UNIVERSITATEA DIN PETROȘANI
ȘCOALA DOCTORALĂ

TEZĂ DE DOCTORAT

CONDUCĂTOR DE DOCTORAT:
Prof. univ. dr. Marius Cioca

STUDENT DOCTORAND:
Schuszter Ioan Cristian

2024

UNIVERSITATEA DIN PETROȘANI
ȘCOALA DOCTORALĂ

UTILIZAREA METODELOR DE ÎNVĂȚARE AUTOMATĂ ȘI BIG DATA PENTRU ÎMBUNĂTĂȚIREA CALITĂȚII SISTEMELOR CRITICE

CONDUCĂTOR DE DOCTORAT:
Prof. univ. dr. Marius Cioca

STUDENT DOCTORAND:
Schuszter Ioan Cristian

2024

1 Rezumat

Această teză de doctorat explorează arhitecturile sistemelor critice, având ca scop îmbunătățirea calității și fiabilității acestora prin cercetare exhaustivă și metodologii inovatoare. Cuprinzând 119 pagini și structurată în șase capitole, această lucrare pornește într-o călătorie de explorare, analiză și implementare, având ca scop final avansarea ingineriei sistemelor critice.

Capitolul 1 oferă un studiu comparativ al arhitecturilor sistemelor existente, oferind perspectiva asupra punctelor lor forte și slabe. Acest lucru pregătește terenul pentru explorările ulterioare, oferind cunoștințe fundamentale.

Capitolul 2 este secțiunea de trecere în revistă a literaturii din această teză și investighează diverse aspecte ale ingineriei sistemelor, tehnicilor de învățare automată și sustenabilitate în sistemele software. Începe cu o revizuire cuprinzătoare care acoperă ingineria sistemelor fiabile, metodologiile de clasificare a textului în învățarea automată, tehnicile clasice de învățare automată și metodele moderne (profunde) de învățare automată. În plus, explorează sistemele software sustenabile, evidențiind importanța practicilor software conștiente de mediu.

În continuare, secțiunea de studii comparative analizează intersecția dintre fiabilitate și sustenabilitate în ingineria sistemelor, investighează abordările inteligente de monitorizare în timp real pentru sistemele de calcul și evaluează aplicarea învățării automate în predicția defecțiunilor și eforturile de sustenabilitate.

Capitolul 3 explorează metodologii și tehnologii noi pentru îmbunătățirea calității sistemelor critice, bazându-se pe o abordare multidisciplinară. Investighează metodele de cercetare inovatoare și arhitecturile pentru a aborda provocările în evoluție.

În **Capitolul 4**, accentul se mută către tehnicile de fiabilitate aplicate metodologiilor de inginerie software. Sunt examinate diverse strategii pentru îmbunătățirea fiabilității în dezvoltarea sistemelor critice din prisma inginerilor software implicați in creearea acestor sisteme.

Capitolul 5 investighează tehnici eficiente de monitorizare pentru gestionarea sistemelor software critice. Prin integrarea uneltelor și metodologiilor avansate de monitorizare, își propune să ofere inginerilor insight-uri pentru o operare fără probleme a sistemelor.

Capitolul 6 pune accentul pe dezvoltarea, implementarea și validarea unui ansamblu pentru îmbunătățirea calității sistemelor critice folosind tehnici de învățare automată și big data. Acest capitol explorează insight-uri bazate pe date și analize predictive pentru a oferi soluții inovatoare.

În cele din urmă, **Capitolul 7** încheie teza prin sintetizarea principalelor constatări, evidențiind contribuțiile personale și oferind perspective privind direcțiile viitoare de cercetare. Servește ca o mărturie a dedicării autorului pentru avansarea ingineriei sistemelor critice.

Dependența umană față de tehnologie a devenit tot mai pronunțată în ultimii ani, ajungându-se la situația în care o bună parte a vieții noastre depinde de interacțiunea cu software-ul, un exemplu recent fiind dependența educației și afacerilor de softul de videoconferință Zoom în timpul epidemiei mondiale de COVID-19 din 2020. O întrerupere a unuia dintre aceste servicii critice poate avea un impact foarte negativ asupra economiei, într-o durată scurtă de timp. Din aceste motive, găsirea și implementarea de soluții bazate pe tehnologii big data și machine

learning pentru îmbunătățirea fiabilității sistemelor ”critice” este un obiect important de vizat, ceea ce această teză își propune.

1.1 Introducere

Astăzi, sistemele software sunt adesea formate din diferite componente scrise într-o gamă largă de limbaje de programare, în funcție de scopul pe care îl are soft-ul, iar problema monitorizării calității acestora este exacerbată datorită naturii lor distribuite și simplitatea prin care acest lucru este realizat folosind un design de tip ”microservicii” și proliferarea tehnologiilor de izolare (containerizare) a proceselor sau de orchestrare a lor. O altă nevoie principală, adesea implementată într-un mod ineficient, este monitorizarea uniformă a tuturor acestor componente, identificarea defectelor cât mai rapidă și ulterior dezvoltarea unei capacități predictive în timp real pentru a evita defectele de la bun început. Există deja anumite lucrări ce tratează subiectul aplicării metodelor de inteligență artificială, în speță procesarea limbajului natural, pentru a identifica anomalii din log-urile anumitor sisteme, dar în principiu efectuate pe sisteme sintetice.

Această teză își propune, inițial, să identifice peisajul și componentele moderne necesare pentru construcția unui sistem software fiabil, cu un studiu de caz efectuat pe un sistem real de Single Sign-On și management al identității la CERN utilizat în producție la CERN. De asemenea, organizația face parte dintr-o federație a institutelor de cercetare și universităților de peste tot din lume, ceea ce face sistemul o verigă critică din tot acest mecanism. Sistemul poate fi considerat unul critic deoarece are un număr de aproximativ 16000 de utilizatori zilnici care depind de el pentru a putea să-și desfășoare activitatea. Se vizează în mod special tehnologiile de tip open source care pot fi utilizate pentru a construi un sistem fără a ne fi teamă de diverse aspecte precum proprietatea intelectuală pe care trebuie să o luăm în considerare, sau costurile plătite pentru licențe.

Teza este împărțită în 3 părți diferite, fiecare concentrându-se pe o fațetă a analizei sistemelor fiabile și îmbunătățirea lor utilizând tehnici moderne big data și de învățare automată. Cele 3 părți sunt dezvoltate în următoarele paragrafe:

1.2 Studiu privind metodele de învățare automată și big data

Această parte a tezei își propune studierea literaturii existente în jurul sistemelor fiabile, pregătind cititorul în domeniu prin expunerea ideilor principale care există în industrie, precum și mecanismele prin care fiabilitatea sistemelor poate fi influențată. Se vorbește despre sisteme distribuite și metodele moderne prin care acestea pot fi compuse. Apoi, se prezintă importanța unor sisteme de monitorizare și alertare conexe care sunt utilizate pentru a avea o viziune de ansamblu asupra sistemelor distribuite care trebuie puse în loc pentru a ne asigura că atingem un nivel de fiabilitate satisfăcător pentru un sistem care rulează în producție fără întreruperi majore în funcționarea sa.

Într-un final, se trece la studiul diferitelor metode de învățare automată prin care datele produse de un sistem și sistemul de monitorizare (adesea textuale), pot fi utilizate pentru a prezice

viitoare probleme care pot fi întâlnite într-un asemenea sistem. Scopul final al tezei este acesta: descrierea arhitecturii software ideale pentru un sistem considerat critic (așa cum pot fi privite cele de electricitate sau aprovizionare a apei), iar ultima pârghie de care ne putem folosi sunt aceste modele predictive care ne permit să anticipăm și să corectăm sistemul înainte ca acesta să eșueze. Cele mai bune metode de implementare ale acestor sisteme sunt identificate prin utilizarea unor studii de caz, unde sunt puse în lumina cercetării diferitele compromisuri ce trebuie realizate în dezvoltarea acestor sisteme.

1.3 Cercetări exploratorii, metode și tehnici de îmbunătățire a calității sistemelor critice

Cea de-a doua parte a tezei se bazează mai mult pe cercetări exploratorii asupra a ceea ce face un sistem distribuit critic să fie superior celorlalte, care suferă adesea de probleme de scalabilitate, sunt greoaie și se auto-vindecă mult prea încet sau deloc. Concret, putem împărți această secțiune în altele 3, mai granulare. În primul rând, ne concentrăm asupra modul în care se poate dezvolta punctul cel mai de jos într-un sistem fiabil: partea care se ocupă de date. Sunt prezentate diferite metode prin care fiabilitatea la nivelul bazei de date este realizată, mai ales în cazul bazelor de date relaționale, care sunt notoriu de greu de crescut peste un anumit nivel. Un studiu de caz este realizat și sunt identificate neajunsuri majore în momentul în care sistemul de stocare este distribuit în mai multe baze de date. Următoarea direcție de cercetare o constau bazele de date nerelaționale, mai ales cele care stochează datele în memorie pentru acces rapid. O implementare a unei baze de date ne-relaționale bazate pe un model de programare numit "actor model" este realizat și comparat cu Redis, una din cele mai folosite implementări de baze de date de acest tip din comunitatea IT. Performanța sistemului dezvoltat este net superioară, după cum poate fi observat și în graficele rezultante.

Un alt punct important de discuție este felul în care serviciile software dezvoltate de către echipă pot fi puse în producție într-un timp cât mai scurt, fără a fi disruptive față de funcționarea de zi cu zi a serviciilor și fără ca utilizatorii să experimenteze o degradare a serviciului utilizat (având în vedere că vorbim de un sistem care se concentrează pe logarea a 16000 de utilizatori în fiecare zi). Openshift este prezentat ca și soluție de deployment a serviciilor, cu un load balancer plasat în fața unui serviciu, astfel încât pentru un client acesta are un punct de intrare unic, când de fapt în spatele load-balancerului se află un număr potențial nelimitat de servere identice.

Odată adresată problema sistemului de stocare și a tehnologiilor care ne permit să scalăm sistemul într-un mod infinit, ne îndreptăm atenția către sistemele de monitorizare masive, care procesează o cantitate enormă de date în timp pseudo-real, astfel încât orice defecte pot ieși la suprafață cât mai rapid și să nu afecteze SLA-urile organizației (Service Level Agreement). O serie de tehnologii și metode de implementare sunt prezentate, având ca scop alegerea arhitecturii și sistemului ideal care poate să proceseze o cantitate suficientă de date fără probleme. Se justifică alegerea unei arhitecturi de ingestie de loguri bazată pe tehnologia open-source Flume, apoi utilizarea Prometheus și Grafana pentru a crea alerte și tablouri de bord pentru a avea o

vedere de ansamblu a întregii plaje de servicii. Sistemul dezvoltat este pus în funcțiune pentru sistemul de producție de la CERN și oferă alerte și o sursă de date continuă privind funcționarea diferitelor servicii software din sistem. Un studiu de caz este prezentat, concentrându-se pe rapiditatea recuperării sistemului dintr-o situație anormală.

1.4 Dezvoltarea, implementarea și validarea unui sistem de îmbunătățire a calității sistemelor critice bazat pe modele de învățare automată și big data

Ultima secțiune a acestei lucrări se concentrează pe aspectul de sisteme predictive, mai precis metodele și tehnicile care pot fi utilizate astfel încât performanța unui sistem să poată fi prezisă înainte ca acesta să sufere de comportament anormal. Problema începe de la analiza datelor, colectarea acestora și stocarea lor într-un format care permite apoi utilizarea lor pentru antrenarea unui algoritm de predicție. Sistemul propus se bazează pe ingestia datelor din sistemul Prometheus, astfel încât să fie agregată schimbarea în utilizarea procesorului pentru un anumit serviciu, la intervale de câte 5 minute distanță. Sistemul de analiză trebuie să proceseze aceste informații și să propună o valoare aproximativă la care serviciul va fi utilizat în următoarele 5 minute, pe baza tuturor datelor istorice. Pentru dezvoltarea și evaluarea completă a sistemului, un set de date colectat de-a lungul unei luni la CERN este utilizat.

Trei arhitecturi bazate pe învățare automată și rețele neuronale adânci sunt propuse pentru implementarea sistemului: CNN, RNN și LSTM. În același timp, pentru a fi siguri că rezultatele sunt concludente, aceste implementări sunt comparate cu un sistem bazat pe metode clasice de analiză a colecțiilor de date time-series, așa cum acestea sunt cunoscute: ARIMA. După evaluarea rețelelor neuronale dezvoltate pentru acest sistem, RNN iese ca un câștigător evident și este pus în comparație cu diferite euristici care ar putea fi folosite pentru același scop, dar este din nou un câștigător evident.

O ultimă analiză detaliată a lucrării se bazează pe posibilitatea de a utiliza acest sistem în mod inedit pentru a prezice utilizarea de CPU a serviciilor și beneficiilor în termeni de sustenabilitate a sistemelor care folosesc această metodă de învățare automată pentru a pune la dispoziție servere mai multe sau mai puține în funcție. În termeni realiști, poate fi observată o economisire a energiei de aproximativ 60% când sunt utilizați acești algoritmi, ceea ce este un rezultat foarte important.

2 Contribuții personale

Se subliniază următoarele contribuții personale (importante) pentru a evidenția conceptele de inovație propus în cadrul acestei teze:

1. Propunerea de considerații arhitecturale pentru un serviciu critic de producție utilizat la CERN, una dintre cele mai mari instituții de cercetare din lume. Lucrarea elucidază cadrul

arhitectural general și componentele cheie ale unui astfel de sistem scalabil, recomandând, de asemenea, tehnologii specifice pentru implementarea sa. În plus, abordează compromisurile implicate în trecerea de la o arhitectură monolitică la una distribuită, toleranta la defecțiuni. (capitolele 2,3)

2. Evaluarea și selectarea unui component al sistemului de monitorizare potrivit, cum ar fi un API REST, pentru implementare alături de fiecare element al sistemului monitorizat. Acest component a fost adaptat pentru a ingestiona și interpreta datele jurnalului în formate diverse, asigurând transmiterea eficientă pentru arhivare și procesare, în timp ce subliniază scalabilitatea pentru a gestiona creșteri potențiale în mesajele de jurnal. (capitolul 3)

3. Realizarea unei evaluări comparative a Apache Flume și Logstash ca soluții software candidat pentru ingestia și procesarea jurnalului. După analiză, Apache Flume a ieșit în evidență ca opțiunea preferată datorită utilizării reduse a memoriei, termenilor de licențiere mai flexibili și performanței comparabile cu cea a Logstash. (capitolul 3)

4. Identificarea unei probleme neabordate referitoare la gestionarea complexității post-implementare, în special în îndeplinirea Acordurilor de nivel de serviciu (SLA) necesare. Provocarea anticipată se învârtește în jurul creșterii timpului și efortului necesar pentru accesarea jurnalelor de serviciu pentru a diagnostica cauzele rădăcină ale erorilor. Deși nu este explorată în această lucrare, această problemă este întâlnită frecvent în arhitecturile distribuite. Viitoarele eforturi de cercetare vor explora soluții potențiale. (după cum este identificat în **capitolul 5**)

5. Implementarea unei baze de date eficiente NoSQL folosind modelul actor. Această abordare de proiectare a bazelor de date valorifică caracteristicile sale de concurență și scalabilitate pentru a îmbunătăți performanța și rezistența, oferind o soluție promițătoare pentru gestionarea datelor la scară largă în sisteme distribuite. (așa cum este propus în **capitolul 3**)

6. Implementarea unui sistem automat de predicție a utilizării resurselor, facilitată de instituirea unei infrastructuri de monitorizare robuste. Acest sistem permite colectarea de date de înaltă calitate, esențiale pentru scalarea operațiilor în mod eficient. (așa cum este identificat în capitolul 4)

7. Dezvoltarea mai multor algoritmi de învățare automată și de învățare profundă destinate validării ipotezei de predicție eficientă a utilizării resurselor. (capitolul 6)

8. Integrarea mecanismelor avansate de detectare a anomaliei, îmbunătățind capacitatea sistemului de a identifica și atenua problemele potențiale înainte ca acestea să escaladeze.

9. Implementarea framework-urilor de analiză a datelor în timp real, permițând luarea rapidă a deciziilor și optimizarea strategiilor de alocare a resurselor.

În plus, în capitolul 4, au fost luate în considerare aspecte mai ”umane” ale procesului de dezvoltare, deoarece resursele umane fac, de asemenea, parte din sistem. Prin urmare, unele constatări și contribuții cheie pe care le-am propus au fost:

1. Am ilustrat influența considerabilă a metricilor robuste și a metodologiilor stabilite, cum ar fi SAFe, asupra performanței echipei și a predictibilității în cadrul sistemelor software critice.

2. Am subliniat importanța armonizării eforturilor de dezvoltare cu obiectivele organizaționale, prin ajungerea la un consens privind metricile de afaceri cheie între membrii echipei.

3. Am evidențiat beneficiile utilizării uneltelor automate în simplificarea eforturilor de suport și în pregătirea terenului pentru integrarea fără probleme a schimbărilor viitoare, reducând preocupările privind regresia.

4. Am dezvoltat instrumente automate care oferă feedback instantaneu pentru activitățile de codificare, îmbunătățind astfel eficiența și fiabilitatea în fluxurile de lucru de dezvoltare a software-ului.

5. Am pledat pentru practici eficiente de gestionare a echipei alături de încorporarea revizuirilor de cod și a testelor automate pentru a consolida predictibilitatea și pentru a minimiza sarcinile echipei.

6. Am avansat în înțelegerea și implementarea metodologiilor și tehnologiilor menite să consolideze fiabilitatea sistemelor software critice în cadrul ingineriei sistemelor.

3 Lucrări de cercetare în perioada de desfășurare a studiilor

În timpul desfășurării cercetării pentru această teză de doctorat, am contribuit la elaborarea a 11 lucrări distincte ca autor sau coautor, dintre care am fost autor principal pentru 7 dintre ele. Este demn de menționat că 4 dintre aceste lucrări au fost publicate în jurnale cotate ISI, iar în cadrul a 3 dintre acestea am avut rolul de prim autor. Celelalte lucrări au fost publicate în lucrările de conferință indexate în Web of Science sau alte baze de date.