

**RST - Raport științific și tehnic**

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

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

**Etapa II Arhitectura de sistem. Algoritmi pentru calculul riscului și de detecție a avariilor, diagnoză și acomodare**

**Cuprins**

<b>1</b>	<b>Introducere</b>	<b>1</b>
<b>2</b>	<b>Elaborarea documentației pentru modelul funcțional</b>	<b>2</b>
2.1	Oportunitatea utilizării platformelor cloud în conducerea proceselor . . . . .	2
2.2	Utilizarea specifică a serviciilor cloud în conducerea proceselor . . . . .	3
2.3	Descrierea arhitecturii sistemului . . . . .	4
2.4	Detalii de implementare . . . . .	6
<b>3</b>	<b>Specificații tehnice pentru componentele sistemului</b>	<b>7</b>
3.1	Structura algoritmilor . . . . .	8
3.2	Model conceptual al sistemului distribuit de control cu componente în cloud . . . . .	9
3.3	Experimentarea modelului client-server al sistemului de control IEC 61499 . . . . .	9
3.4	Serializarea datelor . . . . .	11
<b>4</b>	<b>Proiectare de algoritmi pentru calculul riscului</b>	<b>12</b>
4.1	Măsuri probabiliste pentru calculul riscului . . . . .	14
4.2	Tehnici pentru managementul riscului . . . . .	15
4.3	Analiza riscului și identificarea hazardurilor . . . . .	15
4.4	Analiza alarmelor . . . . .	16
<b>5</b>	<b>Algoritmi de detecție a avariilor, diagnoză și acomodare</b>	<b>17</b>
5.1	Detecția avariilor . . . . .	18
5.2	Algoritmi de reconfigurare și acomodare post-avarie . . . . .	19
<b>6</b>	<b>Concluzii</b>	<b>21</b>
	<b>Bibliografie</b>	<b>21</b>

## 1 Introducere

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

Obiectivele specifice ale proiectului sunt:

- Proiectarea unei platforme care să folosească o interfață prietenoasă de programare adresată mai multor tipuri de utilizator;
- Crearea unor mașini virtuale care să poată găzdui module și aplicații complexe;
- Reducerea costurilor de mentenanță pentru instalațiile industriale;
- Îmbunătățirea relației între mediul academic și cel industrial;
- Îmbunătățirea proceselor și instalațiilor industriale prin metode și servicii accesibile.

În cadrul primei etape de execuție a proiectului, *Etapa I — Cerințele utilizatorului, analiza acceptabilității și platforma colaborativă*, au fost efectuate de către parteneri patru activități principale, având ca rezultate un studiu privind stadiul industriei, cerințele și așteptările sale, un raport de analiză științifică și tehnică cu privire la posibilitățile de implementare a sistemului, elaborarea unei arhitecturi de control a proceselor și, respectiv, analiza posibilităților de interfațare cu utilizatorii și cu procesul.

**Obiectivul etapei de execuție.** Conform planului de realizare a proiectului, în cadrul acestei etape de execuție a proiectului, *Etapa II — Arhitectura de sistem. Algoritmi pentru calculul riscului și de detecție a avariilor, diagnoză și acomodare*, au fost efectuate de către parteneri următoarele activități principale:

- Activitatea II.1 (A2.1): Elaborarea documentației pentru modelul funcțional,
- Activitatea II.2 (A2.2): Elaborare specificații tehnice pentru componentele sistemului,
- Activitatea II.3 (A2.3): Proiectare de algoritmi pentru calculul riscului,
- Activitatea II.4 (A2.4): Algoritmi de detecție a avariilor, diagnoză și acomodare.

Obiectivele propuse pentru această etapă a proiectului au fost atinse în întregime. Toate cele patru activități prevăzute au fost efectuate și au fost orientate spre îndeplinirea obiectivului principal și a obiectivelor specifice menționate mai sus.

Implementarea aplicațiilor de conducere automată avansată a proceselor industriale necesită uzual resurse sporite de stocare și execuție. Raportul prezintă o aplicație bazată pe cloud care îi permite unui utilizator să aibă acces la diferite strategii de conducere folosind o interfață web, să beneficieze de regulatoare virtualizate, care să execute acei algoritmi și să trimită unele comenzi dispozitivelor din proces. Raportul propune o soluție pentru interconectarea modulelor implementând noi servicii, utilizând interfețe RESTful. Abordarea inovativă prezentată presupune virtualizarea unor “baze de date” de regulatoare de proces și îi conferă inginerului automatist dintr-o întreprindere industrială accesul la strategii avansate de modelare, optimizare și conducere, implementate ca funcții bloc IEC 61499.

Raportul investigează, de asemenea, aspecte referitoare la managementul situațiilor de risc, cât și pentru detecția avariilor, diagnoză și acomodare.

Etapa II nu a presupus diseminarea explicită a unor rezultate. Totuși, au fost publicate cinci lucrări elaborate de unii membri ai echipei proiectului (una dintre ele cu alți doi colaboratori, care nu

fac parte din echipă), iar o altă lucrare va fi prezentată la o conferință care va avea loc la Budapesta în luna decembrie 2015 și va fi publicată de către WSEAS (World Scientific and Engineering Academy and Society). Lucrările publicate deja au fost prezentate la conferințe internaționale co-sponsorizate de IEEE (Institute of Electrical and Electronics Engineers) și sunt incluse în reputata colecție IEEE Xplore Digital Library. Toate cele șase lucrări menționează într-o secțiune de mulțumiri suportul financiar parțial din proiectul CALCULOS.

## 2 Elaborarea documentației pentru modelul funcțional

Raportul prezintă un model pentru integrarea unei biblioteci de conducere avansată a proceselor într-un mediu bazat pe cloud. Scopul este de a reduce durata de elaborare a sistemelor automate, a efortului de întreținere și a complexității aplicațiilor de conducere a proceselor, prin adoptarea unei abordări modulare constând din strategii de comandă generice, reutilizabile, conforme unei reprezentări din standardul IEC 61499. Soluția utilizează tehnologii cloud deschise, cum ar fi Docker.

### 2.1 Oportunitatea utilizării platformelor cloud în conducerea proceselor

Evoluția rapidă a platformelor de calcul cloud crează oportunități deosebite pentru întreprinderile industriale, din ce în ce mai complexe, pentru a realiza conducerea mai eficientă a proceselor și reducerea costurilor operaționale. De pildă, adoptarea soluțiilor cloud în sistemele de fabricație permite integrarea pe verticală a diferitelor niveluri în procesul de producție, astfel încât conducerea proceselor poate fi concepută atât la nivelul supervisor (MES — Manufacturing Execution System), cât și la nivelul planificării întreprinderii (ERP — Enterprise Resource Planning), asigurând o performanță îmbunătățită prin reducerea costurilor, creșterea flexibilității sistemului, a toleranței la avarii, scalabilității și agilității.

Calculul în cloud este o paradigmă emergentă care oferă atât echipamente de calcul cât și programe sub formă de servicii. Paradigma posedă multe elemente atractive pentru utilizatori, dar și pentru furnizorii de infrastructură și de servicii ca, de exemplu, elasticitatea resurselor; se crează oportunități pentru reducerea energiei, pe baza virtualizării și a echilibrării sarcinilor de prelucrare [46]. Există studii despre provocările pe care caracteristicile fiecărui tip de aplicații le impune acestui mediu de execuție [19]. În mediile industriale apar provocări speciale. Sistemele industriale actuale de conducere automată memorează mari cantități de date de la senzori de supraveghere, cu scopul de a optimiza procesele. În ultimele decade, au fost proiectate astfel de sisteme în principal în ipoteza că acestea operează în infrastructuri de tehnologie a informațiilor (IT) închise, la nivelul întreprinderii, fără scalabilitate orizontală. Tehnologiile cloud pot fi utilizate în acest context pentru a reduce costurile IT locale și a permite o scalabilitate sporită; totuși, maturitatea acestor sisteme pentru aplicațiile industriale, care au cerințe severe de robustețe și viteză de răspuns (în timp real), nu este încă bine înțeleasă.

În cazul utilizării rețelelor fără fir (Wireless Area Networks—WANs) ca infrastructură de comunicații, latența și lățimea de bandă limitează utilitatea mediului de execuție în cloud ca suport pentru aplicațiile în timp real. Aceași problemă apare când se utilizează Internetul pentru conducerea în rețea; latența Internetului este prea mare pentru funcționarea în timp real a aplicațiilor și nu este posibil întotdeauna să se folosească un client hardware pentru a furniza capacitatea cerută.

Un suport important pentru a depăși aceste dificultăți este oferit de adoptarea SOA (Service Oriented Architecture) ca soluție pentru nivelul superior (de deasupra MES), care conține principalele puncte de integrare cu alți furnizori de materiale și servicii. Folosind SOA, sistemele cloud pot controla și optimiza automat utilizarea resurselor prin oferirea unei capacități de contorizare la

un anumit nivel de abstractizare, corespunzător tipului serviciului (de pildă, memorare, prelucrare, lățimea de bandă și conturile utilizatorilor activi).

Principalul avantaj al folosirii prelucrării în cloud pentru conducerea proceselor constă în conversia sistemelor de calcul tradiționale la mediile cu virtualizare. Virtualizarea unui sistem de calcul înseamnă organizarea și gestionarea în comun a resurselor hardware și software, permițând mai buna lor folosire, auto-servire fără cerere și elasticitate rapidă a capacităților de furnizare de resurse. Importanța acestor elemente a generat eforturi de cercetare susținute. Drept urmare, au fost propuse și implementate (cu finalizare între anii 2013 și 2015), câteva proiecte în cadrul programului FP7 dintre care se menționează 4CAAST, CONTRAIL, LEADS și VISION Cloud.

Abordarea din acest proiect și din lucrările elaborate de echipă [4, 5, 6, 16, 17] este de a furniza puterea de calcul aproape de client, acceptând latențe scăzute, fără a mări cerințele de capacitate de calcul pentru clienți. Această abordare de proiectare permite realizarea unor sisteme eficiente de conducere automată bazate pe Internet, prin extinderea rețelelor SCADA locale de la nivelul supervisor, capabile să implementeze strategii de conducere reutilizabile urmărind o reprezentare IEC 61499 [29]. Aceste strategii, cât și rezultatele execuției lor, sunt făcute disponibile ca servicii cloud. Formatul de date recomandat pentru acest scop este protocolul OPC, care oferă o interfață de comunicație unificată pentru un număr mare de tipuri de echipamente și modele. Rezultatul principal este un model pentru integrarea unei biblioteci avansate de conducere a proceselor într-un mediu bazat pe cloud.

## 2.2 Utilizarea specifică a serviciilor cloud în conducerea proceselor

Standardul IEC 61499 a fost elaborat ca soluție de mare performanță pentru a permite automatizarea inteligentă încorporată în componente software cărora li se asociază funcții bloc (FB), la rândul lor distribuite în sisteme cu rețele [43]. Funcțiile bloc IEC 61499 sunt proiectate natural drept componente reutilizabile, distribuite de la nivelul de conducere inferior până la nivelul cel mai înalt. Abordarea din proiect este de a considera funcțiile realizate prin FB disponibile ca servicii cloud. Așadar, la nivel logic, funcționalitatea este încapsulată în servicii, care pot fi invocate de alte servicii pentru a realiza o sarcină, iar toate datele din proces sunt accesibile independent ca servicii.

Întrucât calculul cloud oferă diverse soluții de punere la dispoziție a unei infrastructuri dinamice și flexibile de resurse de calcul, furnizate ca servicii la cerere, acesta poate fi considerat ca o soluție promițătoare pentru sistemele de automatizare industriale ale viitorului, care trebuie să fie adaptabile și agile. Unul dintre cele mai promițătoare modele arhitecturale este descris în [18]. Modelul constă din două etaje principale. Etajul inferior reprezintă migrarea nivelurilor din proces și de automatizare, în timp ce etajul superior reprezintă nivelurile avansate de automatizare și management. Este propus un cloud de automatizare (AT-Cloud), care furnizează funcții și servicii la nivelul inferior, și aplicații și servicii (IT-cloud) la nivelul superior. Integrarea între aceste două infrastructuri este asigurată de un Model Informațional. Din acest model general de cloud pentru automatizare pot fi derivate atât modelele disponibile cât și cele viitoare de calcul cloud. Pot fi evidențiate următoarele modele de furnizare a serviciilor cloud: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) și Software as a Service (SaaS), cât și modelele de implementare (private, publice și hibride). Funcțiile și serviciile de automatizare speciale pot fi oferite direct ca aplicații SaaS din IT-cloud, cât și din AT-cloud. Alternativ, o aplicație PaaS este utilizată ca platformă de automatizare pentru a furniza necesități specifice pentru integrare.

În continuare, se prezintă structura și funcțiile unei noi arhitecturi conceptuale ierarhice, proiectate astfel încât să evidențieze conceptul de servicii software auto-adaptive deschise, care pot fi evaluate într-un efort de a structura sistemele ICT emergente și de servicii software distribuite, necesare în automatizarea proceselor industriale. Există patru provocări principale care au permis

soluționări adecvate prin monitorizarea aplicațiilor cloud utilizând arhitecturile propuse, fundamental diferite de monitorizarea tradițională a aplicațiilor în întreprinderi industriale.

i. **Virtualizarea.** Pe lângă simpla monitorizare a elementelor cheie ale sistemului la fiecare nod fizic individual, folosirea virtualizării crează o rezervă de capacitate dinamică de resurse de calcul și de stocare, care trebuie monitorizate și administrate într-un mod diferit.

ii. **Determinarea profilului timpilor de răspuns pentru utilizatorul final.** Monitorizarea performanței utilizatorului este diferită într-o aplicație cloud, deoarece operează pe o rețea publică deschisă — fapt ce determină provocări proprii în ceea ce privește capacitatea de a monitoriza efectiv timpii de răspuns.

iii. **Testarea încărcării la scara web.** Soluția propusă înclină în favoarea aplicațiilor cloud a diferențelor de scară, în competiția dintre scara întreprinderii și scara web. Scara întreprinderii este mărginită, predictibilă și măsurabilă, în timp ce scara web, experimentată prin aplicațiile cloud, nu este. Vârfurile de trafic într-o aplicație cloud sunt posibil nelimitate, dar doar aparent impredictibile, deoarece istoricul datelor procesului oferă o informație specială ca Data as a Service (DaaS), care poate fi folosită pentru elaborarea algoritmilor de predicție bazați pe serii de timp.

iv. **Multi-închiriere.** Arhitecturile multi-închiriere furnizează o singură instanță a unei aplicații, pe baza unui model comun de date servind multipli “chiriași”. Această abstracție logică face mai dificilă monitorizarea și profilarea indicatorilor de performanță ai unui client individual.

### 2.3 Descrierea arhitecturii sistemului

Proiectul urmează o abordare de conducere avansată a proceselor, care permite configurarea execuției funcțiilor bloc ale standardului IEC 61499 pe un server la distanță utilizând o interfață web standard și trimite rezultatele la regulatoarele procesului industrial folosind interfețe de proces specifice [33]. Se dorește creșterea eficienței execuției și a flexibilității, prin aplicarea aceluiași concepte într-o structură scalabilă dinamică, cu resurse extinse. Arhitectura sistemului a fost proiectată cu scopul de a simplifica migrarea de la conducerea centralizată la un model distribuit, în care fiecare algoritm de prelucrare poate fi executat pe o mașină diferită. Acest model poate fi implementat folosind o platformă cloud PaaS.

#### Componentele sistemului

Sistemul este proiectat ca o aplicație bazată pe web, care să permită inginerilor automatiști din industrie să acceseze și să execute diferiți algoritmi complecși sau diferite strategii (reglare multi-variabilă, conducere adaptivă, conducere predictivă, identificarea sistemelor etc.), pe baza datelor memorate din proces și să optimizeze parametrii procesului pe baza acestor rezultate. Aplicația web este editată pe o platformă cloud, unde sunt definite două servicii de stocare, Data as a Service și Algorithm as a Service, cât și două servicii de execuție, Model as a Service și Control as a Service.

Data as a Service (DaaS) permite unui utilizator să aibă acces la cloud pentru date istorice, dar cu securitate, scalabilitate și fiabilitate sporite, cât și cu funcționalități de toleranță la defecte sau avarii. Culegerea datelor poate fi făcută fie dintr-o bază de date de proces existentă, fie dintr-o aplicație de istoric SCADA, fie în timp real folosind o interfață OPC-UA cu regulatoarele din proces sau noduri cu senzori inteligenți. Algorithm as a Service (AaaS) furnizează o reprezentare IEC 61499 a algoritmilor disponibili și a strategiilor disponibile, care pot fi ajustate și utilizate în logica existentă a unui regulator de proces. Toți algoritmi sunt furnizați într-o structură generică, ușor de adaptat la procese specifice. Model as a Service (MaaS) poate fi utilizat pentru a executa identificarea sistemului și optimizarea parametrică a procesului folosind datele stocate. Control as a Service (Caas) a fost proiectat pentru a permite execuția de la distanță a algoritmilor disponibili,

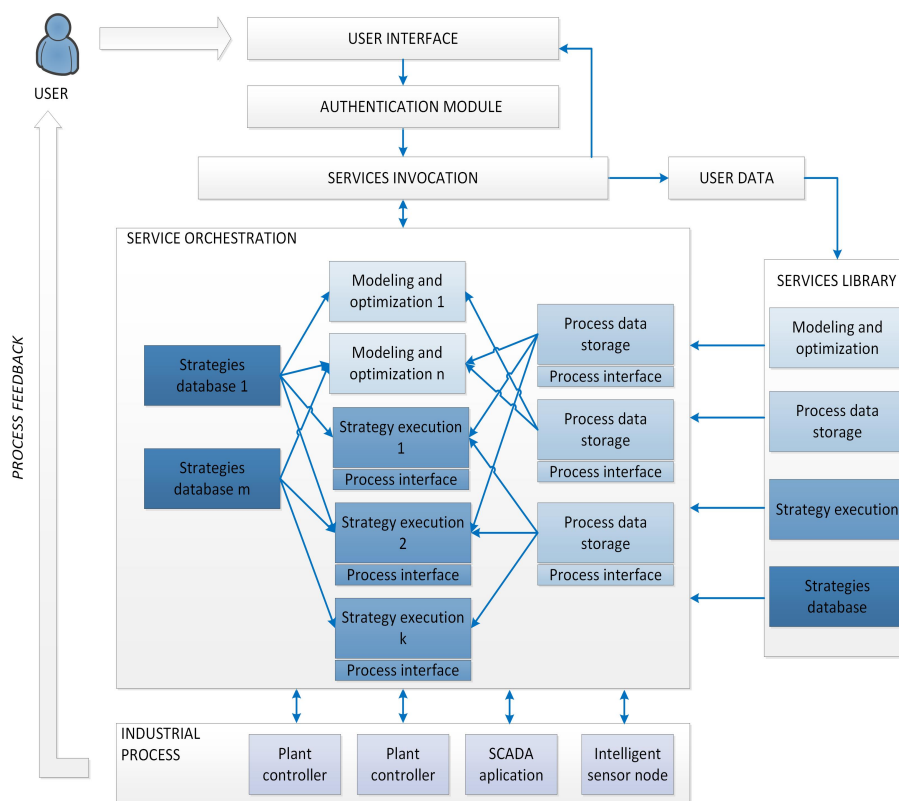


Figura 1: Arhitectura sistemului.

utilizând date de timp real din proces. Datorită naturii impredictibile a resurselor cloud și a vitezei comunicațiilor, acest serviciu nu este adecvat aplicațiilor critice, dar oferă funcționalitate importantă pentru aplicații cum ar fi ajustarea modelului pe baza datelor de timp real din proces, analiza seturilor mari de date din zone larg dispersate geografic etc.

Arhitectura sistemului este prezentată în Fig. 1 [6]. Un utilizator poate accesa interfața web pentru a solicita unul dintre serviciile disponibile. Modulul de Autentificare restricționează accesul unui utilizator doar la informațiile generale și informația sa personală. Serviciile disponibile sunt organizate în patru module: modulul de Modelare și Optimizare oferă MaaS, modulul de Stocare a Datelor din Proces rezolvă cererile DaaS, modulul de Execuție a Strategiilor implementează CaaS, iar modulul Baza de Date de Strategii este responsabil pentru AaaS. Toate aceste module, preconfigurate cu funcționalitatea dorită, sunt disponibile într-o bibliotecă a sistemului și pot fi instanțiate pe baza datelor utilizatorului când se primește o cerere de serviciu.

Platforma cloud este responsabilă pentru orchestrarea serviciilor cererilor active: alocarea dinamică a resurselor virtuale, cât și lansarea în execuție și terminarea aplicațiilor cloud. Pot fi interconectate diferite instanțe, de exemplu, dacă este executată o strategie folosind datele din proces din câteva resurse, sau dacă sunt utilizate simultan diferite instanțe ale aceleiași strategii. Interconectarea cu procesul se realizează folosind interfețe de proces create dinamic pe baza informațiilor furnizate de utilizator privind parametrii de comunicație. În acest mod pot fi utilizate în diferite aplicații strategiile generice disponibile.

## Tehnologii utilizate

Alegerea tehnologiilor utilizate pentru elaborarea sistemului a luat în considerare accesul la funcționalitatea dorită, ușurința implementării și instrumentele de dezvoltare a aplicațiilor centralizate. S-a ales o reprezentare a algoritmilor bazată pe funcțiile bloc ale standardului IEC 61499, datorită caracterului lor deschis, portabilității, independenței de platformă și posibilității de a proiecta arhitecturi distribuite. Regulatele locale dintr-un proces industrial pot utiliza resurse cloud prin conectarea la regulatele virtuale definite în modulul de Execuție a Strategiilor.

Ca mediu de execuție runtime a funcțiilor bloc IEC 61499 s-a ales instrumentul gratuit FBRT, bazat pe Java. Avantajele sale principale sunt compatibilitatea completă cu standardul IEC 61499 și cu alte produse, cât și costul și flexibilitatea oferite pentru rularea sau oprirea aplicațiilor. Pentru a face rezultatele execuției disponibile ca servicii, permițând astfel comunicația inter-module în cloud, s-au definit interfețe RESTful care folosesc servicii web peste HTTP pentru transferul datelor.

La selectarea celei mai bune infrastructuri cloud pentru aplicații de conducere automată avansată s-au luat în considerare disponibilitatea alocării dinamice a resurselor de calcul, fiabilitatea datelor stocate, scalabilitatea și accesul la cerere. Toate acestea pot fi obținute cel mai bine, în această aplicație, utilizând o infrastructură PaaS. Sunt disponibile câteva soluții PaaS, cum ar fi Amazon Web Services (AWS), Microsoft Azure și Google Cloud Platform. Fiecare dintre acestea oferă soluții diferite cu privire la tehnologiile colaterale pentru server, suport pentru instrumentele de dezvoltare și de integrare a aplicațiilor.

**Heroku** este una dintre cele mai de succes soluții PaaS create peste Amazon EC2, care permite punerea în funcțiune, rularea și administrarea aplicațiilor scrise în Ruby, Node.js, Java, Python, Clojure, Scala și PHP. Chiar dacă Heroku aduce multe beneficii în privința ușurinței de utilizare, dependența sa de platforma Amazon EC2 nu permite flexibilitatea și deschiderea așteptate pentru acest sistem de dezvoltare. De asemenea, Heroku nu oferă caracteristici de toleranță la defecte, scalabilitate automată sau fiabilitatea datelor, necesare pentru această aplicație.

**Docker** este o tehnologie deschisă (open source) care oferă o virtualizare la nivelul sistemului de operare, similară mașinilor virtuale, dar cu “balast” semnificativ mai redus. Aceeași aplicație poate fi descărcată pe platforme diferite, decuplându-se astfel cerințele de infrastructură de mediul aplicației. Portabilitatea Docker permite ca utilizatorul să îl integreze în toate soluțiile IaaS/PaaS publice disponibile, cât și în infrastructuri private. În plus, Docker face posibilă automatizarea procesului de includere a unei aplicații în containere și oferă o interfață de înalt nivel pentru administrarea containerelor. Datorită performanței sale, consumului redus de resurse, simplității de împachetare și depunere a unei aplicații în containere, cât și largului suport din industrie, Docker a devenit un standard popular pentru implementarea aplicațiilor în cloud, în special pentru proiectele la care portabilitatea este esențială.

## 2.4 Detalii de implementare

Sistemul cuprinde patru module de bază care sunt instanțiate în funcție de numărul de utilizatori și de serviciile și aplicațiile cerute de aceștia. Se va crea o instanță a fiecărui modul pentru fiecare cerere de serviciu solicitat de un nou utilizator sau o nouă aplicație. Componentele fiecărui modul, prelucrarea datelor și interacțiunea cu alte componente ale sistemului sunt prezentate în Fig. 2 [6]. Fluxul de date CaaS între componentele modulului de Execuție a Strategiilor este ilustrat cu roșu. Partea verde reprezintă administrarea cererilor DaaS utilizând componentele modulului de Stocare a Datelor din Proces. Liniile albastre conectează modulul de Modelare și Optimizare la alte componente ale sistemului. Liniile purpuri furnizează accesul la modulul Bazei de Strategii.

Utilizatorii pot accesa serviciile disponibile utilizând o interfață web. Când un utilizator emite

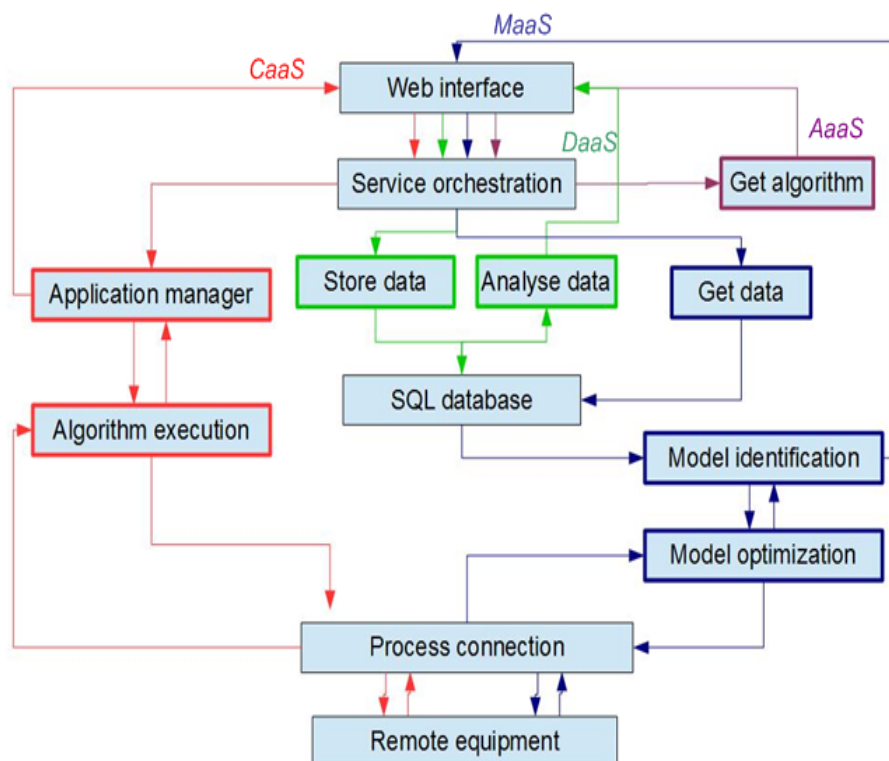


Figura 2: Interconectarea modulelor sistemului.

o cerere de serviciu CaaS, trebuie să selecteze un algoritm specific și să precizeze parametrii de execuție și detaliile de comunicație ale echipamentului de la distanță. Serverul web primește cererea, extrage informația privind detaliile de execuție (secvența ID, comunicația cu dispozitivul de la distanță, parametrii algoritmului de executat) și crează o nouă instanță a execuției containerului.

Un utilizator poate accesa serviciul DaaS ca un “istoric” cloud, permițându-i să memoreze datele din proces pe baza unei adrese specificate. Datele pot fi extrase dintr-un server SQL existent, sau pot fi obținute dintr-o aplicație SCADA folosind OPC. Se poate efectua analiza datelor, se pot determina tendințe, sau genera diferite rapoarte, integrarea în cloud asigurând reducerea timpilor de prelucrare și fiabilitatea sporită a spațiului de stocare.

Echipamentul la distanță poate fi reprezentat de un regulator de proces, o aplicație SCADA, sau chiar de senzori inteligenți. În primele două cazuri, comunicația poate fi făcută folosind o interfață OPC cu procesul. În ultimul caz se preferă o soluție mai eficientă, de pildă un mecanism publicare-abonament implementat în IEC 61499, care va permite, de exemplu, transmiterea datelor doar când apar schimbări, reducând astfel consumul de lățime de bandă.

### 3 Specificații tehnice pentru componentele sistemului

Pentru a realiza implementarea proiectului este necesar un mediu SOA contruit pe servicii web care să furnizeze o interfață pentru dispozitive IEC 61499 și sisteme distribuite de conducere automată. Aceste sisteme automate pot fi abstractizate prin dispozitive virtuale și interfațate cu o interfață REST cu scopul de a furniza blocurile constructive de bază pentru servicii și algoritmi de conducere complecși. În acest mod, pot fi proiectate structuri de conducere automată de înalt nivel, pe baza



unor servicii standard, flexibile, care să poată fi implementate și încorporate în cloud.

Un model conceptual pentru serviciul web de bază pentru conducere automată este prezentat în Fig. 3 din [6]. Sistemele de conducere automată IEC 61499 se bazează pe funcții bloc (FB), care încapsulează date și algoritmi, într-o unitate abstractă de conducere. În plus, mai multe funcții bloc pot fi grupate împreună pentru a crea rețele FB. Aceste funcții bloc sunt asociate cu dispozitive automate, și pot fi încorporate în orice resursă care poate găzdui sistemul “runtime” IEC 61499. De exemplu, trusa runtime și de dezvoltare pentru funcții block, FBDK2, foarte populară și bazată pe Java, poate fi încărcată pe orice sistem care suportă JVM (Java virtual machine).

### 3.1 Structura algoritmilor

Algoritmii vor fi reprezentați sub forma unor funcții bloc compuse, cu o structură deschisă, astfel încât utilizatorul să poată efectua modificări în cazul în care dorește acest lucru. Algoritmii vor fi disponibili sub forma unor fișiere de tip `*.fbt`, compatibile cu toate mediile de dezvoltare care implementează standardul IEC 61499. Fiecare algoritm va fi însoțit de o scurtă descriere care să detalieze funcționalitatea îndeplinită de acesta, dacă este aplicabil numai unor anumite procese, domeniul de utilizare, denumirea și rolul parametrilor de intrare și de ieșire (date și evenimente) și eventual rezultatul așteptat în urma implementării. Algoritmii care implementează funcții mai complexe (de modelare, optimizare, analiză etc.) vor putea fi executați în bibliotecă pentru ca apoi rezultatul să fie trimis către un echipament aflat la distanță. Pentru ca acest lucru să fie posibil, funcția bloc compusă specifică algoritmului va fi introdusă într-o structură de tip configurație de sistem cu extensia `*.sys` având definit un dispozitiv (de tip `PANEL_DEVICE`) și o resursă (de tip `PANEL_RESOURCE`). Aceasta este configurația minimă care permite simularea unui sistem distribuit și oferă în plus facilități de operare specifice echipamentelor industriale, precum pornire la cald, pornire la rece sau oprire.

Blocurile ce trebuie adăugate obligatoriu sunt cele care asigură controlul execuției (prin blocurile `RUNSTOP` și `RS_GATE`) și simulează ceasul unui echipament (prin blocul de tip `_CYCLE`), pornind de la o perioadă de eșantionare stabilită, precum și cele care asigură interfațarea cu aplicația web și cu procesul. Pot fi necesare blocuri de preluare a parametrilor de intrare și de ieșire din algoritm, pentru urmărirea mai ușoară a execuției algoritmilor și recepția corecte a parametrilor de la pagina web din cadrul serverului de aplicație.

Pentru integrarea în sistemul de execuție online al bibliotecii, funcției bloc compuse `i` se adăugă un bloc de `Subscribe` conectat la parametrii de intrare ai funcției bloc, care permite preluarea parametrilor de execuție de la interfața web, și un bloc de `Publish` conectat la parametrii de ieșire ai funcției bloc, care asigură transmiterea rezultatului către echipamentul aflat la distanță în instalație. Sunt cazuri în care algoritmii se execută în funcție de unul sau mai mulți parametri de proces. Pentru aceasta, este necesară adăugarea unui bloc de `Subscribe` suplimentar pentru conectarea la acei parametri.

Pentru ca procesul de generare al IP-urilor de la interfața web să se poată face automat, blocurile de comunicație `Publish-Subscribe` vor avea parametrul `ID` sub forma unui identificator generic, ulterior înlocuit cu valoarea adresei de IP și a portului. Astfel, blocul de `Subscribe` care preia datele de la interfața web va avea ca identificator stringul `__CALCULOS_TO_FBDK__`, blocul de `Publish` care transmite rezultatul către instalație va avea identificatorul `__FBDK_TO_PLANT__`, iar în eventualitatea că există un bloc de `Subscribe` ce preia din instalație valorile parametrilor de proces, acesta va avea identificatorul `__PLANT_TO_FBDK__`.

### 3.2 Model conceptual al sistemului distribuit de control cu componente în cloud

În continuare, se prezintă detaliat un model conceptual pentru un sistem de control distribuit bazat pe utilizarea funcțiilor bloc ale standardului IEC 61499, în care anumite dispozitive sunt reprezentate de aplicații care se execută în cadrul unei infrastructuri cloud. Acestea sunt incluse în containere Docker ce pot fi instanțiate la majoritatea furnizorilor de servicii cloud publice, precum și în sisteme de calcul locale sau private, fiind o soluție standard pentru distribuirea aplicațiilor.

Sistemul distribuit de control are o arhitectură client-sever. Componenta server implementează funcția bloc server sau o rețea de funcții bloc în care este inclusă funcția bloc de tip server. Componenta client implementează funcția bloc client sau o rețea de funcții bloc în care este inclusă funcția bloc de tip client. Aceste funcții bloc, respectiv client și server, oferă facilitatea de comunicare utilizând protocoalele TCP/IP. Pentru implementarea acestei arhitecturi se propune utilizarea mediului de execuție FBDK2. Acesta este implementat în limbajul Java, astfel încât funcțiile bloc pot fi incluse în cadrul unor programe complexe care pot conține elemente de interfață cu utilizatorul sau apeluri de funcții externe pentru execuția unor algoritmi complecși care sunt implementați în alte limbaje de programare. De asemenea, pentru a putea compune aplicații care utilizează paradigma SOA, se propune proiectarea unei interfețe de tip REST, astfel încât componenta client a sistemului distribuit de control să poată fi accesată sub forma unor servicii web de tip RESTful.

În această configurație, atât componenta client cât și componenta server sunt implementate în cadrul unor containere Docker diferite. Acestea pot fi instanțiate în mod separat, neexistând nici o legătură între aceste sisteme din punct de vedere al operării serviciilor. Astfel, componenta client poate să se execute în cadrul unui serviciu cloud public, iar componenta server se poate executa în cadrul unei infrastructuri cloud private. Singura condiție pentru funcționarea sistemului o reprezintă comunicarea directă utilizând protocolul TCP.

Containerele Docker permit execuția mai multor procese, însă cele mai bune practici recomandă execuția doar a unui singur proces în cadrul unui container, și compunerea unor stive de containere pentru executarea unor aplicații care necesită prezența unui server web, a unei baze de date, sau a altor procese auxiliare. O soluție în acest sens este oferită de platforma Kubernetes ce este dezvoltată de Google [25]. Astfel, mai multe containere care furnizează componente necesare unui serviciu sau unei aplicații sunt grupate împreună și formează o entitate denumită *pod*, ce este instanțiată de către platforma care administrează resursele [31]. Această configurație este prezentată în Fig. 3. În acest exemplu, sistemul distribuit de control este executat în cloud, iar interfața sub forma serviciilor web reprezintă o soluție pentru crearea unui model de aplicație de tip CaaS (Control as a Service).

### 3.3 Experimentarea modelului client-server al sistemului de control IEC 61499

Pentru experimentarea acestui model al sistemului de control, s-a utilizat un exemplu propus în cadrul grupului de discuții al platformei FBDK2, respectiv un sistem de control care avea funcția de server și un sistem care îndeplinea funcționalitatea client [15]. Sistemul server a fost instalat pe o mașină virtuală găzduită în cadrul serviciului cloud public Microsoft Azure. Adresa serverului este: `calculos.cloudapp.net`. Pentru conectare trebuie utilizat portul `4444/tcp`. Serverul execută o operație extrem de simplă, respectiv calculează suma a două numere și afișează rezultatul obținut. Pentru execuția clientului s-a folosit comanda: `java -jar demo.jar`. Fereastra client este prezentată în Fig. 4.

Un al doilea experiment a inclus utilizarea unui container Docker și implementarea serverului în cadrul acestuia. Pentru lansarea în execuție a serverului trebuie utilizat containerul Docker denumit `alexcstanciu/calculos_demo2`. Spre exemplu, se utilizează următoarea procedură:

1. Se descarcă imaginea `alexcstanciu/calculos_demo2`

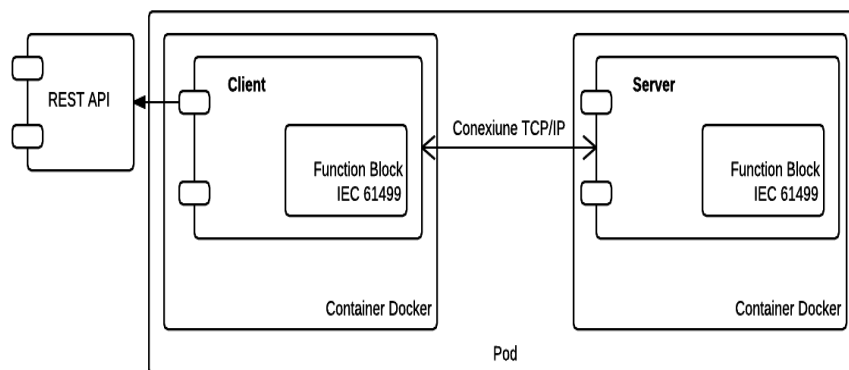


Figura 3: Model conceptual de containere Docker compuse într-o structură comună — pod.

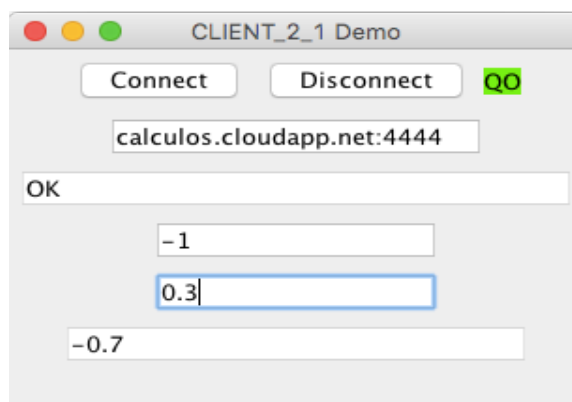


Figura 4: Implementarea unui server în cadrul serviciului cloud Microsoft Azure.

```
$ docker pull alexcstanciu/calculos_demo2
Using default tag: latest
latest: Pulling from alexcstanciu/calculos_demo2
d29d374e51d0: Pull complete
...
Digest: sha256:9797fd50221b74ba93752422291386bd9bccf86a9606384c0cc518226838e27b
Status: Downloaded newer image for alexcstanciu/calculos_demo2:latest
```

## 2. Se instanțiază un container cu această imagine:

```
$ docker run -d -P alexcstanciu/calculos_demo2
23c04aacb423d34cbc35c977702e8875af2b7e5c2876161c8efbf44197aba97d
```

## 3. Pentru conectarea clientului avem nevoie de adresa ip a mașinii virtuale unde rulează Docker, precum și de portul unde este accesibilă aplicația.

```
$ docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED
23c04aacb423   alexcstanciu/calculos_demo2        "/bin/sh -c demo"      16 seconds ago
STATUS        PORTS                                NAMES
Up 16 seconds  0.0.0.0:32768->4444/tcp             goofy_pasteur
```

```
$ docker-machine ip default
192.168.99.100
```

Fereastra client de similară celei din Fig. 4. (Adresa serverului și portul sunt înlocuite cu numărul ip 192.168.99.100, respectiv 32768.

Pentru implementarea serviciului web de tip RESTful în cadrul componentei client trebuie avut în vedere modul în care se va face transferul datelor, respectiv alegerea unui format pentru serializarea datelor. Cele mai populare standarde utilizate în acest sens sunt JSON și YAML. În continuare se va face o scurtă prezentare a acestora.

### 3.4 Serializarea datelor

Serializarea este procesul prin care structuri de date sau obiecte sunt transformate într-un format care poate fi stocat (într-un fișier sau într-un spațiu de memorie) sau poate fi transmis în rețea și care permite apoi reconstruirea datelor în starea inițială pe un alt sistem de calcul [35]. Acest proces se utilizează pentru transferul datelor pe canale de comunicații, pentru salvarea datelor în baze de date sau pe disc, pentru executarea unor apeluri de proceduri la distanță sau apelarea de servicii, etc. Serializarea trebuie să fie independentă de arhitectura sistemului de calcul. Cele mai populare formate pentru serializarea datelor sunt XML, JSON și YAML, acestea având calitatea de a putea fi citite de o persoană, cât și Thrift, Avro, sau Protocol Buffers, care au o reprezentare binară.

#### Standardul JSON

JavaScript Object Notation (JSON) este un format de tip text pentru serializarea datelor structurate. Acesta este derivat din limbajul JavaScript așa cum a fost specificat în cadrul standardului ECMAScript. JSON poate descrie patru tipuri de date fundamentale (șiruri de caractere, numere, boolean și null) și două tipuri de date structurate (obiecte și vectori) [2].

Un șir de caractere reprezintă o secvență formată din zero sau mai multe caractere de tip Unicode. Un obiect “reprint” este o colecție neordonată formată din zero sau mai multe perechi nume/valoare, unde numele este un șir de caractere, iar valoarea poate să fie de tipul șir de caractere, număr, boolean, null, obiect sau vector. Termenii de ‘obiect’ și ‘vector’ provin din limbajul JavaScript. Un *vector* este o listă ordonată din zero sau mai multe valori, fiecare putând să aibă un alt tip de date. Vectorii utilizează notația cu paranteze pătrate și elementele separate cu virgulă. Un *obiect* este o colecție neordonată de perechi nume/valoare în care numele, care mai este denumit și *cheie*, este un șir de caractere. Obiectele sunt utilizate pentru a reprezenta vectori asociativi și de aceea se recomandă ca fiecare cheie să fie unică în cadrul unui obiect. Pentru delimitarea obiectelor se folosesc acoladele, iar pentru separarea perechilor cheie-valoare se folosește virgula.

Obiectivele declarate în proiectarea formatului JSON au fost ca acesta să fie minimal, portabil, textual și să reprezinte un subset al limbajului JavaScript. JSON este un standard deschis bazat pe text, care poate fi citit de o persoană, și care este utilizat pentru transmiterea datelor. Acesta este principalul format folosit în cadrul comunicării asincrone browser-server web (AJAJ), înlocuind formatul XML ce este utilizat în cadrul comunicării de tip AJAX. Deși se bazează pe limbajul JavaScript, JSON este un format independent de limbaj, existând biblioteci și instrumente pentru generarea și extragerea de date în format JSON în majoritatea limbajelor de programare [24].

*Schema* unui document JSON reprezintă un format de tip JSON ce definește structura datelor JSON cu scopul de validare și documentare. Schema unui document JSON este similară unui contract referitor la modul în care sunt descrise datele necesare unei aplicații precum și la modul în care acestea pot fi modificate. Schema JSON se bazează pe concepte din cadrul XML Schema (XSD) însă utilizează elemente JSON. JSON reprezintă o alternativă pentru standardul XML, aceste două

formate de date fiind larg utilizate pentru crearea, citirea și transmiterea datelor. În afară de XML, un alt format foarte popular este YAML.

### **Standardul YAML**

YAML este un format pentru serializarea datelor ce poate fi înțeles ușor de către o persoană. Denumirea YAML provine de la “YAML Ain’t Markup Language”, deși în primele iterații în dezvoltare se numea “Yet Another Markup Language”. Această schimbare de nume s-a făcut pentru a evidenția orientarea sa pentru transmiterea datelor și nu pentru marcarea textului în cadrul unui document [44].

Sintaxa YAML a fost proiectată pentru a putea fi mapată cu ușurință pe tipurile de date cele mai utilizate în cadrul limbajelor de nivel înalt, și anume: listă, vector asociativ și scalar. Având o structură generală bazată pe indentarea nivelurilor inferioare, acest format este adecvat pentru situații în care o persoană trebuie să creeze sau să editeze structuri de date precum fișiere de configurare sau antetul unui document. Un rol deosebit în accesibilitatea acestui format îl are renunțarea la formele de demarcare precum paranteze, acolade, ghilimele, ce sunt dificil de urmărit în cadrul unor ierarhii imbricate. În plus, utilizarea spațiilor și a capătului de rând ca delimitatori îl fac să fie ușor de integrat cu instrumente precum grep/Python/Perl/Ruby.

Atât JSON cât și YAML au ca obiectiv să fie ușor de citit de către o persoană, însă au priorități diferite. Principalul scop în proiectarea formatului JSON a fost să fie foarte simplu și universal. Din acest motiv, JSON este ușor de generat și de prelucrat dar este un format mai greu de citit. De asemenea, JSON se bazează pe un model informațional extrem de simplu, ceea ce îl face să fie procesat de toate mediile de programare moderne. Pe de altă parte, YAML a avut drept obiectiv principal lizibilitatea și posibilitatea de a serializa structuri de date arbitrare. Din acest motiv, YAML permite crearea de documente care sunt ușor de citit, însă este mai dificil de generat și de prelucrat. De asemenea, YAML utilizează modele mai complexe pentru tipurile de date, ceea ce conduce la creșterea efortului de procesare atunci când sunt medii de programare mixte.

YAML poate fi privit ca un superset natural al standardului JSON ce oferă o lizibilitate mai bună și un model informațional superior. Acest lucru este confirmat în practică deoarece fiecare fișier JSON este și un document valid în formatul YAML. Din acest punct de vedere este ușor de migrat de la formatul JSON la YAML dacă sunt necesare facilități suplimentare [45].

## **4 Proiectare de algoritmi pentru calculul riscului**

Teoria riscului este strâns legată de teoria deciziei în condiții de incertitudine [34]. Dintre conceptele de bază, menționăm: riscul, măsura pentru calculul riscului și evaluarea riscului. Subiectul principal al teoriei riscului îl constituie diversele forme de risc suportate de către indivizi, firme, organisme guvernamentale și alte organisme cu rol de decizie confruntate cu necesitatea de a alege între diferite acțiuni posibile în situații cu rezultate nesigure, atunci când acestea sunt guvernate de către mecanisme aleatoare neutre sau de către un adversar inteligent. Riscul există în toate formele activității umane, iar studiul teoretic al naturii riscului și al strategiilor pentru realizarea unor compromisuri a devenit un subiect central în decizia politică, economie (la nivel microeconomic, dar și macroeconomic), inginerie, managementul mediului, teoria asigurărilor și multe alte domenii.

Noțiunea de risc implică aleatoriul, astfel încât instrumentul de bază al teoriei riscului este calculul probabilităților. Exemple de posibile riscuri sunt, de pildă, întreruperea sau funcționarea necorespunzătoare a unui proces tehnologic datorită defectării unei instalații, a unor senzori sau echipamente, căderii tensiunii de alimentare, producerii unor dezastre etc.

Severitatea unui risc depinde atât de probabilitatea lui de apariție, cât și de impactul acestuia (adică, severitatea consecințelor) [32]. Riscuri mici sunt acelea care au o probabilitate mică de

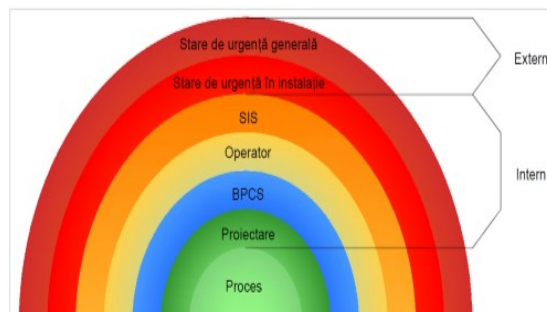


Figura 5: Niveluri de siguranță pentru proces.

aparitiie și impact mic. Similar, sunt definite riscurile moderate sau mari. Un risc cu impact mic este considerat ca risc mic, chiar dacă probabilitatea sa de apariție este mare. Severitatea poate fi apreciată în funcție de produsul dintre probabilitate și impact. În [32] sunt atribuite ponderile 1, 2 și 3 riscurilor cu probabilitate mică, moderată și, respectiv, mare și ponderile 1, 3 și 6 pentru impact mic, moderat și, respectiv, mare, astfel încât ponderea globală a riscului, obținută ca produs al ponderilor pentru probabilitate și impact, are una dintre valorile 1, 2, 3, 6, 9, 12 și 18. Astfel se pot identifica riscurile cele mai critice și pot fi aplicate strategii sau politici de control adecvate.

Riscurile pot fi clasificate în mai multe categorii, dintre care mai importante sunt cele tehnice, programatice (legate de performanțe) și de suportabilitate (legate de mediu). Dintre riscurile tehnice, se menționează: schimbări tehnologice, schimbări ale calității materiei prime, limitări ale productivității, sau schimbări operaționale. Alte riscuri interne, dar netehnice, sunt: întârzieri în aprovizionarea cu materii prime sau materiale, sau schimbări în echipa de operare. Unele riscuri sunt externe, dar nepredictibile, de pildă, inundații, cutremure, catastrofe, colaps financiar, sabotaj. Riscul trebuie privit din două perspective: pe termen scurt și pe termen lung. În general, riscul este controlabil, în sensul că acțiunea umană îi poate modifica forma și efectele, de pildă, elimina efectele negative și întări efectele pozitive.

Siguranță absolută, situația în care riscurile sunt complet eliminate, nu poate fi atinsă. Riscul poate fi doar redus la un nivel acceptabil. De aceea, toate riscurile trebuie tratate pe baza principiului ALARP (As Low As Reasonably Possible), adică urmărind obiectivul de a reduce nivelul de risc la valori cât mai mici posibil. Figura 5 ilustrează utilizarea nivelurilor de protecție (echipamente și/sau măsuri administrative) pentru a reduce riscul la un nivel acceptabil. În funcție de acțiunea lor, nivelurile de protecție pot fi clasificate în niveluri de prevenire și niveluri de atenuare. Primele sunt folosite pentru a opri apariția hazardului, iar celelalte pentru a reduce consecințele apariției unui eveniment de hazard. În funcție de locul în care acționează, nivelurile de protecție sunt împărțite în două categorii: în instalație și în exteriorul acesteia. Metodele care asigură nivelurile de protecție trebuie să fie independente, fiabile, proiectate special pentru riscul în discuție și să poată fi verificate.

Responsabilitatea funcționării în deplină siguranță a unei instalații tehnologice revine personalu-

lui de exploatare și, în principal, inginerului de proces. Definierea riscurilor, identificarea metodelor și mijloacelor de prevenire, cât și stabilirea de proceduri și strategii, se face încă din faza de proiectare. Identificarea, chiar din faza de concepție, a riscului și hazardului poate conduce la măsuri de implementare ce au în vedere reducerea sau eliminarea hazardului prin proiectare orientată către siguranță [22]. Chiar dacă primele măsuri de siguranță se iau din faza de proiectare, toate instalațiile implementează și strategii de gestionare a situațiilor de risc în scopul eliminării riscului, sau, dacă acest lucru nu este posibil, pentru diminuarea efectelor negative ale unui eveniment neprevăzut.

#### 4.1 Măsuri probabiliste pentru calculul riscului

Orice activitate industrială și, în general, orice activitate de producție, comportă elemente de incertitudine. Din punct de vedere matematic incertitudinea se modelează cu ajutorul variabilelor aleatoare sau, mai general, cu ajutorul proceselor stohastice.

Fie  $(\Omega, \mathcal{K}, P)$  este un câmp de probabilitate, unde  $(\Omega, \mathcal{K})$  este un câmp de evenimente ( $\mathcal{K}$  este mulțimea părților mulțimii de selecție  $\Omega$ ), iar  $P$  este o probabilitate pe  $(\Omega, \mathcal{K})$ . Se definește *risc* o variabilă aleatoare  $X : \Omega \rightarrow \mathbb{R}$ . Se notează cu  $V[\Omega]$  mulțimea variabilelor aleatoare definite pe  $\Omega$ . Fie  $M$  o submulțime a lui  $V[\Omega]$ . Se numește *măsură a riscului* o funcție  $f : M \rightarrow \mathbb{R}$ . Prin  $f(X)$  se notează măsura riscului  $X$ . Semnificația lui  $X$  poate fi legată de o serie de activități care au loc în cele mai diverse domenii și care pot influența procesul industrial considerat. De exemplu, semnificația lui  $X$  poate fi: profitul activității industriale; valoarea pagubelor suferite la apariția unei defecțiuni; depășirea valorilor admisibile de către un parametru, în raport cu un anumit prag (de pre-alarmă, de alarmă sau de plauzibilitate). Pentru a măsura riscul trebuie găsite modalități de a i se asocia un număr real. În cazul când semnificația lui  $X$  este efectul unei defecțiuni, atunci riscul apariției unui defect poate fi calculat prin una din următoarele formule:

$$f_1(X) = \sigma(X)^2 =: \text{var}(X) = \mathcal{E}[(X - \mu)^2], \quad \mu := \mathcal{E}(X), \quad (1)$$

unde  $\mu$  este *media*,  $\sigma^2$  este *dispersia* (iar  $\sigma$  este *abaterea medie pătratică*), iar  $\mathcal{E}$  este operatorul de medie;

$$f_2(X) = \mathcal{E}[(\tau - X)_+^p], \quad (2)$$

unde prin  $\tau$  a fost notat un prag, numit *prag de avarie*, iar prin  $p$ , un factor de putere; indicele inferior  $+$  denotă faptul că sunt luate în considerare numai valorile pozitive ale diferenței  $\tau - X$ .

În cazul când semnificația lui  $X$  este depășirea valorii maxime admise de către un parametru în raport cu un anumit prag, atunci riscul poate fi calculat prin următoarea formulă:

$$f_3(X) = \mathcal{E}[(X - \tau)_+^p]. \quad (3)$$

Media depășirilor valorii parametrului peste valoarea acestui prag este considerată ca o măsură pentru riscul de funcționare necorespunzătoare a unei instalații.

Dacă  $F_X$  este funcția de repartiție, iar  $f_X$  este densitatea de repartiție de probabilitate a variabilei aleatoare  $X$ , atunci,

$$\mu(X) = \mathcal{E}(X) = \int_{-\infty}^{\infty} x dF_X(x) = \int_{-\infty}^{\infty} x f_X(x) dx, \quad \sigma(X)^2 = \int_{-\infty}^{\infty} (x - \mu)^2 dF_X(x). \quad (4)$$

Dacă în locul integradului din formula dispersiei (4) se folosește modulul  $|x - \mu|$ , se obține *abaterea medie absolută*, reprezentând *momentul centrat de ordinul 1*.

Alte măsuri ale riscului sunt prezentate, de exemplu, în [34] și referințele citate acolo.

Unele dintre formulele de mai sus pot fi implementate simplu. Dacă mulțimea de evenimente este finită, integralele devin sume. (Alternativ, se utilizează algoritmi numerici de integrare.) În

particular, media și dispersia se calculează folosind formulele uzuale. O mare varietate de algoritmi pentru calcule statistice sunt implementate în bibliotecile comerciale IMSL (International Mathematics and Statistics Library), implementate în limbajele de programare C, Java, C#.NET și Fortran. Este disponibilă, de asemea, și o interfață la limbajul Python.

## 4.2 Tehnici pentru managementul riscului

Instrumente, practici și protocoale pentru managementul riscului, permițând identificarea și evaluarea calitativă și cantitativă a riscurilor în proiecte și programe, din perspectiva responsabililor de proiecte, sunt prezentate pe larg în [32], într-o manieră predominant descriptivă. Sunt examinate într-o structură unitară peste 30 de tehnici de evaluare a riscului și strategii de răspuns la situații de risc. Riscul este perceput ca fenomen viitor care se poate manifesta atât ca amenințare, cât și ca oportunitate. Oportunitatea variază în sens opus variației riscului: un risc mic poate implica o mare oportunitate și invers. Una dintre tehnicile des utilizate este *analiza SWOT* (Strengths, Weaknesses, Opportunities, and Threats), care permite identificarea riscului examinând punctele tari, cele slabe, oportunitățile și amenințările. O altă tehnică de mare reputație este *PERT* (Program Evaluation and Review Technique) în care se evaluează mediile și dispersiile timpilor de execuție a diferitelor activități dintr-un program sau proiect și statisticile pentru cea mai scurtă durată de realizare (sau cel mai scurt traseu de parcurs, adică, *drumul critic*). Media duratei unei activități se calculează cu formula

$$(\text{optimist} + 4 \times (\text{cel mai probabil}) + \text{pesimist})/6,$$

iar dispersia cu

$$(\text{pesimist} - \text{optimist})/6,$$

unde “optimist” și “pesimist” reprezintă estimățiile corespunzătoare celui mai bun, respectiv, celui mai rău caz. Dispersia pentru drumul critic PERT se calculează cu formula

$$\sqrt{\sigma_1^2 + \sigma_2^2 + \dots + \sigma_n^2},$$

unde  $\sigma_i, i = 1, \dots, n$ , sunt dispersiile asociate celor  $n$  componente ale drumului critic.

În general, un risc al unui eveniment cu mare probabilitate de apariție și mare impact trebuie tratat cu urgență, spre deosebire de alte riscuri cu probabilitate și impact asociate mai mici. Riscul tehnologic și fiabilitatea sunt adesea asociate cu amenințări potențiale ale vieții sau mediului [14]. De pildă, întreruperea furnizării de energie electrică poate afecta serviciile de sănătate sau de intervenții de urgență, traficul în orașe mari, controlul traficului etc. Astfel de întreruperi sunt legate de decizii într-un sistem implicând risc, fiabilitate și întreținere. Adesea, aceste decizii includ mai multe obiective care trebuie tratate simultan, folosind modele multi-obiectiv sau multicriteriale.

Multe tehnici și proceduri de modelare și rezolvarea a problemelor de optimizare multicriteriale, inclusiv referitoare la analiza riscului și identificarea hazardului, sunt prezentate în [14]. Când trebuie luată o decizie care implică risc, atunci este necesar să fie evaluate și considerate consecințele, probabilitățile de apariție și preferințele decidentului. Întrucât consecințele sunt adesea multidimensionale (de exemplu, cu dimensiunile: financiar, fiabilitate și siguranță, sau sănătate), este necesară găsirea unui compromis.

## 4.3 Analiza riscului și identificarea hazardurilor

Analiza riscului poate fi definită ca o tehnică pentru identificarea, caracterizarea, cuantificarea și evaluarea hazardului. Evaluarea riscului este un proces sistematic calitativ, cantitativ, sau semicantitativ care descrie natura, probabilitatea și mărimea riscului asociat cu orice substanță, situație, acțiune sau eveniment care include incertitudine.



Măsuri relevante pentru risc pot fi pierderile sau accidentele bazate pe date statistice, dar trebuie privite cu precauție, deoarece multe dintre aceste statistici reprezintă medii, iar nu producerea unui accident specific cu pierderi potențiale.

Proiectarea unei strategii de reducere sau eliminare a riscului începe cu analiza situațiilor de hazard în proces. În faza de proiectare, aceasta conduce la revizuirea procesului tehnologic, a sistemului de control, a procedurilor de operare și a celor de întreținere. În etapa de operare, analiza riscurilor presupune evaluarea stării instalației, identificarea riscurilor posibile și intervenția în scopul prevenirii situațiilor nedorite. Utilizarea unei proceduri de evaluare a riscului și hazardului, cum ar fi HAZOP, conduce la identificarea celor mai bune soluții pentru sistemele cu siguranță mărită [3]. Utilizarea mai multor metode de analiză a riscului cum sunt HAZOP, SIL, RCM și RBI, asigură definirea unitară a riscului și conduce în mod evident la creșterea siguranței în funcționare și trebuie, pe cât posibil, să fie aplicată la fiecare proiect. Pe baza consecințelor posibile, se poate efectua o clasificare a riscurilor în categorii de risc.

Efectuarea unui studiu HAZOP complex presupune o examinare detaliată și riguroasă a procesului de către o echipă multidisciplinară incluzând ingineri de proces, de instrumentație, instalații electrice, instalații mecanice, specialiști în siguranță, dar și reprezentanți ai beneficiarului. Sunt analizate diagramele cauză-efect și riscurile aferente fiecărei funcții și operații din proces. Dacă în urma studiului se constată că sistemul de control al procesului nu este suficient pentru asigurarea condițiilor cerute de beneficiar, se poate solicita un nivel SIS (Safety Instrumented System). Alocarea funcțiilor de siguranță pe niveluri de protecție este etapa care pune bazele întregului sistem de management pentru SIS, atribuind un nivel de siguranță unei funcții de siguranță. Analiza SIL (Safety Integrity Level) este realizată având ca obiectiv alocarea SIL pentru fiecare funcție SIF (Safety Instrumented Function). Standardele permit utilizarea mai multor metode: matricea riscurilor, graful riscurilor, analiza nivelurilor de protecție sau arborele defecțiunilor. În cazul matricei riscurilor, care este cea mai utilizată metodă, se vor alocă niveluri SIL tuturor riscurilor, precum și metode de protecție adecvate încadrării în nivelul tolerabil de risc.

În [14] sunt descrise succint și alte metode, cum ar fi analiza arborelui de avarii, analiza arborelui de evenimente, modelul avariei și analiza efectului etc.

Pentru cuantificarea hazardurilor și riscurilor, fiecărei categorii de risc îi corespund mai multe niveluri de consecințe definite în funcție de severitatea impactului pe care îl are apariția unui risc din categoria respectivă. De asemenea, fiecărui tip de consecință i se asociază un nivel de probabilitate de apariție în intervalul de timp de referință. (Uzual, acest interval este durata de viață a instalației.)

#### 4.4 Analiza alarmelor

Sistemele de alarmare din cadrul unei instalații constituie o parte esențială a interfeței operator de monitorizare și control, întrucât gestionarea corectă a alarmelor într-o situație cu risc ridicat presupune acțiuni de reducere a nivelului de risc. Cu toate că majoritatea alarmelor permit unui operator să detecteze o funcționare anormală și să identifice cauza acesteia, cele mai multe aplicații de monitorizare vor afișa o cantitate foarte mare de alarme într-o situație de risc ridicat, ceea ce va duce la o capacitate scăzută din partea operatorilor de a putea identifica în mod corect cauza acestora. Astfel, alarmele critice se pierd printre cele care nu sunt neapărat necesare. Pentru evaluarea eficientă a stării de alarmă dintr-o instalație, precum și a riscului asociat, este necesară analiza alarmelor la nivel de sistem, prin monitorizarea și gestionarea continuă a acestora folosind algoritmi dedicați. În practică, sunt folosiți mai mulți indicatori de performanță (KPI — key performance indicator) pentru evaluarea nivelului de performanță al unui sistem de alarmare, precum: media ratei de alarmare, rata maximă de alarmare, sau procentul de timp în care ratele de alarmare sunt în afara nivelului acceptat. Rata medie de alarmare se calculează ca fiind numărul total de alarme raportat la numărul

total de perioade de timp. Se recomandă ca acest raport într-o operare normală să fie sub 1 la 10 minute. Rata maximă de alarmare reprezintă numărul de alarme care îi apar unui operator în timpul oricărui interval de 10 minute. Dacă acest raport are o valoare de 10 alarme în 10 minute înseamnă că sistemul nu poate fi gestionat corect de către un operator pe perioade mari de timp. Metoda “cele mai rele 10 alarme” a fost folosită pe scară largă în aplicații industriale în Japonia pentru a reduce numărul de alarme care nu sunt necesare. Această metodă este utilizată pentru a colecta date din istoricul de alarme în timpul operării, pentru a crea o listă a celor mai frecvent generate alarme. Deși această metodă poate reduce efectiv numărul de alarme pe care le vizualizează un operator, este mai puțin eficientă când vine vorba de reducerea numărului de alarme pe măsură ce dispar unele dintre cele zece cele mai rele alarme.

Nishiguchi și Takai [28] au propus o metodă de evaluare bazată pe istoricul datelor, care se referă nu numai la analiza alarmelor, ci și a evenimentelor apărute în funcționarea instalației. Datele de operare sunt convertite în evenimente secvențiale,  $s_i(k)$ . Când apare evenimentul  $i$  între momentele  $(k-1)\Delta t$  și  $k\Delta t$ , atunci  $s_i(k) = 1$ , dar altfel  $s_i(k) = 0$ . Funcția de intercorelare,  $c_{ij}(m)$ , dintre  $s_i(k)$  și  $s_j(k)$ , pentru intervalul de timp  $m$ , se calculează cu ecuația (5). Aici,  $K$  este perioada de timp maximă pentru durata intervalului și  $T$  este durata totală de timp pe parcursul căreia au fost preluate datele.

$$c_{ij}(m) = \begin{cases} \sum_{k=1}^{\frac{T}{\Delta t}-m} s_i(k)s_j(k+m), & 0 \leq m \leq \frac{K}{\Delta t} \\ c_{ij}(-m), & -\frac{K}{\Delta t} \leq m < 0 \end{cases} . \quad (5)$$

Când două evenimente,  $i$  și  $j$ , sunt independente unele de altele, probabilitatea ca ele să apară simultan de mai mult de  $c_{ij}^*$  ori este dată de funcția de intercorelare pe intervalul de timp  $m$  din ecuația (6).

$$P\left(c_{ij}(m) \geq c_{ij}^* \mid -\frac{K}{\Delta t} \leq m \leq \frac{K}{\Delta t}\right) \approx 1 - \left(\sum_{l=0}^{c_{ij}^*-1} \frac{e^{-\lambda} \lambda^l}{l!}\right)^{2K+1} . \quad (6)$$

## 5 Algoritmi de detecție a avariilor, diagnoză și acomodare

Raportul sintetizează și o problematică de mare interes și actualitate, atât pe plan teoretic, cât și pe plan practic, și anume, toleranța la defecte a sistemelor automate. Această direcție de cercetare s-a dezvoltat inițial cu precădere în domeniul aeronautic, având aplicații la conducerea automată a aeronavelor. Totuși, cercetările s-au extins și în alte domenii, de pildă, în industria chimică și petrochimică, energetica nucleară etc. Domeniul abordat a înregistrat o dezvoltare deosebită în ultimii ani, datorită importanței asigurării securității în funcționarea acestor sisteme.

Conducerea optimală robustă și ajustarea modelului admisibil oferă o perspectivă tolerantă la defecte asupra adaptării modelului pentru a coincide cu, sau a aproxima, un model de referință și oferă o abordare admisibilă pentru recuperarea după defecte. În context, model de referință poate fi modelul nominal, în absența defectelor. Este esențială asigurarea stabilității și robusteții, dar și a eficienței computaționale, pentru implementarea în timp real. Se discută o tehnică modificată de pseudo-inversă și se compară rezultatele cu cele obținute folosind metoda cu urmărire a modelului optimal robust, bazată pe un indice de performanță cu factor exponențial, asigurând un grad de stabilitate impus. Tehnica propusă garantează stabilitatea sistemului post-avarie. Se consideră atât acomodarea progresivă la defecte, cât și aproximarea admisibilă și optimală.

## 5.1 Detecția avariilor

Orice sistem fizic construit de om și, cu atât mai mult, un sistem complex, poate să se defecteze. În anumite cazuri, defectarea poate avea consecințe dezastruoase. De aceea, detecția, izolarea și identificarea/estimarea defectelor și *conducerea tolerantă la defecte/avarii* (Fault Tolerant Control—FTC) sunt funcționalități necesare pentru asigurarea operării adecvate și în condiții de siguranță a sistemelor automate complexe. Folosind abordarea FTC un sistem automat poate poseda proprietățile dorite atât în funcționarea normală, cât și în prezența avariilor. Proprietăți dorite des folosite sunt *concordanța modelului* (model matching) sau *urmărirea modelului* (model following). Concordanța modelului presupune că matricea de transfer a sistemului în buclă închisă coincide cu matricea de transfer în buclă închisă a unui *model de referință*. Ideal, urmărirea modelului înseamnă identitatea traiectoriilor de stare ale sistemului și modelului. Nici una dintre aceste proprietăți nu poate fi obținută în general și sunt necesare aproximații.

Sistemele tolerante la defecte și sistemele optimele reconfigurabile au constituit subiectul unor cărți recent publicate și al multor articole apărute în reviste științifice prestigioase [1],[11]–[13],[20, 21, 23, 26, 27],[39]–[42], [47, 48].

Conducerea avansată a unui proces implică acumularea continuă de date de măsură a variabilelor procesului, cât și transmiterea, memorarea și prelucrarea lor, pentru supraveghere (monitorizare), modelare, predicție, administrarea mai bună a resurselor etc. Sunt parcurse diverse etape: achiziția și memorarea datelor, prelucrarea primară, extragerea informației, agregarea datelor, modelarea etc. Trebuie luate în considerare diverse aspecte, incluzând eterogeneitatea, timpul de prelucrare, securitatea și caracterul privat al datelor, interacțiunea cu operatorul uman. Extragerea informației și cunoașterii din mulțimi de date folosește actualmente tehnici de căutare a formelor sau tendințelor (data mining) sau metode de învățare statistice. O metodă care capătă o mare reputație, în special în contextul prelucrării volumelor mari de date (“Big data”) este *învățarea în adâncime* (“deep (machine) learning”), numită și *învățare ierarhică*. Se utilizează niveluri de prelucrare multiple, cu structuri complexe, compuse din reprezentări cu transformări neliniare (implementate, de pildă, în rețele neuronale). Această abordare permite extragerea și învățarea eficientă și nesupravegheată sau semi-supravegheată a trăsăturilor.

*Observațiile* (statistice) *deviante*, clar diferite ca valoare de altele dintr-un eșantion, numite în continuare *șianomalii* (outliers), semnifică uneori riscuri sau oportunități. Sunt necesare mijloace pentru detecția lor. O abordare posibilă se bazează pe testarea ipotezelor statistice. Observațiile deviate pot fi cauzate de abateri trecătoare în cursul achiziției datelor, datorate funcționării necorespunzătoare a aparatelor de măsură, zgomotelor (inclusiv pe canalele de transmitere), sau schimbărilor abrupte ale naturii sau comportării procesului. Aceste observații trebuie eliminate, de pildă, dacă se dorește modelarea funcționării normale a unui proces, dar trebuie analizate când ar putea semnala o comportare anormală, posibil conducând la regimuri de funcționare critice, periculoase sau inacceptabile. Modelarea observațiilor deviate și abstractizarea problemei detecției acestora au trei componente importante: nivelul informației disponibile despre comportarea normală și deviantă, tipul deviațiilor și criteriul pentru identificarea acestora. Abordările existente pentru detecția anomaliilor pot fi clasificate în patru grupe: supervizate, semisupervizate, nesupervizate și complet universale. Abordările supervizate sunt aplicabile când sunt disponibile modele atât pentru observațiile normale, cât și pentru cele deviate, ceea ce este posibil pentru date statice sau modele lent variabile în timp. Această clasă include abordări bazate pe clasificare, rețele neuronale, Bayes și support vector machines (SVM). Abordările semisupervizate folosesc doar un model, fie al datelor normale (în majoritatea cazurilor), fie al celor deviate. Abordările nesupervizate nu utilizează nici o ipoteză despre modelele datelor; exemple sunt abordările discriminative, abordări parametrice și prelucrarea analitică on-line. Abordările complet universale construiesc reguli de decizie cu singura

ipoteză că distribuțiile normală și deviantă sunt diferite.

O prezentare recentă a problematicii de mai sus pentru seturi mari de date se găsește în [38]. Instrumentul principal folosit este testarea ipotezelor statistice.

## 5.2 Algoritmi de reconfigurare și acomodare post-avarie

Mai des folosite au fost două politici pentru FTC, reconfigurarea sistemului și acomodarea la defecte, depinzând de excluderea sau, respectiv, includerea subsistemelor afectate de defecte din/în proiectarea noii legi de comandă, aplicată post-avarie. Obiectivul este fie minimizarea unui criteriu de optim integral (urmărirea modelului), fie aproximarea cât mai bună a modelului ideal (ideal model matching). Una dintre soluții este schema de “acomodare” progresivă pentru problema de conducere liniar-pătratică post-avarie și “suprapunerea” simplificată și extinsă peste modelul optimal robust pentru problema de urmărire a modelului optimal post-avarie [7]–[13].

Se consideră un sistem deterministic liniar invariant în timp (LTI) a cărui comportare nominală, în absența perturbațiilor și avariilor, este descrisă de modelul de stare [13]

$$M_n : \quad \dot{x}(t) = A_n x(t) + B_n u_n^*(t), \quad x(0) = x_0, \quad (7)$$

unde  $A_n \in \mathbb{R}^{n \times n}$  and  $B_n \in \mathbb{R}^{n \times m}$  sunt matricele nominale de stare și, respectiv, de comandă,  $x(t)$  și  $u_n^*(t)$  sunt vectorii de stare și de comandă la momentul  $t$ , iar  $x_0$  este starea inițială. Pentru simplitate, se presupune că vectorul de ieșire a sistemului,  $y(t)$ , coincide cu  $x(t)$ . Se consideră, de asemenea, că în funcționarea normală, se folosește legea de comandă cu reacție de la stare

$$u_n^*(t) = -K_n^* x(t), \quad (8)$$

astfel încât sistemul în buclă închisă, descris de formula

$$M_n^* : \quad \dot{x}(t) = \widehat{A}_n^* x(t), \quad \widehat{A}_n^* = A_n - B_n K_n^*, \quad (9)$$

este stabil și are comportarea dinamică dorită. Modelul  $M_n^*$  din (9) este considerat *modelul de referință*, a cărui dinamică ar trebui reprodusă cât mai fidel posibil în orice alte regimuri de funcționare, diferite de funcționarea ideală. În plus, se impune asigurarea unui *grad de stabilitate* dorit,  $\alpha_n^* > 0$ , adică  $\Re(\lambda_i) < -\alpha_n^*$ ,  $i = 1, \dots, n$ , unde  $\lambda_i$  sunt *polii* sistemului în buclă închisă (valorile proprii ale matricei de stare a acestui sistem).

Apariția unei modificări parametrice a sistemului, de pildă, ca urmare a unei defecțiuni, sau a schimbării regimului de funcționare, face ca modelul nominal (7) să nu mai fie adecvat. Fie

$$M : \quad \dot{x}(t) = Ax(t) + Bu(t), \quad (10)$$

modelul identificat sau estimat de un algoritm de detecție, izolare și identificare/estimare după producerea schimbării înregistrate în proces, și fie legea de comandă

$$u(t) = -Kx(t). \quad (11)$$

Realizarea concordanței exacte cu (9) a sistemului în buclă închisă definit de (10) și (11) ar presupune îndeplinirea condiției:

$$\widehat{A} := A - BK = \widehat{A}_n^*, \quad (12)$$

ceea ce atrage cerința ca subspațiul liniar generat de coloanele matricei  $\widehat{A} - \widehat{A}_n^*$  să fie inclus în subspațiul liniar generat de coloanele matricei  $B$ . Evident, această cerință nu poate fi satisfăcută

în general. Ca atare, problema concordanței (exacte) nu are soluție în general. Se poate obține o soluție aproximativă, folosind pseudo-inversa Moore-Penrose,

$$K^{\text{PI}} = B^\dagger(A - \widehat{A}_n^*). \quad (13)$$

În practică, nu se calculează pseudo-inversa  $B^\dagger$ , ci se rezolvă, în sensul celor mai mici pătrate, sistemul de ecuații algebrice liniare  $BK = A - \widehat{A}_n^*$ , folosind, de pildă, descompunerea după valorile singulare ale matricei  $B$ . Dar rezolvarea aproximativă a problemei concordanței modelului poate însă conduce la sisteme în buclă închisă instabile.

De fapt, nu este necesară impunerea condiției de concordanță a modelelor. Comportarea dinamică a sistemului definit de (7) și (8) este complet descrisă de spectrul matricei  $\widehat{A}_n^*$ ,  $\Lambda(\widehat{A}_n^*)$ . Dacă sistemul (10) este controlabil, se poate obține o matrice  $K$  astfel încât  $\widehat{A}$  din (12) să aibă spectrul  $\Lambda(\widehat{A}_n^*)$ , folosind algoritmi de *alocare a polilor*. Un studiu recent al unor algoritmi de alocare *robustă* a polilor (asigurând, de pildă, sensibilitate redusă la perturbații ale matricelor sistemului) este [30]. O deficiență a algoritmilor de alocare a polilor este că matricea de reacție  $K$  poate avea normă mare, ceea ce poate determina valori posibil inacceptabil de mari ale elementelor vectorului de comandă  $u(t)$ . Reducerea normei matricei  $K$  se poate efectua folosind proceduri de optimizare, fără a reduce semnificativ robustețea. Alternativ, se poate renunța la plasarea exactă a polilor în locații predefinite și realiza plasarea lor în regiuni specificate ale planului complex  $\mathbb{C}$ , de pildă, la stânga abscisei  $-\alpha_n^*$ , asigurând astfel gradul de stabilitate dorit. Acest obiectiv poate fi atins rezolvând o problemă de optimizare liniar-pătratică cu ponderare exponențială în criteriul de optim.

O soluție a problemei menționate mai sus în cazul  $m = 1$  este prezentată în [13]. Se utilizează metoda modificată a pseudo-inversei. Componentele vectorului de reacție corespunzător,  $K^{\text{MPI}} = (k_j^{\text{MPI}}, j = 1, \dots, n)$ , sunt definite astfel:

$$k_j^{\text{MPI}} = \begin{cases} k_j^{\text{PI}}, & \text{adică } |k_j^{\text{PI}}| \leq \widetilde{\delta}^{\text{PI}}, \\ \text{sgn}(k_j^{\text{PI}})\widetilde{\delta}^{\text{PI}}, & \text{în caz contrar,} \end{cases} \quad (14)$$

unde  $\widetilde{\delta}^{\text{PI}}$  reprezintă marginea de stabilitate robustă trunchiată, calculată cu formula  $\widetilde{\delta}^{\text{PI}} = \delta^{\text{PI}} - \epsilon^{\text{PI}}$ , pentru o valoare mică  $\epsilon^{\text{PI}}$ , iar  $\text{sgn}(\cdot)$  reprezintă funcția semn a unui argument real. Calculul lui  $\delta^{\text{PI}}$  este discutat în [13].

Formulele (14) sunt motivate de faptul că, dacă sistemul în buclă închisă folosind reacția de stare cu vectorul  $K^{\text{PI}}$  din (13) este instabil, utilizarea vectorului  $K^{\text{MPI}}$  din (14) îl face stabil, prin definiția marginii de stabilitate; într-adevăr, sistemul închis devine

$$\dot{x}(t) = \widehat{A}^{\text{MPI}}x(t), \quad \widehat{A}^{\text{MPI}} := A - BK^{\text{MPI}} = A - \delta^{\text{PI}}BK^{\text{PI}}, \quad (15)$$

și se poate arăta că  $|k_j^{\text{MPI}}| \leq \widetilde{\delta}^{\text{PI}}$  garantează stabilitatea.

Soluția propusă în [13] pentru acomodarea post-avarie folosește stabilizarea determinată prin rezolvarea unei ecuații algebrice Riccati continue generalizate (corespunzătoare unui criteriu de optimizare integral matrice de ponderare determinate adecvat și asigurând o margine de stabilitate dorită) și o corecție cu metoda modificată a pseudo-inversei. Pentru rezolvarea ecuației Riccati se poate folosi o metodă iterativă de tip Newton [36, 37], care are avantajul că are o convergență pătratică în vecinătatea soluției, iar iteratele sunt stabilizatoare (începând de la a doua). Prima proprietate permite obținerea soluției în două-trei iterații, dacă procedura Newton este inițializată adecvat (de exemplu, folosind soluția ecuației Riccati pentru regimul de funcționare anterior); în acest caz, nu este necesară adoptarea variantei cu căutare liniară a metodei Newton. Ultima proprietate permite folosirea unei soluții suboptimale, obținute înaintea terminării procesului iterativ. Ambele proprietăți fac ca timpul de calcul să fie relativ redus, astfel încât să fie acceptabil pentru o conducere în timp real.

## 6 Concluzii

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

Implementarea aplicațiilor de conducere automată avansată a proceselor industriale necesită uzual resurse sporite de stocare și execuție. Raportul a prezentat o aplicație bazată pe cloud care îi permite unui utilizator să aibă acces la diferite strategii de conducere folosind o interfață web, să beneficieze de regulatoare virtualizate, care să execute acei algoritmi și să trimită comenzi dispozitivelor din proces. Raportul a definit cum modulele implementând aceste noi servicii pot fi interconectate utilizând interfețe RESTful. Abordarea inovativă prezentată presupune virtualizarea unor “baze de date” de regulatoare de proces și îi conferă inginerului automatist dintr-o întreprindere industrială accesul la strategii avansate de modelare, optimizare și conducere, implementate ca funcții bloc IEC 61499.

Au fost investigate, de asemenea, aspecte referitoare la managementul situațiilor de risc, cât și pentru detecția avariilor, diagnoză și acomodare.

Etapa II nu a presupus diseminarea explicită a unor rezultate. Totuși, au fost publicate cinci lucrări elaborate de unii membri ai echipei proiectului (una dintre ele cu alți doi colaboratori, care nu fac parte din echipă), iar o altă lucrare va fi prezentată la o conferință care va avea loc la Budapesta în luna decembrie 2015 și va fi publicată de către WSEAS (World Scientific and Engineering Academy and Society). Lucrările publicate deja au fost prezentate la conferințe internaționale co-sponsorizate de IEEE (Institute of Electrical and Electronics Engineers) și sunt incluse în reputata colecție IEEE Xplore Digital Library. Volumele de lucrări sunt în curs de indexare de către ISI Thomson (ISI Proceedings). Toate cele șase lucrări menționează într-o secțiune de mulțumiri suportul financiar parțial din proiectul CALCULOS.

## Bibliografie

- [1] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki, editors. *Diagnosis and Fault Tolerant Control*. Springer-Verlag, 2006.
- [2] T. Bray. The JavaScript object notation (JSON) data interchange format, 2014. [Online]. Available: <http://tools.ietf.org/html/rfc7159>. [Accessed: 22-Nov-2015].
- [3] Center for Chemical Process Safety, American Institute of Chemical Engineers. Introduction to inherently safer design, 2009.
- [4] O. Chenaru, G. Stamatescu, I. Stamatescu, and D. Popescu. Towards cloud integration for industrial wireless sensor network systems. In *Proceedings of the 9th International Symposium on Advanced Topics in Electrical Engineering, 7-9 May, 2015, Bucharest, Romania*, pages 917–922, 2015.
- [5] O. Chenaru, A. Stanciu, D. Popescu, G. Florea, V. Sima, and R. Dobrescu. Modeling complex industrial systems using cloud services. In *Proceedings of The 20th International Conference on Control Systems and Computer Science (CSCS20), May 27-29, 2015, Bucharest, Romania*, pages 565–571, 2015.
- [6] O. Chenaru, A. Stanciu, D. Popescu, G. Florea, V. Sima, and R. Dobrescu. Open cloud solution for integrating advanced process control in plant operation. In *Proceedings of the 23rd Mediterranean Conference on Control and Automation (MED 2015), Torremolinos, Spain, June 16th-19th, 2015*, pages 973–978, 2015.
- [7] B. Ciubotaru and M. Staroswiecki. Fault tolerant control of the B747 short-period mode using progressive accommodation. In *Proceedings of the 2006 IEEE International Conference on Control Applications, Munich, Germany, October 4-6*, pages 3288–3294, 2006.
- [8] B. Ciubotaru, M. Staroswiecki, and C. Christophe. Fault tolerant control of the boeing 747 short-period mode using the admissible model matching technique. In *Proceedings of the 6th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes, Tsinghua University, P.R. China, August 29 - September 1, 2006*, pages 819–824, 2006.

- [9] B. D. Ciubotaru. *Perspective on Fault Tolerance in Flight Control*. Phd thesis, Department of Automatic Control and Computer Science, Polytechnic University of Bucharest, Bucharest, Romania, Nov. 2009.
- [10] B. D. Ciubotaru and M. Staroswiecki. Anytime algorithm for parametric faults accommodation under handling quality constraints. In *Proceedings of the 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes, Barcelona, Spain, June 30 - July 3, 2009*, pages 887–892, 2009.
- [11] B. D. Ciubotaru and M. Staroswiecki. Comparative study of matrix Riccati equation solvers for parametric faults accommodation. In *Proceedings of the 10th European Control Conference, 23-26 August 2009, Budapest, Hungary*, pages 1371–1376, 2009.
- [12] B. D. Ciubotaru and M. Staroswiecki. Extension of modified pseudo-inverse method with generalized linear quadratic stabilization. In *Proceedings of the 2010 American Control Conference, Marriott Waterfront, Baltimore, MD, USA, June 30-July 02, 2010*, pages 6222–6224, 2010.
- [13] B. D. Ciubotaru, M. Staroswiecki, and N. D. Christov. Modified pseudo-inverse method with generalized linear quadratic regulator for fault tolerant model matching with prescribed stability degree. In *Proceedings of the 2011 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC), Orlando, FL, USA, December 12-15, 2011*, pages 1583–1588, 2011.
- [14] A. T. de Almeida, C. A. V. Cavalcante, M. H. Alencar, R. J. P. Ferreira, A. T. de Almeida-Filho, and T. V. Garcez. *Multicriteria and Multiobjective Models for Risk, Reliability and Maintenance Decision Analysis*, volume 231 of *International Series in Operations Research & Management Science*, Camille C. Price and Joe Zhu (Series editors). Springer, Heidelberg New York Dordrecht London, 2015.
- [15] FBDK client server — Google groups. [Online]. Available: <http://groups.google.com/forum/#!msg/fbdk/storw0Op3vo/J5S8q6nMAQUJ>. [Accessed: 22-Nov-2015].
- [16] G. Florea, O. Chenaru, D. Popescu, and R. Dobrescu. A fractal model for power smart grids. In *Proceedings of The 20th International Conference on Control Systems and Computer Science (CSCS20), May 27-29, 2015, Bucharest, Romania*, pages 572–577, 2015.
- [17] G. Florea, O. Chenaru, D. Popescu, and R. Dobrescu. Evolution from power grid to smart grid: Design challenges. In S. Caraman, M. Barbu, and R. Solea, editors, *Proceedings of the 19th International Conference on System Theory, Control and Computing, October 14-16, 2015, Cheile Gradistei - Fundata Resort, Romania*, pages 912–916, 2015.
- [18] O. Givehchi, H. Trsek, and J. Jasperneite. Cloud computing for industrial automation systems — A comprehensive overview. In *Proceedings of the 18th IEEE International Conference on Emerging Technologies and Factory Automation, 10-13 Sep 2013, Cagliari, Italy*, pages 1–4, 2013.
- [19] T. Goldschmidt, A. Jansen, H. Koziolk, J. Doppelhamer, and H. P. Breivold. Scalability and robustness of time-series databases for cloud-native monitoring of industrial processes. In *Proceedings of the IEEE 7th International Conference on Cloud Computing, June 27 - July 2, 2014, Alaska, USA*, pages 602–609, 2014.
- [20] I. Hwang, S. Kim, Y. Kim, and C. E. Seah. A survey of fault detection, isolation and reconfiguration methods. *IEEE Trans. on Control Systems Techn.*, 18(3):636–653, 2010.
- [21] R. Isermann. Trends in the application of model-based fault detection and diagnosis of technical processes. *Control Engineering Practice*, 5(5):709–719, 1997.
- [22] A. Jaya. Safety in overpressure relieving systems (Engineering design guideline). Technical report, KLM Technology Group, Practical Engineering Guidelines for Processing Plant Solutions, 81200 Johor Bahru, Malaysia, 2011. Rev: 01. Available: [http://kolmetz.com/pdf/EDG/ENGINEERING\\_DESIGN\\_GUIDELINE\\_safety\\_over\\_pressure\\_Rev\\_web.pdf](http://kolmetz.com/pdf/EDG/ENGINEERING_DESIGN_GUIDELINE_safety_over_pressure_Rev_web.pdf).
- [23] J. Jiang. Fault-tolerant control systems -- An introductory overview. *Acta Automatica Sinica*, 31(1):161–174, 2005.
- [24] JSON — Wikipedia, the free encyclopedia. [Online]. Available: <https://en.wikipedia.org/wiki/JSON>. [Accessed: 22-Nov-2015].
- [25] Kubernetes by Google. [Online]. Available: <http://kubernetes.io>. [Accessed: 22-Nov-2015].
- [26] J. Lunze and J. H. Richter. Reconfigurable fault-tolerant control: A tutorial introduction. *European J. of Control*, 14(5):359–386, 2008.
- [27] J. F. Magni, S. Bannani, and J. Terlouw, editors. *Robust Flight Control: A Design Challenge*. Springer-Verlag, 1997.
- [28] J. Nishiguchi and T. Takai. IPL2 and 3 performance improvement method for process safety using event correlation analysis. *Computers & Chemical Engineering*, 34(12):2007–2013, 2007.

- [29] L. A. Ocheană, O. I. Rohat, D. A. Popescu, and G. G. Florea. Library of reusable algorithms for Internet-based diagnose and control system. In *Proceedings of the 14th IFAC Symposium on Information Control Problems in Manufacturing, 23-25 May 2012, Bucharest, Romania*, Part 1, pages 1407–1412, 2012. Available: <http://www.ifac-papersonline.net/Detailed/53961.html>.
- [30] A. Pandey, R. Schmid, T. Nguyen, Y. Yang, V. Sima, and A. L. Tits. Performance survey of robust pole placement methods. In *Proceedings of the 2014 IEEE 53rd Annual Conference on Decision and Control (CDC 2014), Los Angeles, California, USA, December 15–17 2014*, pages 3186–3191. IEEE, 2014.
- [31] Pods. [Online]. Available: <http://kubernetes.io/v1.1/docs/user-guide/pods.html>. [Accessed: 22-Nov-2015].
- [32] C. L. Pritchard. *Risk Management. Concepts and Guidance*. CRC Press, Boca Raton, FL 33487-2742, fifth edition, 2015.
- [33] O. Rohat and D. Popescu. Remote web-based execution of IEC 61499 function blocks. In *Proceedings of the 6th International Conference on Electronics, Computers and Artificial Intelligence, Oct 23-25, 2014, Bucharest, Romania*, pages 33–38, 2014.
- [34] M. Rădulescu, S. Rădulescu, and C. Z. Rădulescu. *Modele Matematice pentru Optimizarea Investițiilor Financiare*. Editura Academiei Române, București, 2006.
- [35] Serialization — Wikipedia, the free encyclopedia. [Online]. Available: <https://en.wikipedia.org/wiki/Serialization>. [Accessed: 22-Nov-2015].
- [36] V. Sima. Efficient computations for solving algebraic Riccati equations by Newton’s method. In M. H. Matcovschi, L. Ferariu, and F. Leon, editors, *Proceedings of the 2014 18th Joint International Conference on System Theory, Control and Computing (ICSTCC 2014), October 17-19, 2014, Sinaia, Romania*, pages 609–614, 2014. ISSN 978-1-4799-4602-0.
- [37] V. Sima and P. Benner. Numerical investigation of Newton’s method for solving continuous-time algebraic Riccati equations. In J.-L. Ferrier, O. Gusikhin, K. Madani, and J. Sasiadek, editors, *Proceedings of the 11th International Conference on Informatics in Control, Automation and Robotics (ICINCO-2014), 1-3 September, 2014, Vienna, Austria*, volume 1, pages 404–409. SciTePress — Science and Technology Publications, Portugal, 2013. ISSN 978-989-758-039-0.
- [38] A. Tajer, V. V. Veeravalli, and H. V. Poor. Outlying sequences detection in large data sets. *IEEE Signal Proc. Mag.*, 31(5):44–56, Sept. 2014.
- [39] V. Venkatasubramanian, R. Rengaswamy, and S. N. Kavuri. A review of process fault detection and diagnosis. Part II. Qualitative models and search strategies. *Computers and Chemical Eng.*, 27(3):313–326, 2003.
- [40] V. Venkatasubramanian, R. Rengaswamy, S. N. Kavuri, and K. Yin. A review of process fault detection and diagnosis. Part III. Process history based methods. *Computers and Chemical Eng.*, 27(3):327–346, 2003.
- [41] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri. A review of process fault detection and diagnosis. Part I. Quantitative model-based methods. *Computers and Chemical Eng.*, 27(3):293–311, 2003.
- [42] M. Verhaegen, S. Kanev, R. Hallouzi, C. Jones, J. Maciejowski, and H. Smaili. Fault tolerant flight control – A survey. In C. Edwards, T. Lombaerts, and H. Smaili, editors, *Fault-Tolerant Flight Control – A Benchmark Challenge*, chapter 2, pages 47–89. Springer-Verlag, 2010.
- [43] V. Vyatkin. IEC 61499 as enabler of distributed and intelligent automation: State-of-the-art review. *IEEE Trans. Ind. Inform.*, 7:768–781, 2011.
- [44] YAML — Wikipedia, the free encyclopedia. [Online]. Available: <https://en.wikipedia.org/wiki/YAML>. [Accessed: 22-Nov-2015].
- [45] YAML ain’t markup language (YAML™) Version 1.2. [Online]. Available: <http://www.yaml.org/spec/1.2/spec.html>. [Accessed: 22-Nov-2015].
- [46] Q. Zhang, L. Cheng, and R. Boutaba. Cloud computing: State-of-the-art and research challenges. *Journal of Internet Services and Applications*, 1(1):7–18, 2010.
- [47] Y. Zhang and J. Jiang. Bibliographical review on reconfigurable fault-tolerant control systems. *Annual Reviews in Control*, 32(2):229–252, 2008.
- [48] A. Zolghadri. The challenge of advanced model-based FDIR techniques for aerospace systems: The 2011 situation. In *Proc. 4th European Conference for Aerospace Sciences*, 2011.