utilizând orice bibliotecă compatibilă cu Linux, fără a cauza conflicte din cauza versiunilor diferite utilizate de către alte aplicaţii. În plus, un container Docker poate fi exportat, arhivat şi executat pe un mediu diferit de cel folosit la implementare.

În afară de facilităţile deja menţionate, respectiv furnizarea de medii computaţionale complete şi izolate, Docker mai oferă şi alte funcţionalităţi care îl fac să fie extrem de atractiv pentru implementarea aplicaţiilor ştiinţifice. În [36] se exemplifică câteva din beneficiile tehnologiei Docker, dintre care menţionăm:

- Utilizarea imaginilor Docker rezolvă problema dependenţelor între diverse versiuni de aplicaţii şi biblioteci. Există o singură imagine binară, bazată pe sistemul de operare Linux, în care toate componentele software necesare sunt pre-instalate, configurate şi testate.
- Portabilitatea şi partajarea aplicaţiilor prin gestionarea distribuirii şi execuţiei unui container: acesta rulează în acelaşi mediu indiferent de sistemul gazdă şi expune acelaşi set de interfeţe de reţea sau de stocare. Partajarea imaginilor este facilitată de către infrastructura Docker Hub.
- Utilizarea versiunilor pentru containere, permţând restaurarea simplă a unei versiuni anterioare şi încărcarea/descărcarea incrementală a imaginilor folosind diferenţele între două imagini.
- Existenţa unei liste a celor mai bune practici pentru utilizarea containerelor Docker.

Docker are un potenţial important să devină o tehnologie standard, nu numai în industrie, dar şi în mediul ştiinţific şi de cercetare. Cu toate acestea, există mai multe limitări, de pildă: nu oferă o soluţie completă de virtualizare; este limitat la sisteme gazdă cu arhitectură de tip 64 bit; trebuie executat în cadrul unei maşini

virtuale Linux pe sistemele Mac şi Windows; problemele potenţiale de securitate trebuie încă evaluate.

4. Elaborarea unui cadru arhitectural standardizat comun

Arhitectura sistemului. Progresul eficient şi sustenabil, atât în domeniul controlului proceselor, cât şi al activităţii de cercetare-dezvoltare, poate fi asigurat prin utilizarea unor mijloace adecvate în dezvoltarea de algoritmi, integrând reacţiile de la implementările în industrie, formulând problemele legate direct de cazuri reale şi găsind rezolvări la aceste probleme. Cele mai eficiente sisteme automate dedicate pentru controlul proceselor industriale sunt în prezent caracterizate printr-o structură ierarhică presupunând existenţa a cel puţin două niveluri de automatizare: un nivel executiv, responsabil de controlul clasic al parametrilor principali de process; şi un nivel de supraveghere, responsabil de monitorizarea instalaţiilor şi de luare a deciziilor. Cele mai multe dintre mediile industriale necesită un sistem cu un nivel de control superior, capabil să îmbunătăţească şi să optimizeze funcţionarea instalaţiei.

Arhitectura sistemului este prezentată în Fig. 2, în care sunt evidenţiate cele trei componente principale ale acestuia: modulul de dezvoltare şi testare, biblioteca online şi subsistemul de execuţie online. Sistemul va funcţiona ca o aplicaţie web standard având o bază de date relaţională, care va permite accesul la fişierele cu diverşi algoritmi, prin organizarea acestora în funcţie de tipul algoritmilor şi de procesele cărora li se pretează. La această aplicaţie web se adaugă un modul responsabil cu implementarea şi testarea algoritmilor. Acest modul poate folosi FBDK pentru elaborarea algoritmilor, pornind de la un fişier descriptor ce conţine diagrame de execuţie, formule, diagrame de stare etc. Pentru fiecare algoritm se va adăuga o descriere care să definească concret domeniul de utilizare, să detalieze performanţele şi limitările. Dacă este cazul, se va specifica dacă un anumit algoritm foloseşte alte funcţii din bibliotecă, dacă au existat implementări ale acestuia în mediul industrial, caracteristicile procesului şi rezultatele obţinute.

Biblioteca permite două moduri de utilizare a componentelor sale: descărcare pentru a putea fi integrate într-un controller de proces (implementare offline) sau execuţie directă în bibliotecă (execuţie online). În cazul utilizării offline, utilizatorului i se pune la dispoziţie un fişier ce conţine o funcţie bloc simplă sau compusă, ce implementează funcţionalitatea dorită. În cazul utilizării online, reprezentarea va fi sub forma unei configuraţii de sistem care să permită rularea şi conectarea la o aplicaţie existentă. Pentru aceasta vor fi dezvoltate funcţii bloc speciale necesare pentru interfaţarea cu procesul. Biblioteca online va fi alcătuită dintr-un sistem de baze de date care va asigura stocarea coerentă a algoritmilor şi accesarea lor într-un mod optim, un sistem de asamblare şi prezentare a informaţiilor folosit pentru interacţiunea cu exteriorul şi validarea introducerii algoritmilor/blocurilor funcţionale, un modul de comunicaţie care va prezenta algoritmii stocaţi folosind standardul ales (fie el IEC 61499 sau orice alt standard ulterior preferat), astfel încât algoritmii să poată fi implementaţi în sistemul de control al procesului pentru execuţia lor optimizată. În plus, este necesară existenţa unui modul de simulare/testare, fie înglobat în bibliotecă, fie extern acesteia, care să se ocupe cu validarea şi simularea rulării acestor date pentru a identifica oportunitatea folosirii unui anume algoritm în situaţii concrete. Reala valoare a acestui modul constă în culegerea datelor din timpul rulării şi integrarea acestora în cadrul informaţiilor despre algoritmi conţinute în biblioteca de algoritmi, oferind un istoric util în clasificarea eficienţei, împreună cu rata de succes/eşec a fiecărui algoritm în parte.

Pentru implementarea efectivă a bibliotecii, este preferată abordarea modulară, care să permită un nivel crescut de portabilitate şi o abstractizare a cadrului în care rulează. Astfel se poate folosi o implementare bazată pe maşini virtuale (VM) integrate în mediul cloud, conţinând fiecare toate resursele necesare rulării modulului/modulelor incluse în acel VM, lucru ce permite portarea bibliotecii între diferiţii ofertanţi de soluţii cloud, fie ele soluţii publice, cât şi private sau hibride, conform paradigmei PaaS. Alternativa este folosirea containerelor linux (LXC), care permite izolarea resurselor fie ele CPU, memorie, I/O, reţea sau chiar a proceselor care rulează pe acea maşină, făcându-se astfel posibilă separarea şi portarea fără a mai fi nevoie de folosirea unei maşini virtuale pentru fiecare componentă/proces din bibliotecă. Avantajul acestei soluţii este înaltul grad de integrare cu soluţiile cloud deja existente pe piaţă şi flexibilitatea portabilităţii, inclusiv în timpul funcţionării, fără a afecta disponibilitatea serviciului conform paradigmei SaaS.

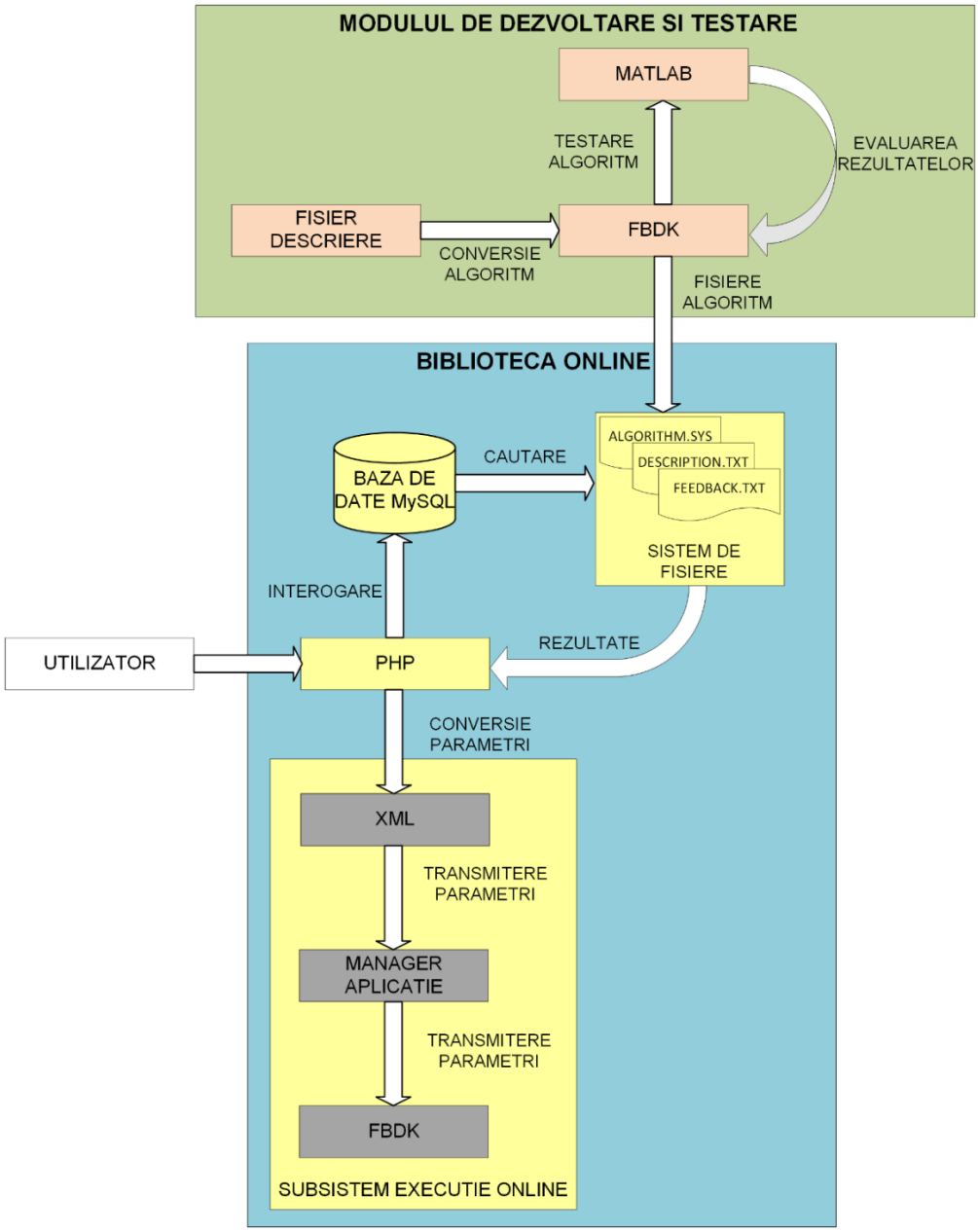

Fig. 2. Arhitectură sistem

Sistemul online trebuie să îndeplinească anumite funcţii:

- Să permită stocarea algoritmilor reutilizabili de control.
- Să permită accesul online la algoritmii disponibili în bibliotecă.
- Să ofere posibilitatea de executare online a unor algoritmi din bibliotecă şi furnizarea rezultatului execuţiei acestora către un controller aflat la distanţă;
- Să pună la dispoziţia administratorilor de sistem mecanismele necesare gestionării înregistrărilor (algoritmi şi utilizatori) astfel încât să nu existe duplicate;
- Să asigurare corectitudinea şi îndeplinirea criteriilor de performanţă ale algoritmilor prin verificarea şi validarea acestora înainte de a deveni disponibili utilizatorilor;
- Să definească limitele de utilizare ale algoritmilor disponibili (tipul de aplicaţie, numărul de variabile de intrare/ieşire etc.).

Biblioteca online va avea patru tipuri de utilizatori: utilizatori neînregistraţi, utilizatori privilegiaţi, utilizatori avansaţi şi administratori de sistem. Cele patru tipuri de utilizatori au un nivel de privilegii diferit, după cum a fost ilustrat în Fig. 3.

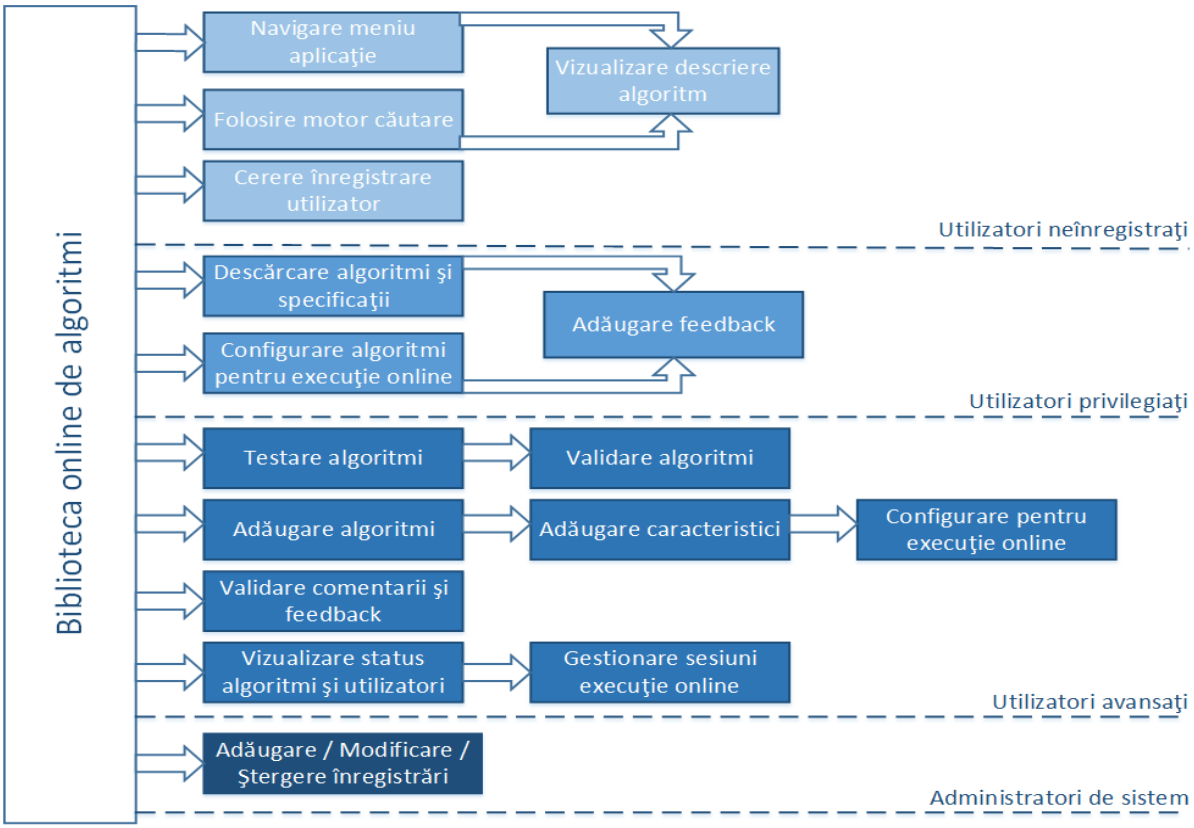

Fig. 3. Funcţiile sistemului

Utilizatorii neînregistraţi au acces la interfaţa de bază a aplicaţiei. Ei pot căuta algoritmi, vizualiza caracteristicile acestora şi eventualele rezultate obţinute în urma implementării lor într-o aplicaţie. Ei se pot înregistra, completând un formular, pentru a putea avea acces la funcţiile specifice utilizatorilor privilegiaţi. Utilizatorii privilegiaţi pot în plus să descarce algoritmii pentru a-i folosi în aplicaţii de control al proceselor sau pot configura execuţia lor online astfel încât rezultatul să fie trimis la un echipament aflat la distanţă. Ei îşi pot accesa istoricul acţiunilor (ce algoritmi au descărcat, ce feedback sau comentarii au adăugat şi ce algoritmi au avut sau au în execuţie online, precum şi caracteristicile acestora). Aceşti utilizatori pot, de asemenea, introduce comentarii sau fişiere înregistrând răspunsul procesului în care au fost implementaţi. Utilizatorii avansaţi au la dispoziţie aceleaşi funcţii ca şi utilizatorii privilegiaţi şi pot, în plus, să vadă care sunt algoritmii aflaţi în execuţie, să vadă istoricul activităţii utilizatorilor privilegiaţi, să adauge algoritmi noi şi să vizualizeze algoritmii adăugaţi, dar încă nevalidaţi. De asemenea, ei pot gestiona algoritmii online, astfel că pot opri execuţia fără a fi necesar acordul utilizatorului privilegiat care pornise execuţia, el fiind înştiinţat de acest lucru. Aceşti utilizatori pot aplica metode de verificare, validare sau optimizare a algoritmilor prin folosirea unor platforme de simulare. Administratorii bibliotecii au drepturi depline asupra înregistrărilor din baza de date. Ei pot adăuga sau şterge noi categorii de procese sau algoritmi, pot adăuga sau şterge utilizatori şi sunt responsabili de asigurarea integrităţii bibliotecii şi a componentelor acesteia.

Platforma colaborativă. Se propune ca algoritmii să fie scrişi în standardul IEC 61499 pentru sisteme de control distribuite şi să reprezente suportul şi mijlocul de creare a unor funcţii bloc distribuite, reutilizabile, care să poată fi integrate în strategia de control cu mai puţin efort, bazându-se pe strategiile stocate în baza de date. Strategiile vor putea fi introduse în procesul de control pentru a funcţiona alături de

sistemul existent. Rezultatul obţinut după integrarea funcţiilor bloc va fi prelucrat astfel încât algoritmii să se poată parametriza în funcţie de cazul concret. Fig. 4 prezintă structura platformei colaborative (CP). Aceasta poate fi utilizată pentru a asigura schimbul de informaţii între partenerii de proiect şi utilizatorii înregistraţi care doresc să contribuie la analiza specificaţiilor utilizatorilor şi să testeze rezultatele proiectului chiar din faza de dezvoltare. Utilizatorii vor fi încurajaţi să descarce funcţiile bloc şi să le testeze pentru instalaţii din diferite domenii industriale, utilizând infrastructura de modelare bazată pe tehnologii cloud ce suportă simulare „in-the-loop". Pe baza răspunsului primit de la utilizatorii finali, se va efectua un proces de selectare, astfel încât numai cei mai eficienţi algoritmi vor fi păstraţi în biblioteca bazei de date, în timp ce algoritmii neperformanţi vor fi eliminaţi sau optimizaţi, pentru a îndeplini criteriile utilizatorilor. Aceste date vor fi stocate utilizând etichetele semantice corespunzătoare pentru căutari ulterioare mai eficiente.

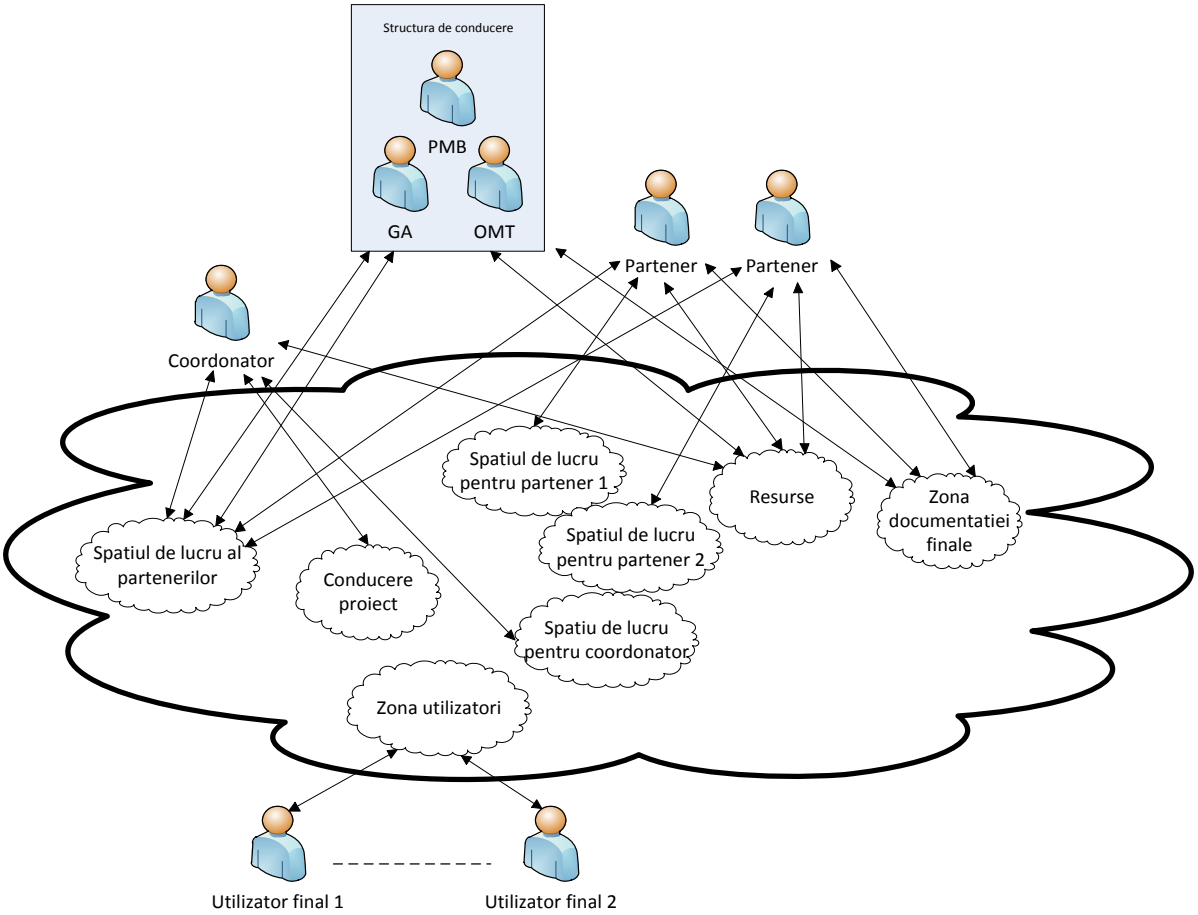

Fig. 4. Platforma Colaborativă

Administratorul Bazei de Date va fi responsabil de monitorizarea drepturilor şi statuturilor utilizatorilor; el va asigura şi suportul tehnic şi mentenanţa pe acest server. Când un utilizator final/client furnizează date despre proces în vederea creării unui algoritm de control pentru o anumită instalaţie, informaţiile vor fi strict confidenţiale. Inginerii specializaţi pot coopera în diverse faze ale procesului de elaborare a unui algoritm, pentru a identifica cele mai bune soluţii de implementare a funcţionalităţii acestuia, dar vor pune la dispoziţie şi serviciul de mentenanţă pentru modulul în care sunt specializaţi. Ei au acces numai la anumite module specifice şi la modulele de validare, simulare şi testare. Un alt tip de utilizator este potenţialul client, care are acces la modulul de bază de date, poate vedea algoritmii, îi poate testa şi primi răspuns/reacţii. Interfaţa cu utilizatorul ghidează clienţii şi informează clienţii potenţiali despre ofertele de suport tehnic. Platforma Colaborativă este susţinută de către Platforma Cloud.

Fluxul informaţiilor în sistem foloseşte o structură ierarhizată ilustrată în Fig. 5. Modulul de Baze de Date, amplasat la nivelul 0, serveşte la stocarea informaţiilor încărcate de către utilizatori prin API. Acest modul va prezenta o aplicaţie de data-mining, capabilă să facă un prim pas în descoperirea informaţiilor în baza de date (Knowledge Discovery in Databases − KDD) şi pre-procesarea informaţiilor. De asemenea, platforma asigură un Modul de Maşină Virtuală, în care utilizatorii privilegiaţi interesaţi pot testa algoritmii şi oferi reacţii, sau, în cazul unor utilizatori avansaţi, chiar pentru a contribui la extinderea bazei de date. Acest lucru va servi la exploatarea rapidă a rezultatelor şi la obţinerea unei valori adăugate mari prin implementarea unui algoritm adecvat unui procesul industrial. Mai mult, având un punct valid de plecare pentru rezolvarea problemelor specifice, abordarea va conduce la obţinerea unor produse mai fiabile, îmbunătăţite, într-o gamă largă de sisteme de control al proceselor (discrete sau cu funcţionare continuă).

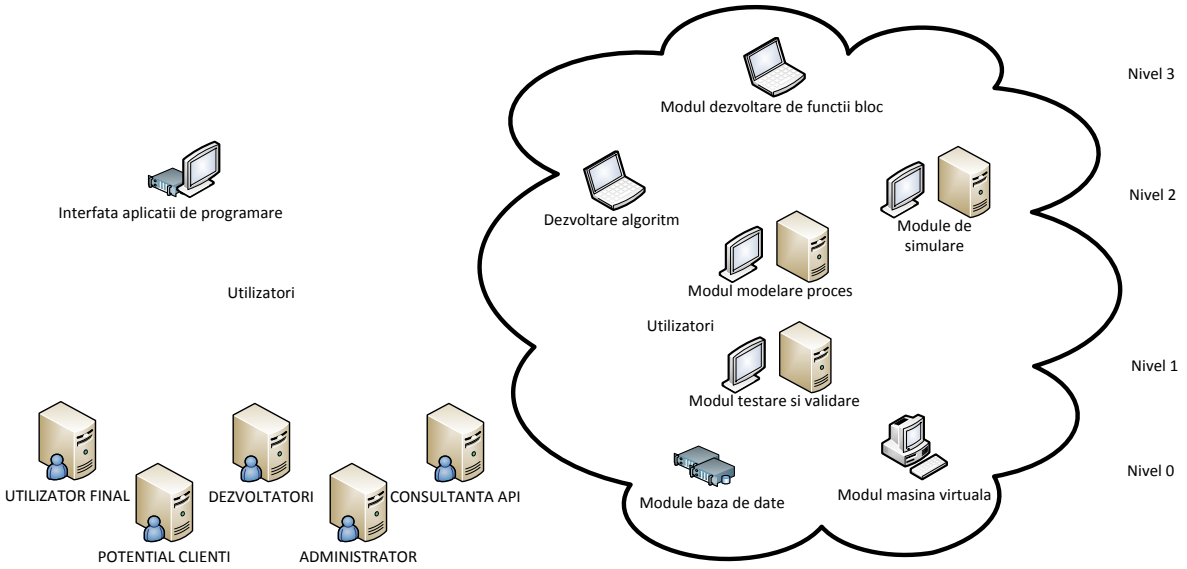

Fig. 5. Interfaţa Aplicaţiei de Programare

Modulul de Testare şi Validare, localizat la nivelul 1, este capabil să verifice informaţia colectată de la nivelul inferior şi să trimită un răspuns. Informaţiile pre-procesate sunt trimise la nivelul 2, Modulul de Modelare a Proceselor, unde se creează structura procesului pe baza informaţiilor prelucrate la nivelul inferior. Se poate folosi un Modul de Identificare Parametrică, pentru a obţine coeficienţii modelului, pe baza informaţiilor de intrare/ieşire şi a altor informaţii încărcate de către utilizatori.

Structura sistemului de control include Modulul de Dezvoltare a Algoritmilor şi Modulul de Simulare; rezultatele sunt accesibile utilizatorilor prin interfaţa aplicaţiei. Modelarea şi simularea sunt necesare pentru configurarea unui algoritm pentru o anumită instalaţie, respectiv, verificarea aplicabilităţii algoritmului pentru un sistem specific, iar apoi răspunsul obţinut din proces va fi folosit în optimizarea algoritmului. Nivelul superior este reprezentat prin Modulul de Dezvoltare a Funcţiilor bloc.

Infrastructura de preluare în cloud a datelor de la senzori fizici. Se propune o infrastructură dedicată, denumită Sensor-Cloud Infrastructure (SCI), care permite virtualizarea unui senzor sau a unei reţele de senzori fizici pentru acces în cloud, care devine astfel parte componentă în cloud. În particular, a fost tratată doar problema virtualizării reţelelor de senzori wireless (WSN). Interacţiunea dintre cloud şi lumea reală reprezentată de WSN este decuplată, în sensul că toate operaţiile executate în cloud se fac cu date furnizate de senzorul virtual. Ca atare, interacţiunile pot fi grupate în două categorii: cloud vs. senzor virtual şi, respectiv, senzor virtual vs. nod din reţeaua de senzori reali. Soluţia propusă are mai multe avantaje: nu mai este necesară instalarea de sisteme SCADA pe calculatoarele locale; utilizatorii au acces la diverse tehnologii informatice cu suport off-side; se oferă un spaţiu scalabil de server într-un cloud privat; pot fi satisfăcute la parametri de calitate superioară cerinţele de timp real.

Arhitectura SCI. Arhitectura software SCI are şapte componente:
1) Client: Utilizatorii pot accesa interfaţa utilizator a SCI prin browsere Web.
2) Portal: Portalul furnizează interfaţa utilizator a SCI. Când se conectează la portal, utilizatorul trebuie să-şi precizeze statutul (utilizator final, proprietar al reţelei de senzori, sau administrator SCI), pentru a determina operaţiile admise. Pentru utilizatorii finali, serverul portal indică meniurile de conectare/deconectare, cererile de alocare sau de anulare a grupurilor de senzori virtuali, precum şi procedurile de monitorizare şi control ale acestora. Pentru proprietarii reţelelor de senzori, serverul indică meniurile de conectare/ deconectare şi de înregistrare sau ştergere a senzorilor fizici.
3) Alocare: Punere în funcţiune/alocare automată a senzorilor virtuali. Acest server transmite grupurilor de senzori virtuali cererile de alocare de la serverul portal. Totodată, el defineşte fluxurile de lucru pe care le execută în ordinea prestabilită.
4) Managementul resurselor: SCI utilizează resurse IT pentru senzorii virtuali.
5) Monitorizare: SCI furnizează mecanisme dedicate de supraveghere şi monitorizare. Serverul de Monitorizare primeşte date despre senzorii virtuali de la agenţii din serverele virtuale, date care sunt preluate de administratorul SCI.
6) Gruparea Senzorilor Virtuali: SCI permite gruparea senzorilor la utilizatorii finali. Un grup de senzori virtuali este alocat automat unui server virtual de serverul de alocare. El poate fi activat sau dezactivitat de proprietarul grupului şi poate fi controlat de acesta direct sau printr-un browser Web.
7) Senzori: Senzori fizici (reali) utilizaţi în SCI.

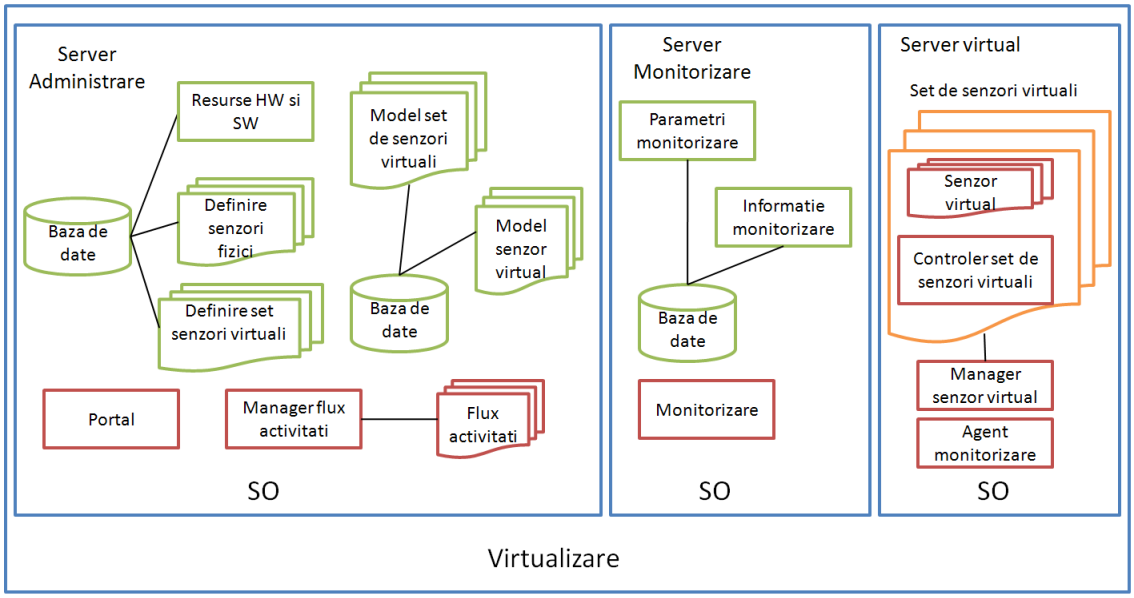

Fig. 6. Prototip SCI

Detalii de implementare. Fig. 6 ilustrează schema de implementare. In această variantă, serverul de administrare asigură funcţiile atât pentru serverul portal cât şi pentru cel de alocare, asigurând şi managementul bazei de date. Baza de date stochează caracteristicile senzorilor fizici (ID-ul proprietarului, tipul de senzor şi sursa datelor preluate de senzor), ale grupurilor de senzori virtuali (ID-urile grupului de senzori virtuali, ale utilizatorului final şi respectiv serverului virtual, cât şi datele create) şi ale resurselor IT (date despre servere şi despre servere virtuale, ca de exemplu, adresa IP sau numele gazdei). Totodată, se crează un registru depozit care stochează formularele tipizate pentru senzorii virtuali (un astfel de formular conţine biblioteca de programe şi fişierele cu reguli de prelucrare şi clasificare a datelor).

Fluxul de activităţi pentru alocarea grupurilor de senzori virtuali cuprinde:
1) Conectare.
2) Selectarea de formulare (*templates)* pentru grupurile de senzori virtuali.
3) Solicitarea unui grup de senzori virtuali. Portalul apelează la rândul său serverul de alocare.

4) Rezervarea resurselor IT.
5) Alocarea grupului de senzori virtuali pe serverul virtual selectat.
6) Notificarea reușitei alocării.

5. Analiza posibilităţilor de interfaţare

Interfaţarea cu sisteme SCADA. O mare contribuție la succesul arhitecturii prezentate este adusă de modul în care este executată interfațarea. Așa cum s-a menționat anterior, pentru structurarea algoritmilor de control se va folosi conceptul de funcții bloc distribuite. Similar cu circuitele integrate utilizate în proiectarea circuitelor electronice, o funcție bloc încorporează o anumită funcționalitate și poate fi conectată la alte funcții bloc prin intrările și ieșirile sale. In dezvoltarea algoritmilor se va utiliza Standardul IEC 61499, ce definește o arhitectură deschisă pentru noile generații de control distribuit și automatizări. La elaborarea standardui, IEC a luat în considerare proprietățile de portabilitate, reutilizare, interoperabilitate și reconfigurare a aplicațiilor distribuite. Spre deosebire de predecesorul sau, IEC 61131-3, o funcție bloc în IEC 61499 rămâne pasivă până în momentul în care apare un eveniment ce armează funcționalitatea respectivă și este executată, producând evenimente de ieșire și date. Dacă inițial această interfață a evenimentelor a fost criticată pentru că făcea scrierea aplicațiilor mai complicată comparativ cu IEC 61131, faptul că se permite specificarea explicită a secvenței de execuție a funcțiilor bloc dă dezvoltatorilor un nou nivel de flexibilitate inexistent anterior.

Sistemele de control industrial de tip SCADA (Supervisory Control and Data Acquisition) se află în centrul majorității industriilor moderne, cum ar fi cele de producție, energie electrică, rețele de utilități și transport etc. In orice domeniu actual, se vor găsi versiuni de sisteme SCADA, acestea implicând diverse tehnologii ce permit organizațiilor atât funcții de comandă cât și de monitorizare, colectare și prelucrare a datelor extrase din procesele industriale. Aceste sisteme variază de la configurații simple la proiecte de amploare, majoritatea acestora utilizând software de tip HMI (human-machine interface) ce permite utilizatorilor să interacționeze și să controleze dispozitivele implicate (valve, pompe, motoare etc.).

SCADA primește informațiile de la RTU-uri (remote terminal units) sau PLC-uri (programmable logic controllers), care la rândul lor sunt alimentate cu informații de către senzori sau cu valori introduse manual. Datele colectate sunt apoi monitorizate și prelucrate cu scopul principal de a crește eficienţa și eficacitatea instalațiilor. Majoritatea sistemelor moderne SCADA permit accesarea datelor în timp real și de la mare distanță facilitând luarea deciziilor imediate. Utilizarea de către programele SCADA a standardelor și practicilor IT de ultima generație, precum bazele de date puternice și integrarea cu sisteme de tip MES și ERP, permite, pe lângă o creștere a eficienţei, securității și productivității, și un mai facil flux al datelor.

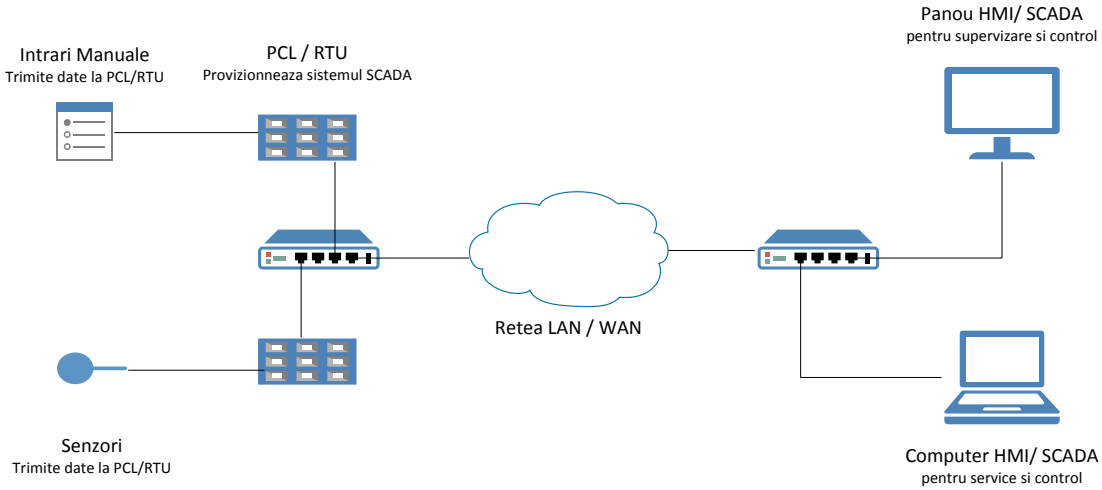

Fig. 7. Integrarea arhitecturii SCADA de bază

Sistemele SCADA au evoluat în timp distingându-se trei generații, o arhitectură de bază fiind prezentată în Fig. 7. În soluția de față se dorește ca platforma de Cloud Computing să preia atribuțiile sistemelor SCADA, adoptând tehnologia numita IoT (Internet of Things). Ca rezultat, sistemul SCADA poate raporta starea aproape în timp real și se poate folosi de proprietatea scalabilității orizontale, pe care mediul Cloud Computing o pune la dispoziție, pentru a implementa algoritmi de control mai complecși decât s-ar putea implementa în mod tradițional în PLC. Utilizarea protocoalelor deschise de rețea ca TLS (intrinsec IoT) oferă în mod implicit o arie de securitate mai mare decât în cazul utilizării unui mix de protocoale de rețea proprietare, cum se întâmplă în cazul multor implementări de sisteme SCADA descentralizate. O analiză a performanţei unui sistem SCADA distribuit în cloud este efectuată în [37], implementarea utilizând patru maşini virtuale. Analiza este realizată atât din punctul de vedere al performanţei, cât şi al securității. Se distinge un tip particular de cloud, anume cloud hibrid, ce este capabil să satisfacă cerințele de mentenanţă, acces la distanţă şi management al datelor necesare unui sistem SCADA.

Cea mai mare provocare rămâne totuși cea a asigurării securității, ținând seama ca unele tipuri de breșe sunt încă mai pregnante în mediul distribuit cloud, cum ar fi partajarea resurselor și imposibilitatea păstrării datelor senzitive sub controlul direct al organizației. Partajarea resurselor în cloud este realizată prin tehnologii de virtualizare, care poate fi de tip Full-Virtualization sau Para-Virtualization. În primul caz, mediul virtual este situat direct peste mediul hardware, pe când în cazul al doilea, mediul virtual este implementat indirect prin așa numitele middleware (hipervizoare) situate între sistemul de operare și hardware. O soluție care preîntâmpină unele considerente în materie de securitate asupra serviciilor oferite de furnizori este adoptarea de cloud privat şi utilizarea canalelor VPN (Virtual Private Networks).

Fiind încă în stadiu incipient pe plan mondial, adoptarea de sisteme SCADA "în cloud" este privită rezervat, paradigma abstractizării complete a arhitecturii fizice și transferul responsabilității dezvoltatorului și utilizatorilor doar către furnizorul de servicii fiind posibilă după o periodă inițială de adaptare și probare. Se propune ca, pentru sistemele SCADA complexe existente, trecerea să se facă treptat, păstrând o parte din funcționalitățile esențiale în sisteme de rezervă.

Interfaţarea M2M pentru integrare în cloud. În accepțiunea IoT, sunt interconectate trei tehnologii uzuale, senzori wireless, internet şi cloud computing, pentru a crea comunicaţia de tip M2M (machine-to-machine). Comunicația M2M este făcută posibilă de către senzori ce captează anumite evenimente care sunt apoi transmise către aplicațiile ce le vor procesa. Senzorii inteligenți sunt încorporați în dispozitive aflate la distanţă și transmit parametrii precum temperatura, locația, viteza, gradul de luminozitate, parametrii medicali, etc. Senzorii includ adaptoare de comunicație wireless, fiind capabile să primească și să transmită date prin rețelele de comunicații către un server central sau distribuit, unde aplicația le traduce în informație semnificativă ce poate fi și prelucrată. Comunicația M2M poate fi utilizată pentru aplicații din domenii variate:

- Producție: Senzorii monitorizează instalațiile de producție;
- Transport și Logistică: Urmărirea în timp real a vehiculelor, furnizând informaţii precum viteza, distanţa parcursă, consum de combustibil;
- Energie și Utilități: Contoare „smart" ce transmit consumul înregistrat în interval de câteva secunde;
- Servicii Publice: Orașe inteligente în care luminile stradale și semafoarele ajută la ghidarea vehiculelor ce intervin în caz de urgenţă;
- Medical: Echipament medical ce poate monitoriza pacienţii la distanţă, etc.

Dispozitivele M2M generează tipare diferite de trafic de la caz la caz ce pot include tipare de periodicitate, bazate pe eveniment, fluxuri multimedia etc. Acest fapt rezultă de cele mai multe ori în seturi de date mari și complexe (Big Data), greu de prelucrat într-un sistem nedistribuit. Fig. 8 ilustrează amestecul de tipare de trafic prezent în comunicațiile M2M aparținând diferitelor aplicații. Semnalele S1−S3 sunt exemple de tipare constante, bazate pe eveniment, iar semnalele S4 sunt periodice, generate de senzori aparținând, de pildă, aplicațiilor de supraveghere video, urgențe, contorizare. Trebuie ținut seama și faptul că acest trafic trebuie preluat și manevrat de mediul de comunicație. Prin mecanismele puse la dispoziție, rețelele de telecomunicații mobile pot asigura nivelul de QoS (Quality of Service) dorit.

Fig. 9 rezumă cele două variante posibile de implementare a comunicației M2M. Se propune o structură cu doi senzori virtuali (VS A, VS B), prezentă în platforma cloud, care vor putea fi controlați prin funcții

standard. Alocarea și gruparea este făcută dinamic, ca răspuns la cerințele utilizatorilor sau ale aplicației. Pentru cazurile în care senzorii sunt localizați în zone ce nu pot fi acoperite în mod normal de către rețeaua radio a furnizorului de servicii de comunicații mobile (de pildă, ecranare sau interferență cu aparatura existentă), se poate adopta soluția amplasării dispozitivelor ad-hoc, numite femtocell. Acestea pot opera în locația și frecvența adecvată pentru a maximiza capacitatea de comunicare.

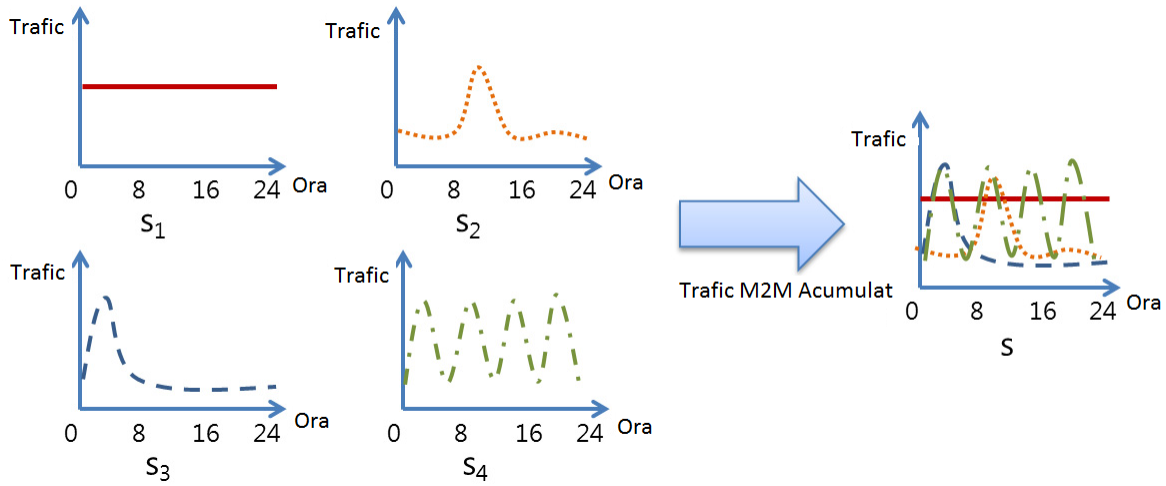

Fig. 8. Exemple de tipar de trafic în comunicația M2M

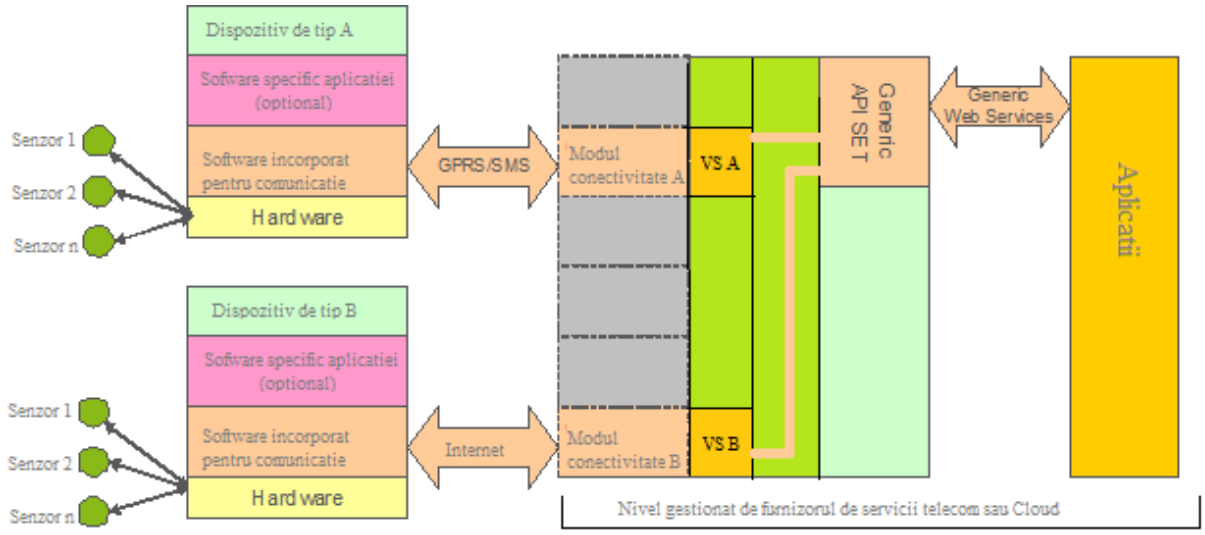

Fig. 9. Comunicație M2M între senzorii wireless și o aplicație din cloud

În [38] este făcută o analiză a principalelor dificultăți întâlnite la realizarea comunicației M2M a unei rețele de senzori și este prezentat un exemplu de Arhitectură de Rețea Orientată pe Serviciu (SONA − Service Oriented Network Architecture) pentru rețele de senzori ce comunică M2M. Sunt atinse aspecte privind scalabilitatea, latența, și irosirea resurselor sistemelor de comunicații datorate naturii distribuite a sistemului. Soluția propusă este înglobată într-o infrastructureă orientată pe serviciu, numită AAN (Application Assist Network). Structura acesteia este stratificată pe patru nivele logice, iar pentru a atinge eficiența maximă, în stratul de rețea este folosit un controller ce are ca funcționalitate optimizarea algoritmilor și găsirea punctului cel mai apropiat de sursa comunicației pentru a fi executați. O modalitate

interesantă de reutilizare a algoritmilor este adusă în discuție prin conceptul de date "cache" în nodurile aflate „în apropierea utilizatorilor".

Pentru soluția de față, instanțierea dinamică a senzorilor sau grupului de senzori virtuali (priviți ca resurse) poate fi coordonată astfel încât comunicația să nu încarce inutil rețeaua de date. Această abordare vine totuși cu o complexitate crescută, fiind dificil de realizat în acest moment, în special în cazul în care sistemul este distribuit pe platforme aflate sub controlul a diferite entități. Un model de soluție ce merită menționat, bazat pe arhitectura M2M a organismului de standardizare în telecomunicație ETSI (European Telecommunications Standards Institute), este prezentat în [39]. Această arhitectură este construită atât pe baza standardelor verificate ale ETSI, dar și ale altor entități, cum ar fi IETF, 3GPP, OMA (Open Mobile Alliance) și Broadband Forum. Este făcută o analiză a unui model comercial generator de platforme M2M în cloud, fiind pus accentul pe faptul că implementările comerciale adoptate de operatorii actuali trebuie inter-relaționate și aduse la un standard comun, denumit "oneM2M". Dezvoltarea soluției în cloud cu sisteme și software standardizat aduce implicit economii de scară, dar și specificații clare ale componentelor arhitecturale, interfețelor, aplicațiilor, tehnologiilor de acces, QoS, securității și caracteristicilor de gestionare a ciclului de viața (lifecycle management).

Interfațarea cu arhitecturi unificate OPC. OPC UA (OLE for Process Control Unified Architecture) este un protocol industrial de comunicație M2M, dezvoltat de OPC Foundation și folosit pentru interoperabilitate. Acesta este succesorul lui OPC, dar diferă semnificativ față de predecesor, deși este dezvoltat de aceeași organizație. Scopul fundației în legătură cu acest proiect a fost să furnizeze o continuare a modelului original de comunicație, OPC (bazat pe tehnologia Microsoft COM/DCOM), la o arhitectură SOA (Service Oriented Architecture) pentru controlul proceselor, independentă de platformă și, în același timp, ajutând la sporirea securității și furnizând un model informațional.

OPC UA supportă două protocoale: unul de tip binar ce implica un minim de resurse, permițând activarea simpla prin firewall-uri și un protocol de tip Web Service (SOAP) ce folosește porturile standard HTTP/HTTPS. Grație beneficiilor acestui nou protocol, se poate observa o tendință crescătoare a aplicațiilor industriale ce au adoptat OPC UA, atât în industriile de automatizări tradiționale OPC-centrice, cât și în cele emergente, cum ar fi cele energetice.

Modelul anterior „Classic OPC" necesită ca sistem de operare Microsoft Windows pentru a implementa funcționalitatea de server COM/DCOM. Utilizând SOA și Servicii Web, OPC UA este un sistem independent de platformă, iar prin SOAP/XML peste HTTP, OPC UA poate fi implementat într-o varietate de sisteme integrate, ce utilizează sisteme de operare în timp real deterministe, Fig. 10.

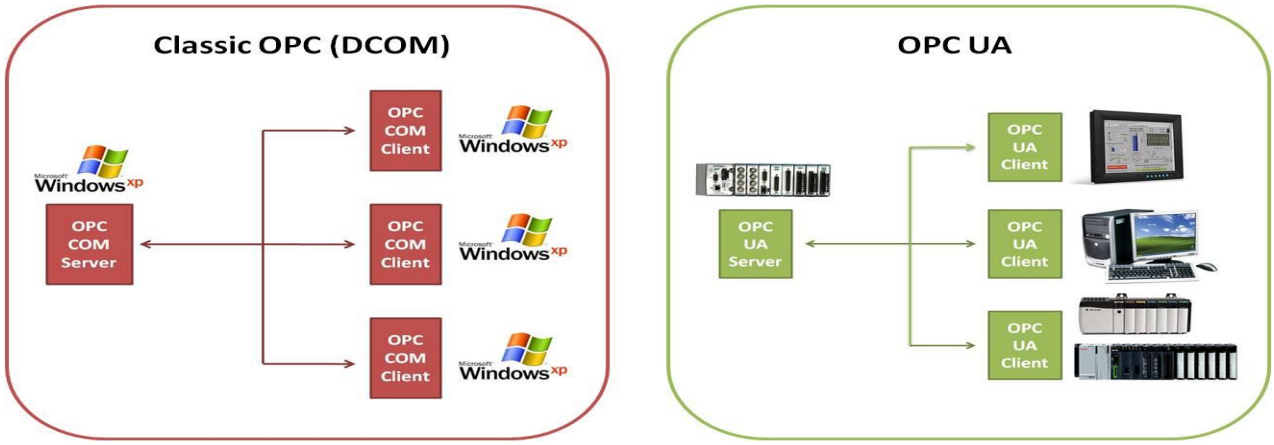

Fig. 10. OPC UA poate comunica direct cu sistemele integrate OPC UA de pe PLC

În [40] este făcută o prezentare detaliată a metamodelului OPC UA, conceptul de bază fiind nodul. Sunt definite diverse clase de noduri extinzând și specializând clasa de bază. Fiecare nod deține o mulțime fixă de

atribute, depinzând de clasă, unele dintre ele fiind opționale. Relațiile dintre noduri sunt realizate prin referințe de diverse tipuri cu ierarhii extensibile. Sunt expuse, de asemenea, 34 de servicii OPC UA, grupate în seturi de servicii. Un aspect important de considerat este procedeul de migrare a serverelor bazate pe Classic OPC (COM/DCOM) la noua tehnologie UA bazată pe Servicii Web. În mod tipic, serverele OPC bazate pe tehnologia COM utilizează interfețe proprietare pentru a accesa datele dispozitivelor în rețeaua de control. Se disting două abordări ce pot fi adoptate: împachetarea serverelor existente (wrapping) sau accesare directă a datelor dispozitivului. Împachetarea poate avea un timp mai scurt de implementare.

DPWS (Devices Profile for Web Services) definește un set minimal de restricții de implementare pentru a permite dispozitivelor cu constrângeri de resurse accesul la Servicii Web sigure. În cadrul proiectului SIRENA, sub auspiciile inițiativei de cercetare Europene ITEA, Schneider Electric a produs o implementare inițială a DPWS având ca țintă dispozitivele încorporate (embedded). Aceasta a devenit ulterior open-source prin platforma SOA4D.org (SOA for Devices). De asemenea, pornind de la proiectul SIRENA, WS4D.org (Web Services for Devices) furnizează informații și știri despre trei implementări DPWS. SODA (Service Oriented Device and Delivery Architecture) a continuat dezvoltarea și implementarea unei stive DPWS încorporate pentru dispozitive și instrumentele asociate. Actualmente, proiecte precum SOCRADES, alcătuit din companii ca ABB, SAP, Schneider Electric și Siemens, și AESOP se concentrează în implementarea, testarea și crearea de prototipuri de dispozitive DPWS în domeniul automatizării industriale.

Tehnologiile OPC UA și DPWS au câteva puncte distinctive, cum ar fi faptul că provin din domenii disjuncte − dacă OPC UA provine din mediul industrial, DPWS s-a născut în lumea IT și a fost gândit ca o modalitate de a implementa Servicii Web pe dispozitive mici, sau că au fost construite având abordări diferite, ca "discovery" și "eventing", însă au un larg set de asemănări, de exemplu, amândouă sunt bazate pe SOAP. Datorită acestui fapt, este posibil sa se găsească cazuri de utilizare unde cele două tehnologii pot lucra în comun sau pot beneficia una de pe urma celeilalte, iar o fuziune a acestora ar putea fi implementată într-un dispozitiv încorporat.

O cheie a succesului IoT pentru aplicațiile de automatizare industrială o reprezintă scalarea OPC UA pe dispozitive cu resurse limitate, cum sunt senzorii. Se investighează aspectul implementărilor curente ale OPC UA, prin faptul că sunt înzestrate cu caracteristici numeroase, ceea ce le face voluminoase, unele consumând chiar și 10MB de memorie. Aceasta se datorează faptului că memoria consumată este direct proportională cu complexitatea și densitatea modelului informațional specific aplicației. Este resimțită nevoia de putere înaltă de calcul, scalabilă și de infrastructuri de stocare masivă de date pentru prelucrarea, păstrarea și analiza datelor WSN, atât online cât și offline. În acest context, platforma Cloud Computing devine alegerea naturală, care furnizează o stivă flexibilă de calcul, stocare și servicii software.

6. Concluzii

Prezentarea făcută mai sus dovedește că obiectivele propuse pentru această etapă a proiectului, Etapa I, au fost atinse în întregime. Toate cele patru activități prevăzute au fost efectuate. Investigațiile întreprinse se vor dovedi foarte utile pentru realizarea etapelor următoare.

Etapa I nu a presupus diseminarea explicită a unor rezultate. Totuși, în ultimele trei luni au fost publicate trei lucrări elaborate de unii membri ai echipei proiectului, cu subiecte pe tematica proiectului. Una dintre lucrări a apărut într-o revistă indexată ISI, iar alte două lucrări au fost publicate în „proceedings", anume la o conferință internațională, care a avut loc la Sinaia și care a fost co-sponsorizată de IEEE. Lucrările conferiței vor fi accesibile de pe platforma IEEExplore, iar volumul de lucrări va fi indexat de ISI.

Bibliografie citată

1. "Function Blocks—Part 1 Architecture". International Electrotechnical Commission, Geneva, International Standard IEC 61499-1, 2005.
2. Vyatkin, V. "IEC 61499 as Enabler of Distributed and Intelligent Automation: State of the Art Review". IEEE Transactions on Industrial Informatics, 7 (4), pp. 768-781, 2011.

3. Vyatkin, V., Christensen, J. and Lastra, J.L. "An Open, Object-Oriented Knowledge Economy for Intelligent Distributed Automation". IEEE Transactions on Industrial Informatics, 1 (1), pp. 4-17, 2005.
4. Vyatkin, V., Hanisch, H.-M., Pang, C. and Yang, J. "Application of Closed-Loop Modelling in Integrated Component Design and Validation of Manufacturing Automation". IEEE Transactions on Systems, Man and Cybernetics - C, 39 (1), pp. 17-28, 2009.
5. Yang, C.-H. and Vyatkin, V. "Transformation of Simulink Models to IEC 61499 Function Blocks for Verification of Distributed Control Systems". Control Engineering Practice, 20 (12), pp. 1259-1269, 2012.
6. Aggelogiannaki, E. and Sarimveis, H. "A Simulated Annealing Algorithm for Prioritized Multiobjective Optimization – Implementation in an Adaptive Model Predictive Control Configuration". Systems, Man, and Cybernetics, 37 (4), pp. 902-915, 2007.
7. Khalgui, M., Mosbahi, O., Carpanzano, E. and Valente, A. "An Automated Approach for Adaptive Control Systems". International Journal of Intelligent Mechatronics and Robotics (IJIMR), 2 (3), pp. 58-71, 2012.
8. Schubert, L., Jeffery, K. (Eds.). "Advances in Clouds – Research in Future Cloud Computing". Expert Group Report Public version 1.0, European Union, 2012, http://cordis.europa.eu/fp7/ict/ssai/docs/future-cc-2may-finalreport-experts.pdf.
9. Brusaferri, A., Ballarino, A. and Carpanzano, E. "Distributed Intelligent Automation Solutions for Self-adaptive Manufacturing Plants". Balanced Automation Systems for Future Manufacturing Networks, IFIP Advances in Information and Communication Technology, 322, pp. 205-213, 2010.
10. Strasser, T., Rocker, M., Hegny, I., Wenger, M., Zoitl, A., Ferrarini, L., Dede, A. and Colla, M. "A Research Roadmap for Model-Driven Design of Embedded Systems for Automation Components". In: Proceedings of the 7th IEEE International Conference on Industrial Informatics (INDIN'09), June 23-26, 2009, Cardiff, Wales, United Kingdom.
11. Zoitl, A., Sunder, C., Strasser, T. and Colla, M. "A Device and Resource Execution Model for IEC 61499 Control Devices". In: Proceedings of 5th IEEE Int. Conference on Industrial Informatics (INDIN'07), pp. 1143-1149, Vienna, Austria, 2007.
12. Sunder, C., Zoitl, A., Christensen, J. H., Colla, M. and Strasser, T. "Execution Models for the IEC 61499 elements Composite Function Block and Subapplication". In: Proceedings of 5th IEEE Int. Conference on Industrial Informatics (INDIN'07), pp. 1169-1175, Vienna, Austria, 2007.
13. Leidi, T., Heeb, T., Colla, M. and Thiran, J.-P. "Component-based and Model-driven Development of Data-intensive, Time-critical Applications for Multi-core Embedded Systems". International Journal of Discrete Event Control Systems (IJDECS), 1 (2), Serial Publications, 2010.
14. Colla, M. and Leidi, T. "Tools Integration through a Central Model and Automatic Generation of Multi-Platform Control Code". In: Proceedings of First IEEE Workshop on Industrial Automation Tool Integration for Engineering Project Automation (IATPA 2011), pp. 1-6, Toulouse, France, 2011.
15. Vyatkin, V., Karras, S. and Pfeiffer, T. "Architecture For Automation System Development Based on IEC 61499 Standard". 2005 3rd IEEE International Conference on Industrial Informatics (INDIN'05), 10-12 Aug. 2005, pp.13-18, 2005.
16. Colla, M., Leidi, T., Kunt, M. and Thiran, J.-P. "CEC Designer: Domain Specific Modelling for the Industrial Automation Based on the IEC 61499 Standard". In: Proceedings of 13th IEEE Int. Conference on Emerging Technologies and Factory Automation (ETFA'08), Hamburg, Germany, 15-18 September 2008.
17. Joseph, J. "Cloud Computing - Patterns for High Availability, Scalability, and Computing Power with Windows Azure". MSDN Magazine, May 2009, http://msdn.microsoft.com/en-us/magazine/dd727504.aspx
18. Florea, Gh., Ocheana, L., Dobrescu, R. and Popescu D. "Emerging Technologies – the Base for the Next Goal of Process Control – Risk and Hazard Control". 11th WSEAS International Conference on Systems Theory and Scientific Computation (ISTASC '11), ISBN: 978-1-61804-027-5, 2011.
19. Ghosh, A. and Woll, D. "Business Issues Driving Safety System Integration". ARC Advisory Group, 2006.
20. Hatch, D. and Stauffer, T. "Operators on Alert. Alarm Standards, Protection Layers, HMI Keys to Keep Plants Safe". InTech, 9, 2009.
21. Mastronardi, N., Kressner, D., Sima, V., Van Dooren, P. and Van Huffel, S. "A Fast Algorithm for Subspace State-Space System Identification via Exploitation of the Displacement Structure". J. Comput. Appl. Math., 132 (1):71-81, 2001.
22. Sima, V., Sima, D.M. and Van Huffel, S. "High-Performance Numerical Algorithms and Software for Subspace-Based Linear Multivariable System Identification". J. Comput. Appl. Math., 170 (2), 371-397, 2004.
23. Sima, V. "Algorithms for Linear-Quadratic Optimization". Pure and Applied Mathematics: A Series of Monographs and Textbooks, vol. 200, Marcel Dekker, Inc., New York, 1996.
24. Benner, P., Mehrmann, V., Sima, V., Van Huffel, S. and Varga, A. "SLICOT – A Subroutine Library in Systems and Control Theory". In: B.N. Datta (Ed.), Applied and Computational Control, Signals, and Circuits, 1, ch. 10, pp. 499-539, 1999.
25. Van Huffel, S., Sima, V., Varga, A., Hammarling, S. and Delebecque, F. "High-Performance Numerical Software for Control". IEEE Control Syst. Mag., 24 (1), 60-76, 2004.
26. Benner, P., Sima, V., Voigt, M. "L_∞-norm Computation for Continuous-time Descriptor Systems Using Structured Matrix Pencils". IEEE Trans. Automat. Contr., 57(1), pp.233-238, 2012.
27. Isermann, R. "Trends in the Application of Model-Based Fault Detection and Diagnosis of Technical Processes". Control Engineering Practice, 5 (5), pp. 709-719, 1997.
28. Hwang, I., Kim, S., Kim, Y. and Seah, C.E. "A Survey of Fault Detection, Isolation and Reconfiguration Methods". IEEE Transactions on Control Systems Technology, 18 (3), pp. 636-653, 2010.
29. Ciubotaru, B.D., Staroswiecki, M. and Christov, N.D. "Modified Pseudo-Inverse Method with Generalized Linear Quadratic

Regulator for Fault Tolerant Model Matching with Prescribed Stability Degree". In: Proceedings of the 50th IEEE Conference on Decision and Control, and the 11th European Control Conference, paper MoC03.5, 2011.

30. Sima, V., Benner, P. (2014). "Numerical Investigation of Newton's Method for Solving Continuous-time Algebraic Riccati Equations". Proceedings of the 11th International Conference on Informatics in Control, Automation and Robotics (ICINCO-2014), Vienna, Austria, 1-3 September, 2014 (CD-ROM), Vol. 1, pp. 404-409.

31. Sima, V. "Efficient Computations for Solving Algebraic Riccati Equations by Newton's Method". In: Proceedings of the 2014 18th International Conference on System Theory, Control and Computing, October 17-19, 2014, Sinaia, Romania, pp.609-614, 2014.

32. Kriz, P. "Survey on Open Source Platform-as-a-Service Solutions for Education. " Proceedings of the 18th International Database Engineering & Applications Symposium. ACM, 2014.

33. Zhizhong, Z., Wu, C., and Cheung, D.W.L.. "A Survey on Cloud Interoperability: Taxonomies, Standards, and Practice. " ACM SIGMETRICS Performance Evaluation Review 40.4 (2013): 13-22.

34. Swan, C. "Docker Drops LXC as Default Execution Environment". InfoQ http://www.infoq.com/news/2014/03/ docker_0_9, 2014.

35. Chamberlain, R., Invenshure, L.L.C. and Schommer, J.. "Using Docker to Support Reproducible Research". Technical Report 1101910, figshare, 2014. http://dx. doi. org/10.6084/m9. Figshare. 1101910, 2014.

36. Boettiger, C. "An Introduction to Docker for Reproducible Research, with Examples from the R Environment". arXiv preprint arXiv:1410.0846, 2014.

37. Ocheana, L.A., Rohat, O.I., Popescu, D.A. and Florea, Gh.G. "Library of Reusable Algorithms for Internet-Based Diagnose and Control System". Information Control Problems in Manufacturing, 14th IFAC Symposium on Information Control Problems in Manufacturing, 14, Part# 1, 1407-1412, 2012, http://www.ifac- papersonline.net/Detailed/53961.html.

38. Holmström, K. "Practical Optimization with the TOMLAB Environment in Matlab". Applied Optimization and Modeling Group, Mälardalen University, Västerås, Sweden, 15.09.2001, 19 pag.

39. Lofberg, J. "YALMIP : a Toolbox for Modeling and Optimization in MATLAB". 2004 IEEE International Symposium on Computer Aided Control Systems Design, Taipei, 4-4 Sept. 2004, pp. 284-289.

40. Fojcik, M., Folkert, K., Kwiecien, A., Gaj, P. and Stera P. (Eds.). "Introduction to OPC UA Performance". Computer Networks 2012, Communications in Computer and Information Science (CCIS) 291, Springer-Verlag, Berlin Heidelberg, pp. 261-270, 2012.