Experiences with a Course on Scientific Communication in Computer Science

Danny Weyns

Katholieke Universiteit Leuven, Belgium
danny.weyns@kuleuven.be

Abstract. Scientific communication is a key skill in any academic education. At the Katholieke Universiteit Leuven, students of the final year of the Bachelor in Computer Science have a course dedicated to scientific communication. This report documents the results of this course organized in the academic year 2022-2023. The aim of the document is to provide insights for teaching staff in a particular approach to organize such a course. On the other hand, the document provides insights for future students on communicating scientific results. We start with outlining the approach with goals, learning activities, assignments, and evaluation. Then we reflect on experiences with the course that can be considered for future instances of such a course. Finally, we provide the work products produced by the students during the course.

1. Outline of the Approach

We start with outlining the goals of the course and briefly explain the learning activities to realize the goals. Then we explain the assignments. We conclude with evaluation and a brief reflection on the course.

1.1. Goals of the course

The overall aim of the course is to learn students to communicate about science both in their native language (Dutch) and English. Concretely, the students should be able:

- To learn the basic principles of scientific integrity;
- To learn the criteria to which scientific and popular science texts should comply;
- To apply the criteria for scientific and popular science texts when writing such texts;
- To be able to clearly identify and/or formulate scientific problems, the settings of the problems, scientific methods, approaches with evaluation results, validity of results, and open challenges;
- To apply a correct source reference as used in the scientific field;
- To communicate scientific and popular scientific results orally.

1.2. Learning activities

The learning activities are centered on:

- Exercises: principles of scientific integrity;
- Writing: starting from scientific material (e.g., a technical report) produce (1) a scientific paper and (2) a popular science text;
- Oral communication: present the scientific paper and/or the popular science text to the target audience (represented by fellow students).

By means of exercises, writing, and presentation assignments, students learn to communicate in writing and orally about their own discipline within computer science.

The course start with an introductory seminar. During this seminar the students are introduced to the course. The basis principles of scientific integrity are explained and how these principles should be applied in scientific communication. Finally, an explanation is given of the assignments to be carried out and the evaluation of the course.

1.3. Assignments

The course includes four main assignments:

- Obtain the certificate for scientific integrity.
- Writing a scientific text of three pages.
- Writing a six-page text for people in information technology industry.
- Present the papers.

Obtain the certificate for scientific integrity via an online assessment. The topics of the assessment include correct usage of citations to other work and own previous work, references to online material, plagiarism, reproducibility, author responsibilities, respect, among others. The assessment evaluates the student's understanding of these basic principles. Obtaining the certificate is a prerequisite to pass the course.

Writing a scientific text of three pages in Dutch (plus an extra page for references). The input to the assignment consists of: (1) a protocol[1] of a scientific study, and (2) an report with the analysis[2] results of the study. The text should be written using the standard IEEE style for conferences with double columns. The scientific article should consider the following parts: setting of the research, related work, research problem, methodology, contributions, evaluation, validity, conclusions and future work, references. In addition, a personal reflection can be added. The text can consider the study as a whole or zoom in on a part of it. After submission, the scientific text is reviewed both by a fellow student and by the docent. The criteria for the review are the accuracy with which the different parts are described. A final paper can then be submitted that implements the review feedback.

Writing a popular science text of six-page in English (plus extra page for references). This assignment is based on the same protocol and report. The target audience of the text is people active in information technology industry. Articles of the "IEEE Software" magazine[3] serve as guideline. The text can again consider the study as a whole or zoom in on a part of it. The quality of the text will be determined based on the quality criteria provided by IEEE Software.[4] After submission, the text is reviewed and revised like the first writing task.

Presentations. Finally, the students give presentations for the group of fellow students. The students form groups of two or three students to present either the three-pages scientific paper or the six-pages paper targeted at an audience from industry. Students provide presentations of 10 to 15 minutes and evenly divided the work between the students. After the presentations there is five minutes time for questions and answers. The presentation of the scientific paper should emphasize the scientific approach, while the presentation of the paper targeted at industry should highlight importance of the study for people in industry.

1.4. Evaluation

The evaluation is based on the activities performed during the course. It consists of three parts as follows:

- Certificate scientific integrity: go/no go
- Scientific text (three pages): 35%
- Text directed to industry (6 pages): 35%
- Presentations: 25%

---

[1] https://people.cs.kuleuven.be/danny.weyns/papers/SAinIndustryProtocol.pdf
[2] https://people.cs.kuleuven.be/danny.weyns/papers/SAinIndustryBasicAnalysis.pdf
[3] https://www.computer.org/csdl/magazine/so
[4] https://www.computer.org/csdl/magazine/so/2007/05/s5005/13rRUxYIMT1

1.5. Reflection

Developing skills for communicating science is essential to academic education. We presented our experiences with a course that specifically targets these skills. Putting communication of science in the centre of attention of a course has the advantage that specific attention can be given to different aspects, including integrity in communication, the target audience of the communication, and the essential parts of the communication. The commitment of the students in this course demonstrated the inherent interest of students to learn the skills to communicate. Yet, we believe that the skills for science communication should not be limited to one course but be a red threat throughout academic education. In a world where communication is pervading every aspect of our life, these skills have become more important as ever before. The recent pandemic showed the importance of science communication to the broader society; the challenges with climate change we are facing is another example. It is obvious that we will need to broaden the skill set of methods of communicating science in the future to align and evolve with the ways people communicate.

2. Papers and presentations

The remainder of this report contains the papers produced by the students at Kulak KU Leuven. All students provided consent to make their work public.

Self-Adaptation in de praktijk: een beknopte analyse

1st Ruben Anseeuw
Informatica
Wetenschap en Technologie
KULAK
Kortrijk, België
ruben.anseeuw@student.kuleuven.be

Samenvatting—Deze paper is een bespreking van een onderzoek door middel van een vragenlijst, gedaan door Danny Weyns et al. naar het praktisch gebruik van self-adaptive software in verschillende gebieden van de industrie . Doorheen deze paper zal gebruik gemaakt worden van de 'we' vorm om het onderzoek te bespreken. Voor een uitgebreidere en gedetailleerdere bespreking van het onderzoek, verwijzen we graag naar de originele paper [1].

In deze korte bespreking van het onderzoek zullen we eerst een korte inleiding geven over self-adaptation, vervolgens verschillende aspecten van het onderzoek kort bespreken, zoals onderzoeksvragen, onderzoeksmethode en de samengevatte resultaten. Uiteindelijk focussen we op de verschillen tussen kleine en grote bedrijven die opdoken tijdens het analyseren van de resultaten.

Introductie

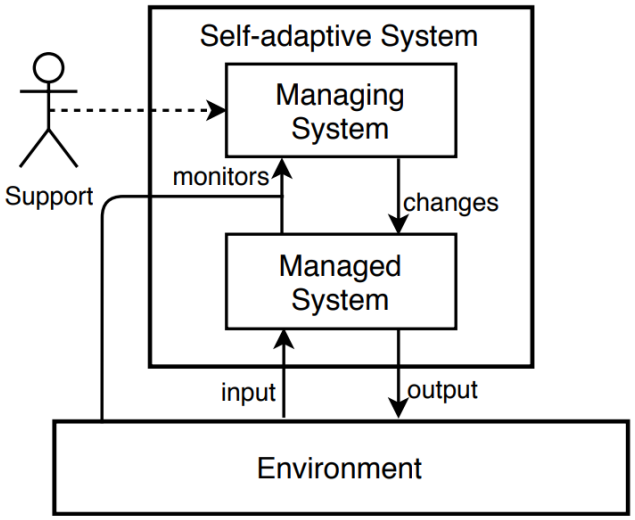
Self-adaptation is een relatief nieuwe vorm van software ontwikkeling, waar een veel onderzoek naar gedaan wordt en waarvoor de mogelijkheden nog lang niet allemaal verkend zijn. Het kan efficiënt kosten verminderen en prestaties verbeteren, maar is moeilijker te implementeren dan traditionele software.

Vandaar dat Danny Weyns et al. [1] een vragenlijst hebben opgestuurd naar een verspreide groep aan vakmensen binnen de industrie met kennis van zaken, en vervolgens de antwoorden grondig geanalyseerd en in kaart gebracht, om zo een zicht te krijgen of self-adaptation wel degelijk wordt gebruikt in de industrie, of het louter een academisch concept is.

1 Self-adaptation

Aangezien waarschijnlijk niet iedereen vertrouwd is met het concept van self-adaptation en om een eenduidige definitie te geven van een self-adaptive system, geven we hier een korte inleiding, zoals we ook gaven aan het begin van de enquête. [2]

Een self-adaptive system is elk software systeem dat uit 2 delen bestaat: een managing system en een managed system. Het managed system kan elke klassiek soort van software-intensief systeem zijn, of een deel van zo'n systeem. Het managing system neemt informatie uit de omgeving of enviroment en kan het managed system wijzigen, bijvoorbeeld om de prestaties te verbeteren wanneer de omgeving wijzigt. Deze 2-deligheid staat mooi afgebeeld op Figuur 1. Een veelgebruikt voorbeeld van een self-adaptive systeem is een Cloud met auto-scaling, die automatisch meer of minder opslagcapaciteit, netwerk en rekenkracht voorziet, zodat het zo goedkoop mogelijk kan blijven werken onder een veranderende vraag.



Figuur 1. schematische voorstelling van een self-adaptive system, overgenomen uit [2]

2 Onderzoeksvragen

Het algemene doel van onze studie was om beter het praktische gebruik van self-adaptation in kaart te brengen, alsook specifiek de motivatie erachter, hoe vakmensen het

gebruiken, welke problemen ze ondervinden en welke toekomstige mogelijkheden ze nog zien.

Deze onderzoeksvragen zijn opgedeeld in 4 categorieën, die ook in verschillende hoofdstukken van de enquête bevraagd werden:

- RQ1: Waarom wordt self-adaptation vooral gebruikt in de praktijk?
- RQ2: Hoe zouden vakmensen self-adaptation karakteriseren?
- RQ3: Hoe wordt self-adaptation toegepast in de software-intensieve systemen in de industrie-sector?
- RQ4: Wat zijn de meningen en opmerkingen over self-adaptation door vakmensen?

In de enquête werd ook nog een extra categorie ingevoerd, categorie 0, waar er inleidende vragen werden gesteld over bijvoorbeeld grootte van het bedrijf, functie erbinnen, het aantal jaren ervaring, enz. Zo kunnen we antwoorden linken aan kenmerken van bedrijven, wat we gaan gebruiken in de discussie van de resultaten.

3 Methodologie

De methode van onderzoek was een online vragenlijst, uitgestuurd naar 355 vakmensen in 21 landen. Deze personen zijn geselecteerd omdat ze actief zijn in het domeinen waar software-intensieve systemen gebruikt worden en hebben genoeg expertise om te antwoorden. Uiteindelijk zijn 184 enquêtes ingevuld, teruggezonden en opgenomen in deze studie.

Om objectiviteit te maximaliseren, hebben we alle open vragen geclassificeerd onder kernwoorden die niet op voorhand vastgelegd waren. Zo kunnen we toch statistiek toepassen op vrije tekst, zonder vooraf de antwoorden te beperken.

Een tweede methode die gebruikt is om een zo objectief mogelijk resultaat te bekomen, is de antwoorden laten classificeren door meerdere mensen. In deze studie is elke vraag geanalyseerd door een team van 2 of 3 mensen, die eerst elke vraag individueel gecategoriseerd hebben, en dan met elkaar hun resultaten vergeleken. Elk resultaat is vervolgens nog eens geclassificeerd door 2 andere researchers en indien nodig hebben deze nog eens samengezeten met het originele team om tot een consensus te komen.

4 Gerelateerd werk en bijdragen

In de paper [1] claimen we dat er al veel onderzoek is gedaan naar self-adaptation, maar nog nooit systematisch zoals in dit onderzoek gebeurd is. Dit onderzoek is dus uniek, maar komt natuurlijk niet uit de lucht vallen: enkele auteurs van deze paper schreven ook over de richtlijnen voor industrie-relevant onderzoek over self-adaptation [3] over self-adaptive software evalueren [4], maar ook door andere groepen, bijvoorbeeld een ander onderzoek via vragenlijst over self-adaptation, maar de onderzoekskant van het verhaal [5].

Hopelijk zal dit onderzoek kunnen bijdragen aan de verder ontwikkeling en onderzoek in het veld van self-adaptive software. Ook de praktijk-wereld van die software zouden we ook kunnen helpen met dit onderzoek.

Concreet hopen we een overzicht van het gebruik van self-adaptation te geven, inzicht te brengen voor researchers om hun onderzoek in het licht te zien van de industriële toepassingen, inzicht te brengen voor vakmensen om hun huidige niveau van gebruik van self-adaptive software te reflecteren, en nieuwe mogelijkheden aan het licht te brengen voor samenwerkingen tussen industrie en onderzoekswereld.

5 Resultaten

Uit de eerste categorie van vragen kunnen we concluderen dat self-adaptation wijd verspreid is in verschillende domeinen van de industrie en vooral gebruikt wordt om prestaties te verbeteren, de tevredenheid van gebruikers te vergroten en om kosten uit te sparen.

Uit de categorie van RQ2 kunnen we besluiten dat self-adaptive software in de praktijk zowel voor grote systemen als voor deelsystemen wordt gebruikt, voornamelijk voor auto-scaling, auto-tuning en monitoring. De elastische cloud en auto-scalers zijn de belangrijkste toepassingen die leiden tot het praktische gebruik van self-adaptation. Dit komt goed overeen met de theoretische inleiding die we gaven aan het begin van de vragenlijst en van deze paper, waardoor we kunnen besluiten dat de theoretische en praktische doelen grotendeels overeenkomen.

Uit de derde categorie van vragen kunnen we besluiten dat de meest gebruikte monitoring types resource usage en system load zijn, vaak gesignaleerd door sensors in de environment. Veelgebruikte mechanismes voor het realiseren van self-adaptation zijn data analyse en vergelijken met de thresholds, met als doel auto-scaling en reconfiguration. Vaak doen gebruikers beroep op tools voor self-adaptation zoals Kubernetes en AWS, om het ontwikkelen en onderhouden van systemen gemakkelijker te maken. Met hetzelfde doel worden dezelfde oplossingen vaak opnieuw gebruikt in andere systemen. De mens is echter nog niet altijd weg te denken uit het systeem, aangezien een toeziend oog buiten de machine een goed vangnet is en vertrouwen geeft om vaker self-adaptation te gebruiken.

In de vierde categorie hebben we gefocust op problemen en opmerkingen met self-adaptation. We kunnen besluiten dat een meerderheid van de vakmensen moeilijkheden ondervinden met het ontwerpen of onderhouden van self-adaptive systemen. Ongeveer de helft nemen risico's bij het gebruiken van deze systemen, voornamelijk omtrent foutieve functionaliteit, verminderde prestaties en duurdere kost. Ook ongeveer de helft van de gebruikers zouden bijstand willen van researchers bij hun problemen en zien nieuwe, toekomstige opportuniteiten voor self-adapting software, bijvoorbeeld in het veld van machine learning.

Grootte bedrijf	Tools gebruiken	eigen mechanismes
1-10	5 (56%)	4 (44%)
11-20	3 (27%)	8 (73%)
21-50	5 (36%)	9 (64%)
51-100	4 (40%)	6 (60%)
< 100	17 (39%)	27 (61%)
>100	7 (13%)	47 (87%)

Tabel 1
Kruisvergelijking van de grootte van bedrijven versus de mechanismes van self-adaptation

6 Discussie: Grootte bedrijven

Wanneer we de grootte van bedrijven, die we opgevraagd hebben bij Q0.3, kruisvergelijken met andere statistieken, kunnen we nieuwe uitspraken doen over verschillen tussen kleine en grote ondernemingen. Vooreerst definiëren we een groot bedrijf als een bedrijf met meer dan 100 werknemers, en een klein bedrijf met minder dan 100 werknemers.

Als eerste merken we op dat er geen groot verschil zit in de risico-situaties tussen grote en kleine ondernemingen: beide nemen soms risico's om met self-adaptive software te werken. Wel valt op te merken dat grote bedrijven vaker fouten rapporteren dan kleine bedrijven, nl. 47% van grote versus 9% van kleine ondernemingen.

Wanneer we de mechanismes vergelijken, zien we dat grote bedrijven veel vaker eigen oplossingen gebruiken voor problemen met self-adaptation, terwijl kleine vaker teruggrijpen naar beschikbare tools en infrastructuur. Dit wil niet zeggen dat grote bedrijven geen gebruik meer maken van tools, noch dat kleine bedrijven geen eigen oplossingen bedenken, maar het verschil is wel significant. De exacte gegevens kunnen nagelezen worden in tabel 1.

Dit is niet geheel onlogisch, aangezien grote bedrijven meer middelen hebben om zelf mechanismen te ontwerpen, en kleinere eerder oplossingen gaan zoeken in tools die al bestaan.

7 Threats to validity

Als eerste kunnen we ons afvragen of de vragen in de enquête duidelijk genoeg waren gesteld, zonder onopgemerkte misinterpretatie. Hiervoor bevatte de enquête bij de eerste 10, vooraf bepaalde deelnemers een aantal meta-vragen over de vragenlijst zelf: over de duidelijkheid van de terminologie, de relevantie en duidelijkheid van de vragen en de reikwijdte van de vragenlijst. Door deze feedback is de inleiding aangepast, de vragen zijn echter hetzelfde gebleven aangezien zo door iedereen als duidelijk werden vernomen.

Als laatste vraag (Q5.1) is aan iedereen gevraagd hoe zeker ze waren van hun antwoorden. Bijna iedereen gaf een positief tot sterk positief antwoord, wat de resultaten meer betrouwbaarheid geeft.

We kunnen ons ook afvragen of we onze resultaten mogen generaliseren, m.a.w. of onze steekproef goed genoeg gekozen was, aangezien we geen probabilistische selectiemethode gebruikt hebben. Toch hebben we zo willekeurig mogelijk personen gekozen om alle gebieden van self-adaptation aan bod te laten komen, vanuit verschillende landen en vanuit verschillende functies binnen verscheidene groottes van bedrijven. Om de expertise van de bevraagde personen te verzekeren, hebben we vragen gesteld over hun persoonlijke ervaringen met self-adaptive software.

8 Conclusie en verdere werken

We hopen dat door middel van deze studie we een beeld hebben kunnen schetsen van het praktische gebruik van self-adaptation in de industrie, om zo zowel vakmensen als onderzoekers te helpen hun werkwijzen en studies met elkaar te vergelijken en op elkaar of te stemmen.

We hopen ook om verder studies in dezelfde aard aan te moedigen, zodat onderzoek en praktijk steeds beter op elkaar wordt afgestemd. In het bijzonder is er nog de vraag of de industrie-sector niet alleen self-adaptive software gebruikt, maar ook of ze gebruik maken van de resultaten door studies.

Referenties

[1] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, Grace Lewis, Marin Litoiu, Angelika Musil, Juergen Musil, Panos Patros, and Patrizio Pelliccione. Self-adaptation in industry: A survey. *ACM Trans. Autonom. Adapt. Syst*, 1(1):43, November 2022.

[2] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, Grace Lewis, Marin Litoiu, Angelika Musil, Juergen Musil, Panos Patros, and Patrizio Pelliccione. Self-adaptation in industry – state-of-the-practice and opportunities. *ACM Trans. Autonom. Adapt. Syst*, 1(1):23, June 2020.

[3] Danny Weyns, Ilias Gerostathopoulos, Barbora Buhnova, Nicolás Cardozo, Emilia Cioroaica, Ivana Dusparic, Lars Grunske, Pooyan Jamshidi, Christine Julien, Judith Michael, et al. Guidelines for artifacts to support industry-relevant research on self-adaptation. *ACM SIGSOFT Software Engineering Notes*, 47(4):18–24, 2022.

[4] Ilias Gerostathopoulos, Thomas Vogel, Danny Weyns, and Patricia Lago. How do we evaluate self-adaptive software systems?: A ten-year perspective of seams. In *2021 International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 59–70. IEEE, 2021.

[5] Claes Wohlin, Aybuke Aurum, Lefteris Angelis, Laura Phillips, Yvonne Dittrich, Tony Gorschek, Hakan Grahn, Kennet Henningsson, Simon Kagstrom, Graham Low, et al. The success factors powering industry-academia collaboration. *IEEE software*, 29(2):67–73, 2011.

[6] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, Grace Lewis, Marin Litoiu, Angelika Musil, Juergen Musil, Panos Patros, and Patrizio Pelliccione. Analysis report survey on self-adaptation in industry. *ACM Trans. Autonom. Adapt. Syst*, 1(1):19, November 2022.

Analyse van het gebruik van zelf-adaptieve systemen in de industrie

Arthur Cremelie
Computerwetenschappen
KU Leuven KULAK
Kortrijk, België
arthur.cremelie@student.kuleuven.be

***Samenvatting*—Computersystemen en software spelen een cruciale rol in de huidige maatschappij. Die software wordt doorheen de tijd steeds complexer. Zelf-adaptieve systemen bevatten een feedback loop zodat taken die normaal worden uitgevoerd door operators, nu worden geautomatiseerd. Er is echter geen weet over wat de state-of-the-practice van zelf-adaptieve systemen is. Om een antwoord te bieden op waarom zelf-adaptieve systemen worden ingezet, welke problemen er worden opgelost en wat de risico's zijn, deden we een grootschalige survey (verder in de tekst hebben we het over *vragenronde*). In dit document analyseren we een aantal resultaten.**

***Index Terms*—Zelf-adaptatie, state-of-the-practice, vragenronde, industrie**

I. Introductie

In deze tekst beschrijven we een uitvoerige studie uitgevoerd in [2]. Software, die zo goed als overal aanwezig is in onze huidige samenleving, kan heel complex worden. Het raakt fabrieken, de gezondheidszorg, telecommunicatie en meer. Daarom is het cruciaal dat deze systemen robuust en betrouwbaar werken. Juist door de complexiteit van die systemen ontstaan er nieuwe uitdagingen en innovatieve benaderingen om de software-intensieve systemen in goede banen te leiden. Dit kan onder andere bereikt worden door het voorzien van een feedback loop. Deze feedback loop analyseert het systeem en zijn omgeving, redenen voor het gedrag van het systeem en zijn doelen. Het systeem voert waar nodig aanpassingen uit om de doelen te bereiken. De doelen kunnen heel wat zaken zijn, zoals het afhandelen van problemen veroorzaakt door externe factoren en het minimaliseren van de operationele kost onder een bepaalde werkdruk.

Academici hebben een groot aantal literatuurstudies en vragenrondes over zelf-adaptieve systemen uitgevoerd. Zie bijvoorbeeld [5]–[7]. Ook zijn er reeds een aantal industriële toepassingen van feedback loops te vinden in bijvoorbeeld elastische clouds en geautomatiseerd management [3], [4]. Daarom proberen we een beter zicht te verkrijgen over zelf-adaptatie in de industrie.

Volgens academisch onderzoek is zelf-adaptatie ontstaan om twee problemen op te lossen. Het eerste is het automatiseren van het management van complexe, software-intensieve systemen gebaseerd op hoog niveau doelen. Het tweede is het omgaan met operationele omstandigheden die moeilijk te voorspellen zijn voor de uitrol van het systeem. Er zijn vier belangrijke taken voor zelf-adaptatie: self-healing, self-optimisation, self-protection en self-configuration.

II. Gerelateerd werk

Wie zien dus dat zowel academici als de industrie bezig is met de principes van feedback controle. Het uitgevoerde onderzoek door academici is duidelijk gedocumenteerd, terwijl er over de toepassingen ervan in de industrie weinig geweten is. Voor zover we inzien, is er nog nooit eerder een grootschalig en goed gedocumenteerd onderzoek geweest als [2].

III. Onderzoeksvragen en methode

A. Onderzoeksvragen

Om een beter zicht te krijgen over de toepassingen van zelf-adaptatieve systemen in de industrie, proberen we een antwoord te vinden op de volgende opgestelde onderzoeksvragen:

- OV1 Wat drijft gebruikers om zelf-adaptatie toe te passen in software-intensieve industriële systemen?
- OV2 Hoe karakteriseren gebruikers zelf-adaptatie?
- OV3 Hoe passen gebruikers zelf-adaptatie toe in software-intensieve industriële systemen?
- OV4 Wat zijn de ervaringen van gebruikers bij het toepassen van zelf-adaptatie en zien zij opportuniteiten?

Vermits niet alle gebruikers de term zelf-adaptatie kennen, moeten we ze eerst een korte introductie geven over het begrip met een aantal concrete voorbeelden. Het doel van OV1 is om de motieven van de gebruikers van zelf-adaptatie te begrijpen en het type problemen zij hiermee oplossen. Het doel van OV2 is om te achterhalen wat de industrie begrijpt onder het concept *zelf-adaptatie*. Hoofdzakelijk vragen we ons af hoe gebruikers zelf-adaptatie beschrijven als iets dat een systeem in staat stelt zichzelf aan te passen tijdens gebruik. Hiervoor vragen we concrete voorbeelden over wat zij onder zelf-adaptatie verstaan. Aan de hand van OV3 proberen we te achterhalen hoe zelf-adaptatie wordt gebruikt in de industrie. De nadruk ligt hier vooral op de methodes, technieken, tools, benchmarks en processen gebruikt om zelf-adaptatie mogelijk te maken. Als laatste is het doel van OV4 om de moeilijkheden en risico's die de industrie ervaart te onderzoeken. Hier kijken we ook of onderzoek kan helpen bij deze problemen. Dit alles zal ons toelaten om de samenhang tussen de academische wereld en de industrie voor het concept zelf-adaptatie in kaart te brengen.

B. Onderzoeksmethode

De populatie voor ons onderzoek betreft gebruikers die actief meewerken in het uitwerken van industriële software-intensieve systemen in gelijk welk veld. Er werden 355 gebruikers van verschillende bedrijven gecontacteerd via het netwerk van onderzoekers betrokken bij [2]. Er werden twee voorwaarden opgesteld om gebruikers te bevragen. Eerst moesten we een goede representatie hebben van domeinen die gebruik maken van zelf-adaptatie. Ten tweede moesten gebruikers de nodige kennis hebben om de vragen te kunnen beantwoorden. We contacteerden de uitgenodigde gebruikers via gepersonaliseerde emails.

De vragenlijst bestond uit zowel gesloten als open vragen. Bij gesloten vragen kunnen deelnemers enkel een vast aantal voorgedefinieerde opties selecteren. Bij open vragen daarentegen, kunnen deelnemers in detail en naar keuze antwoorden. De vragenronde werd aan de hand van een anonieme, online vragenlijst georganiseerd. Dit omdat we dan een grote set aan deelnemers konden bevragen aan een lage kost. De vragen werden door twee teamleden uit [2] opgesteld op basis van de onderzoeksvragen besproken in subsectie III-A. Daarna werden de vragen geverifieerd door de andere teamleden.

Om de vragenlijst te valideren deden we beroep op acht willekeurig gekozen deelnemers die aan onze criteria voldeden. De vragenlijst voor validatie bevatte ook extra vragen over de vragenlijst zelf. Zo konden we evalueren of de opgestelde vragen duidelijk, relevant en correct waren. Geen enkele deelnemer gaf aan dat een vraag zou moeten verwijderd of gewijzigd worden. Op basis van de gekregen feedback werd de introductie van zelf-adaptatie gewijzigd zodat deze minder bias bevat.

De vragenlijst bestond uit vijf delen. De details van alle vragen zijn te vinden in [1]. Het eerste deel bevatte vragen over of de deelnemer al dan niet zelf-adaptatie toepast en algemene demografische info. Het tweede deel bevatte vragen omtrent OV1. Zaken als problemen die worden opgelost met zelf-adaptatie, de bedrijfsmotivatie voor het toepassen van zelf-adaptatie en de voordelen behaald door het toepassen van zelf-adaptatie. Het derde deel van de vragenlijst bevatte een vraag over OV2. We vroegen om een concreet voorbeeld van een zelf-adaptief systeem te beschrijven waarmee hij of zij mee gewerkt heeft. Het vierde deel bevatte vragen omtrent OV3. We waren geïnteresseerd in hoe gebruikers zelf-adaptatie toepassen. Het vijfde en laatste deel bevatte vragen omtrent OV4 over de moeilijkheden, risico's, uitdagingen en opportuniteiten van zelf-adaptatie. Daarna werd de vragenlijst beïndigd met drie extra vragen over het vertrouwen die de deelnemers hadden bij het beantwoorden van de vragen en of ze nog iets wensten toe te voegen.

Om de data te analyseren werden twee verschillende aanpakken gebruikt. Voor de gesloten vragen werd beschrijvende statistiek en kantitatieve data analyse gebruikt. Voor de open vragen werd kwalitatieve data analyse gebruikt.

IV. Contributies

Het onderzoek omvat vier hoofdbijdragen:

- Een empirisch gegrond overzicht van state-of-the-practice toepassingen van zelf-adaptatie;
- Inzichten voor onderzoekers om hun huidig onderzoek te plaatsen in relatie met de industrie;
- Inzichten voor gebruikers om hun huidige graad van toepassing van zelf-adaptatie in perspectief te plaatsen;
- Bijkomende perspectieven voor het toepassen van zelf-adaptatie in de praktijk en opportuniteiten voor samenwerkingen tussen industrie en onderzoek.

V. Resultaten

A. Demografische informatie

Van de 355 uitgenodigden hebben 184 deelnemers de vragenlijst ingevuld (51.8%). 100 van de 184 deelnemers (54.4%) deelden mee dat ze ervaring hebben met het gebruik van zelf-adaptieve systemen. Als we kijken naar de software systemen gebouwd door de organisaties, behoren volgende typen tot 52.5% van alle systemen: web/mobile, embedded, cyber-physical, IoT systemen, data managemnt en cloud. Ongeveer de helft van de organisaties heeft meer dan 100 personeelsleden die werken als software engineer. De andere helft heeft een aantal dat gelijk is verdeeld tussen de 1 en 100. Verder hebben de deelnemers gemiddeld 1.6 rollen in hun organisatie. De deelnemers die zelf-adaptatie toepassen hebben gemiddeld 1.5 rollen, meestal als programmeur en project manager. Ongeveer een op de drie gebruikers werken als een ontwerper of architect. Als we kijken naar de jaren ervaring van de deelnemers die software engineer zijn, zien we dat een meerderheid minstens 9 jaar ervaring heeft (69.6 % van het totaal en 76.0% van de gebruikers).

B. Motieven voor zelf-adaptatie (OV1)

Deze subsectie analyseert de antwoorden voor OV1 over wat de motieven zijn om zelf-adaptatie toe te passen en welke problemen ze oplossen. Het blijkt dat zelf-adaptatie in de industrie hoofdzakelijk wordt gebruikt om de performantie van het systeem te optimaliseren en taken te automatiseren. Het wordt minder gebruikt om aan de slag te gaan met veranderingen in organisatiedoelen. Zelf-adaptatie wordt in verschillende domeinen ingezet om om te gaan met een veranderende omgeving. De bedrijfsmotivatie om zelf-adaptatie in te zetten zijn: het verbeteren van de gebruikerstevredenheid, kostenreductie en het vermijden van risico's. Daarnaast zien we dat de meest voorkomende voordelen van het inzetten van zelf-adaptatie bestaan uit een verbeterd nut en gebruik, besparingen, interactie met de mens en omgaan met dynamiek. De problemen die worden opgelost dankzij zelf-adaptatie zijn in het algemeen gelijk aan de onderzochte problemen door academici. Het belangrijkste verschil vinden we terug in het gebrekkig belang van gebruikers om zelf-adaptatie gebruiken om onzekerheden te vermijden, waar de focus ligt in onderoek. Als laatste keken we ook nog naar de taken die zelf-adaptatie invullen. Academici bestuderen 4 taken (self-healing, self-optimising, self-protecting en self-configuring), terwijl gebruikers daarnaast de nadruk leggen op het belang

van het verbeteren van de gebruikerservaring, kostenreductie en risicovermijding.

C. Kenmerken van zelf-adaptatie (OV2)

We trachten nu antwoorden te analyseren van OV2. We zien dat zelf-adaptatie wordt ingezet van volledige systemen tot delen van een systeem of ondersteunende systemen. Auto-scaling, auto-tuning en monitoring zijn dominante typen van zelf-adaptatie ingezet in de industrie. De oorzaken van adaptatie in de industriële software intensieve systemen zijn veranderingen in kenmerken van systemen en hun omgeving, verschillen in system load, het gebeuren van acties en acties ondernomen door gebruikers. Zelf-adaptatie vinden we het vaakst terug bij het gebruik van technologiën als elastische clouds, auto-scaling en containerisatie zoals Kubernetics.

D. Toepassingen van zelf-adaptatie (OV3)

Het gebruik van middelen en load op het systeem zijn de meest voorkomende zaken die worden gemonitord. Deze statistieken worden meestal verzameld door sensoren in de omgeving en in het systeem. Data analyse en vergelijkingen met grenswaarden zijn de meest gebruikte werkwijzen om analyses uit te voeren om zelf-adaptatie mogelijk te maken. Daarnaast zien we dat heel wat mechanismen worden gebruikt om zelf-adaptatie mogelijk te maken. De belangrijkste mechanismen zijn auto-scaling en reconfiguratie. We zien ook dat gebruikers sterk vertrouwen op systemen zoals AWS of Kubernetes voor het uitbouwen van zelf-adaptieve systemen. Daarbij hergebruiken gebruikers heel wat oplossingen in de vorm van code, ontwerp artifacten en specificaties bij de opbouw van de zelf-adaptieve systemen. Als we kijken naar hoe men vertrouwen heeft in de ontwikkelde industriële zelf-adaptieve systemen, zien we dat er enorm veel wordt getest, aan runtime monitoring en alerting wordt gedaan. Maar soms komt er ook nog menselijke supervisie bij kijken.

E. Moeilijkheden en opportuniteiten (OV4)

We bekijken de belangrijkste inzichten die we verkegen bij het analyseren van de antwoorden van OV4. Een meerderheid van de gebruikers heeft moeilijkheden bij het ontwerpen en onderhouden van zelf-adaptieve systemen. We zien deze moeilijkheden vooral bij het betrouwbaar ontwikkelen, de complexiteit van het systeem en debugging. Daarnaast komt ongeveer de helft van de gebruikers in aanraking met de risico's van zelf-adaptatie. Gebruikers hadden het meeste last van risico's die betrekking hebben op een incorrecte werking en de moeilijkheid om om te gaan met de omgevingsonzekerheid. Ook speelt een gedegradeerde performantie en hogere kost een rol. Ongeveer de helft van de gebruikers melden dat zij hulp van onderzoekers zouden appreciëren. Dit zou vooral nuttig zijn bij problemen gerelateerd aan het ontwerpen van zelf-adaptieve systemen en het beheer van de data. Nog eens de helft van alle deelnemers geeft aan dat zij opportuniteiten zien om zelf-adaptatie toe te passen. Zeker voor autonome operaties, databeheer en machine learning.

VI. Validatie

We hebben een aantal maatregelen genomen om mogelijke misinterpretaties te vermijden. Zelf-adaptatie werd ingeleid aan de hand van een standaard model met een feedback loop en daarbijhorende voorbeelden. De feedback van de vragenlijst voor validatie werd toegepast om vragen te verduidelijken. De deelnemers werden zodanig gekozen dat zij voldoende ervaring hadden. De deelnemers gaven mee dat zij voldoende vertrouwen hadden in hun antwoorden.

Om te vermijden dat de resultaten gegeneraliseerd worden en representatief zijn, hebben we deelnemers uitgenodigd uit het netwerk van [2]. Deelnemers moesten voldoende ervaring hebben. Uit de demografische resultaten blijkt dat dit het geval is. Dankzij de verschillende teams die in [2] het onderzoek hebben gevoerd, zitten we ook met een gebalanceerde populatie over de wereld heen.

VII. Discussie

A. Voordelen voor het gebruik van zelf-adaptatie

Het optimaliseren van performantie en het omgaan met de veranderende omgeving zijn de grootste problemen die worden opgelost dankzij zelf-adaptatie. Dit is vooral in het domein van embedded/cyber-physical/IoT. In de cloud gebruikt men zelf-adaptatie om taken te automatiseren en het systeem te reconfigureren. Het hergebruik van zelf-adaptieve systemen valt vooral op in de domeinen van productie, databeheer en embedded/cyber-physical/IoT. Er is geen specifiek artifact dat meer wordt hergebruikt dan een ander.

B. Moeilijkheden en risico's verbonden aan zelf-adaptatie

We kijken ook eens naar de moeilijkheden en risico's de worden ervaren bij zelf-adaptatie. Hier zien we dat vooral het ontwerpen van zelf-adaptatie een moeilijkheid is. De grootste risico's vinden we terug bij het ontwikkelen, de operations (werking) en de impact op organisaties. Grotere bedrijven ondervinden grotere risico's met betrekking tot fouten veroorzaakt door zelf-adaptieve systemen. De moeilijkheden bij het ontwerpen is voor iedereen een probleem. Echter is het debugging belangrijker voor grotere bedrijven. Hier tegenover staat dat kleinere en medium bedrijven hogere aandacht schenken aan de limieten van tools en methodes.

VIII. Conclusie

In deze paper analyseerden we het gebruik van zelf-adaptatie in de industrie aan de hand van 184 deelnemers verspreid over verschillende landen en domeinen. 100 van de 184 deelnemers hebben ervaring in het ontwerpen van zelf-adaptieve systemen. We zien dat zelf-adaptatie veel gebruikt wordt in de industrie door programmeurs, projectleiders, architecten en ontwerpers. Zelf-adaptatie wordt het vaakst toegepast bij auto-scaling, auto-tuning en monitoring. Daarnaast staan gebruikers zeker open voor hulp vanuit de academice wereld voor zekerheden en databeheer. Verder onderzoek kan gaan over autonome werking, het gebruik van machine learning en databeheer.

REFERENTIES

[1] D. Weyns, I. Gerostathopoulos, N. Abbas, J. Andersson, S. Biffl, P. Brada, T. Bures, A. Di Salle, M. Galster, P. Lago, G. Lewis, M. Litoiu, A. Musil, J. Musil, P. Panos en P. Pelliccione, "Analysis Report Survey on Self-Adaptation in Industry," https://people.cs.kuleuven.be/ danny.weyns/papers/SAinIndustryBasicAnalysis.pdf, accessed: February 2023.

[2] D. Weyns, I. Gerostathopoulos, N. Abbas, J. Andersson, S. Biffl, P. Brada, T. Bures, A. Di Salle, M. Galster, P. Lago, G. Lewis, M. Litoiu, A. Musil, J. Musil, P. Panos en P. Pelliccione, "Self-Adaptation in Industry: A Survey, " arXiv, 2022. [Online]. Available: https://arxiv.org/abs/2211.03116

[3] B. Beyer, C. Jones, N. Murphy, en J. Petoff, "Site Reliability Engineering, How Google Runs Production Systems, " O'Reilly Inc., 2016.

[4] A. Spyker, "Disenchantment: Netflix Titus, Its Feisty Team, and Daemons, "InfoQ, 9/2020. [Online]. Available: https://www.infoq.com/presentations/netflix-titus-2018/

[5] D. Weyns, U. Iftikhar, S. Malek, en J. Andersson, "Claims and supporting evidence for self-adaptive systems: A literature study, " International Symposium on Software Engineering for Adaptive and Self-Managing Systems, 2012.

[6] Z. Yang, Z. Li, Z. Jin, en Y. Chen, "A systematic literature review of requirements modeling and analysis for self-adaptive systems, " equirements Engineering: Foundation for Software Quality. Springer, 2014.

[7] D. Weyns, U. Iftikhar, D. de la Iglesia, en T. Ahmad, "A survey of formal methods in self-adaptive systems, " Fifth International C* Conference on Computer Science and Software Engineering, ser. C3S2E '12. NewYork,NY,USA:AssociationforComputingMachinery,2012, p. 67–79. [Online]. Available: https://doi.org/10.1145/2347583.2347592

[8] D. Gray, "Doing research in the real world." SAGE Publications Ltd., 2013.

Analyse van resultaten vragenlijst self-adaptation in de industrie

Matias Daneels
KU Leuven

Kortrijk, België
matias.daneels@student.kuleuven.be

***Samenvatting*—Software wordt steeds belangrijker in de hedendaagse wereld. Software wordt gebruikt in medische toepassingen, industriële toepassingen en zoveel meer. Self-adaptation systems zijn software systemen met een feedback loop dat taken automatiseert die anders door beheerders worden uitgevoerd. Voorlopig is het nog onduidelijk hoe vaak self-adaptation wordt gebruikt in de praktijk, noch wat de uitdagingen en problemen ermee zijn. Om hier duidelijkheid over te creëren hebben werd een vragenlijst samengesteld. Deze vragenlijst werd verspreid over 21 landen, met 184 ontvangen antwoorden.**

I. Introductie

Het gebruik van computer systemen is enorm toegenomen in de industrie in de voorbije decennia. Wanneer software centraal staat in het computer systeem spreken we over een software-intensief systeem [1]. De complexiteit van het beheer van deze systemen wordt steeds complexer omwille van de grote vraag van de systemen, onzekere toestanden, etc. Omwille van deze redenen wordt er constant gezocht naar alternatieven voor het beheer van software-intensieve systemen.

Een mogelijke oplossing is het gebruik van een feedback-loop. Deze gaat aan de hand van informatie over het systeem en de omgeving, het systeem beheren om de doelen van het systeem te bereiken. Dit wordt self-adaptation genoemd [2].

In universiteiten is er al veel onderzoek gedaan naar de implementatie, uitdagingen, mogelijkheden, etc. van self-adaptation. Deze ontwikkeling heeft zich ook voorgedaan in de industrie. Het gebruik, kennis, problemen, etc. van de implementatie van self-adaptation in de industrie is voorlopig nog onduidelijk. Om hier duidelijkheid over te krijgen hebben we een vragenlijst opgesteld en verspreid over 21 verschillende landen. Concrete richtlijnen zijn te vinden in [3]. De concrete vragen zijn te vinden in [4].

II. Gerelateerd werk

Bij onderzoekers is er al veel onderzoek gedaan over self-adaptation zoals de voordelen ervan [5], de vereisten voor self-adaptation [6], de onzekerheid bij self-adaptation [7] [8] en meer. Bij ons weten is er nog geen onderzoek gedaan naar het gebruik, uitdagingen, kennis, etc. over self-adaption systemen in de industrie.

III. Onderzoeksprobleem en methode

Met de vragenlijst proberen we antwoorden te vinden om volgende vragen:

OV1: Wat motiveert beoefenaars om self-adaptation te gebruiken in industriële software-intensieve systemen?

OV2: Hoe omschrijven beoefenaars self-adaptation?

OV3: Hoe implementeren beoefenaars self-adaptation in industriële software-intensieve systemen.

OV4: Wat zijn de ervaringen van beoefenaars met het implementeren van self-adaptation en zien ze mogelijkheden voor onderzoek?

Self-adaptation systemen zijn software-intensieve systemen dat zichzelf kunnen bijsturen tijdens uitvoering. Het bekendste voorbeeld is bij het gebruik van de cloud, wanneer de cloud capaciteit automatisch aangepast wordt naar de noden van de gebruiker.

Om antwoorden te vinden om de bovenstaande onderzoeksvragen gebruiken we data gehaald van een representatieve steekproef van de beoefenaars in de industrie. Het is zeer belangrijk dat de data representatief is voor de populatie. Om de data te bekomen maken we gebruik van een online vragenlijst en van interviews. De methoden zijn geselecteerd aan de hand van de richtlijnen van Lethbridge et al. [9] en Robson [10].

De vragenlijst bestaat uit zowel open vragen als gesloten vragen. Bij gesloten vragen zijn er enkele vooropgestelde antwoordmogelijkheden. Bij open vragen wordt aan de deelnemer gevraagd iets uit te leggen. We maken gebruik van een online vragenlijst om het aantal deelnemers te maximaliseren en de kosten de minimaliseren.

We maken ook gebruik van interviews omwille van de non-verbale communicatie en de mogelijkheid voor bijvragen. Bij een interview kunnen er bijvragen gesteld worden indien een antwoord niet duidelijk genoeg is.

De informatie behaald door de vragenlijst en interviews wordt samengevat en daar worden de conclusies uit getrokken.

IV. Resultaten

In het totaal waren er 184 deelnames aan de vragenlijst. We vatten de verkregen antwoorden kort samen per onderzoeksvraag.

A. Motivatie

OV1: Wat motiveert beoefenaars om self-adaptation te gebruiken in industriële software-intesive systemen? Om op deze vraag een antwoord te krijgen gebruiken we de antwoorden van 100 van de 184 deelnemers die duidelijk maakten dat ze werken met concrete self-adaptation systemen.

We merken dat self-adaptation vaak wordt gebruikt in de industrie in verschillende domeinen. Het wordt hoofdzakelijk gebruikt om resultaten zo goed mogelijk te bereiken, taken te automatiseren en voor het omgaan met veranderingen in de omgeving. Self-adaptation wordt ook gebruikt voor het opsporen en oplossen van errors en het configureren van systemen.

De hoofdzakelijke motivatie van organisaties om self-adaptation te gebruiken zijn tevredenheid van gebruikers verhogen, reduceren van kosten en het verlagen van risico's.

De voordelen die gehaald worden door middel van self-adaptation zijn het verbeteren van de prestaties, besparen op uitgaven, verbeteren van interactie met het systeem en het omgaan met een dynamische omgeving.

B. Omschrijving van self-adaptation

OV2: Hoe omschrijven beoefenaars self-adaptation? Van de 100 deelnemers aan de vragenlijst die ervaring hebben met een concreet self-adaptation systeem hebben 99 een geldige omschrijving gegeven van self-adaptation. We zochten naar karakteristieken van self-adapatation in de antwoorden, en we konden deze categoriseren als onderwerp, type en trigger.

Het onderwerp zegt ons of de self-adaptation op niveau van het volledige systeem of een deel van het systeem is. Het type geeft aan welke soort adaptation gedaan wordt. De trigger geeft aan wat aanleiding geeft tot de adaptation.

Uit de antwoorden kunnen we concluderen dat self-adaptation op verschillende niveaus wordt gebruikt. Self-adaptation wordt gebruikt op het niveau van een volledig systeem en op het niveau van delen van het systeem.

De hoofdzakelijke types van self-adaptation zijn auto-scaling, auto-tuning en analyse.

De meest voorkomende triggers voor self-adaptation in industriële software-intensieve systemen zijn veranderingen in eigenschappen van het systeem en/of de omgeving, dynamiek in de systeembelasting, relevante evenementen en input van de gebruiker. Elastische cloud en auto-scalers zijn belangrijke factoren voor de realisatie van self-adaptation in de praktijk.

C. Implementatie

OV3: Hoe implementeren beoefenaars self-adaptation in industriële software-intensieve systemen? Dit is een zeer ruime vraag, hiervoor werden ook verscheidene vragen voor voorzien in de vragenlijst.

Om het systeem en de omgeving te analyseren en controleren kan het self-adaptation systeem gebruik maken van verschillende mogelijkheden. De meest gebruikte manieren van controle zijn middelen gebruik en belasting van het systeem. Sensoren in het systeem en de omgeving wordt het meest gebruikt om deze waarden op te meten.

In de industrie wordt gebruik gemaakt van een breed aanbod aan mechanismen en hulpmiddelen voor het analyseren van de voorwaarden voor het self-adaptation systeem. Het gebruik van data analyse en drempelwaarden zijn de voornamelijkste.

Auto-scaling en configuratie/reconfiguratie zijn de hoofzakelijke mechanismen gebruikt door self-adaptation systemen om aanpassingen te doen aan de systemen.

De graad van automatisering van de self-adaptation systemen is zowel volledig automatisch, semi-automatisch als een mix van beide. Een mix van volledige automatisering en semi-automatisering is het meest voorkomend.

Bij het implementeren van self-adaptation wordt er vaak hergebruik gemaakt van oplossingen met voornamelijk hergebruik van code, ontwerp en specificaties.

Het vertrouwen opgelegd aan de self-adaptation systemen wordt hoofdzakelijk gevalideerd door middel van uitbundig testen, controle tijdens runtime en menselijke supervisie.

We kunnen concluderen dat de implementatie van self-adaptation op veel verschillende manieren gebeurt in de industrie.

D. Ervaringen bij implementatie

OV4: Wat zijn de ervaringen van beoefenaars met het implementeren van self-adaptation en zien ze mogelijkheden voor onderzoekers? We willen weten of het implementeren van self-adaptation moeilijk is en welke moeilijkheden er worden ervaren.

De meerderheid van de deelnemers aan de vragenlijst geeft aan dat ze moeilijkheden ervaren bij het implementeren/onderhouden van self-adaptation. Enkele moeilijkheden zijn een betrouwbaar/optimaal ontwerp ontwerpen, de complexiteit van het ontwerp en tuning/debuggen.

Ongeveer de helft van beoefenaars van self-adaptation ondervinden risico's bij het implementeren/onderhouden van self-adapation systemen. De hoofdzakelijke risico's zijn foute resultaten, de moeilijkheid om te gaan met onzekerheid in de omgeving, verminderde prestaties en toenemende kosten.

Ongeveer de helft van beoefenaars ervaart problemen waarvoor ze graag hulp zouden krijgen van onderzoekers. Deze problemen gaan dan meestal over het ontwerpen van self-adaptation, zekerheden en het omgaan met data. Hier zien we dus enkele mogelijkheden voor toekomstig onderzoek.

Ongeveer de helft van beoefenaars zien mogelijkheden voor gebruik van self-adaptation in de organisatie die momenteel nog niet vervuld worden. Deze mogelijkheden zijn voornamelijk in verband met autonoom opereren, datamanagement en machine learning.

We kunnen concluderen dat er mogelijkheden zijn voor onderzoekers om de industrie te helpen rondom het domein van self-adaptation.

V. Discussie

A. Observaties

We merken dat self-adaptation regelmatig wordt gebruikt in de industrie. De problemen aangekaart door de industrie liggen ongeveer in dezelfde lijnen met wat onderzocht wordt

door onderzoekers. We merken echter dat veranderingen in de doelen van een organisatie zelden wordt opgelost aan de hand van self-adaptation. Onze hypothese is dat self-adaptation nog niet tot zijn volledige mogelijkheden wordt gebruikt in de industrie. Dit is echter enkel een hypothese, verder onderzoek is nodig om dit te bevestigen.

B. Voordelen van self-adaptation in de industrie

Wanneer we kijken naar de vernomen voordelen bij het gebruik van self-adaptation vinden we dat het verbeteren van de ervaring van de gebruiker en vermindering van kosten de meest voorkomende zijn. Deze voordelen werden maar liefst 70 % van de keren waargenomen. Optimaliseren van prestaties en omgaan met veranderingen in de omgeving zijn de 2 meest voorkomende problemen waarvoor self-adaptation wordt gebruikt.

C. Nadelen/moeilijkheden van self-adapation in de industrie

De moeilijkheden rond self-adaptation liggen hoofdzakelijk bij het ontwerp van self-adapation. Risico's bij het gebruik van self-adaptation zijn er vooral door de impact op de organisatie.

Grote bedrijven hebben vaker te maken met risico's bij het gebruik van self-adaptation. Grote bedrijven (minstens 100 werknemers) hebben meer moeite rond debugging, terwijl (algemeen) kleinere bedrijven zich meer zorgen maken om de limitaties van middelen en processen.

D. Ondersteuning door onderzoekers

Een groot deel van beoefenaars gaf aan dat ze hulp van onderzoekers zouden appreciëren. De hulp is gewenst rond auto-scaling, auto-tuning en analyse. De hulp is gewenst op verschillende niveaus: systeem, module en applicatie.

Module en applicatie niveau omvatten ongeveer 2/3 van de problemen waarbij beoefenaars mee te maken krijgen en waarvoor ze graag hulp zouden krijgen van onderzoekers.

VI. Bedreigingen voor validiteit

We bespreken mogelijke gevaren voor de validiteit aan de hand van de richtlijnen uit [11]. We bekijken constructieve validiteit, externe validiteit en betrouwbaarheid.

A. Constructieve validiteit

Bij de vragenlijst gingen we ervan uit dat de deelnemers enige voorkennis hadden over self-adaptation systemen. Bij het selecteren van de deelnemers voor de vragenlijst werden personen gebruikt die al mogelijks met self-adaptation gewerkt hebben. Hierdoor verhogen we de kans op expliciete voorkennis over het onderwerp. Er werd ook een vraag toegewezen naar het gebruik van self-adaptation. In het begin van de vragenlijst werd kort verduidelijkt wat met self-adaptation bedoeld werd. We kunnen afleiden aan de hand van de antwoorden verkregen dat de deelnemers een basiskennis hadden over self-adaptation.

B. Externe validiteit

Een mogelijke bedreiging voor de validiteit van de resultaten is de generalisatie. Als onze steekproef niet representatief is voor de populatie kunnen we verkeerde generalisatie doen voor heel de populatie. De deelnemers aan de vragenlijst waren zo geselecteerd zodat ze verspreid waren over de hele wereld. De deelnemers werden ook geselecteerd uit verschillende domeinen en uit verschillende rollen binnenin de bedrijven. We veronderstellen dat de steekproef wel representatief is voor de populatie.

C. Betrouwbaarheid

Data analyse, vooral van quantitatieve analyse is enigsinds subjectief. Het coderen van de antwoorden met vrije text kan andere resultaten opleveren indien het werk door verschillende programmeurs wordt gedaan. Om de persoonlijkheid van de programmeurs uit de resultaten te halen werd er afzonderlijk door 3 verschillende programmeurs codering van de antwoorden gedaan. Waar de programmeurs verschillende resultaten vonden werd er overlegd tot er een overeenkomst was over de analyse. Hierdoor proberen we de bevonden resultaten zo objectief te maken.

VII. Conclusie

We maakten een online vragenlijst dat correct beantwoord werd door 184 deelnemers waarvan 100 concrete ervaring hadden met self-adaptation.

Aan de hand van analyse van de antwoorden hebben we een state-of-practice kunnen vastleggen over self-adaptation in de industrie. Self-adaptation wordt regelmatig gebruikt in de industrie voor een grote verscheidenheid van systemen. De dominante types van self-adaptation zijn auto-scaling, auto-tuning en monitoring/analyse. We merkten ook dat menselijke supervisie nog vaak nodig is. Ongeveer de helft van beoefenaars ervaart risico's bij het gebruik van self-adaptation. Ongeveer de helft van beoefenaars zou het appreciëren om hulp te krijgen van onderzoekers bij self-adaptation.

De vragenlijst biedt een inkijk in de kennis van de industrie over self-adaptation wat gebruikt kan worden door onderzoekers om het domein te bepalen van toekomstig onderzoek.

We hopen dat deze resultaten een goede inkijk kunnen geven voor zowel de industrie als onderzoekers zodat ze samen kunnen werken om het domein van self-adaptation tot zijn volste te benutten in de industrie.

Referenties

[1] Matthias Hölzl, Axel Rauschmayer, and Martin Wirsing. *Engineering of Software-Intensive Systems: State of the Art and Research Challenges*, pages 1–44. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[2] Danny Weyns. *An Introduction to Self-adaptive Systems: A Contemporary Software Engineering Perspective*. John Wiley & Sons, 2020.

[3] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, et al. Self-adaptation in industry–state-of-the-practice and opportunities. 2020.

[4] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, et al. Analysis report survey on self-adaptation in industry. 2022.

[5] Danny Weyns, M Usman Iftikhar, Sam Malek, and Jesper Andersson. Claims and supporting evidence for self-adaptive systems: A literature study. In *2012 7th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 89–98. IEEE, 2012.

[6] Zhuoqun Yang, Zhi Li, Zhi Jin, and Yunchuan Chen. A systematic literature review of requirements modeling and analysis for self-adaptive systems. In *Requirements Engineering: Foundation for Software Quality: 20th International Working Conference, REFSQ 2014, Essen, Germany, April 7-10, 2014. Proceedings 20*, pages 55–71. Springer, 2014.

[7] Sara M Hezavehi, Danny Weyns, Paris Avgeriou, Radu Calinescu, Raffaela Mirandola, and Diego Perez-Palacin. Uncertainty in self-adaptive systems: A research community perspective. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 15(4):1–36, 2021.

[8] Sara Mahdavi-Hezavehi, Paris Avgeriou, and Danny Weyns. A classification framework of uncertainty in architecture-based self-adaptive systems with multiple quality requirements. In *Managing Trade-Offs in Adaptable Software Architectures*, pages 45–77. Elsevier, 2017.

[9] Timothy C Lethbridge, Susan Elliott Sim, and Janice Singer. Studying software engineers: Data collection techniques for software field studies. *Empirical software engineering*, 10:311–341, 2005.

[10] Colin Robson. *Real world research: A resource for social scientists and practitioner-researchers*. Wiley-Blackwell, 2002.

[11] Claes Wohlin, Aybuke Aurum, Lefteris Angelis, Laura Phillips, Yvonne Dittrich, Tony Gorschek, Hakan Grahn, Kennet Henningsson, Simon Kagstrom, Graham Low, et al. The success factors powering industry-academia collaboration. *IEEE software*, 29(2):67–73, 2011.

Gebruik van zelf-adaptatie in de praktijk omgeving bij software intensieve systemen

Dieter Demuynck
Wetenschappen & Technologie
Katholieke Universiteit Leuven Afdeling KULAK Kortrijk
Kortrijk, België
dieter.demuynck@student.kuleuven.be

***Samenvatting*—Computer software wordt hedendaags steeds meer toegepast in allerhande systemen, waaronder verkeersregeling, productie, gezondheidszorg, enzovoort. Systemen waarbij software een belangrijke rol speelt in design, constructie en werking van dit systeem, worden software intensief (software-intensive) genoemd. Bij zelf-adaptatie (self-adaptation) wordt in een software intensief systeem een feedback-lus geïmplementeerd die taken van mensen kan overnemen of met onzekerheden kan rekening houden[6]. Deze paper bespreekt een enquête die doelt om het gebruik en kennis van zelf-adaptatie in software intensieve systemen in de praktijk op te meten, welke problemen deze doelen op te lossen, en de voordelen en risico's van zelf-adaptieve systemen in de praktijk te bepalen. In deze enquête gaven 184 vakmannen uit 21 verschillende landen geldige antwoorden op de enquête. Deze antwoorden worden geanalyseerd zodat onderzoekers een beter idee hebben van de noden en vragen van vakmannen in de praktijk.**

INTRODUCTIE

Software intensieve systemen worden steeds meer toegepast in de praktijk, maar om deze systemen betrouwbaar en duurzaam te behouden wordt steeds moeilijker omdat deze systemen steeds meer complex worden om te maken, onderhouden en integreren. Men zoekt dus naar methoden om dit te vergemakkelijken, zoals DevOps, waarbij ontwikkeling (**Dev**elopment) en werking (**Op**eration) in elkaar verweven worden.[1] Daarom worden steeds meer zelf-adaptieve systemen gebruikt, die de taak van de mens kan overnemen om het systeem aan te passen wanneer nodig. Zo kan het systeem onzekerheden in de omgeving aan, of kan het zelf fouten verwerken, of kan het kosten verminderen door minder middelen te gebruiken.

De academische wereld heeft al veel onderzoek gedaan naar de principes van feedback controle in software intensieve systemen. Zo is er al veel kennis van de principes, modellen, talen, processen en methoden. Zo werden vier klassieke doelen opgesteld door wetenschappers, namelijk zelf-optimalisatie, zelf-heling, zelf-bescherming en zelf-configuratie[6]. Gelijkertijd werden deze principes ook bestudeerd en toegepast in de industrie. Nu hebben wetenschappers de vraag of hun kennis vanuit de academische wereld wordt toegepast in de industrie.

I. GERELATEERD WERK

Enkele gerelateerde werken zijn: *An Introduction to Self-adaptive Systems, A Contemporary Software Engineering Perspective*[4], die doelt om een uitgebreid overzicht te creëren van de verworven informatie over zelf-adaptieve systemen in de afgelopen twee decennia. Ook *A survey on engineering approaches for self-adaptive systems*[3], die een nieuwe kijk probeert te geven aan zelf-adaptieve systemen. Het laatste meegegeven gerelateerd werk is *Dealing with Drift of Adaptation Spaces in Learning-based Self-Adaptive Systems using Lifelong Self-Adaptation*[2], die doelt om een nieuwe techniek te beschrijven. deze zou kunnen omgaan met veranderingen van de factoren die een zelf-adaptief systeem kan aanpassen.

II. ONDERZOEKSPROBLEMEN

Het voornaamste doel van de enquête is om een beter begrip te hebben van de stand van zaken van zelf-adaptatie in de industrie, waaronder dus de motivering van het gebruik, de problemen die het oplost, het design en ontwikkeling van zulke systemen, en ook de moeilijkheden en risico's en de toekomstige mogelijkheden die de industrie ziet bij het gebruik van zelf-adaptatie[6]. Om deze punten te onderzoeken stelt de enquête vier grote onderzoeksvragen (**R**esearch **Q**uestions):

- **RQ1:** Wat motiveert vakmannen om zelf-adaptatie te gebruiken in software intensieve systemen?
- **RQ2:** Hoe karakteriseren vakmannen zelf-adaptatie?
- **RQ3:** Hoe passen vakmannen zelf-adaptatie toe in industriële software intensieve systemen?
- **RQ4:** Wat zijn de ervaringen van de vakmannen bij toepassing van zelf-adaptatie, en zien zij toekomstige mogelijkheden voor hoe en waar zij zelf-adaptatie toepassen?

Met RQ1 doelt men ook de types van industriële systemen die zelf-adaptatie toepassen te bepalen, alsook de problemen die zo een systeem oplossen. Men wil weten of deze dezelfde zijn als deze die de academische wereld verwacht, namelijk beheren van complexe (software intensieve) systemen gebaseerd op hoog-niveau doelen of het behandelen van onzekerheden tijdens de werking van het systeem. Bij RQ2 wil men onderzoeken hoe vakmannen zelf-adaptatie karakteriseren, waaronder welke terminologie zij gebruiken, en ook of er opkomende standaard praktijken ontstaan. Bij RQ3 wordt er nadruk gelegd op mechanismen, hulpmiddelen, maatstaven en processen. Ook wordt de samenhang van de academische kennis met de industriële praktijken bepaald. Ook de maat van vertrouwen die de vakman heeft in zo'n systeem is een onderzoekspunt van RQ3. Met RQ4 wil men de moeilijkheden en risico's bepalen die vakmannen ondervinden tijdens

het toepassen van zelf-adaptatie in de praktijk. Hierbij wil men bepalen waar vakmannen hulp van wetenschappers kan gebruiken, om de kloof tussen de academische en industriële wereld te versmallen.

III. Contributie

Deze paper doelt de paper *Self-Adaptation in Industry: A Survey* door Danny Weyns, Ilias Gerostathopoulos et al[6]. te vertalen en samen te vatten. De contributies van deze paper zijn dus gelijkaardig aan de contributies van die paper, die bedoelt om een empirisch gegrond overzicht te geven van de stand van zaken in de toepassing van zelf-adaptatie, inzichten voor wetenschappers over de noden van de industrie en voor vakmannen over hun niveau bij het toepassen van zelf-adaptatie, en extra informatie en mogelijkheden in verband met zelf-adaptatie in de praktijk.

IV. Methodologie

De onderzoeksvragen worden onderzocht aan de hand van een enquête. De onderzoekers van de originele paper[6] contacteerden in totaal 355 vakmannen in verschillende bedrijven, uit 21 verschillende landen, en vroegen hen om een online enquête in te vullen. Deelnemers moesten actief zijn in verschillende domeinen in verband met software intensieve systemen en moesten elk ook de benodigde expertise hebben.

De deelnemers worden bekend gemaakt met de term zelf-adaptatie en wat het inhoudt. Hierna krijgen zij een aantal open en gesloten vragen. Deze laatste hebben veelal een extra tekstveld waarin deelnemers een extra optie kunnen meegeven. De gesloten vragen worden geanalyseerd aan de hand van beschrijvende statistiek en kwantitatieve data-analyse. Hierbij werden veelal enkel frequenties en percentages weergegeven, alsook de relevante relaties tussen antwoorden door gebruik van kruistabellen. De open vragen werden geanalyseerd door kwalitatieve analyse. Inductieve redenering werd gebruikt om codes en categorieën op te stellen, waarna de voorkomens van codes werden opgeteld en gegroepeerd per categorie. De codes werden opgesteld door de auteurs van de originele paper[6] in groepen van twee tot drie codeurs, die elk individueel werkten, en dan samen hun codes opstelden. Dit werd dan nagekeken door twee andere onderzoekers. De vragen zijn opgedeeld volgens de onderzoeksvraag die ze helpen beantwoorden. Ook waren er vragen in het begin die de deelnemers dienen te categoriseren of ze zelf-adaptatie hebben gebruikt, en op het einde die het vertrouwen van de deelnemers in hun antwoorden peilt. Om een duidelijke enquête te maken werd een pilot verstuurd naar 8 kandidaten die extra feedback konden geven op de gestelde vragen en de duidelijkheid van de term zelf-adaptatie. Uit die pilot werd de enquête dan lichtjes aangepast zodat de enquête duidelijker werd.

V. Resultaten

A. Demografische informatie

Van de 355 kandidaten hebben er 184 (geldige) antwoorden gegeven op de enquête. Van de 184 deelnemers zijn er 100 die melden te hebben gewerkt met zelf-adaptieve systemen. 181 deelnemers gaven een duidelijke beschrijving van de systemen waarmee ze werken. Uit de antwoorden kon men het systeem classificeren volgens het type en de focus van het systeem. De meest voorkomende types zijn web/mobile, Internet of Things en Data management, en de meest voorkomende focussen waren controle, services en kwaliteit. Ongeveer de helft van alle deelnemers werkten in bedrijven met meer dan 100 software ingenieurs, met de rest ongeveer gelijk verdeeld tussen de categorieën 1 tot 10, 11 tot 50, 51 tot 100. Deze verhouding blijft ook ongeveer gelijk bij enkel deelnemers die software maken met zelf-adaptatie. Van de 184 deelnemers duidden 129 aan dat ze één rol hebben binnen het bedrijf, de rest geeft aan dat ze meerdere hebben. Hierbij zijn ongeveer 45% programmeurs, 40% project manager en 30% designer of architect. De waarden blijven ongeveer gelijk wanneer we ons limiteren tot enkel deelnemers die met zelf-adaptatie werken, met de uitzondering dat 9 op 10 onderzoekers met zelf-adaptatie werken. 69,6% heeft minstens negen jaar ervaring als software ingenieur, dat wordt 76,0% bij enkel die met zelf-adaptatie.

B. RQ1: Motivering voor zelf-adaptatie

De vragen in RQ1 dienen te peilen welke problemen men oogt op te lossen aan de hand van zelf-adaptatie, wat de grootste motivaties zijn, en welke voordelen het kan bieden of heeft geboden. De sleutelconclusies uit de resultaten zijn als volgt: Vele domeinen in de industrie passen zelf-adaptatie toe in verschillende systemen. Vakmannen passen zelf-adaptatie toe voor betere performantie, het automatiseren van taken en het verwerken van veranderingen in de implementatie omgeving. De meest voorkomende zakelijke doelen zijn het verbeteren van gebruikerstevredenheid en het verminderen van kosten, en vaak ook het beperken van risico's. De belangrijkste voordelen zijn verbeterd gebruik, besparingen, verbeterde interactie met de mens en omgaan met dynamiek.

C. RQ2: Karakterisatie van zelf-adaptatie

Aan de 100 deelnemers die met zelf-adaptatie werken werd gevraagd wat voor systemen deze ontwikkelden. Uit hun antwoorden werden drie categorieën van karakteristieken geïdentificeerd, namelijk *onderwerp*: het systeem of deel dat werd geadapteerd, *type*: de soort van de adaptatie en *oorzaak*: de reden tot aanpassing. De meest voorkomende onderwerpen zijn *systeem* en *module*. De meest voorkomende types zijn *automatisch schalen* en *automatisch afstemmen*. De meest voorkomende oorzaken zijn *systeemeigenschappen* en *omgevingseigenschappen*. Uit de resultaten van RQ2 merken we dat zelf-adaptatie op verschillende niveau's wordt toegepast, van volledige systemen tot enkele onderdelen. De belangrijkste types van adaptatie zijn automatisch schalen, afstemmen en controle. Aanpassingen gebeuren bij veranderingen in het systeem en implementatie omgeving, dynamieken, relevante gebeurtenissen en gedrag van de gebruiker. Een elastische cloud en automatisch schalen zijn sleutel factoren tot het realiseren van zelf-adaptatie in de praktische omgeving.

D. RQ3: Toepassing van zelf-adaptatie

Wanneer gevraagd hoe een zelf-adaptief systeem zich controleert, werden drie categorieën geïdentificeerd, namelijk *meetwaarden* met voornaamste gebruik van middelen, *controlemechanismen* zoals omgevingssensoren en *controleermiddelen* met Kubernetes monitoring en Prometheus als meest genoemde. De meest vernoemde analysemechanismen waren data analyse methoden en vergelijking met drempelwaarden, en de meest vernoemde hulpmiddelen voor analyse waren *AWS analysis tools* en *Kubernetes stack*. Aanpassingen werden vooral gedaan a.d.h.v. schaalmechanismen en reconfiguraties, of door gebruik van *Kubernetes* of *AWS*. Meeste van de 100 deelnemers melden dat ze vooral werken met een mix van deels en volledig geautomatiseerde systemen, en een grote meerderheid van 71 hergebruikt oplossingen soms tot altijd voor de realisatie van zelf-adaptieve systemen. Men hergebruikt vooral code, maar ook (van meest tot minst gebruikt) design artefacten, specificaties, IT-infrastructuur en procedures. De grootste hindernissen tot hergebruik blijkt vooral het verschil te zijn in de problemen die men wil oplossen. Vakmannen verzekeren het vertrouwen in de werking van hun systemen vooral a.d.h.v. tests en verificatie, maar ook soms door een belanghebbende centraal te plaatsen, vooral supervisie door een toeziende, en via online technieken.

E. RQ4: Moeilijkheden, Probleemondersteuning en Mogelijkheden

De vragen bij RQ4 hebben als doel de moeilijkheden die de deelnemers hadden bij toepassing van zelf-adaptatie te peilen en hoe wetenschappers deze problemen kunnen helpen omzeilen. Uit de 100 deelnemers die met zelf-adaptatie werken blijkt dat 41 soms problemen heeft bij het ontwikkelen van zelf-adaptieve systemen, met nog eens 24 die frequent tot altijd problemen heeft. Veelal zijn dit problemen met het design, maar ook met de levenscyclus en looptijd, en ook mens en proces problemen. Maar 18 van de 100 deelnemers melden dat ze frequent tot altijd risico's ondervinden tijdens de ontwikkeling. 34 melden dat ze soms risico's ondervinden, en 48 zelden tot nooit. De risico's die het meest voorkwamen waren mogelijke fouten, maar ook moeilijkheden met ontwikkeling, en de impact op kwaliteiten en op het bedrijf. Om deze risico's te ontmijnen werden vooral belanghebbendegecentreerde technieken gebruikt, maar ook offline en online technieken. Van 166 deelnemers merkt men dat ongeveer de helft soms tot vaak problemen heeft waarbij hulp door wetenschappers gewenst is. Deze problemen zijn vooral op vlak van engineering, maar gaan ook over garanties, data (bestuur en toegang) en gebruikersinteractie. Van de 184 deelnemers melden 101 dat er niet-toegepaste mogelijkheden zijn voor zelf-adaptatie, waaronder systeem activiteit en eigenschappen (zoals autonome werking resp. kwaliteitsverbetering), engineering activiteiten (zoals onderhoud en hergebruik) en menselijke betrokkenheid (personalisering).

F. Vertrouwen

Bijna alle deelnemers melden dat ze vertrouwen hebben in de antwoorden die zij gegeven hebben.

VI. Discussie en bedreigingen voor de validiteit

In het algemeen zijn de academische studies gelijklopend met de problemen bemerkt in de industrie, met de uitzondering dat het omzeilen van onzekerheden weinig nadruk heeft in de praktijk, wat mogelijks komt omdat vakmannen andere terminologie gebruiken in hun antwoorden. Bedrijfsdoelen lijken ook een minder voorkomende vereiste te zijn bij het implementeren van zelf-adaptatie. Vakmannen leggen ook nadruk op gebruikerstevredenheid, besparingen en het verminderen van risico's. Ze gebruiken veelal hulpmiddelen en infrastructuur voor de realisatie van hun systemen. Hergebruik kan ook nog meer nadruk krijgen in de academische wereld.

Betere performantie en verwerken van veranderende omgevingsfactoren zijn de belangrijkste problemen die men wil oplossen met zelf-adaptatie bij embedded systems en Internet of Things. In de cloud is dit vooral taken automatiseren en reconfiguratie, terwijl hergebruik (vooral van modules) het meest voorkomt bij fabricage, data management en Internet of Things. De grootste moeilijkheden zijn design van zelf-adaptatie en mens en processen op het gehele systeem, en de grootste risico's hebben te maken met ontwikkeling en werking. Grotere bedrijven hebben meer risico's op fouten, en hebben meer moeite met debugging, terwijl kleinere bedrijven meer de focus leggen op de limieten van hulpmiddelen en methoden. Het merendeel van vakmannen wenst hulp van academici, vooral op vlak van automatisch schalen en afstemmen, controle en analyse. Mogelijkse misinterpretatie van zelf-adaptatie werd vermeden a.d.h.v. een pilot met feedback. Het gebruik van non-probabilistische bemonstering kan leiden tot onrepresentatieve antwoorden voor de algemene populatie, wat werd omzeild door verschillende software ingenieurs uit verschillende landen, bedrijven en met verschillende rollen uit te kiezen. Subjectiviteit bij kwalitatieve data analyse werd omzeild door een uitgebreide methode waarbij verschillende ploegen van twee auteurs onafhankelijk deze analyse uitvoerden en daarna kruiscontroleren.

VII. Conclusies en toekomstig werk

Uit de enquête volgt dat zelf-adaptatie al veel wordt toegepast in de praktijk, door programmeurs, projectleiders, architecten en designers, voor betere werking, het automatiseren van taken en veranderingen in de implementatie omgeving, wat resulteert in makkelijker gebruik, betere gebruikerstevredenheid en besparingen. Automatisch schalen en afstemmen bij veranderingen in de omgeving of dynamische belasting zijn de meest voorkomende vormen van zelf-adaptatie. Hulp van onderzoekers is gewenst bij het realiseren van zelf-adaptatie, het hebben van zekerheden en datamanagement. Autonome werking, datamanagement en gebruik van machine learning zijn nog onaangetaste mogelijkheden.

Referenties

[1] Christof Ebert e.a. “DevOps”. In: *IEEE Software* 33.3 (2016), p. 94–100. DOI: 10.1109/MS.2016.68.

[2] Omid Gheibi en Danny Weyns. *Dealing with Drift of Adaptation Spaces in Learning-based Self-Adaptive Systems using Lifelong Self-Adaptation*. 2022. DOI: 10.48550/arXiv.2211.02658. arXiv: 2211.02658 [cs.LG]. URL: https://doi.org/10.48550/arXiv.2211.02658.

[3] Christian Krupitzer e.a. “A survey on engineering approaches for self-adaptive systems”. In: *Pervasive and Mobile Computing* 17 (2015). 10 years of Pervasive Computing’ In Honor of Chatschik Bisdikian, p. 184–206. ISSN: 1574-1192. DOI: https://doi.org/10.1016/j.pmcj.2014.09.009. URL: https://www.sciencedirect.com/science/article/pii/S157411921400162X.

[4] Danny Weyns. *An Introduction to Self-adaptive Systems: A Contemporary Software Engineering Perspective*. John Wiley & Sons, 2020.

[5] Danny Weyns e.a. “Preliminary Results of a Survey on the Use of Self-Adaptation in Industry”. In: *Proceedings of the 17th Symposium on Software Engineering for Adaptive and Self-Managing Systems*. SEAMS ’22. Pittsburgh, Pennsylvania: Association for Computing Machinery, 2022, p. 70–76. ISBN: 9781450393058. DOI: 10.1145/3524844.3528077. URL: https://doi.org/10.1145/3524844.3528077.

[6] Danny Weyns e.a. *Self-Adaptation in Industry: A Survey*. 2022. DOI: 10.48550/arXiv.2211.03116. arXiv: 2211.03116 [cs.SE]. URL: https://doi.org/10.48550/arXiv.2211.03116.

Huidige toestand en implementatie van zelf-aanpassende systemen in de industrie

Maxim Ketels
Groep Wetenschap & Technologie Kulak
Informatica
Email: maxim.ketels@student.kuleuven.be

***Abstract*—Software gedreven computersystemen worden vandaag de dag steeds groter in complexiteit. Om deze systemen te blijven onderhouden en beheren kan worden gebruik gemaakt van zelf-aanpassende systemen. Dit houdt in dat het systeem wordt uitgerust met een feedback loop die taken automatiseert die anders door een menselijke operator zou worden uitgevoerd. In dit verslag wordt gekeken naar een studie die de huidige toestand van zelf-aanpassende systemen in de industrie onderzoekt. Deze studie bevroeg 184 gebruikers gespreid over 21 landen en ging over: kennis van de gebruikers, redenen waarom men deze systemen gebruikt, hoe deze systemen worden toegepast en welke risico 's en uitdagingen men hierbij ervaart.**

1. Inleiding

Software gedreven computersystemen zijn vandaag de dag in zo goed als elke industrie aanwezig. De steeds groter wordende complexiteit van deze industriële software gedreven systemen maakt het onderhouden en beheren van deze systemen problematisch. De meerdere voorgestelde oplossingen voor dit probleem hebben allemaal een gemene eigenschap: het systeem wordt uitgerust met een feedback loop die taken automatiseert die anders door menselijke operators zouden worden uitgevoerd. Dit worden zelf-aanpassende systemen genoemd. Deze zelf-aanpassende systemen hebben al meerdere praktische implementaties met als typisch voorbeeld een elastische cloud. Zowel industrie als de academische wereld werken beide aan zelf-aanpassende systemen maar hebben niet altijd dezelfde focus.

In dit verslag wordt er gekeken naar de huidige staat van zelf-aanpassende systemen in de industrie. Dit wordt gedaan aan de hand van een studie [2] [1] [3] die een bevraging uitvoerde op 184 deelnemers in 21 landen. Deze studie had meerdere onderzoeksvragen met als hoofdpunten: de kennis van de gebruikers, redenen waarom men deze systemen gebruikt, hoe deze systemen worden toegepast en welke risico 's en uitdagingen men hierbij ervaart.

2. Gerelateerd werk

Er zijn meerdere studies gedaan over zelf-aanpassende systemen in de industrie maar nog nooit eerder is er op deze manier en schaal een bevraging gedaan. Aangezien de studie dus uniek is zal deze in de paper puur op zichzelf bekeken worden.

3. Onderzoeksvragen en methodologie

Het algemene doel van de studie is het onderzoeken van het huidig gebruik van zelf-aanpassing. Dit voor industriële software-intensieve systemen. Dit onderzoek zal worden gedaan vanuit het perspectief van de gebruikers in de industrie.

3.1. Populatie en doelpubliek

Voor dit onderzoek werd gezocht naar een specifiek doelpubliek. De deelnemers moesten actieve gebruikers zijn. Dit houdt in dat de deelnemers mensen waren die actief betrokken zijn in het ontwikkelen van industriële software-intensieve systemen. Ook moest de deelnemer een minimum van drie jaar technische expertise hebben in het praktisch ontwikkelen van industriële software systemen.

3.2. Onderzoeksvragen

De vier vragen die dit onderzoek hoopt te beantwoorden zijn de volgende:

- **RQ1:** Hoe karakteriseren gebruikers zelf-aanpassing?
- **RQ2:** Voor welke redenen passen gebruikers zelf-aanpassing toe in industriële software-intensieve systemen?
- **RQ3:** Hoe passen gebruikers zelf-aanpassing toe in industriële software-intensieve systemen?
- **RQ4:** Welke risico 's en uitdagingen ondervinden gebruikers bij het toepassen van zelf-aanpassing in industriële software-intensieve systemen?

Vraag 1 heeft als doel het controleren van de kennis van de gebruiker over zelf-aanpassing. De gebruikers zijn misschien niet volledig familiair met het begrip "zelf-aanpassing".

Met vraag 2 wordt gekeken naar het type problemen waarvoor zelf-aanpassing wordt toegepast als oplossing en

waarom. In academisch onderzoek wordt zelf-aanpassing voor voorgesteld als oplossing voor twee problemen:

- Het automatiseren van complexe software-intensieve systemen gebaseerd op complexe doeleinden geleverd door de operatoren.
- Het omgaan met condities die moeilijk te voorspellen zijn en die tijdig moeten behandelt worden.

Indien aan de hand van vraag 2 blijkt dat zelf-aanpassing voor andere doeleinden wordt toegepast zal dit toekomstige onderzoeken naar zelf-aanpassing dus in andere richtingen sturen.

Vraag 3 bekijkt de huidige implementatie van zelf-aanpassing. Dit verwijst dus naar welke methoden, technieken, werktuigen, benchmarks en processen er worden gebruikt om zelf-aanpassing toe te passen. In het bijzonder wordt er gekeken naar de graad van automatisering en de nood aan menselijke operatoren tijdens run-time. Ook hier is het interessant om de industriële oplossingen te vergelijken met de academische oplossingen.

Vraag 4 staat ons toe te de huidige risico 's en uitdagingen te begrijpen. Er wordt bovendien gevraagd hoe de gebruikers het vertrouwen verkrijgen in hun oplossingen met zelf-aanpassing die ze toepassen. Meer kennis over deze uitdagingen kan opnieuw de academische wereld in de juiste richting wijzen voor meer gericht en relevant onderzoek.

3.3. Onderzoeksmethoden

Om bovenstaande onderzoeksvragen te beantwoorden werd gebruik gemaakt van een online bevraging.

Elk van bovenstaande vragen werd opgedeeld in meerdere kleinere, meer gerichte vragen [3]. Al deze kleinere vragen vormen nu samen een set van open en gesloten vragen die de online bevraging zullen vormen.

Er werd gekozen voor een online bevraging omdat dit toelaat van een zo groot mogelijk publiek te bereiken met een lage tijds- en financiële kost. Het invullen van de bevraging zou ongeveer een half uur duren.

Voor het versturen van de online bevraging naar alle deelnemers werd eerst een pilot bevraging gestuurd naar 10 gebruikers. Hierin werden extra vragen toegevoegd over duidelijkheid van de terminologie en relevantie, domein en duidelijkheid van de vragen. Aan de hand van deze eerste 10 resultaten kon dan de uiteindelijke vragenlijst worden aangepast. Aan de hand van dit commentaar werd de introductie van de bevraging aangepast om eventuele bias te vermijden. De vragen werden niet aangepast aangezien deze duidelijk en grondig werden ervaart.

4. Contributies

Het doel van het onderzoek is het verhelderen van de huidige staat van zelf-aanpassende systemen in de industrie, wat relevant is voor zowel de onderzoekers als de gebruikers. Dit zal onderzoekers in staat stellen om te weten of hun onderzoek relevant is voor de huidige noden van de industrie, het zal de huidige gebruikers in staat stellen om het niveau van hun gebruik van zelf-aanpassing te peilen en het zou eventuele mogelijkheden voor samenwerkende studies kunnen opleveren.

5. Evaluatie en resultaten

In totaal werden 355 bevragingen verstuurd. Hiervan werden er 184 ingevuld. De antwoorden van deze 184 deelnemers werden dan nog verdeeld in twee groepen:

- De antwoorden van alle 184 deelnemers
- De antwoorden de 100 deelnemers die uitdrukten gewerkt te hebben met zelf-aanpassende systemen

5.1. Resultaten van vraag 1

Uit de bevraging konden meerdere conclusies getrokken worden omtrent onderzoeksvraag 1. Zelf-aanpassing wordt toegepast op een wijde variatie van domeinen in de industrie. De meestvoorkomende hieronder waren: performance optimisation, automatiseren van taken en het omgaan met veranderingen van de omgeving.

De meestvoorkomende handels-motivatie om zelf-aanpassing toe te passen zijn: verbeteren van gebruiker ervaring, het verminderen van kosten en het verminderen van risico's.

De meestvoorkomende voordelen van zelf-aanpassing die werden vermeld zijn: robuustere systemen en betere prestaties, betere kost en middelen efficiëntie, betere menselijke interactie en het omgaan van dynamische omstandigheden.

De industrie zal over het algemeen dezelfde problemen oplossen met zelf-aanpassing als degene die werden bestudeerd door de academische wereld met een merkbare uitzondering op vlak van zelf-aanpassing voor het verminderen van onzekerheden, wat een sterk onderzocht domein is in de academische wereld.

5.2. Resultaten van vraag 2

De bevraging geeft ons meerdere inzichten omtrent onderzoeksvraag 2. Zelf-aanpassing blijkt toegepast te worden op allerhande niveaus van industriële software-intensieve systemen. Het wordt gebruikt voor zowel complete systemen tot kleinere delen van een geheel systeem en support systemen.

De overheersende types van toepassingen in de industrie zijn: auto-scaling, auto-tuning, monitoren en analyse.

De toepassingen van zelf-aanpassing in de industrie worden geactiveerd door veranderingen in eigenschappen van de systemen en hun omgevingen, dynamieken in de systeembelasting, het voorkomen van bepaalde gebeurtenissen en gebruikers acties.

Technologieën zoals elastische cloud, auto-scaling en container-orchestration systems zoals Kubernetics zijn belangrijke facilitators voor de ontwikkeling van zelf-aanpassing in de praktijk.

5.3. Resultaten van vraag 3

De belangrijkste inzichten omtrent onderzoeksvraag 3 zijn als volgt: De meestvoorkomende parameters die worden gemonitord zijn resource usage en systeemsbelasting. Deze parameters worden voornamelijk gemonitord door sensoren in de omgeving en het systeem.

Meerdere analyse mechanismen worden toegepast door gebruikers in industriële settings. Data analyse en vergelijkingen met drempelwaarden zijn hier de meestvoorkomende.

Een wijd domein van mechanismen worden gebruikt om zelf-aanpassing toe te passen in industriële systemen met auto-scaling en herconfiguratie als de top mechanismen.

Gebruikers zijn sterk afhankelijk van tools zoals Kubernetes en AWS om de toepassing van verschillende functies van zelf-aanpassing te ondersteunen.

Industriële systemen passen een zowel semi- als volledig automatische toepassingen van zelf-aanpassing toe.

Het merendeel van de gebruikers hergebruiken eerdere oplossingen wanneer ze zelf-aanpassing toepassen. Dit vooral in de vorm van code, design artifacts en specificaties.

Het vertrouwen in zelf-aanpassende systemen wordt vooral verkregen door middel van uitgebreid testen, run-time monitoren en alarmeren en menselijke overzicht.

5.4. Resultaten van vraag 4

Uit de bevraging konden meerdere conclusies getrokken worden omtrent onderzoeksvraag 4.

Een merendeel van de gebruikers ervaart moeilijkheden wanneer zelf-aanpassende systemen worden ontwikkeld of onderhouden. Dit vooral met betrouwbaar of optimaal design, design complexiteit en tuning of debugging.

Ongeveer de helft van de gebruikers stuit op problemen met zelf-aanpassing. De hoofd-risico's zijn gerelateerd aan incorrecte functionaliteit, de moeilijk te beheren omgeving onzekerheid en eventueel verminderde prestaties en hogere kost.

Ongeveer de helft van de gebruikers melden dat ze steun van de onderzoekers zouden appreciëren voor de problemen die ze ervaren. In het bijzonder de problemen omtrent de ontwikkeling van zelf-aanpassende systemen en de verzekering en beheer van data.

Ongeveer de helft van de gebruikers ziet toekomstige mogelijkheden voor toepassing van zelf-aanpassing. In het bijzonder gerelateerd aan automatische operatie, data beheer en machine learning.

6. Validiteit

Er zijn meerdere potentiële dreigingen naar de validiteit van dit onderzoek. De eerste is dat men er van uitgaat dat de deelnemers voldoende familiair zijn met de concepten van zelf-aanpassing. Uit de verkregen resultaten blijken de deelnemers een goed basis begrip hebben van deze concepten. Ook werden meerdere maatregelen genomen om eventuele misinterpretaties te vermijden.

Een andere dreiging is de generalisatie van de onderzoeksresultaten. Dit probleem stamt uit de selectie van het doelpubliek. Indien de populatie niet representatief is kunnen de resultaten onnauwkeurig zijn en dus niet generaliseerbaar. Er werd geen probabilistische steekproefmethode gebruikt en dus is er een mogelijk risico dat de populatie bevooroordeeld is en niet representatief. Om dit probleem te minimaliseren werden deelnemers gezocht uit de netwerken met industrie van de onderzoekers. De resultaten van de populatie toont dat de deelnemers allemaal actieve gebruikers zijn met voldoende expertise in verschillende rollen in verschillende bedrijven van verschillende groottes. Bovendien werd er gewerkt met met 8 teams die deelnemers verzamelden over de hele wereld. Dit levert een gebalanceerde populatie op een globaal niveau.

Nog een dreiging is het feit dat caritatieve analyse van de data tot een zekere hoogte subjectief is. Deze analyse kan beïnvloed worden door de persoon die de analyse uitvoert. Om eventuele subjectiviteit te minimaliseren werd een grondig codeerschema gevolgd. Alle codeertaken werden uitgevoerd door teams van twee personen die elk individueel de data analyseerden en hierna waar nodig overlegden tot een overeenkomst werd gevonden. Hierna werden alle codeertaken opnieuw verdeeld tussen twee onderzoekers en opnieuw geanalyseerd. De resultaten werden dan vergeleken met de originele resultaten en weer overlegd tot een consensus werd bereikt. Uiteindelijk werd alle originele data van de bevraging publiek gemaakt.

7. Conclusie

Het doel van deze studie was het onderzoeken en karakteriseren van het huidige gebruik van zelf-aanpassende systemen in de industrie. Dit vanuit het oog van de gebruiker van deze systemen. Uit deze studie is gebleken dat zelf-aanpassing vooral werd gebruikt voor het optimaliseren van prestaties, het automatiseren van taken en het omgaan met veranderingen in omgevingen. Dit alles leidt tot verbeterd gebruik, verbeterde user satisfaction en lagere kosten.

Zelf-aanpassing werd vooral toegepast door parameters te monitoren zoals resource gebruik en systeemsbelasting. Deze werden vooral geanalyseerd door data analyse methoden en werden vergeleken met drempelwaarden.

Deze systemen zijn vooral een mix van semi- en volledig automatisch en hebben dus minder menselijke input nodig. Het vertrouwen in deze systemen werd verkregen door middel van uitgebreide tests en menselijk toezicht houden tijdens runtime.

De grootste uitdagingen bij deze systemen zijn het betrouwbaar en optimaal designen van de systemen, de design complexiteit, het tunen en debuggen. De belangrijkste risico's bevatten incorrecte functionaliteit, moeilijk te beheren omgeving onzekerheden en slechtere prestaties.

Mogelijke vooruitzichten zijn eventuele steun van onderzoekers bij het oplossen van problemen bij het ontwikkelen van zelf-aanpassing.

8. Toekomstig werk

Zelf-aanpassing in software-intensieve systemen is een domein dat enkel maar groter gaat worden in de nabije toekomst. Deze studie helpt de brug te leggen tussen de industrie en de academische wereld en zal hopelijk leiden meer gericht onderzoek of tot mogelijkse samenwerkingen tussen beide partijen.

References

[1] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, Grace Lewis, Marin Litoiu, Angelika Musil, Juergen Musil, Panos Patros, and Patrizio Pelliccione. Analysis report survey on self-adaptation in industry, 2022.

[2] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, Grace Lewis, Marin Litoiu, Angelika Musil, Juergen Musil, Panos Patros, and Patrizio Pelliccione. Self-adaptation in industry: A survey. 2022.

[3] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, Grace Lewis, Marin Litoiu, Angelika Musil, Juergen Musil, Panos Patros, and Patrizio Pelliccione. Self-adaptation in industry – state-of-the-practice and opportunities (survey protocol), 2022.

Self-Adaptation in practice: a short analysis of a survey

Ruben Anseeuw
Informatics
Science and Technology
KULAK
Kortrijk, Belgium
ruben.anseeuw@student.kuleuven.be

Abstract—This paper is a short analysis of the study done by Danny Weyns et al. [1] by survey, about the use of self-adaptive software in various sections of the industry. Self-adaptation is a new form of software-intensive systems which can change part of systems or architecture of systems to maximize performance and minimize costs, depending on changing extern factors. A classical example of a self-adaptive software is an auto-scaling cloud.

The purpose of the study of Danny Weyns et al. was to sketch an image of the practical use of self-adaptive software in the industry. After all, most research about self-adaptation was purely academical, and it is not known how much of this research is actually used in practice.

In this paper, we will summarize the questions, methodology and results of the original study. At the end, we will also discuss the differences between small and large companies who use self-adaptation

INTRODUCTION

Computers have become irreplaceable in our daily life, in factories, in communication, in the financial sector, and so forth. Logically, we would like those software-driven systems to be trustworthy, sustainable and reliable, while still flexible and adaptable to changing needs. Yet, building such systems require much effort and resources, and still are difficult to be flexible and adaptable the regular way.

This is the reason systems which are self-adaptable were researched and developed. These systems come from regular software-intensive systems with control from humans, and have become semi or fully automatized by equipping systems with feedback loops to automate tasks that otherwise need to be performed by humans.

Self-adaptation is a relative new form of software development, introduced by Oreizy et. al. in 1998 [2], which is still heavily researched today and of which the possibilities are far from all explored. It can reduce costs, improve performance, dealing with errors caused by external factors, etc. We will dive deeper into self-adaptive systems in section 1

This is the reason Danny Weyns et al. [1] have created a survey and sent them to a variety of practitioners with knowledge of the state-of-practice of self-adaptation in the industry. They analyzed the questions carefully and thoroughly and published a paper to sketch an image of the practical use of self-adaptation in industry, and to know if self-adaptation is something more than just an academical concept.

This paper is a short analysis of an investigation by survey, by Danny Weyns et al. . Throughout this paper, the we/us pronouns will be used to describe the methodology and results of that paper. For more detailed results and explicit data, we would recommend to check out the original paper. [1]

In this short paper, we will first give a short introduction to self-adaptation, followed by the research questions and methodology of the original study. We will discuss the contributions and related work, and thereafter the actual results of the study. We will crosscheck the size of the companies with these results and end with threats to validity and a conclusion.

1 SELF-ADAPTATION

Probably, not everybody reading this paper is familiar with the concept of self-adaptation. Some of you

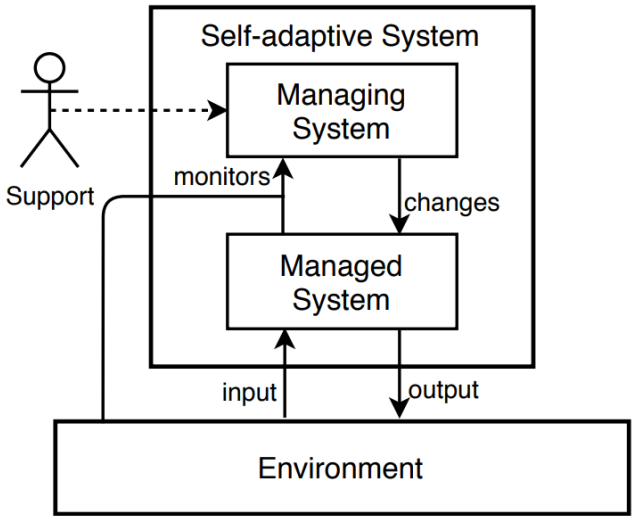


Figure 1. diagram of a self-adaptive system, taken from [3]

may vaguely know what self-adaptation mean, so we will explain its concept in a short introduction using some examples and globally accepted terms. We also gave this kind of introduction at the beginning of the survey, so every participant knew what we were talking about. [3]

1.1 Parts

A self-adaptive system is any software system consisting of 2 parts: a managing system and a managed system.

The managing system can be any regular kind of software-intensive or a part of it. This can be an entire system, like a car, a sub-system, like an engine, a feature, like a steering system, etc. It is essential the managed system can be easily changed by other systems.

The managing system takes information from its environment and can change the managed system, depending on the signals or inputs from the environment. This could be to improve performance when conditions in the environment change, or to deal with errors.

The environment can consist of other software systems, hardware, users, communication networks etc. Note that, while the environment sends signals to the managing system to change the managed system, the environment cannot be changed by the managing system.

These 3 parts are nicely pictured in Figure 1.

1.2 Examples

A classic, commonly used example of a self-adaptive system is a Cloud with auto-scaling.

Here, the managed system is the Cloud itself, providing e.g. computing power and data-storage. The environment are users and users programs that produce irregular and unpredictable workloads. The managing system is auto-scaling framework that uses scaling rules to minimize operational costs, while still providing enough for user satisfaction.

Other examples of self-adaptive systems are elastic systems, digital twins, Internet of Things based systems.

2 Research Questions

The overall objective is to better understand the state of the practice of self-adaptation in industry through a large scale survey. More specific, we liked to show what problems practitioners solve with self-adaptation, what motivates them to apply self-adaptation, how they design such systems, whether they follow some existing models, what difficulties they face, what risks they took and what future opportunities they see for self-adaptation.

These questions were bundled in 4 main research questions (RQ), which were further translated in 4 different categories with subquestions. If interested, these can be read in the [1]

- RQ1: What drives practitioners to apply self-adaptation in software-intensive systems?
- RQ2: How do practitioners characterize self-adaptation?
- RQ3: How do practitioners apply self-adaptation in industrial software-intensive systems?
- RQ4: What are the experiences of practitioners with applying self-adaptation and do they see opportunities for how and where to apply self-adaptation?

In the questionnaire, an extra category was added, category 0, in which some introductory questions were asked about the partaker, the size of their company, their role within, their amount of experience etc. This way, we crosscheck these results with the results of questions 1 through 4 and link categories of answers to characteristics of companies. We will discuss this kind of conclusions later in this paper.

3 Methodology

3.1 Population

The survey was intended for practitioners who are "actively involved in the engineering of industrial software-intensive systems in any domain" [1]. This way, we hoped to reach people with enough active knowledge about self-adaptation in practice, to sketch a correct image of the overall use of self-adaptation in the industry.

Concretely, 355 practitioners were contacted from 21 different countries via the researches of our study by personalized emails. In the end, 184 people sent back their completed surveys, which are all included in the data-analysis of this study.

3.2 Survey

The questionnaire itself was made and reworked to maximize objectivity and clarity. First of all, we started the survey with a gentle introduction to self-adaptation with basic terminology and multiple examples, to make sure every partitioned is using the same definition of self-adaptation and self-adaptive systems.

Secondly, in closed questions, such as yes/no and multiple choice, every participant had the opportunity to add an extra option, to ensure each could communicate what they wanted to say, and not push them towards a predefined choice.

Lastly, we validated the questionnaire in a pilot with eight randomly chosen practitioners, by adding extra meta-questions about the clarity, terminology, relevance and scope of the questions.

3.3 Method of analysis

To maximize objectivity, we used 2 general methods. Firstly, although we classified open question into categories for analysis, we did not construct the categories beforehand, nor the number of them. This way, we can analyze the results, while not limiting or influencing the answers.

The second method we used to maximize objectivity is by letting multiple groups of multiple people categorize the answers. Concretely, every question in this survey is analyzed and categorized by a team of 2 or 3 people, who first categorized the questions individually, second they compared their finding with each other. Every question is then independently classified by 2 other researchers, and if necessary, also compared and discussed with the original team, until a consensus was reached.

4 Contributions

Through this survey and the image in provides, we hope to help both researchers and practitioners on multiple levels. Concretely, we hope to provide an overview of the present use of self-adaptation, give insight to researchers and practitioners to assess their current research in relation to the practical use, and their current level of practice respectively.

We also believe this paper can create and sustain collaborations between research and industry.

5 Related work

In our paper [1] we claim "There is no clear and documented view of why and how the principles of self-adaptation are applied in practice, and what challenges practitioners face when realizing self-adaptation." This research is thus a unique study, so hopefully we can contribute to the development and research in the field of self-adaptive software.

Yet this research doesn't come out of thin air: some of the same authors wrote a paper about the guidelines to support industry relevant research on self-adaptation, [4] about evaluating self-adaptive software [5], but also other groups, e.g. another survey about self-adaptive software, but the research side of the story [6]

6 Results

6.1 introductory questions

The introductory questions provide an overview of the participants of this survey. About half (54.4%) have worked with concrete self-adaptive systems.

The roles within the companies are widely distributed, most frequently programmer, project manager and designer. These roles do not vary much when we only consider the practitioners who have worked with self-adaptation

Most people have between 9 and 20 years of experience (45.1%), followed by people with more than 20 years of experience (24.5%) and people with between 4 and 8 years of experience (29.6%). These statistics also stay roughly the same when only considering those who have worked with self-adaptation

About half (48.9%) of the participants work in a big company (more than 100 employees), and half in smaller companies. This last will be important later in this paper.

We can conclude that our sample has a broad variety of people, and that the roles, sizes and experience probably independent is from whether they have worked with self-adaptive systems

6.2 RQ1: drivers

The questions in category 1 were focused on the drivers for applying self-adaptation, like kind of problems, motivations and benefits.

We can conclude that self-adaptation is widely applied in industry across a wide variety of domains. The main kind of problems are optimizing performance, automate tasks, and deal with changes in the deployment environment.

The primary motivations are improving user satisfaction, reducing costs and mitigating risks.

The most important benefits provided of applying self-adaptation are improved utility (in robustness and performance), savings (costs and resources), improved human interaction (user experience and engineers support), and handling dynamics (in the context and system load).

6.3 RQ2: Characterization of self-adaptation

This part of the survey focused more on characterization of self-adaptation by practitioners, through questions like explaining a concrete self-adaptive system they worked with.

We found that self-adaptation is applied at different levels of industrial software-intensive systems, complete systems as well as parts of a system and support systems.

The most common types of adaptations applied in industry include auto-scaling, auto-tuning, and monitoring/analysis, and these adaptations are triggered by changes in system and environment properties, dynamics in system load, relevant events, and user actions.

Additionally, the study highlighted that key technologies such as elastic cloud and auto-scalers play a critical role in enabling the practical implementation of self-adaptation in industry.

6.4 RQ3: Application of self-adaptation

Section 3 of the survey focused more on tools, mechanisms, automation, re-usability of solutions and trust.

We noticed that resource usage and system load are the main types of monitoring metrics used in practice, and that these metrics are primarily tracked by sensors in the environment and the system.

Practitioners use various mechanisms for analysis in realizing self-adaptation, as well as mechanisms to enact self-adaptation in industrial systems, with data analysis methods and comparison to thresholds as main mechanisms for the analysis, and auto-scaling and reconfiguration to enact.

Tools such as Kubernetes, Prometheus and AWS are heavily used to support the realization of different functions of self-adaptation.

Industries apply a mix of semi and full automated adaptation, and a majority of the group reuses solutions when applying self-adaptation, mainly in the form of code, design artifacts and specifications.

Ensuring trust in self-adaptive systems is primary achieved through extensive testing, runtime monitoring and alerting, and human supervision.

6.5 RQ4: difficulties, problem support, and opportunities

Section 4 of the questionnaire focused on difficulties during engineering and maintaining, risks taken and reduced, support from researchers and unexplored applications of self-adaptations.

We can conclude that a majority of participants sometimes face difficulties when engineering or maintaining self-adaptive systems, mainly with reliable and optimal design, design complexity, tuning and debugging.

About half of the participants sometimes encounter risks when using self-adaptation, which mainly relate to incorrect functionality and difficulty to manage environment uncertainty, as well as degraded performance and increased cost.

Also about half of the practitioners report that they would appreciate support from researchers to deal with problems they face, in particular problems related to the engineering of self-adaptive systems, guarantees, and management of data.

Lastly, about half of the participants see future opportunities for applying self-adaptation, in particular in relation to autonomous operation, data management and machine learning.

7 Discussion: size of companies

7.1 Motivation

As mentioned in 6.1, the size of the companies the participants work for varies, with about half of the partakers working for a large company (more than 100 employees), and the other half working in small companies (less than 100 employees). Therefore, it is meaningful to crosscheck the results gathered in chapters 1-4 of the survey with the size of the company, and make statements about the differences between small and big companies

Size company	Tools/infrastructure	Custom mechanisms
1-10	5 (56%)	4 (44%)
11-20	3 (27%)	8 (73%)
21-50	5 (36%)	9 (64%)
51-100	4 (40%)	6 (60%)
< 100	17 (39%)	27 (61%)
>100	7 (13%)	47 (87%)

Table 1
Crosscheck analysis of h-the size of companies against mechanisms of self-adaptation, split in 2 categories: relying on tools/infrastructure and custom mechanisms

7.2 Risks

First of all, we note that there is no significant difference in the frequency of risk associations take across large and small companies: both take some risks to work with self-adaptive software. It should be noted, however, that large companies report errors more often than small companies, namely 47% of large companies versus 9% of small companies in this survey.

7.3 Difficulties

Secondly, large and small organizations are equally concerned about difficulties with design and tool support, yet the latter in a different way.

Both types of companies are also concerned about tool support, but in different ways: large companies lay more emphasis on difficulties with debugging, while small companies find limitations on tools and methods more important

7.4 Mechanisms

Lastly, when crosschecked against mechanisms used to realize self-adaptation, we see that large companies are much more likely to use custom solutions for self-adaptation problems, while small companies are more likely to rely on available tools and infrastructure.

This is not to say that large companies are not using tools, nor that small companies are not coming up with their own solutions, but the difference is noteworthy. The exact data can be read in the table 1

It is not hard to see the logic in this, considering large companies have more resources and budget to design their own, custom solutions and mechanisms, while small companies will much quicker search for and adapt existing solutions and tools.

8 Threats to validity

First of all, we can wonder if the questions in the questionnaire were clear enough, with no unnoticed possible misinterpretation. To this purpose, a pilot was added to the questionnaire and answered by a randomly chosen group of 10 people, containing meta-questions about the questionnaire itself. These questions were about the clarity of the terminology, the relevance and clarity of the questions, and the scope of the questionnaire.

Because of this feedback, we changed the introduction, yet the questions remained unchanged because they were received by all as clear and understandable.

Secondly, at the end of the questionnaire, everyone was asked how sure they were about their answers. Almost everyone gave a positive to strongly positive answer. This gives the results more credibility.

We could also wonder if we could generalize our results, in other words if our sample was chosen good and general enough, since we did not choose a probabilistic selection method. Yet, we tried to choose the practitioners randomly and from all field in the self-adaptation and areas in the industry, from different countries and different roles within companies of various sizes.

Lastly, to ensure the expertise of the partakers and the clarity of the questions, we often asked them to specify their answer with an example they faced, to ensure we and them were on the same page about the question, and to ensure their multiple choice answer reflected their experience, written in free text

9 Conclusion and further research

We sincerely hope, through this study, we have sketched a correct image of the practical use of self-adaptation in industry. We hope to help both practitioners and researchers to compare and coordinate their projects and research with each other

We hope to encourage further studies in the same line, so research and practice can help each other grow. We hope to steer research in the way industry needs, and to encourage industry to ask and employ researchers in their companies.

Still, there is a question unanswered and not yet researched: we now know how and why the industry uses self-adaptation, but we are left to wonder if they use the direct results of research.

REFERENCES

[1] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, Grace Lewis, Marin Litoiu, Angelika Musil, Juergen Musil, Panos Patros, and Patrizio Pelliccione. Self-adaptation in industry: A survey. *ACM Trans. Autonom. Adapt. Syst*, 1(1):43, November 2022.

[2] Peyman Oreizy, Michael M Gorlick, Richard N Taylor, Dennis Heimhigner, Gregory Johnson, Nenad Medvidovic, Alex Quilici, David S Rosenblum, and Alexander L Wolf. An architecture-based approach to self-adaptive software. *IEEE Intelligent Systems and Their Applications*, 14(3):54–62, 1999.

[3] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, Grace Lewis, Marin Litoiu, Angelika Musil, Juergen Musil, Panos Patros, and Patrizio Pelliccione. Self-adaptation in industry – state-of-the-practice and opportunities. *ACM Trans. Autonom. Adapt. Syst*, 1(1):23, June 2020.

[4] Danny Weyns, Ilias Gerostathopoulos, Barbora Buhnova, Nicolás Cardozo, Emilia Cioroaica, Ivana Dusparic, Lars Grunske, Pooyan Jamshidi, Christine Julien, Judith Michael, et al. Guidelines for artifacts to support industry-relevant research on self-adaptation. *ACM SIGSOFT Software Engineering Notes*, 47(4):18–24, 2022.

[5] Ilias Gerostathopoulos, Thomas Vogel, Danny Weyns, and Patricia Lago. How do we evaluate self-adaptive software systems?: A ten-year perspective of seams. In *2021 International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, pages 59–70. IEEE, 2021.

[6] Claes Wohlin, Aybuke Aurum, Lefteris Angelis, Laura Phillips, Yvonne Dittrich, Tony Gorschek, Hakan Grahn, Kennet Henningsson, Simon Kagstrom, Graham Low, et al. The success factors powering industry-academia collaboration. *IEEE software*, 29(2):67–73, 2011.

[7] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, Grace Lewis, Marin Litoiu, Angelika Musil, Juergen Musil, Panos Patros, and Patrizio Pelliccione. Analysis report survey on self-adaptation in industry. *ACM Trans. Autonom. Adapt. Syst*, 1(1):19, November 2022.

Characterizing the application of self-adaptive systems in industry

Arthur Cremelie
Department of Computer Science
Katholieke Universiteit Leuven, Kulak
8500 Kortrijk, Belgium
arthur.cremelie@student.kuleuven.be

***Abstract*—Software has an important presence in a lot of aspects of life nowadays. We can find it in manufacturing, traffic control, health care and financial systems. These systems are more and more referred as software-intensive. The use of self-adaptation implies that a system is equipped with a feedback loop that either automates tasks that otherwise need to be performed by human operators or deals with uncertain conditions. This idea has gained traction in some practical applications such as elastic clouds to adapt computing resources and automated server management to respond quickly to business needs. This article aims to discuss motivations for applying self-adaptation in practice, the problems solved using self-adaptation and the difficulties and risks that the industry faces when adopting self-adaptation. This is done with the help of a large-scale survey with 184 valid responses from practitioners spread over 21 countries. Practitioners can compare their current practice in applying self-adaptation.**

***Index Terms*—Self-adaptation, state-of-the-practice, survey, industry, learning**

I. Introduction

This article is based on previous work and research in [4]. We will discuss it as if we performed the research. Nowadays, we see software systems that keep our factories working, control the traffic systems, ensure telecommunication, support financial systems, and so forth. When software plays a vital role in their design, construction, and operation, these systems are often referred to as software-intensive systems. As the industry continues to innovate, it becomes more and more challenging tguaranteeing trustworthiness and sustainability. These systems face growing demands, continued integration, uncertain operation conditions, and a fast pace of technological progress. The industry has to adapt to the problems it faces under these conditions. A common approach today is what is called DevOps. DevOps means end-to-end automation in software development and delivery, building on lean and agile practices. Development and operation are blended, allowing system components to be easily evolved and redeployed without impacting their operation.

One classic approach to handling the increasing demand and complexity of software-intensive systems is to transfer control from humans to software components by equipping systems with feedback loops. These feedback loops monitor the system and its environment, reason about the system's behavior and its goals, and adapt the system to ensure its goals under changing conditions, or gracefully degrade if necessary. Such goals can be very diverse, ranging from ensuring a required level of performance under uncertain workload conditions, dealing with errors caused by external services that are difficult to predict, or defending the system against malicious attacks and the problems they may cause.

The classical approach of feedback control has been applied in industry: IBM launched its initiative on autonomic computing back in 2001. Here, systems manage themselves according to an administrator's goals, inspired by the autonomic nervous system of the human body. There are four classic goals: self-optimization, self-healing, self-protection, and self-configuration. The aim is to reduce the time required by operators to resolve system complexities and other maintenance tasks. We have seen practical applications of systems based on feedback loops, for example, elastic clouds to adapt computing resources of server parks to deal with changing business needs.

A. Survey

In order to have a view of the state-of-practice of self-adaptation in industry, we performed a large-scale survey in the form of a questionnaire with active practitioners. This helps us discover what motivates practitioners for self-adaptation, the kind of problems they solve using self-adaptation, how practitioners design and develop self-adaptive systems, and the difficulties and risks they face in adopting self-adaptation. We used two criteria to invite people for the survey: (1) participants should be active in different domains that are representative of software-intensive systems, and (2) participants have the required expertise to answer the questions.

II. Practitioners of Self-Adaptation

Out of 184 participants of the survey, 54.4% expressed to have worked with self-adaptive systems. 15% of systems that use self-adaptations is focused on web/mobile. Then, 13% has its focus on embedded/cyber-physical/IoT. Another 11% is occupied with the cloud. We see that participants that worked with self-adaptation reported on average 1.5 roles. The most frequent roles are programmer and project manager/coordinator. About one in three participants works as a designer or architect. Most participants of the survey are experienced software engineers.

III. Characterisation of Self-Adaptation

A. Concrete self-adaptive systems

We identified three categories for self-adaptive system characterisation:

- subject
- type
- trigger of adaptation.

The first one, subject, is the system or part of it that is adapted. The second characterization, type, means what kind of adaptation that is applied. Finally, the trigger of adaptation refers to the source that initiates adaptation. The industry says that system is the most frequented subject, followed by module, which is a part of a system. The platform layer and application layer are less frequented as a subject for self-adaptation. Auto-scaling, auto-tuning are the most frequented types of applied self-adaptation. There is a smaller but still important group that reports focusing solely on monitoring and analysis. The main triggers originate from system properties and environment properties. Changes in the system load, events, and user actions are other types of triggers for adaptation.

B. Drivers for Self-Adaptation

Self-adaptation can deal with a diversity of problems. The main problems addressed by self-adaptation in industry is optimizing performance and automating tasks. Dealing with changes in business goals is less frequently solved using self-adaptation. If we take a look at the main business motivations to apply self-adaptation, we see that improving user satisfaction, reducing costs, and mitigating risks are the most selected motivations. The industry sees different benefits of self-adaptation. The first one is improved utility. There is better robustness, for example fault tolerance and better error handling. Performance and availability are also improved. Second we see savings in costs and resources. Self-adaptation can make the maintenance of systems more efficient (cloud service), but also make it possible to use less resources. For example, when traffic is low, self-adaptation can scale down resources without any manual interference. Third, there is an improved human interaction. On the one hand there is improved user experience, and on the other hand self-adaptation removes most of the optimization burden for programmers, it reduces workload on human operators. Finally, the industry sees a benefit in handling dynamics. For example, each machine is unique and its optimal operational parameters change over time due to ware, location, task and seasonal factor.

C. Application of Self-Adaptation

1) Monitoring: To enable self-adaptive systems to function effectively, it is essential to have mechanisms or tools in place for monitoring the managed system during operation. We have identified three categories of monitoring:

- monitoring metrics
- monitoring mechanisms
- monitoring tools.

The most used monitoring metric is resource usage, typically CPU and Memory usage. In addition to resource usage, load is also commonly monitored, for example, the number of queries or requests. The industry also uses reliability metrics, such as the AWS lambda error metric, and performance metrics such as the response times for the users' requests. Regarding monitoring mechanisms, environment sensors such as Lidar, Camera, GPS are the most popular. Followed by logging mechanisms and system sensors. Finally, Kubernetes monitoring, Prometheus and grafana, and AWS monitoring are popular monitoring tools for self-adaptation in the industry. Other monitoring tools mentioned by the industry include Azure monitoring, Datadog, Splunk, cAdvisor, and Elasticsearch. One participant even mentioned that human involvement is necessary in monitoring.

2) Analysis: A self-adaptive system also uses mechanisms or tools to analyse conditions of a managed system during operation. Here, we identified two categories:

- analysis mechanisms
- analysis tools.

The most frequently mentioned mechanism are data analysis methods, such as interference, statistical data analysis, what-if analysis, and search-based methods. Shortly followed by comparison to threshold, metric calculation and learning, such as machine learning. With comparison to threshold we mean hard coded critical boundaries or comparing the error rate with constant or dynamic thresholds. An example of metric calculation is failure rate or the measurement of the performance of nodes. The following is an example of machine learning as an analysis mechanism. Each node has a kind of edge computing component that performs some analysis based on machine learning results. Other mechanisms are custom rules and auto-scaling policies. Regarding the analysis tools, the second category, the most popular choice is the AWS analysis tool. Here, analytic functions native to the cloud environment from AWS in which the systems runs are used. Second, a Kubernetes stack is used, often built-in. Other analysis tools are Dynatrace, rule-based systems like Splunk, tooling from Azure or Kibana.

3) Modification: The self-adaptive system requires a mechanism or tool to modify a managed system or its parts while it's in operation. We can divide these into two categories:

- change mechanisms
- change enacting tools.

One of the most commonly used change mechanisms is scaling. Another is reconfiguration, which allows for the adaptation logic to be altered, the network to be reconfigured, parameters to be adjusted, or load balancing to be applied. For example, depending on the context, controlled variables can be managed through different automation systems. There are also cases in which automation is not used for these tasks. In such cases, an engineer must approve the results obtained from the tool, which are then acted upon. Alerts are generated, and humans are expected to resolve the error manually based on

suggestions. This can be a safety protocol in some situations. Other change mechanisms include restarting or deploying systems or subsystems (such as restarting an unhealthy workload) and migration. The second category, change enacting tools, mostly consists of Kubernetes and AWS. Examples of such tools include the AWS ElasticLoadBalancer and triggering actions via AWS Lambda functions. The industry also mentions other tools like IBM ITM, Log Analyzer, TCAM, UC4 Automation Engine workflows that orchestrate Kubernetes clusters, and built-in OpenShift mechanisms.

4) Degree of Automation: If we analyze the degree of automation of the majority of the self-adaptive solutions, mixed automation is most frequently used in the industry. This means both semi-automation and full automation. Mixed automation is followed by semi-automation and a smaller group uses full automation.

5) Reuse of Self-Adaptive Systems: The majority of the participants reused, at least sometimes, solutions in self-adaptive systems. The results of the survey demonstrate that reuse in self-adaptation is a common practice. More concretely, there are five categories of subjects for reuse:

- code
- design artifacts
- specifications
- IT infrastructure
- procedures.

In the first category, code, modules are most frequently reused. For example, one participant stated that the self-adaptation mechanisms used for speech recognition, are also used for computer assisted coding solutions. Another example in a robotics company, is that different parts of a behavior tree, can be reused in different robots. Scripts and algorithms are also regularly reused. This is often because it is the easiest way to create new systems with a constant lack of time. Threshold algorithms can be reused, only by changing the threshold value to suit the specific use case. Design artifacts is the second category in which patterns and architecture are reused most frequently. Design patterns (e.g., auto-scaling) can be used for cloud native applications. For architecture, an AWS stack can be used as a generic template for different applications that are based on a job processing. Also, models, for example machine learning cost models, can be reused by different systems. The third category is about specifications. The most popular reuse of specifications is the reuse of policies and rules. For example, auto-scaling policies can have a standard definition that can be reused in different systems or use cases. The reuse of configuration files has the same popularity in our survey. We find less reuse of templates such as configuration templates and in metrics specifications. The category IT infrastructure contains frameworks and platforms, and tools. The last category, procedures, contains processes, pipelines and schedules. Here, reuse is less represented in the industry. Further investigation into what hinders practitioners from reusing elements, shows that the main hurdle is the difference in problems. Other obstacles are a lack of experience or maturity in applying self-adaptation within the company, the complexity of the system and organizational concerns.

6) Trust Self-Adaptive Solutions: Another important facet of self-adaptive systems is ensuring that you can trust it. The techniques for this problem can be divided in three categories:

- testing and verification
- stakeholder-centred techniques
- online techniques.

We see that trust is mainly achieved through extensive testing, runtime monitoring and alerting, and human supervision. If we take a closer look at the first and most popular category, extensive testing is the most important method. Another testing and verification method for self-adaptive systems is benchmarking. The second category consists of stakeholder-centered techniques. In this category, human supervision is the most important technique. This technique is most commonly applied in the release phase of the system. Human supervision can be gradually released when things are going as planned. Rigorous design and development are another, but less popular, technique used by the industry. Other factors that ensure trust in the self-adaptive systems are trust in third-party software and operational constraints such as concrete actions that are exactly specified by the user. The last category is about online techniques. An important technique is runtime monitoring and alerting. For example, one can deploy some alerts to track high-level properties of systems. Other online techniques are continuous testing during operation, such as gradual canary testing in a real production system, and mitigation strategies such as automation, which can rollback automatically if there is any issue.

D. Difficulties and Problem Support

We see that either small or medium organizations (¡ 100) are concerned about difficulties with design. If we take a closer look at the concerns about tool support, large organizations (¿ 100) have more difficulties with debugging. This is different for smaller and medium-sized organizations. Here, the limitations of tools and methods are a bigger difficulty. The frequency of encountering risks is not different for large, small, and medium organizations. Smaller and medium organizations will rely more on tools and infrastructure in order to realize self-adaptation, while large organizations will more often use custom solutions. Almost all companies that apply self-adaptation have mechanisms in place for monitoring but not necessarily for analysis and change.

1) Main Difficulties: Creating self-adaptive systems mostly doesn't come without problems. Only four percent of the participants in the survey reported never having problems. If we take a look at some examples of difficulties when engineering or maintaining self-adaptive systems, on average, each participant has to deal with nearly two difficulties. There are four categories of difficulties:

- design issues
- lifecycle issues
- runtime issues
- people and process issues.

The industry has the most difficulties with the design of self-adaptation, in particular reliable or optimal design, followed by design complexity. The industry finds complexity in defining the adaptation rules and conditions that are not always obvious. Engineering a system with self-adaptation, mostly implies that self-adaptation or resilience have to be taken into consideration at each stage of the workflow. The problem is that not every programmer and developer knows the concepts of self-adaptation. Regarding lifecycle issues, tuning and debugging are the most important difficulties. Debugging the cause of a self-adaptation failure can be time-consuming. The second group of life cycle issues is about the limitations of tools and methods. For example, available metrics are not always fully transparent and built with auto-scaling in mind. The last lifecycle issue is the system's or environment's evolution. For example, if self-adaptation is not implemented from the beginning, it can be a lot of work to implement later. The third category is runtime issues. Here, runtime uncertainty is the most important difficulty. Some self-adaptive systems are based on unproven heuristics. It can be hard to make a guess about how much the environment can affect the system, or to extend parameters to cover a whole production. Other runtime issues are data collection or evaluation, the required resources, and delayed or missing runtime changes. The last category consists of people and process issues. Here, as mentioned before, skills or experience are the most important difficulty. For example, self-adaptive systems must be tuned up, which can be tricky and requires highly skilled engineers. Also, the Kubernetes or Openshift cloud and centralized log storage require experienced administration staff and vast knowledge of many networking concepts. Another issue is process and management, for example, it can be a challenge to convince the IT department to implement self-adaptation, design the system, and master the technology.

2) Risks: About half of the participants encounter risks when using self-adaptation. The risks can be grouped in four categories:

- faults
- difficulties with development or operations
- impact on qualities
- impact on business.

The most frequently mentioned risks in the industry relate to faults. For example, incorrect functionality in automation can lead to unexpected values. Wrong results, misconfiguration, and network failures are other faults mentioned by the industry. The second category is about difficulties with development or operation. Here, it can be difficult to manage environment uncertainty. For example, one can underestimate environment variability, or the incoming event stream can be completely unpredictable and have differences in data for a considerable period of time. Testing can also be difficult, for example, when doing reliable performance testing in non-production environments. Thus, if one does not test the self-adaptive system, the system might introduce some risks. Also, building self-adaptive systems can be difficult. It can take a longer time to implement and design self-adaptive systems, and the costs of building a self-hosted environment can be high. The third category is impact on qualities. Here, the risk of degrading performance instead of improving it is the most important group of risks. As a result, the user experience is also degrading. This is followed by the risks of reduced availability, safety and security threats, and extra resource consumption. The risks of reduced availability is mostly a problem when auto-scaling is involved in the self-adaptive system. And, if a system is self-adaptive, it can be difficult to ensure that it is safe during the production of a product. If the self-adaptive system is using machine learning, it is more difficult to secure its safety. The risks of excessive resource consumption arise when resources are consumed by a self-adaptive process. The last category is impact on business. The most important group here are the increased costs. For example, when using auto-scaling, a main issue can be that it fails, increasing the infrastructure cost of the users due to bugs in the system. If one loses control over the size of the system, total costs can end up higher than what was agreed upon with the customer. Another issue is trust. For example, flexible manufacturing systems have autonomous behavior, and clients can initially be very skeptical and not trust the system. Another issue is that the systems become more complex and fewer people understand all the details of their behavior, making them harder to understand or fix. Some businesses find self-adaptation not useful, because it might not perform better than the baseline when dealing with dynamic shapes, and the cost model might not be generic enough to predict the performance. To end this subsection, a cross analysis of subjects of adaptation versus difficulties and risks shows that the reported difficulties and risks are similarly distributed across subjects of adaptation.

3) Mitigation of Risks: The industry has some techniques for mitigating risks when engineering self-adaptive systems. We can categorize the techniques in three groups:

- stakeholder-centred
- offline
- online.

Stakeholder-centered techniques are the most important category, with rigorous design and development as the most important technique. One can do careful engineering, so that there are open doors for manual intervention, when necessary, without losing system availability or hindering the automation mechanisms. Having design sessions and possibly enhancing the design in the early phases of development is also done in the industry. Other possible techniques are code review (with an incident as a trigger, for example), human supervision, outsourcing (for example, outsourcing the cloud operation with RedHat or AWS), post mortem analysis, hiring experts, working in pairs, and having clear and sufficient documentation. Offline techniques are the second most frequented category of techniques, with extensive testing being the most important technique. This can include testing each action in isolation before it is provided to the system, automated and human

testing, and running parallel, correlated analyses. Other offline techniques are setting operational boundaries (for example, defining a maximum amount of resources a system is allowed to consume), and encryption. The last category containing online techniques has runtime monitoring and analysis as its most important technique. For example, alerts can track high-level properties that can give some assurance that the system is working fine. Another online technique is different roll-out or roll-back strategies. For example, a slow roll strategy where one only sends the new system traffic in small increments until production baselines are established for load, actual latency, ... This can help with the determination of good parameters. Another technique is to run the system during non-business critical hours, like at night. In this way, there is less chance of interference with business critical systems.

IV. Benefits of applying Self-Adaptation

Further analysis of adaptation problems and kinds of systems leads to the observation that most adaptation problems are applied to all kinds of systems. Each adaptation problem is applied to one or two champion kinds of systems. The most frequently addressed adaptation problem is to optimize the system's performance, except for transportation, finances, and manufacturing. In transportation, detecting and resolving errors is the main adaptation problem. In finances, dealing with changes in the environment is the main problem. In manufacturing, automating tasks is the main problem. The top kinds of systems to optimize system performance can be found in the categories embedded, cyber-physical, and IoT, just like for dealing with changes in the environment. The adaptation problem of detecting and resolving errors has web or mobile as the top kind of system. Looking at the adaptation problems with the benefits, improving user satisfaction and reducing costs are by far the most frequently perceived benefits across all types of problems solved with self-adaptation in the industry. Finally, the top domains where solutions are frequently reused are data management and embedded, cyber-physical, and IoT systems. In manufacturing, solutions are frequently reused.

V. Machine Learning and Self-Adaptation

As stated in III-C2, machine learning is becoming a more popular approach to realizing and enhancing self-adaptation in the industry. Because of its rising popularity, it deserves a section in this article. The recent increase in computer power doesn't slow down this process either. Machine learning has been used for a variety of reasons, ranging from learning a model of the environment of a system during operation to filtering large sets of possible configurations before analyzing them [1]. Data collected by [1] shows that machine learning in papers is mostly used for updating adaptation configurations to improve system qualities, and managing resources to better balance resources. The methods used in machine learning are classification, regression, and reinforcement learning. In this section, we take a closer look at architecture-based adaptation. This type of self-adaptation relies on a feedback loop that monitors the system and its context and adapts the system to ensure its goals, or degrades gracefully if necessary [1]. It is critical to use runtime models that enable the system to reason about change and make adaptation decisions. In the industry, architecture-based adaptation is used for the management of renewable energy production plants, information systems for public administration, and automation of the management of IoT applications. Online verification, mentioned in III-C6, of rigorously specified runtime models helps the system make adaptation decisions. Machine learning can be used to filter the configurations before starting analysis, as formal verification of runtime models can be time-consuming.

A. MAPE-based Self-Adaptation

Architecture-based adaptation consists of a managed system that is controllable and subject to adaptation, and a managing system that performs the adaptations of the managed system. The managed system operates in an environment that is non-controllable, while the managing system realizes a feedback loop that comprises four essential functions: Monitor-Analyze-Plan-Execute that share Knowledge. This is often referred to as MAPE-K or MAPE [2]. The monitor monitors the managed system and the environment of the system. It also updates the knowledge. The analyzer uses this up-to-date knowledge to evaluate the need for adaptation. The planner then selects the best option based on the adaptation goals and generates a plan to adapt the system. Finally, the executor executes the adaptation actions of the plan. The management system is based on a MAPE feedback loop that solves an adaptation problem, and the MAPE feedback loop is supported by a machine learner that solves a particular learning problem.

B. Learning and MAPE Functions

Learning is dominantly used (in papers) to support analysis in order to make decisions in the feedback loop. This is particularly done where machine learning is used to reduce large adaptation spaces so that only the relevant options need to be analyzed. Learning is also often used to support the planning of the feedback loop. One can, for example, adopt an instance-based learning method to implement a hybrid planning approach that selects an optimal planning strategy among the possible strategies. In monitoring, learning can be used to pre-process data using a lightweight classifier in order to learn optimization rules. Learning is not used for the execution functions of MAPE. All types of learning problems are tackled in support of monitoring, analysis, and planning. The dominant implementation of learning is used for updating and changing adaptation rules and policies to support analysis and planning. This is followed by predicting and analyzing resource usage. The majority of the papers use supervised or interactive learning. The dominant learning method used in support of self-adaptation is model-free reinforcement learning. Other popular learning methods used in self-adaptation are support vector machines, traditional artificial neural networks, and linear regression. In one example, a support vector machine was used to detect network attacks in cyber-physical systems.

In another example, an artificial neural network was used to predict qualities of services such as response time and throughput. Finally, linear regression was applied to predict performance and power consumption in yet another example. Supervised and interactive learning have been frequently used over the years, while unsupervised learning has only been used in a limited number of papers. However, unsupervised learning can detect novelty in data without any labeling, which can play a key role in managing complex types of uncertainty.

C. Design Process for Using Machine Learning in Self-Adaptive Systems

[1] presents an initial design process that can help guide designers when applying machine learning to self-adaptive systems that are based on MAPE feedback loops. The process of designing a self-adaptive system begins by defining the adaptation problem. Then you have to design the MAPE feedback loop of the managing system in order to solve the adaptation problem. This involves identifying the knowledge maintained by the feedback loop, the functionality required to monitor the managed system and its environment, analyze runtime data, plan actions for adaptation, and enact these actions. After this, domain knowledge is used to identify the learning problem that supports the MAPE feedback loop. The learning problem is delegated to and solved by a machine learner. Now choose a learning task that fits the learning problem. This choice depends on the learning type and the characteristics of the problem. For example, if there is no labeled data, supervised learning is not possible. The learning type is influenced by the domain characteristics and the MAPE feedback loop. Next, map the learning problem to a learning task. Then, select a specific learning method to solve the task. Finally, implement the managing system and learning method and integrate them with the feedback loop. Test the system and deploy it if accepted. This process provides a high-level guide for integrating machine learning into MAPE-based self-adaptive systems. The flow of activities is conceptual and may be applied iteratively in practice. One can turn this into a practical engineering process with supporting tools.

VI. Research Support

The problems addressed by industry are, in general, similar to those studied by academics. A lot of practitioners of self-adaptation in the industry would place high value on support from research. Certainly, there are problems when self-adaptation is applied at the system level, a module of the system, or the application layer. These problems are mainly related to auto-tuning, auto-scaling, monitoring, and analysis.

VII. Conclusion

Self-adaptation is widely applied in the industry in a wide range of systems and is developed by companies of all sizes. From the survey, the key people concerned with realizing self-adaptation are programmers, project leads, architects, and designers. The companies applying self-adaptation do so because they want to optimize their performance, automate some tasks, and deal with changes in the environments of the system and company. At the end, self-adaptation results in improved utility, user satisfaction, and reduced costs. It is mainly applied at the level of the system or a part or module of it. The most frequent types of self-adaptation are auto-scaling and auto-tuning. These self-adaptation systems are triggered by changing properties of systems and the environment, as well as dynamics in load. Practitioners apply self-adaptation with the help of tools and infrastructure. The most frequent mechanisms for monitoring are resource usage and load on the system. For analysis, the main approaches are data analysis methods and comparison to thresholds. The top mechanisms are auto-scaling and reconfiguration. Ensuring trust in industrial self-adaptive systems is achieved through extensive testing, runtime monitoring, and human supervision. The industry broadly confirms that the use of self-adaptation improves robustness and performance while reducing costs and required resources, and it improves the user experience while reducing the burden of engineers. In research, machine learning in MAPE-based self-adaptive systems is a rapidly growing interest. The problems in self-adaptation that are solved using machine learning are: updating and changing adaptation rules and policies, predicting and analyzing resource usage, keeping runtime models up-to-date, and reducing large adaptation spaces. This is done with support, analysis, and planning. Supervised and interactive learning dominate, primarily to solve regression, classification, and reinforcement learning tasks. We then defined an open process for applying machine learning to self-adaptation that can be extended. Finally, the industry would appreciate support from research for certain problems it faces with self-adaptive systems.

References

[1] O. Gheibi and D. Weyns and F. Quin, "Applying Machine Learning in Self-Adaptive Systems: A Systematic Literature Review," arXiv, 2020. [Online]. Available: https://arxiv.org/abs/2204.01834

[2] J. Kephart and D. Chess. 2003. "The vision of autonomic computing." Computer 1 (2003), 41–50.

[3] D. Weyns, I. Gerostathopoulos, N. Abbas, J. Andersson, S. Biffl, P. Brada, T. Bures, A. Di Salle, M. Galster, P. Lago, G. Lewis, M. Litoiu, A. Musil, J. Musil, P. Panos and P. Pelliccione, "Analysis Report Survey on Self-Adaptation in Industry," https://people.cs.kuleuven.be/ danny.weyns/papers/SAinIndustryBasicAnalysis.pdf, accessed: February 2023.

[4] D. Weyns, I. Gerostathopoulos, N. Abbas, J. Andersson, S. Biffl, P. Brada, T. Bures, A. Di Salle, M. Galster, P. Lago, G. Lewis, M. Litoiu, A. Musil, J. Musil, P. Panos en P. Pelliccione, "Self-Adaptation in Industry: A Survey, " arXiv, 2022. [Online]. Available: https://arxiv.org/abs/2211.03116

Self-Adaptation in Industry

Matias Daneels
KU Leuven

Kortrijk, België
matias.daneels@student.kuleuven.be

***Abstract*—Computing systems get more important everyday in the industry. Computing systems get used in many areas of our society, from healthcare to construction, banks, etc. Self adaptation is a form of a computing system using a feedback loop that automates tasks otherwise performed by operators. Self-adaptation is getting more popular in a variety of domains. Elastic cloud, where the storage of the cloud can change during operation, is the most known example. It is still unclear how much self-adaptation is used in the industry and what the knowledge about it is. To get insight in this information, we performed a large scale survey. The survey was done by practitioners across 21 countries. We got 184 valid responses from the survey. This report does a basic analysis of the results.**

I. Introduction

Computing systems serve as the fundamental building blocks for a plethora of industries, ranging from healthcare and finance to construction. When software assumes a pivotal role in the design and operation of such systems, they are commonly referred to as software-intensive systems. Ensuring that these systems are trustworthy and sustainable poses an immense challenge, given their intricacy, uncertain conditions, and rapid pace.

One approach to tackling the escalating complexity of software-intensive systems is to incorporate a feedback loop within the software system. This loop assumes the responsibility of tasks that were traditionally executed by practitioners, constantly monitoring the system and its environment, as well as the system's goals, and adapting accordingly. The objectives of these systems can span from decreasing costs under uncertain workloads to ensuring optimal performance, addressing errors, and beyond.

The principles of feedback loops and self-adaptation have been a topic of research within the academic industry for numerous years. The basics of self-adaptation can be found in [1], whereas a more comprehensive background is available in [2]. All figures used are from [3]

To obtain insight into the current state-of-practice of self adaptation within the industry, we conducted a survey spanning 21 countries and receiving 184 valid responses. In the subsequent sections, we will delve into the outcomes of this survey, highlighting the findings and their implications.

II. Introduction to Self-Adaptation

Self-adaptation is a powerful approach to designing and building software systems that can autonomously and intelligently adjust their behavior in response to changes in their operating environment. This capability allows self-adaptive systems to operate effectively and efficiently even as their environment changes or as new requirements arise.

Self-adaptation involves the use of sensors that collect data about the system and its environment, and of feedback loops that use this data to make decisions about how the system should respond to changes in its environment. These decisions can be based on pre-defined rules, machine learning algorithms, or a combination of both.

The use of self-adaptation is particularly important in today's fast-paced and dynamic world, where software systems must be able to adapt to new conditions quickly and efficiently. It has become an essential tool for building complex and distributed systems, such as those found in cloud computing, cyber-physical systems, and the Internet of Things.

One of the key benefits of self-adaptive systems is their ability to improve their own performance over time. By continually monitoring their own behavior and the behavior of their environment, self-adaptive systems can adjust their behavior to optimize performance, reduce costs, and minimize risks.

However, self-adaptation is not without its challenges. Designing and engineering self-adaptive systems can be complex and require a deep understanding of the system and its environment. It can also be difficult to ensure the correctness and reliability of self-adaptive systems, particularly in safety-critical applications.

One example of self-adaptation is in cloud computing, where self-adaptive systems can adjust their resource allocation according to the current workload. This helps to optimize the system's performance and reduce costs. Another example is in the field of autonomous vehicles, where self-adaptive systems can detect changes in the environment, such as road conditions or weather, and adjust their behavior accordingly to ensure safe and efficient driving. In the healthcare industry, self-adaptive systems can personalize treatment plans for patients based on their individual needs and responses to treatment. This can lead to better outcomes and reduced costs. These examples demonstrate how self-adaptive systems can be applied in different domains, addressing various challenges and delivering benefits.

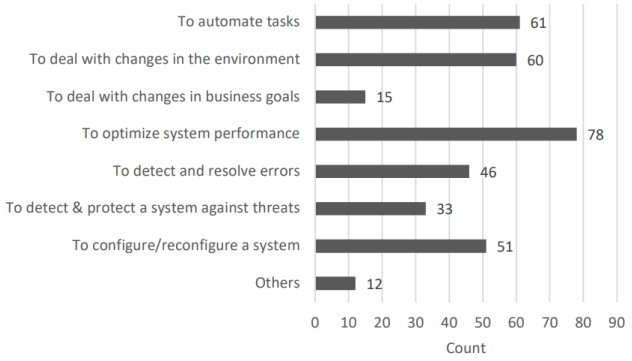


Figure 1: Problems to apply self-adaptation.

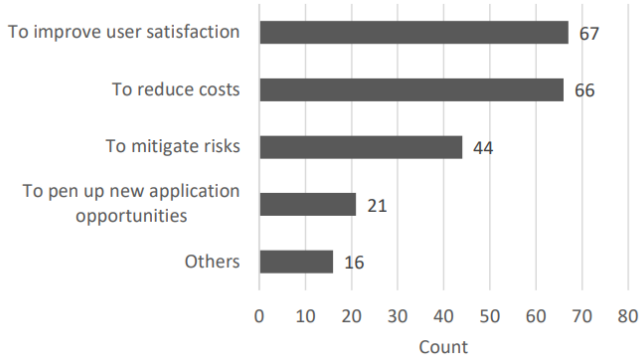


Figure 2: Business motivations to apply self-adaptation.

III. Results of Survey

A. Drivers for Applying Self-Adaptation

First of all, we want to know what drives practitioners to apply self-adaptation and the types of problems they solve using self-adaptation.

For which problems does self-adaptation get applied? On average, the participants gave 3.6 types of problems. The results can be found in figure 1. Based on the outcomes derived from this question, we can see that practitioners tend to utilize the approach of self-adaptation across a multitude of problem domains. The technique of self-adaptation is predominantly leveraged by practitioners for the purpose of optimizing the performance of a system, automating tasks, and addressing changes or fluctuations that occur within the environment. Such findings highlight the importance and effectiveness of self-adaptation as a reliable and versatile method in the problem-solving toolkit of practitioners. We can see that self-adaptation is a versatile approach that has the potential to be employed in a vast range of problems.

What are the main business motivations to apply self-adaptation? On average, the participants gave 2.1 business motivations to apply self-adaptation. The results can be found in figure 2. The business motivations that drive practitioners to employ self-adaptation in their business operations are very versatile. The main business motivations practitioners use self-adaptation for are to improve user satisfaction, to reduce costs or to mitigate risksk. Furthermore, the adoption of self-adapation can also enable practitioners to pen up new application opportunities that may not have been previously possible using traditionel approaches

What could be benefits of applying self-adaptation? We got a lot of interesting answers on this question. The pracititoners have reported experiencing an improvment in utility when applying self-adaptation. In addition, they also see improved human interaction and cost savings. The advantages of employing self-adaptation span across a diverse array of domains and are not limited to any one specific area of application. The versatility and flexibility of self-adaptation make it a highly adaptable approach that can yield positive outcomes in a wide range of problem-solving scenarios.

After a close analysis of the results from the first set of questions, it is clear that self-adaptation is a widely adopted practice that is employed in a wide range of domains in the industry. One of the most crucial applications of self-adaptation is its ability to optimize performance, automate tasks, and tackle changes in the environment, thus providing a tailored and flexible system for the organization. Companies have identified several business motivations to incorporate self-adaptation into their systems, which include enhancing user satisfaction, trimming down expenses, and managing risks more efficiently. The implementation of self-adaptation offers numerous benefits, including improved utility, cost savings, and enhanced human interaction, which can ultimately lead to a more robust and sustainable business model. In summary, the use of self-adaptation techniques in the industry has become increasingly vital, given its ability to provide customized and efficient solutions to tackle the ever-changing landscape of the industry.

B. Characterisation of Self-Adaptation

Explain a concrete self-adaptive system you worked with. We can devide the answers in 3 categories:

- Subject of Adaptation
- Type of Adaptation
- Trigger of Adaptation

The majoritiy of practitioners utilize either a complete system or a modules as the subject of adaptation. Other possibilities that prarctitioners use as the subject of adaptation are platform the platform layer and the application layer. While the complete system and module are more commonly used as the subject of adaptation, the platform layer and application layer can also offer valuable opportunities for adaptation. The platform layer, for instance, provides a foundation upon which applications can be built, this means that adaptation at this layer can potentially impact multiple applications. Meanwhile, adaptation at the application layer can target specific functionality within a given application.

Practitioners in various problem domains typically leverage three primary types of adaptation to improve systems. The first of these is auto-scaling, which is particularly popular in cloud

management applications. Auto-scaling enables practitioners to dynamically adjust the resources allocated to a system based on changes in demand, ensuring that the system remains responsive and efficient even during periods of high usage. The second primary type of adaptation employed by practitioners is auto-tuning. Auto-tuning involves the automatic adjustment of system parameters to optimize performance and minimize resource usage. This technique is commonly used in a wide range of applications, including databases, web servers, and network management systems. At last, practitioners also utilize self-adaptation techniques for monitoring and analysis, which can provide valuable insights into system behavior and help identify potential areas for improvement.

After analysis of results of the second set of questions, we can conclude that self-adaptation is a widely practiced technique utilized in various levels of industrial software systems. This technique can be implemented at different levels, such as system level, parts of systems, or support systems, depending on the specific requirements of the organization. The most prevalent forms of adaptation employed in the industry are auto-scaling, auto-tuning, and monitor/analysis, which enable organizations to tailor their systems to meet their unique needs. Adaptations in software-intensive systems are primarily triggered by either system properties or environmental properties, this emphasizes the significance of designing adaptable systems that can withstand unexpected changes in the environment. In summary, self-adaptation has become an indispensable tool for the industry, as it offers a flexible and responsive approach to managing complex software systems.

C. Application of Self-Adaptation

What are mechanisms or tools used to monitor a managed system during operation? The participants gave a total of 146 instances. We divide the answers in 3 categories:

- Monitoring Metrics
- Monitoring Mechanisms
- Monitoring Tools

We observed that practitioners primarily rely on resource usage as a metric for monitoring when employing adaptive techniques. By monitoring resource usage, participants can gain valuable insights into the performance and efficiency of a system, allowing them to identify potential areas for optimization and improvement. Additionally, participants also commonly utilize load and reliability as key metrics for monitoring. Monitoring the load on a system enables participants to determine how much demand the system is experiencing at any given time, allowing for more accurate resource allocation and capacity planning. Meanwhile, reliability monitoring is crucial for ensuring that a system remains available and functional at all times, which is particularly important for critical applications or systems.

We observed that practitioners tend to prioritize the monitoring of resource usage, followed closely by load and reliability metrics. Resource usage serves as a critical metric for assessing the performance and efficiency of a system, enabling practitioners to identify areas where resources may be underutilized or overutilized, and make informed decisions regarding resource allocation and optimization. Load monitoring, on the other hand, provides insights into the level of demand being placed on a system, and can help inform decisions related to capacity planning and allocation. Finally, reliability monitoring is crucial for ensuring that a system remains available and functional at all times, and can be a key factor in maintaining user satisfaction and trust in the system.

The most common monitoring mechanisms used by practitioners are environment sensors and system sensors. These sensors can give a good insight in the system which give a good foundation to make adaptations when necessary.

Practitioners mostly use Kubernetes and Prometheus as a tool for monitoring. Kubernetes is an open-source container orchestration system that provides a robust platform for deploying, scaling, and managing containerized applications across clusters of machines. Meanwhile, Prometheus is an open-source monitoring system that collects and stores metrics from various systems, providing a comprehensive view of system behavior and performance.

What are mechanisms or tools used to analyze conditions of a managed system during operation? On average, there was given 1.5 mechanisms/tool per participant. We divide the answers in 2 groups:

- Analysis Mechanisms
- Analysis Tools

When it comes to the techniques used by practitioners for analysis in the context of self-adaptation, several approaches have emerged as popular choices. Data analysis is the most popular, allowing practitioners to examine large datasets and identify patterns or trends that may be relevant to system performance. Another commonly used approach is comparison to threshold, whereby practitioners establish predefined thresholds for system performance metrics and compare actual performance against these thresholds to determine whether adaptation is necessary. Metric calculation and learning are also popular techniques for analysis, with practitioners leveraging machine learning algorithms and other techniques to gain insights into system behavior and identify opportunities for optimization.

For analysis, practitioners use a variety of different tools. AWS analysis tools and Kubernetes stack are the most popular. Dynatrace is also used by practitioners.

What are mechanisms or tools used to change a managed system or parts of it during operation? The participants provided an average of 1.3 mechanism/tool per participant.

As for mechanisms used to change a managed system or parts of it during operation, 2 mechanisms are popular. Scaling mechanisms and reconfiguration are the most commonly used. Scaling can be performed horizontally, by adding additional instances of a component to distribute load across multiple nodes, or vertically, by increasing the resources allocated to a single instance of a component to improve its processing capabilities. Practitioners commonly use automated scaling mechanisms to enable rapid, responsive adjustments to resource allocation, based on real-time monitoring of system

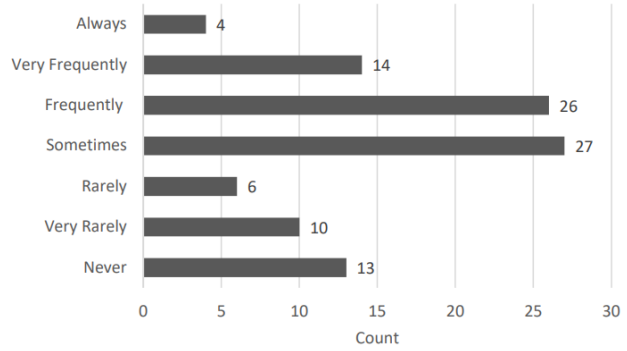


Figure 3: Do the practitioners reuse solutions?

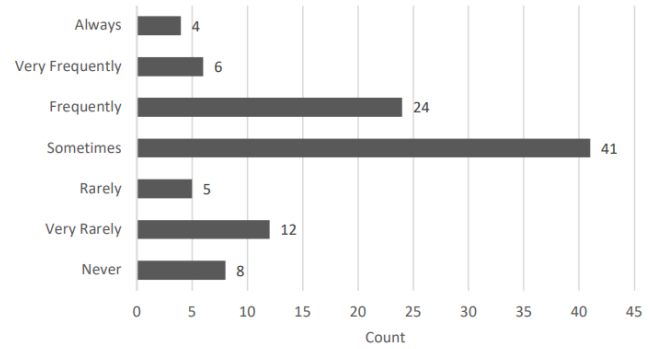


Figure 4: Did the practitioners encounter difficulties when engineering or maintaining self-adaptive systems?

behavior and performance. Reconfiguration involves modifying the configuration of a system or its components to optimize performance or address specific issues. Reconfiguration may involve modifying the configuration of individual components, adjusting system parameters or settings, or changing the overall architecture of the system to better align with specific performance goals or requirements.

Do you reuse solutions to realise self-adaptation in systems you work with? The results can be found in figure 3. The majority of the practitioners (71/100) said they reuse at least sometimes solutions in self-adaptive systems. The others rarely or never reuse solutions. From the 29 that rarely or never reuse solutions, 11 mentioned they don't reuse solutions because of the difference in problems.

How do you ensure that you can trust the self-adaptive solutions you build?

Most practitioners ensure that they can trust their self-adaptive solutions by testing and verification. By testing and verification they can change certain parameters where needed. Stakeholder centered techniques are also used to gain confidence in the self-adaptation systems. These techniques may involve soliciting feedback from end-users or other stakeholders to ensure that the system is meeting their needs and expectations, or engaging in user-centered design practices to ensure that the system is intuitive and user-friendly. Finally, human supervision is also widely used by practitioners to ensure that self-adaptive systems are performing as intended. This may involve manual intervention to override or adjust system behavior in certain scenarios, or ongoing monitoring and review of system performance to identify potential issues and areas for improvement.

After examining the results of the third set of questions, it is clear that resource usage and system load are the primary monitoring metrics utilized in self-adaptation. To track these metrics, sensors in the environment and the system are the most widely used monitoring mechanisms. To achieve self-adaptation, practitioners use a variety of analysis mechanisms, including auto-scaling and reconfiguration. The adaptation can be a mix of semi or fully automated, depending on the organization's requirements. Practitioners often reuse solutions to enable them to customize their systems to meet their unique needs. Building trust in the system is critical and is mainly achieved through rigorous testing, runtime monitoring and alerting, and human supervision, which provides organizations with a sense of control over their systems. These results show that is important for organizations to implement effective monitoring and analysis mehcanisms that enable them to track critical metrics and ensure the system reliability and efficiency.

D. Difficulties, Problem Support, and Opportunities

Did you encounter particular difficulties when engineering or maintaining self-adaptive systems you worked with? The results can be found in figure 4. Around 40% of the practitioners said they somtimes encounter difficulties when applying self-adaptation. 30% of the practitioners said they encounter difficulties with applying self-adaptation frequently, while the rest rarely encounter difficulties.

The participants described a total of 140 difficulties. We divide the problems in 4 categories:

- Design Issues
- Lifecycle Issues
- Runtime Issues
- People and Process Issues

Most of the difficulties reported are problems with the design of self-adaptation, more specificly, problems with reliable/optimal design and with design complexity. This complexity can arise from a variety of factors, including the need to incorporate a wide range of sensors and monitoring mechanisms, the need to adapt to rapidly changing system and environmental conditions, and the need to integrate multiple subsystems and components into a cohesive whole. In terms of reliable and optimal design, practitioners may struggle with identifying the most effective strategies for adapting system behavior in real-time, particularly in complex and dynamic environments.

The second most diffcuties ecountered were lifecycle issues. One of the most common lifecycle issues reported by practitioners is the need for tuning and debugging. This can involve identifying and correcting errors in the self-adaptive

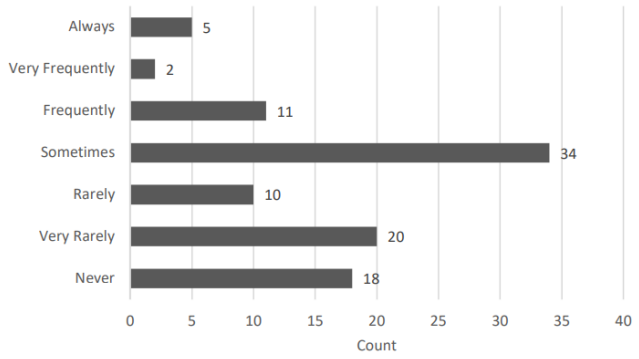


Figure 5: Did the practitioners face any risks when engineering self-adaptive systems?

system, as well as fine-tuning its behavior to ensure optimal performance in different contexts. Tuning and debugging can be particularly challenging in large or complex systems, where there are many interdependent components and subsystems that must be carefully balanced and optimized.

Another common lifecycle issue encountered by practitioners is the limitations of available tools and methods. Despite significant advances in self-adaptive technology in recent years, there are still many situations where existing tools and methods may not be sufficient to address the needs of a particular application or domain. This can lead to difficulties in implementing self-adaptive solutions that are effective, efficient, and robust.

Runtime issues and runtime as well as people and process issues were less common issues but still reported by some practitioners.

Did you face any risks when engineering self-adaptive systems? The results can be found in figure 5.

34% of the pracitioners reported facing risks at times when engineering seslf-adaptive systems. 18% frequently to always ecounter risks and 48% rarely to never face risks. It is noteworthy that over 50% of the pracitioners reportedd facing risks with the application of self-adaptation at least some of the time.

The participants provided a total of 66 instances of risks they encounter while engineering self-adaptive systems. We can divide the answers in 4 categories:

- Faults
- Difficulties with Development/Operation
- Impact on Qualities
- Impact on Business

The survey results indicate that 20% of the risks encountered by practitioners can be attributed to faults. These faults are characterized by incorrect functionality, wrong results, misconfiguration, and network failure. It is worth noting that these types of faults can have significant implications for the performance and reliability of self-adaptive systems. Therefore, it is crucial for practitioners to identify and address these faults in a timely and effective manner to minimize their impact on the system's overall performance.

In addition to the risks associated with faults, practitioners also reported encountering difficulties with the development and operation of self-adaptive systems. These difficulties can be attributed to several factors, including challenges in managing environmental uncertainty, testing the system's behavior, and building the system itself. Such challenges can lead to delays in the development process, as well as increased costs and reduced system performance. Therefore, it is essential for practitioners to develop effective strategies and tools to manage these difficulties and ensure the smooth development and operation of self-adaptive systems.

The practitioners also reported the risk of multiple qualtites being impacted, like the performance, the availability, safety, and security. These risks were identified as potential concerns that could arise from the application of self-adaptation techniques.

Lastly, practitioners also reported that increased costs and loss of control and trust are significant risks associated with the application of self-adaptation, which can have a negative impact on the business. These last risks are more on the scale of the business than on the scale of the individual.

How did you mitigate the risks that you faced? To manage the risks, the participants mentioned 87 valid methods to mitigate the risks that they faced. The answers can be grouped in 3 groups:

- StakeHolde-Centered Techniques
- Offline tehcniques
- Online techniques

The most common way used by practitioners to mitigate the risks they face are stakeholder-centered techniques. These techniques mostly consist of rigorous desing and development, code review, and human supervision. These techniques are deemed crucial for ensuring the quality, reliability, and security of the developed system, and for mitigating potential risks and negative impacts on stakeholders.

Another way pracititoners can mitigate the risks they face when applying slef-adaptation is offline techniques. An example of this can be extensive testing.

Lastly, online techniques can also be used to mititgate risks faced when applying self-adaptation. Here, runtime monitoring is the most common.

Have you faced or seen any problems of self-adaptation for which you would appreciate support from researchers? The results can be found in figure 6.

Partipants provided a total of 113 problems for which they would appreciate support from researchers. We divided those instaces in 4 categories:

- Engineering
- Quarantees
- Data Interaction
- User Interaction

Out of all the reported problems (42.5% of the total), 48 problems are related to the engineering of self-adaptive systems.

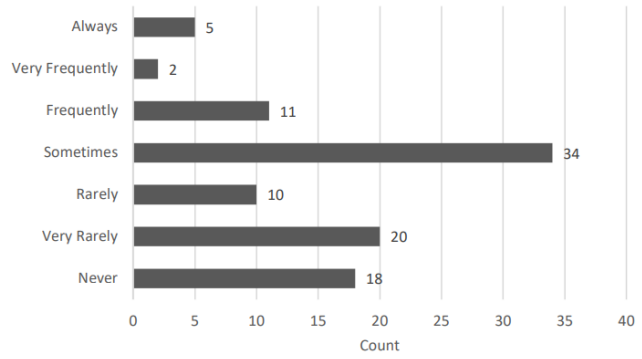


Figure 6: Did the practitioners face or seen any problems of self-adaptation ofr which they would appreciate support from researchers?

The majority of these problems are associated with architecture and design, followed by the adoption of self-adaptation. This underscores the significance of addressing these challenges to improve the efficacy and efficiency of self-adaptive systems. By addressing these challenges, practitioners can ensure that self-adaptive systems perform optimally and are better equipped to meet the evolving demands of the environment in which they operate.

In the reported problems, a significant portion of them are related to guarantees, providing trustworthiness and dealing with unknowns. Another common issue reported by practitioners is related to problems with data, which includes data access, data governance, and machine learning. Additionally, user interaction was also identified as a concern, specifically automation and user experience. It is essential to address these issues to ensure that self-adaptive systems meet the needs and expectations of their users and are easy to use and interact with. There are still many domains left to explore in this area, and researchers have ample opportunities to delve deeper into these issues.

Out of the 184 participants, 101 (54.4%) perceive new opportunities for utilizing self-adaptation, whereas 83 do not. This suggests that a significant portion of the participants recognize the potential benefits and applications of self-adaptation in various domains.

The results of the fourth and final set of the survey show that a majority of participants experience difficulties when engineering or maintaining self-adaptive systems. The main challenges are with reliable/optimal design, design complexity, and tuning/debugging. Also, about half of the participants encounter risks with self-adaptation, including incorrect functionality, difficulty managing environmental uncertainty, and degraded performance and increased costs.

Almost half of the practitioners report that they would appreciate support from researchers to address the problems they face, particularly related to the engineering of self-adaptive systems, guarantees, and management of data.

About half of the participants see future opportunities for applying self-adaptation, particularly in relation to autonomous operation, data management, and machine learning. This is positive for the future of self-adaptation in the industry and the future of research around self-adaptation.

IV. Conclusion

This paper presents a survey-based study on the use of self-adaptation in industry, with valid responses obtained from 184 practitioners worldwide, including 100 with experience in engineering self-adaptive systems. Through close analysis of the data, the study offers several key observations, including the extensive application of self-adaptation in various domains and the primary types of adaptations applied, such as auto-scaling, auto-tuning, and monitoring/analysis. The study highlights the importance of tools and infrastructure for realizing the different functions of self-adaptation, and the critical role of human supervision in ensuring trust in industrial self-adaptive systems. However, risks associated with applying self-adaptation in practice include incorrect functionality, difficulty in managing environmental uncertainty, degraded performance, and increased cost.

The results offer valuable insights for researchers and practitioners alike. Researchers can use these findings to better align their research with industry needs, taking into account the importance of improving user satisfaction, reducing costs, and mitigating risks in addition to uncertainty mitigation. Practitioners can assess the level of their current practice in applying self-adaptation and use the results to guide their future efforts.

The study identifies several prospects for applying self-adaptation in practice, including realizing full autonomous operation, exploiting machine learning, improving quality and security, and applying self-adaptation for maintenance. Moreover, the study identifies key opportunities for industry-research collaborations, such as consolidating best practices, modeling paths for adoption of self-adaptation in industry, supporting advanced features, ensuring trustworthiness, and governance of data. The study advocates for moving from a human in the loop to a human on the loop approach, where humans oversee the system to ensure trust.

References

[1] Danny Weyns. *An Introduction to Self-adaptive Systems: A Contemporary Software Engineering Perspective*. John Wiley & Sons, 2020.

[2] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, et al. Analysis report survey on self-adaptation in industry. 2022.

[3] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, et al. Self-adaptation in industry: A survey. *arXiv preprint arXiv:2211.03116*, 2022.

Uses of self-adaptation in practice

Dieter Demuynck
Science & Technology
Catholic University of Leuven campus Kulak Kortrijk
Kortrijk, Belgium
dieter.demuynck@student.kuleuven.be

***Abstract*—Computer software is increasingly used in all kinds of systems, including traffic regulation, manufacturing, healthcare, etc. Systems in which software plays an important role in the design, construction and working of the system are called software-intensive. When a software-intensive system is equipped with a feedback loop to take over tasks from humans or to mitigate uncertainties, we call these systems self-adaptive [5]. This paper aims to document the insights gained from a survey, which aims to measure the use and knowledge of self-adaptation in software-intensive systems in practice, as well as the problems it aims to solve and the benefits and risks of the use of self-adaptive systems in practice. In this survey, 184 practitioners from 21 different countries gave valid answers to the survey.**

Introduction

Software-intensive systems are increasingly applied in practice, but to keep these systems reliable and sustainable is an increasingly difficult task. This is because these systems are more and more complex, which makes it more complicated to create, maintain and integrate. Thus, practitioners and academics are looking for methods to facilitate these processes. One such example is DevOps, where an organisation integrates the two worlds of **Dev**elopment and **Op**eration, to, among other things, lower the chance of problems due to miscommunication and to have existing problems be resolved faster [1].

Another increasingly common approach is to delegate control of these complicated tasks from humans to software, through the use of a feedback loop. The system can then adapt itself during runtime, which allows it to work with uncertainties in its environment, as well as handle unexpected errors, or even to reduce costs by reducing the amount of used resources [5].

Academics have already done a considerable amount of research on the principles of feedback control in software-intensive systems. As a result, a substantial body of research has been conducted on the principles, models, languages, processes, and methods involved, yielding extensive knowledge on the topic. Moreover, researchers have identified four classic goals, namely self-optimisation, self-healing, self-protection and self-configuration [5]. Concurrently, these principles were also studied and used in the industry. Researchers have raised the question of whether their knowledge from the academic world is being applied in the industry. In this paper, we document the insights of this research.

I. Related items

An example of an interesting related work is *An Introduction to Self-adaptive Systems, A Contemporary Software Engineering Perspective* [3], which aims to create a comprehensive overview on the field of self-adaptation from the past two decades. Another example is *Dealing with Drift of Adaptation Spaces in Learning-based Self-Adaptive Systems using Lifelong Self-Adaptation* [2], which aims to describe a new technique which could work with changing the factors that a self-adaptive system can adapt.

II. Research questions

The main purpose of the survey is to get a better understanding of the state of practice of self-adaptation in the industry. The main goals of the survey are to shine a light on the motivations of practitioners to apply self-adaptation, the type of problems solved by self-adaptation, how self-adaptive systems are designed and developed, as well as the difficulties and the risks, and possible future opportunities the industry sees in the use of self-adaptation [5].

To better understand these topics, the survey is constructed with four main **R**esearch **Q**uestions in mind:

- **RQ1:** What drives practitioners to apply self-adaptation in software-intensive systems?
- **RQ2:** How do practitioners characterise self-adaptation?
- **RQ3:** How do practitioners apply self-adaptation in industrial software-intensive systems?
- **RQ4:** What are the experiences of practitioners with applying self-adaptation and do they see opportunities for how and where to apply self-adaptation?

RQ1 aims to investigate the motivations of practitioners to use self-adaptation, as well as the types of industrial systems where it is applied, and the problems solved by it. This could answer the question if these problems are the same as the academic world expects. The expected problems are management of complex, software-intensive systems based on high-level goals, and dealing with operating conditions that are difficult to predict before deployment, i.e. mitigating uncertainties during the runtime of the system.

With RQ2 researchers aim to investigate how practitioners characterise self-adaptation, including the used terminology and any (emerging) standard practices. This should shine a light on if there are any differences in the viewpoints on what constitutes self-adaptation, or if self-adaptation is considered useful in general.

RQ3 aims to examine how self-adaptation, with an emphasis on the mechanism, tools, benchmarks and processes employed in the industry. There is also a focus on the degree of automation as well as the role of humans in the runtime adaptation. The industrial practices are also compared to the solutions developed by academics. The survey then polls the practitioners on their trust in their systems with self-adaptation. These answers will provide insights into the best practices for self-adaptation, what academic solutions are already adopted by practitioners, as well as highlight opportunities for future research and improvements to practical applications.

RQ4 tries to determine the difficulties and risks that practitioners experience with the design, implementation, etc. of self-adaptation in practice. We also attempt to get more information on how academics could best help practitioners, by asking where practitioners which problems they would like support with from researchers and what future opportunities they see for self-adaptation that are not yet exploited. The answers to RQ4 will fill the gap between industry and academia, and identify problems and risks on which collaborative studies could be done to address these challenges.

III. Aim of this article

This article aims to summarize *Self-Adaptation in Industry: A Survey* by Danny Weyns, Ilias Gerostathopoulos et al [5], as well as give insights relevant to industry. This article will give empirically grounded overview of the state-of-the-practice in the application of self-adaptation, as well as insights for practitioners to assess the level of their current practice in applying self-adaptation and additional prospects for applying self-adaptation in practice and opportunities for industry-research collaboration.

IV. Methodology

The research questions are answered through the use of an online questionnaire. The researcher in the original paper [5] contacted in total 355 practitioners from different companies, from 21 different countries, and invited them to fill in an online survey. The participants had to be active in different domains that are representative of software-intensive systems, and each needed to have the required expertise to fill in the questionnaire. The invitations were sent by personalized e-mails in different batches from November 30, 2020 until July 31, 2022.

The participants are first familiarised with the term self-adaptation, after which they receive a set of both open and closed questions. There is often an extra text field appended to the closed questions' answers such that the participants are capable of giving an extra option not yet listed in the possible answers. Open questions have only a text field.

The closed questions are analysed using descriptive statistics and quantitative data analysis. Open questions were analysed through qualitative data analysis, particularly through inductive reasoning to construct codes and infer categories from the data by labelling occurrences of codes and grouping them into categories. These codes were constructed by the original authors of the paper [5] in sub-teams of two to three coders. Codes were added to small coherent fragments of the comments given by the participants.

V. Results and insights

A. Demographic information

In the survey, 100 out of 184 participants claim they have worked with self-adaptive systems before. Considering the participants are selected in such a way that they often work in different companies, or at the very least have different functions within one company, we can conclude that a large portion of companies already apply self-adaptation to their systems. The participants also gave an insight into what kind of systems they develop. The most common types of systems were web or mobile, embedded systems, cyber-physical, Internet of Things (IoT), data management, and cloud. These types were mentioned by half of the participants.

We notice that self-adaptation is more commonly applied when working in ICT communications and networks, IT infrastructure, finances, and especially transportation and cloud. If your company develops systems in these fields, it might be worth investigating and perhaps applying self-adaptation. In contrast to these fields, the system type "web/mobile" seems to apply self-adaptation significantly less. Of the 35 participants (19.3% of total participants), only 15 (15.0% of participants who work with self-adaptation) apply self-adaptation in this field. If you work in this field, there might be an opportunity to be one of the first to apply self-adaptation.

On the other hand, monitoring, analytics, and control, as well as services, and quality and security, seem to be the most common focuses for systems.

Roughly half of the companies of the participants have more than 100 employees which work as software engineers. This still holds true when we filter the participants to only select those participants that work with self-adaptive systems. The other half works for smaller to medium sized companies.

On average, each participant appear to have 1.6 roles within their company, while those that work with self-adaptive systems have 1.5 roles on average. We notice all roles (programmer, project lead/manager, ...) are equal among the total and only those that work with self-adaptation, with the only exception to this being the role of researcher: 9 out of the 10 participants who work as researchers within their company work with self-adaptive systems.

Lastly, we note that roughly 70% of participants have at least nine years of experience as a software engineer, and this changes to 76% when only selecting those that work with self-adaptive systems.

B. RQ1: Motivation for Self-adaptation

The questions in RQ1 aim to poll which problems the participants aim to solve using self-adaptation, what the most prominent motivations of its use are, and what benefits self-adaptation offers or has provided in its application.

From the answers of the 100 participants who work with self-adaptation, we conclude that the main problems to apply

self-adaptation for are optimizing system performance (78%), automating tasks (61%), dealing with changes in the environment (60%), and (re-)configuring a system (51%).

The most prominent motivation for the use of self-adaptation appears to be user satisfaction (67%), reducing costs (66%) and mitigating risks (44%). Penning up new application opportunities was indicated the least, with 21 indications.

Participants mentioned the benefits they experienced with the application of self-adaptation. The most common benefit appears to be savings in costs, with an example quote: "the cheaper bills for running this in an efficient manner in e.g. a cloud service". Self-adaptation was also often applied for robustness, creating a better error-handling mechanism, without the need for human intervention. It can also improve user experience through e.g. prompt website responses, as well as support the engineers by "[removing] most of the optimization burden from programmers, so they can be more productive."

Key insights on **RQ1** from the results are that there are many domains in industry which apply self-adaptation in numerous systems. Practitioners apply self-adaptation with the goal of improving performance, automating tasks and dealing with changes in the environment. The most common business goals are improving user satisfaction and reducing costs, and often also mitigating risks. The most prominent advantages of using self-adaptation is improved utility, savings on costs and resources, improved human interaction and handling dynamics.

C. RQ2: Characterisation of Self-adaptation

The 100 participants who worked with self-adaptation were asked what types of systems they developed. From their responses, three categories of characteristics were identified. These categories are *subject*, *type* and *trigger*. Subject refers to the system or part that is adapted. Type refers to the kind of adaptation that is applied. Trigger refers to the source that results in the system initiating an adaptation.

The most common subjects are the system and module, i.e. a part of a system. These are followed by the platform layer, the application layer, and the cluster. Less commonly, the network, and a mixture of different subjects were mentioned as the subject for adaptation. The most commonly mentioned types are auto-scaling and auto-tuning. An example of auto-scaling in practice is the auto-scaling of a cluster based on the resource usage, and an example of auto-tuning would be a feeding robot, where the amount of food is adjusted to a set of rules and the amount left from a previous feeding. Monitor and analysis is also a common type of adaptation. When conditions are met, the system will more closely monitor and analyse the data. Here, other adaptation functions are possibly done using the human in the loop. One participant claims he uses AWS alarms to monitor performance, where monitoring starts when a significant amount of HTTP 400/500 errors are received. Automated reconfiguration is a little less common, but still mentioned quite often, as a type of adaptation. The most prominent triggers for adaptation are system properties, such as when the network resources start to get exhausted, and environment properties, such as a water leak in an IoT system, for household insurance.

Key insights on **RQ2** from the results are that, at the very least for industrial level software-intensive systems, self-adaptation is applied at different levels of a system, from the system as a whole to parts of the system and other support systems. The dominating types of adaptation are auto-scaling, auto-tuning, and monitoring and analysis. Triggers for adaptations in industrial software-intensive systems are primarily the properties of systems and their environment, dynamics in system load, relevant events, and user interactions. We also note that technologies such as elastic cloud and auto-scalers are key enablers for the realisation of self-adaptation in practice.

D. RQ3: Application of Self-adaptation

1) Monitoring the system: When asked what mechanisms or tools the participants that work with self-adaptive systems use to monitor and manage systems during deployment, the participants gave answers that seemed to focus on both "what" is being monitored, and "how". We could define three categories, namely *monitoring metrics*, *monitoring mechanisms*, and *monitoring tools*.

The most common monitoring metric is resource usage, such as CPU and memory usage, and time and power consumption. The load is also quite common, including but not limited to the amount of incoming HTTP requests, or the amount of queries to run. The third most mentioned metric is a reliability metric, such as the AWS lambda metric keeping track of the sum of 400/500 errors in the past few minutes.

The most common monitoring mechanism is the use of environment sensors, such as Lidar, camera, GPS, etc. Also note-worthy is the use of logging systems, where e.g. errors as well as successful operations are logged.

The most common monitoring tools are Kubernetes monitoring, and Prometheus. Just slightly less common is AWS monitoring. Some other mentions are Azure monitoring, Datadog, Splunk, cAdvisor and Elasticsearch.

2) Analysing conditions of the system: Another interesting topic is what mechanisms the participants use to analyse the conditions of the self-adaptive system. Two categories emerged in the answers, namely analysis mechanisms and analysis tools.

As for analysis mechanisms, data analysis methods were the most common, such as rolling averages and similar algorithms, or statistical inferences. Almost as common was the use of threshold values, to which the monitored values would be compared. Other common methods are metric calculations, such as measurements of traffic load, CPU utilization, etc., and also learning, such as machine learning.

The most common tools for analysis were AWS analysis tools and Kubernetes stack. Dyntrace, as well as Splunk, JMX, Jasmina, Kibana, Openshift, and Azure were also mentioned.

3) Changing (parts of) the system: Next up are the mechanisms and tools used by practitioners to adapt the system, or a part of it, based on the measurements.

The most prominent mechanism is a scaling mechanism. This means adding more instances, servers, or workers to the system whenever it is needed, to e.g. decrease average response time for the users. The mechanism of reconfiguration is also commonly applied. An example of this is to adapt the time between packages being sent over a network, as well as the number of packages sent, possibly to reduce network usages when the system is under a high load, or to increase response time when the system is under a small load. Some non-automated mechanisms are also used. The machine could ring an alarm, and possibly even give a suggestion on the next action, which an engineer could then either approve, or perform manually. Lastly, the system may simply restart certain components, or fall back on previous versions, when errors or unhealthy workloads are noticed.

The most common tools for enacting change are Kubernetes and AWS. Openshift and Dynatrace were also mentioned a few times.

4) Degree of automation: The degree of automation applied in roughly half of the companies where self-adaptation is applied in their systems, seems to mostly be a mixture of fully-automated systems and semi-automated systems. A little less common, at only three tenths of the aforementioned companies, is the use of only semi-automated systems, then at roughly two tenths there's only use of fully-automated systems. It's noteworthy that one participant claims their systems are fully-automated until first incident. This form of automation could possibly help guarantee safety and robustness of a system.

5) Reuse of solutions: It's clear that the majority of participant reuse solutions for self-adaptation. A whopping 71% reuse their solutions at least sometimes. The most commonly mentioned subject of reuse was modules. Self-adaptation mechanisms programmed for one system could possibly be used by another system. Scripts and algorithms are also often reused, such as threshold algorithms, where the threshold would be adapted for the specific use case. Internal libraries are also mentioned quite often. Apart from code, design artifacts are also quite commonly reused. One example is the use of design patterns, e.g. for auto-scaling in multiple cloud native applications. Architecture such as AWS stacks are also reused. Lastly, specifications, such as policies, configuration files, and templates, can also be reused, as well as IT infrastructure (frameworks & platforms) and procedures (such as processes and pipelines).

Of course, there can also be important hurdles for reusing solutions. Participants who very rarely to never reuse solutions do this seemingly because the problems they want to solve are simply too different to allow reusing solutions. A little less commonly, a lack of experience, or maturity with the subject of self-adaptation, seems to be an issue. This could possibly be overcome by hiring an individual who's proficient at applying self-adaptation. Though uncommon, the system structure could be a problem due to e.g. high coupling. This could be an argument for decoupling system modules as much as it is reasonably possible. Equally rarely, organisational concerns could prevent reuse of solutions, due to legal reasons.

6) Trust in the system: A system must be trustworthy to be used in practice. To ensure trust in their system, practitioners appear to mostly use extensive testing, such as unit, module, and system tests. Runtime monitoring and alerting is also quite common. Examples of this are results being tracked over time when new capability is added to a system, or the use of alerts to track the high-level properties of a system. Human supervision is the third most common method. In some cases, people would supervise the system until a level of confidence in the system was reached, and/or human supervision would gradually be reduced after the system going live. Similar methods for ensuring trust fall under the categories *testing and verification*, *stakeholder-centred techniques*, and *online techniques*, in descending order of frequency.

It is noteworthy to mention that, despite (formal) verification being an important focus of research, this was only mentioned three times in the survey.

7) Key insights: **Key insights** on **RQ3** from the results are that resource usage and system load are the main types of monitoring metrics used in practice, and are tracked by sensors (both in environment and in the system). Practitioners use various mechanisms for analysis, data analysis and comparisons to thresholds being the most prominent. Enacting self-adaptation is done by a wide range of mechanisms, but mostly auto-scaling and reconfiguration. Tools such as Kubernetes and AWS are extensively used in the realisation of self-adaptation. Most systems apply a mixture of fully-automated and semi-automated adaptation. The majority of practitioners reuse solutions, with code, design artifacts and specifications being the most common. Lastly, ensuring trust in self-adaptive systems is mainly achieved through extensive testing, runtime monitoring and alerting, and human supervision.

E. RQ4: Difficulties, Problem Support and Opportunities

The questions for RQ4 aim to create a better understanding of the difficulties that the participants have experienced during the application of self-adaptation in practice. The questions also poll the participants for the problem support they get or want from the academic world, and the opportunities they see in self-adaptation that have yet to be exploited.

1) Difficulties: The participants appear to have difficulties with applying self-adaptation relatively often: exactly three quarters of all participates indicate they experience difficulties sometimes to always.

The main type of issues faced by the participants appears to be reliable or optimal design. One participant mentions that they must make the system provide service, even in times where parts of the system fail. Tuning and debugging also seems to be a common issue. With one participant, debugging the root cause of a scaling failure was a very time consuming process. The design of self-adaptive systems can also be quite complex. Defining adaptation rules for a system in an uncertain

environment could be hard to do. The uncertainty during runtime is also a common issue. Heuristics which self-adaptive modules are built upon may not always be proven to work in a specific system's environment. How much the environment affects the system could also be uncertain, making it hard to robustly apply self-adaptation. The concepts of self-adaptation may also not yet be fully grasped by developers, perhaps due to a lack of experience.

2) Risks: While issues with applying self-adaptation may be very common, risks can also pop up. More than half of the participants claim they sometimes to always experience risks when applying self-adaptation.

Faults, more specifically incorrect functionality, is the most common risk for self-adaptive systems. Automating these systems may lead to unexpected values, or perhaps other bugs could kill the system entirely. Difficulties with development or operation, such as managing environment uncertainty, is the second most common risk. This could be potentially underestimating environment variability, or incoming data suddenly becoming unpredictable, with large difference spikes, for a prolonged period of time. Another risk is the impact on qualities, such as the risk of degrading performance rather than improving, as a result of applying self-adaptation. The impact self-adaptive systems have on the business, such as increased cost, can also be a risk worth noting.

Of course, businesses attempt to mitigate these risks. While techniques in the the category of stake-holder centred techniques appear to be the most common, the specific technique of purely extensive testing is the most common. This includes testing each action in isolation before providing it to the system, running parallel correlated analysis, injecting non-determinism in a test-suite, and on-site testing at the costumer to fine-tune the models. The aforementioned stakeholder-centred techniques include rigorous design and development, code review, human supervision (presumably during runtime), and outsourcing operations to a specialized provider. Some online techniques were also mentioned, mostly including runtime monitoring and analysis.

3) Problem support: Interestingly, we note that a large portion of participants (roughly 41%) face problems for which they would appreciate support from academics, at least sometimes to always.

The most common problem for which academic support would be appreciated is trustworthiness of the system. Formal verification of algorithms for the system as a whole, safety protocols for machine learning, and mechanisms to identify malicious input are all requests the participants made on this topic. Overall, the engineering problems seem to be the most common requests. These include support on the architecture & reuse, such as best practices. One participant even suggest appending a few techniques to *Kazman's Architecture Tactics* checklist. Adoption of the concepts and techniques by workers is also such an engineering problem: participants request organisational structures and workflows that lead to more self-adaptive and resilient platforms. Next, a problem on the topic of user interaction, is automation. If human supervision is part of the system, according to some participants, this can become the bottleneck of the system. Support on how to automate certain tasks without the need for human intervention could help speed up plenty of systems. Lastly, data governance and data access are also problems for which support would be appreciated.

4) Opportunities: We also asked the participants for their ideas of opportunities for self-adaptation that they think are not yet exploited. With 72 of 147 mentions of opportunities, system activities were the most common. System activities include autonomous operation, with one example being smart heating and lighting systems which takes people's habits into consideration. Data management & machine learning also belongs to this category, as one participant mentions that manual fine-tuning seems to be delay timely new releases for them, and methods which could handle changes in machine learning models and efficiently deploy them could help resolve this.

Participants also gave a lot of system property opportunities. Quality improvement was the most common, with examples being fault tolerance, power consumption, resource optimization, etc. Security improvement was also mentioned often, with some examples being mobile devices that could adapt based on locally identified threats. Detecting in-vehicle threats, and when a system is compromised, are also examples.

Lastly, some engineering activities and human involvement opportunities were mentioned. Maintenance and reuse is the most mentioned opportunity, including software provisioning and automatic updates, preventive maintenance, and self-adapting infrastructure based on demand.

5) Key insights: **Key insights** on **RQ4** from the results are that the majority of practitioners experience difficulties with self-adaptation, especially with reliably/optimal design, design complexity, tuning and debugging. Another large portion, roughly half, experience risks, such as incorrect functionality, difficulty to manage environment uncertainty, degraded performance and increased costs. Again, roughly half report they would like support from academics on certain problems, especially when it comes to engineering self-adaptive systems, the guarantees, and management of data. Once again, roughly half of the participants see future opportunities for utilising self-adaptation, namely autonomous operation, data management and machine learning.

F. Confidence in the answers

Lastly, the participants were polled on their confidence in their answers. Nearly all participants noted they are confident their answers are correct and that their answers to the survey portray a good picture of their experiences in practice.

VI. Other related insights and observations

We shortly note some observations in the data. Practitioners don't seem to put emphasis on mitigating uncertainties, which might be because they simply use different terms, such as "conditions are not always obvious". Changes in business goals aren't as frequently solved by self-adaptation as other

problems, perhaps because self-adaptation focuses more on lower-level problems, or because self-adaptation has yet to be fully utilised in industry to deal with bigger system changes. Besides the classic goals of self-healing, self-optimising, self-protecting, and self-configuring studied by academics, practitioners also put emphasis on improving user satisfaction, reducing costs, and mitigating risks. It might be a good idea for researchers to put emphasis on tools and infrastructure for realising self-adaptation, as these are commonly used in practice. More attention should be put on how to reuse solutions, such as reference architectures and patterns. In practice, software-intensive systems are usually not fully-automated, and humans still play a crucial role by either providing parts of functions or supervising the system. This is usually the first step towards fully-automating the system, but some stay at this stage such that they can trust the system to function properly, causing us to think that humans will continue to play a crucial role in self-adaptive systems. Because roughly 50% of participants claim they at least sometimes experience risks, and roughly 50% claim they would appreciate help from academics, we assume engineering efficient and trustworthy self-adaptive systems could benefit from joint efforts between industry and academics.

As for the benefits of applying self-adaptation in practice, The main problems solved by self-adaptation in the embedded/cyber-physical/IoT domain seems to be optimising performance and dealing with changes in the environment. In the cloud, this would be automating tasks and reconfiguration of the system. Reuse is mostly applied in manufacturing, data management, and embedded/cyber-physical/IoT, with the main artifact of reuse being system module.

As for difficulties and risks of applying self-adaptation in practice, the main difficulties are the design, and the people and processes at the system level. The most prominent risks are also on the system level, and these relate to development and operation, and impact on the business. Large companies appear to face a higher risk for faults in the system. Large companies also have stronger concerns for debugging, while small/medium companies are more concerned about limitations of tools and methods. Companies of all sizes consider difficulties with design to be important.

As for research support to address problems in practice, the majority of practitioners appear to welcome help from academics on realising self-adaptation on the system level, system module, or the application layer. The most prominent problems practitioners would appreciate support on are auto-tuning, auto-scaling, and monitoring and analysis. Triggers for adaptation would be dealing with system and environment properties, and system load. These problems are slightly more prominent in the system type of ICT communication and networks.

VII. Threats to validity

To reduce misinterpretations, the notion of self-adaptation was introduced at the start of the survey using a standard model with a feedback loop. A pilot was sent out, of which the feedback was used to clarify some questions and enhance the description. Experienced participants were selected from various domains. The confidence in the answers confirmed participants' trust in their responses.

Since the participants were selected non-probabilistically, there's a possibility that the survey's results aren't representative enough of reality. To make sure that the pool of participants was general enough, a wide net of experienced software engineers was selected through the networks of the authors of the original paper [5], and questions were added to the survey to poll the participants on their experience.

To mitigate a potential interpretation bias in the qualitative analysis, a thorough coding scheme was established. The coding tasks were distributed into teams of two of the authors of the original paper [5], each person coded individually, then discussed the coding among the team. This happened twice, with two teams individually coding the free texts, after which the results were compared by the latter team. Finally, the coding was crosschecked with the original coders to reach consensus.

VIII. Conclusions and key insights

A survey was conducted and analysed to poll practitioners on the use of self-adaptation in practice. 184 participants replied to the survey, with 100 of these having experience with engineering self-adaptation. The analysis of the data allowed for an empirically-grounded overview of the state-of-the-practice in the application of self-adaptation.

The key insights for practitioners are as follows: Self-adaptation appears to increase robustness and performance while reducing costs and required resources, and it improves user experience while reducing the burdens of engineers. A wide range of mechanisms are used to realise self-adaptation in industrial systems. Tools and infrastructure (auto-scaling, container-orchestration platforms, etc.) are available and commonly used in the realisation of self-adaptation. The greatest challenges in engineering self-adaptation are: reliable/optimal design, design complexity, and tuning/debugging. Lastly, there is a relevant match between industry practices, and the work produced by the research community, in realising self-adaptation.

The key prospects for applying self-adaptation in practice are the following: realising full autonomous operation, exploiting machine learning, improving quality and security, and applying self-adaptation for maintenance.

The key opportunities for industry-research collaboration are the following: consolidating best practices (such as architecture, patterns, and reuse), modelling paths for the adoption of self-adaptation in industry, supporting advanced features to realise self-adaptation (such as dealing with the evolution of self-adaptive systems), rigorous methods for ensuring trustworthiness of self-adaptive systems, governance of data, and lastly, moving the human in the loop (from performing adaptation functions) to the human on the loop (to overseeing the system to ensure trust).

References

[1] Christof Ebert et al. “DevOps”. In: *IEEE Software* 33.3 (2016), pp. 94–100. DOI: 10.1109/MS.2016.68.

[2] Omid Gheibi and Danny Weyns. *Dealing with Drift of Adaptation Spaces in Learning-based Self-Adaptive Systems using Lifelong Self-Adaptation*. 2022. DOI: 10.48550/arXiv.2211.02658. arXiv: 2211.02658 [cs.LG]. URL: https://doi.org/10.48550/arXiv.2211.02658.

[3] Danny Weyns. *An Introduction to Self-adaptive Systems: A Contemporary Software Engineering Perspective*. John Wiley & Sons, 2020.

[4] Danny Weyns et al. “Preliminary Results of a Survey on the Use of Self-Adaptation in Industry”. In: *Proceedings of the 17th Symposium on Software Engineering for Adaptive and Self-Managing Systems*. SEAMS ’22. Pittsburgh, Pennsylvania: Association for Computing Machinery, 2022, pp. 70–76. ISBN: 9781450393058. DOI: 10.1145/3524844.3528077. URL: https://doi.org/10.1145/3524844.3528077.

[5] Danny Weyns et al. *Self-Adaptation in Industry: A Survey*. 2022. DOI: 10.48550/arXiv.2211.03116. arXiv: 2211.03116 [cs.SE]. URL: https://doi.org/10.48550/arXiv.2211.03116.

The current state of self-adaptive systems

Maxim Ketels
Group Science & Technologie Kulak
Informatics
Email: maxim.ketels@student.kuleuven.be

***Abstract*—Self-adaptation equips a software-intensive system with a feedback loop. This feedback loop then automates tasks that otherwise would need to be done by a human operator. This article looks at the results of a survey done on the state and usage of self-adaptive systems in industry. In particular it looks at how these systems are used, why they are used and what problems may occur when relying on these types of systems.**

1. Introduction

Software-driven computer systems have rapidly taken over nearly every industry. This along with the ever-increasing complexity of these systems have made the managing of these systems a difficult task. To overcome this problem more and more are beginning to make use of self-adaptive systems. These so-called self-adaptive systems, or SA-systems, are equipped with a continuous feedback loop. These feedback loops will closely monitor the system and its environment and make decisions based on these parameters to adapt the system when needed. This way it removes the need for constant human supervision for certain tasks. A typical example of one such system is an elastic cloud, which can grow or shrink depending on the current requirements, thus freeing up resources that can be used elsewhere.

This article will go over a study [2] surveying 184 users spread over 21 countries. The study mainly aims to answer what users saw self-adaptive systems as, why and when they used SA-systems and what risks and challenges these systems brought with them.

2. Related works

Self-adaptation as a principle of applying a feedback control to software-intensive systems has long been the subject of study in academia. Such as in 1998, when Oreizy et. al [1] introduced the notion of ”self-adaptation”. This study however does not research the technical aspects of self-adaptation but instead is a systematic study to understand the current state of practice of self-adaptation in industry. As this is the first study to research this particular question there are no previous studies to compare the results to.

3. Research questions and methodology

The purpose of the study is to create a better understanding of the state of practice of self-adaptation in industry. To achieve this goal, a large-scale survey was done on active practitioners. In this survey, this goal of a better understanding of the state of practice of self-adaptation was split up into four main research questions and each of these questions into multiple sub-questions.

3.1. Population and sampling

The study was aimed at a certain target audience: the participants had to be active users. This entails that they were actively engaged in the development of industrial software-intensive systems. This includes architects, designers, developers, testers, maintainers, operators and other people that are actively involved in the development and maintenance of software-intensive systems.

The second requirement was that the participants had to have a minimum of three years of technical expertise in the development of industrial software systems.

Given these requirements, a total of 355 practitioners were contacted via the networks of the researchers involved. These 355 practitioners were from a wide variety of companies and spread over 21 countries.

3.2. Research Questions

To best draw a picture of what the current state is of self-adaptation in industry, the researchers set up four concrete research questions:

- **RQ1:** What drives practitioners to apply self-adaptation in software-intensive systems?
- **RQ2:** How do practitioners characterise self-adaptation?
- **RQ3:** How do practitioners apply self-adaptation in industrial software-intensive systems?
- **RQ4:** What are the experiences of practitioners with applying self-adaptation and do they see opportunities for how and where to apply self-adaptation?

Question 1 aims to investigate ”why” practitioners apply self-adaptation, what kinds of industrial systems it is applied to and the types of problems they solve.

ID	Question
Q1.1	For which problems do you or your organisation apply self-adaptation capabilities, i.e., a managing system that monitors and adapts a managed system to achieve some objectives?
Q1.2	What are the main business motivations for you or your organisation to apply self-adaptation?
Q1.3	What could be the benefit ofself-adaptation in one of the systems you worked with? Please explain briefly.

TABLE 1. QUESTIONNAIRE Q1

ID	Question
Q2.1	Think of a concrete self-adaptive system you worked with. Name this system and briefly explain its purpose (please use this system to answer the following three questions)

TABLE 2. QUESTIONNAIRE Q2

Question 2 looks at how practitioners view the concept of ”self-adaptation”.

Question 3 aims to examine how self-adaptation has been applied in the industy. What methods, techniques, tool, benchmarks and processes are currently used.

Question 4 will look at the difficulties and risks practitioners experience when working with or on self-adaptive systems. Additionally there will be looked at whether or not practitioners would like support from researchers on certain problems.

3.3. Online survey

Since each of the four main research questions has a very broad range of answers, all questions were split up into groups of several smaller questions with a much more concrete answer option. These questions ranged from multiple choice questions to free text. These smaller questions would allow for an easier analysis than the alternative of larger open questions.

Additionaly before any of the four main research question groups there was an initial question group pertaining to whether the participant had experience with self-adaptation (Q0.1), confirmed a good coverage of of kinds of software-intensive systems across participants (Q0.2), the size of the companies (Q0.3), the participant's role (Q0.4) and years of experience (Q0.5).

To answer all these questions, the study made use of an online survey. This allows for a large outreach with a minimal cost (both financial and time). The completion of the survey would take approximately half an hour.

Before the real survey was sent to all the participants, a pilot survey took place with 8 participants [3]. This pilot survey contained all the normal questions but also some extra meta-questions pertaining to the survey itself: about the clarity of terminology, clarity of the questions, relevance of the questions and scope of the questionnaire. Doing this further ensures the quality of the survey for the coming participants. Based on the answers of the meta-questions in the pilot it was concluded that questions would not need to be revised as they were perceived as clear and well scoped.

ID	Question
Q3.1	What mechanisms or tools does the self-adaptive system you worked with use to monitor a managed system during operation? By monitor, we mean tracking properties of the system or its environment.
Q3.2	What mechanisms or tools does the self-adaptive system you worked with use to analyse conditions of a managed system during operation? By analyse, we mean examining conditions of the system or its environment and determining whether any adaptation is required or not.
Q3.3	What mechanisms or tools does the self-adaptive system you worked with use to change a managed system or parts of it during operation? By change, we mean adjusting parameters of the system, or adding, removing or changing any parts of it.
Q3.4	What is the degree ofautomation ofthe majority ofthe self-adaptive solutions you work with in your organisation?
Q3.5	Do you reuse solutions to realise self-adaptation in systems you work with?
Q3.6	Please provide a concrete example of reuse you used to realise selfadaptation?
Q3.7	Why do you not often reuse solutions when realising self-adaptive systems? What hinders the reuse, please provide a short answer.
Q3.8	How do you ensure that you can trust the self-adaptive solutions you build? Examples could be extensive testing or human supervision, but you may use other means. Please describe briefly.

TABLE 3. QUESTIONNAIRE Q3

ID	Question
Q4.1	Did you encounter particular difficulties or challenges when engineering or maintaining self-adaptive systems you worked with?
Q4.2	Please give one or two examples of the difficulties or challenges that you encountered when engineering or maintaining self-adaptive systems.
Q4.3	Did you face any risks when engineering self-adaptive systems you worked with?
Q4.4	Please briefly describe one or two risks that you faced when engineering self-adaptive systems.
Q4.5	How did you mitigate the risks that you faced? Please explain briefly.
Q4.6	Have you faced or seen any problems of self-adaptation for which you would appreciate support from researchers?
Q4.7	For which problems of self-adaptation would you appreciate support from researchers? Please briefly explain one or two such problems.
Q4.8	In your organisation or in industry in general, do you see application opportunities for self-adaptation that are currently not exploited?
Q4.9	Please describe or give examples of the application opportunities for self-adaptation that are currently not exploited.

TABLE 4. QUESTIONNAIRE Q4

3.4. Data Analysis

The questions as seen in tables 1, 2, 3, 4 consist of both open and closed questions. The closed questions were analysed using descriptive statistics and quantitative data analysis. As such the answers are mostly reported by frequency, percentages relative to the respective number of responses and relationships between answers to questions based on the categorisations of answers. These relationships are only reported if they lead to relevant insights.

The open questions are analysed using qualitative anal-

ysis. Inductive reasoning was used to construct codes and to derive categories from the data.

4. Contributions

The study mainly aims to create a better understanding of the state of practice of self-adaptation in industry, which would be relevant for both the academic world and the practitioners. It would allow researchers to know whether their research is relevant according to the current needs of the industry. It would allow the current practitioners to gauge the level of their current practice in applying self-adaptation and would open up possibilities for potential collaborative studies.

5. Results

There were a total of 355 surveys sent to practitioners. Of these 355 a total of 184 were completed which results in a response rate of 51.8%. These 184 were further split up into two groups based on the results of Q0.1 (Experience with self-adaptation): those that expressed to have worked with self-adaptive systems (100 participants) and those that didn't (84 participants).

5.1. Results for RQ1: Motivations for applying SA

Given how all four research questions are about self-adaptation in industry it is evident that all resulting data for these questions comes from the 100 participants that expressed to having worked with self-adaptation in (Q0.1).

5.1.1. RQ1.1: For which problems do you apply SA?. The average participant applied self-adaptation for 3.6 types of problems from a predefined list. The most common ones being in order: to optimize system performance (78), to automate tasks (61), to deal with changes in the environment (60) and to configure/reconfigure a system (51).

5.1.2. RQ1.2: What are the main business motivations for you or your organisation to apply self-adaptation?. The average participant expressed 2.1 business motivations for self-adaptation with the most common ones being in order: improving user satisfaction (67), to reduce costs (66) and to mitigate risks (44).

5.1.3. RQ1.3: What could be benefits of applying self-adaptation?. Of the 100 viable participants, 92 provided meaningful descriptions of benefits of self-adaptation with an average of 1.8 per participant. Summarising these findings leaves us with the most common benefits being: improved utility (61), savings in cost and resources (38) and improved human interaction (37).

5.1.4. Summarizing RQ1. Summarizing the answers to give an answer to the first research question: motivations for applying self-adaptation. Self-adaptation is used on a wide variety of problems with its most common application being: optimisation of performance and automation of tasks. The overarching business motives are mainly to improve user satisfaction and reducing costs with benefits of improved utility.

5.2. Results for RQ2: Characterisation of self-adaptation

RQ2 aims to answer see how practitioners view the concept of self-adaptation. This research question only had one sub-question as seen in table 2.

5.2.1. RQ2.1: Explain a concrete self-adaptive system you worked with. This question was answered by all but one of the 100 participants with actual experience with self-adaptive systems. Based on the given responses, the answers were split into different categories based on subject of adaptation, type of adaptation and trigger for adaptation.

The most common subjects were systems (28), modules (22) and platform layers (13).

The most common type of adaptation were auto-scaling (33), auto-tuning (28) and monitoring/analysis (22).

Finally the most common triggers were: system properties (27) and environment properties (18).

5.2.2. Summarizing RQ2. The results of RQ2 provided insights on several aspects. Such as self-adaptation being used on many different levels of software-intensive systems such as complete systems to singular modules. The most common types of adaptation are auto-scaling and auto-tuning with the most common triggers for this self-adaptation being the changes in properties of systems and their environments.

5.3. Results for RQ3: Application of self-adaptation

RQ3 looks at the current implementation of self-adaptation. This means looking at which methods, tools, processes etc. are currently being used and in particular at the degree of automation and the need for human supervision during run-time.

5.3.1. RQ3.1: What mechanisms or tools does the self-adaptive system you worked with use to monitor a managed system during operation?. There were a total of 146 mechanisms or tools provided that participants used for monitoring in self-adaptive systems which is an average of 1.5 mechanisms/tools per person. Again the responses were split into different categories based on: monitoring metric (75), monitoring mechanisms (20) and monitoring tools (34).

The most common monitoring metrics being: resource usage (23) and load (18).

The most common monitoring mechanisms being: environmental sensors (9) and logging mechanisms (6).

And the most common monitoring tools being: Kubernetes monitoring (9), Prometheus (9) and AWS monitoring (8).

5.3.2. RQ3.2: What mechanisms or tools does the self-adaptive system you worked with use analyse conditions of a managed system during operation?. A total of 115 mechanisms or tools for analysing conditions of a SA-system were provided with an average of 1.5 mechanisms/tools per person. The responses are split into mechanisms (73) and tools (23). The most common mechanisms: data analysis methods (18) and comparison to threshold (16). And the most common tools for analysis are: AWS analysis tool (9) and Kubernetes stack (7).

5.3.3. RQ3.3: What mechanisms or tools does the self-adaptive system you worked with use to change a managed system or parts of it during operation? . There were a total of 83 mechanisms and 19 tools provided for applying changes which is 1.3 mechanism/tool per person. The most common mechanisms being: Scaling mechanics (36) and reconfiguration (25). The most common change enacting tools being: Kubernetes (9) and AWS (7).

5.3.4. RQ3.4: What is the degree of automation of the majority of the self-adaptive solutions you work with in your organization?. Of the 100 viable participants all 100 provided an answer with most having a "mixed" automation (47), followed by some having semi-automation (31) and some having full automation (19). Finally there were 2 participants with no automation and a singular participant that stated there was "full automation till first incident".

5.3.5. RQ3.5: Do you reuse solutions to realise self-adaptation in systems you work with?. Again all 100 participants responded with most (71) participants at least sometimes reusing solutions in self-adaptation. The other 29 participant will reuse solutions rarely to never.

5.3.6. RQ3.6: Please provide a concrete example of reuse you used to realise self-adaptation?. The 67 answers can be categorized in: code (33), design artifacts (22), specifications (18), IT infrastructure (11) and procedures (7). These answers show that reuse of self-adaptation solutions is a common practice.

5.3.7. RQ3.8: How do you ensure that you can trust the self-adaptive solutions you build?. There were 91 valid answers that can be categorized in: testing and verification (71), stakeholder-centred techniques (45) and online techniques (45). With for each its most common example being: Extensive testing (58), Human supervision (22) and run-time monitoring and alerting (27).

5.3.8. Summarizing RQ3. We can summarize the answers to these questions about application of self-adaptive systems. The most common monitored parameters are resource management and system load. These parameters are mainly being monitored by sensors in the environment and in the system.

Multiple mechanisms and tools are used for analysis with the most common being data analysis and threshold comparisons.

A multitude of mechanisms are used to apply self-adaptation with the most common ones being auto-scaling and auto-tuning.

Practitioners are strongly reliant on tools such as Kubernetes and AWS to support the different functions of self-adaptive systems.

Most practitioners will use a mix of semi- and fully automated self-adaptation.

Most practitioners will reuse previous self-adaptation solutions when possible. This mostly in the form of code, design artifacts and specifications.

Trust in self-adaptive systems is created by means of extensive testing, run-time monitoring and human supervision.

5.4. Results of RQ4: Difficulties and risks

RQ4 aims to understand the difficulties and risks, if there are any, that can be experienced when self-adaptation is chosen and whether or not the practitioners would appreciate support from researchers.

5.4.1. RQ4.1: Did you encounter particular difficulties when engineering or maintaining self-adaptive systems you worked with?. Of the 100 viable participants there were 41 that reported to sometimes facing difficulties when applying self-adaptation. 30 reported encountering difficulties frequently or very frequently. 17 reported rarely or very rarely having difficulties and 4 reported to always have problems. Finally 8 participants reported never having problems.

5.4.2. RQ4.2: Please give one or two examples of the difficulties that you encountered when engineering or maintaining self-adaptive systems. There were 74 participants gave answers totalling 140 difficulties resulting in an average of 1.9 per person. These problems are categorized in: Design issues (43) with the most common issue being reliable/optimal design, Lifecycle issues (42) with the most common being tuning/debugging, Runtime issues (30) with the most common being runtime uncertainty and people and process issues (19) with the most common issue being skills/experience.

5.4.3. RQ4.3: Did you face any risks when engineering self-adaptive systems?. Of the 100 participants, most (34) reported to sometimes facing risks. 18 reportedly always facing risks and 48 rarely to never face risks.

5.4.4. RQ4.4: Briefly describe one or two risks that you faced when engineering self-adaptive systems. . 60 participants provided an answer totalling 66 instances of risks faced when engineering self-adaptive systems with an average of 1.3 per person. The reported risks can be categorized into:

- Faults (20), most common being incorrect functionality (7)

- Difficulties with development/operation (16), most common being difficult to manage environment uncertainty (6)
- Impact on qualities (5), most common being performance degradation (5)
- Impact on business (14), most common being increased cost (5)

5.4.5. RQ4.5: How did you mitigate the risks that you faced?. 55 responses containing 66 instances of risk mitigating techniques were reported averaging at 1.3 per person. These techniques can be categorized as follows:

- Stakeholder-centered techniques (25), most common being rigorous design and development (8).
- Offline techniques (18), most common being extensive testing (15)
- Online techniques (9), most common being run-time monitoring and analysis (6)

5.4.6. RQ4.6: Have you faced or seen any problems of self-adaptation for which you would appreciate support from researchers?. Out of 166 participants, 33 frequently to always experience problems with self-adaptation for which they would appreciate support from researchers. 43 sometimes face such problems. But the 108 participants making the majority never faces problems for which they would like support.

5.4.7. RQ4.7: For which problems of self-adaptation would you appreciate support from researchers? Please briefly explain one or two such problems. This question was aimed at the participants that expressed a need/appreciation for possible support in the previous question. There were 65 participants describing in total 113 problems. These problems are categorized as follows:

- Engineering (48), most common being architecture & reuse (16)
- Guarantees (25), most common being trustworthiness (20)
- Data (21), most common being data governance (8)
- User interaction (19), most common being automation (9)

5.4.8. RQ4.8: In your organisation or in industry in general, do you see application opportunities for self-adaptation that are currently not exploited?. Of the 184 participants, 101 reported opportunities for applying self-adaptation, while 83 do not report any. The numbers within these two groups is almost equally split among participants who have worked with self-adaptive systems and those that haven't.

5.4.9. RQ4.9: Please describe or give examples of the application opportunities for self-adaptation that are currently not exploited. 85 participants reported a total of 147 unexploited opportunities for applying self-adaptation making an average of 1.7 per person. These can be categorized as follows:

- System activities (72), most common being autonomous operation (37)
- System properties (47), most common being quality improvement (26)
- Engineering activities (21), most common being maintenance & reuse (15)
- Human involvement (7), most common being personalization (4)

5.4.10. Summarizing RQ4. RQ4 aimed to understand the difficulties and risks that practitioners face. Also probing whether practitioners face problems for which they would appreciate support from researchers. Summarizing the previous answers we can conclude that a majority of practitioners will face difficulties when engineering or maintaining self-adaptive systems. These problems mainly pertain to reliable or optimal design, design complexity and tuning or debugging. About half of the participants reported encountering risks when using self-adaptation with most risks relating to incorrect functionality and difficulty to manage environment uncertainty.

Just shy of half the practitioners reported they would appreciate support from researchers with some of the problems they face, particularly problems related to the engineering of self-adaptive systems, guarantees and data-management. Finally about half of the participants see future opportunities for applying self-adaptation, particularly those related to autonomous operation, data management and machine learning.

6. Validity

There are a couple of potential threats to the validity of this study. The first being that it is assumed that the participants are adequately familiar with all the concepts and terms of self-adaptation. Given the provided answers to the questions this is unlikely that the participants are not familiar with the concepts and terms. However there were still precautionary measures made to avoid possible confusion: the notion of self-adaptation is introduced at the start of the survey and feedback on the pilot-survey was used to further make the questionnaire clear.

Another potential threat is the possible generalisation of the study results. This problem is derived from the selection of the target audience. There may be inaccuracies in the results should the population turn out to not be representative. The practitioners were not chosen by a probabilistic sampling method, thus the threat remains. To minimize this threat to validity, the practitioners were chosen from the networks of the researchers and were asked questions about their experience with engineering self-adaptive systems to guarantee their adequate level of experience.

A final potential threat is the reliability of the data analysis. Qualitative data analysis (the coding of answers with free text) is to a certain extent a subjective task that

could introduce a bias of sorts. To counteract this possibility, all coding tasks were distributed among teams of two authors. Each author performed the coding independently and discussed where needed until agreement was reached. Then the same coding tasks were given to a different group of two and the coding was repeated independently from the initial coding. Again the results were discussed until an agreement was reached. Finally all material of the survey is made public including the raw data. This ensures an as fair coding as possible.

7. Conclusion

The aim of the survey was to research and characterize the current state of self-adapting systems in industry. This was done from the perspective of the practitioners of these systems. This study showed that self-adaptation is mainly used for optimising performance, automating tasks and the handling of changes in the environment. All this ultimately leads to better utility and user satisfaction and lower costs.

Self-adaptation is mainly being used by monitoring parameters such as resource usage and system load. These parameters are mainly being monitored by data analysis methods and threshold comparisons.

These systems are a mix of semi- and fully automatic and need less human input to operate. Trust in these system is gained through rigorous testing and human supervision during run-time.

The greatest challenges these systems create are reliable or optimal design, their design complexity and tuning or debugging. The most common risks are incorrect functionality, difficulty to manage environment uncertainty and degraded performance.

Almost half of the surveyed practitioners expressed that they would appreciate support from researchers in the solving of problems with realising self-adaptation. As such this survey provides prospects for potential future collaborations.

References

[1] R.N. Taylor P. Oreizy, M.M. Gorlick and Others. An architecture-based approach to self-adaptive software. intelligent systems and their applications, 1998.

[2] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, Grace Lewis, Marin Litoiu, Angelika Musil, Juergen Musil, Panos Patros, and Patrizio Pelliccione. Self-adaptation in industry: A survey. 2022.

[3] Danny Weyns, Ilias Gerostathopoulos, Nadeem Abbas, Jesper Andersson, Stefan Biffl, Premek Brada, Tomas Bures, Amleto Di Salle, Matthias Galster, Patricia Lago, Grace Lewis, Marin Litoiu, Angelika Musil, Juergen Musil, Panos Patros, and Patrizio Pelliccione. Self-adaptation in industry – state-of-the-practice and opportunities (survey protocol), 2022.