Software Center

Posters from the Software Center Reporting workshop December 2021

- Accelerating Digitalization Through Data
- Enterprise Scale Continuous Integration and Delivery
- Strategic Ecosystem-Driven R&D Management
- Engineering Knowledge Flows in Large-Scale Agile System Development
- Using Digital Twin to Detect Cyber-Attacks On Cyber-Physical Systems
- Analysis of Timing Properties in System of Cyber-Physical Systems
- Flaky Tests & Software Center: In a Nutshell
- Managing Model Inconsistencies
- Automated Recovery of Data Pipelines
- Noise Handling For Improving Machine Learning-Based Test Case Selection
- Towards Federated Learning
- Challenges in developing and deploying AI in the engineering, procurement and construction industry
- Transforming Automo/ve Architecture with Assistance from AI
- Architectural Design and Verification/Validation of Systems with Machine Learning Components

Software Center

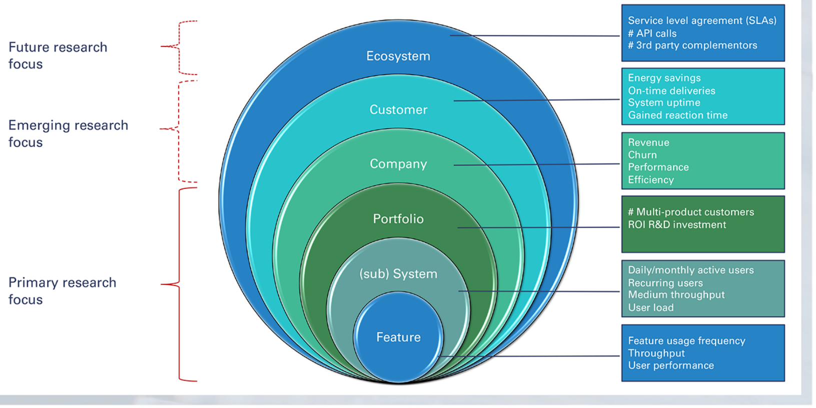
Accelerating Digitalization Through Data

Research theme:
Customer Data- and Ecosystem-Driven Development

Project number: Software Center #5

Partners:
Jeppesen Systems AB
Saab AB
Siemens AB
Volvo Car Corporation
Volvo Truck Corporation
Wärtsilä
Grundfos Holding A/S

Chalmers University of Technology
Malmö University



Summary

In this project, we help companies advance their adoption of *data driven development practices* with the overall intention to ensure rapid and continuous delivery and improvement of customer value. We focus our research on *'value design'* with the intention to help companies identify and agree on what they optimize for, align metrics at different levels and transition from a qualitative towards a quantitative understanding of customer value.

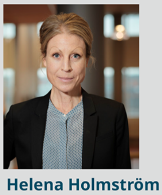
Value Design: What?

- Value design is a process intended to help explicate assumptions and move towards more quantitative assessment of customer value
- Value design helps organizations ensure customer value and align low-level and high-level business metrics
- Value design recognizes the multiple stakeholders and their different perspectives; (1) user perspective, (2) technology and product perspective, and (3) business and market perspective

Value Design: Why and How?

- The purpose of value design is to move from implicit assumptions about the value of a feature, a product, a portfolio, a service offering etc., to explicating those assumptions
- The process starts from a purely qualitative assessment of value factors but with the intention to translate this into quantitative and/or quantitative measures

Contact us:



Helena Holmström Olsson
helena.holmstrom.olsson@mau.se
+46-40-6657319

Jan Bosch
jan.bosch@chalmers.se
+46-31-7725716

www.software-center.se

CHALMERS · UNIVERSITY OF GOTHENBURG · MÄLARDALEN UNIVERSITY SWEDEN · MALMÖ UNIVERSITY · LINKÖPING UNIVERSITY · SAAB · SCANIA · SIEMENS · TOYOTA MATERIAL HANDLING · WÄRTSILÄ · VOLVO · VOLVO · qamcom · JEPPESEN · ERICSSON · BOSCH · GRUNDFOS · CEVT · AXIS COMMUNICATIONS · ESAB · DEIF · zenseact · Tetra Pak

Software Center

Torvald Mårtensson
SAAB

Daniel Ståhl
ERICSSON
LiU

Antonio Martini
UiO : University of Oslo

Jan Bosch
CHALMERS UNIVERSITY OF TECHNOLOGY

Enterprise Scale Continuous Integration and Delivery

Project acronym: ESCID

Research theme: Continuous Delivery

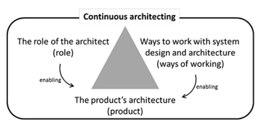
Project number: Software Center Project #6

Partners: AXIS COMMUNICATIONS, VOLVO, SAAB, SIEMENS, BOSCH, ERICSSON, GRUNDFOS, VOLVO TRUCKS

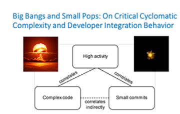
Enable more frequent integration of software

The EMFIS model

	Developers	Enablers
Activity planning and execution		
Work breakdown	2.5 ⚠	2.8
Teams and responsibilities	3.8	4.2
Activity sequencing	2.9 ⚠	2.3
System thinking		
Modular and loosely coupled architecture	4.7	4.2
Developers must think about the complete system	4.1	4.0
Speed		
Tools and processes that are fast and simple	1.9 !	3.2 ≠
Availability of test environments	4.2	4.4
Test selection	4.0	3.8
Fast feedback from the integration pipeline	3.8	3.7
Confidence through test activities		
Test before commit	3.5	3.2
Regression tests on the mainline	3.2	3.6
Reliability of test environments	1.5 !	1.3



Continuous architecting



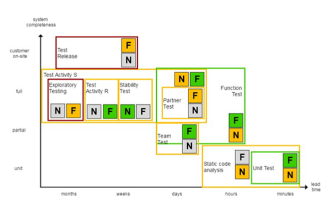
Cyclomatic complexity

Mob programming

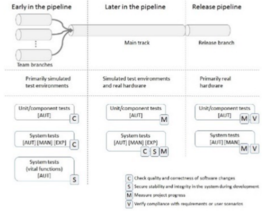
Fig. 2. Thematic map of interviews with experienced practitioners.

Designing the CI/CD pipeline

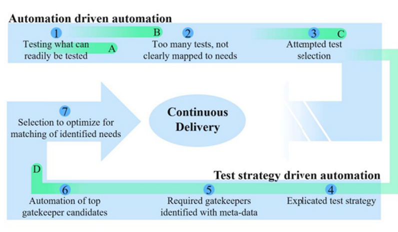
The CINDERS model



The TAS model

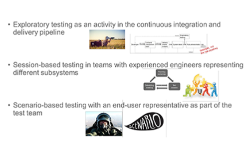


The Tapco model



Exploratory testing in the CI/CD pipeline

ET test method



Nine key factors

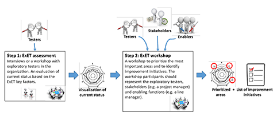
The testers' knowledge, experience and personality
1) The testers know how **the system** is built, and the correct behavior of the system
2) The testers know how the product is used by **the end-user** (or the end user is represented in the test team)
3) The testers are **curious and want to learn** about the system

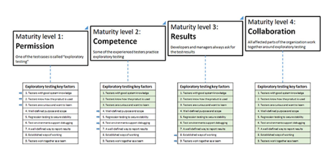
Purpose and scope
4) A well-defined **purpose and scope for the tests** (system functions ready to be tested) which the testers can transform into e.g. scenarios or focus areas
5) Regression testing secure **basic stability and integrity** in the system (before exploratory testing)

Ways of working
6) An **established way of working**, including planning meetings, preparations, test strategies and heuristics (a balance between structure and freedom)
7) Testers with different experiences and competences work **together as a team**, helping each other with new ideas and knowledge about different parts of the system

Recording and reporting
8) Test environments which support **debugging and recording**
9) A well defined way to **report the test results**, including a description of areas covered by the tests and a list of identified problems

The ExET model





The MaLET model

CHALMERS, UNIVERSITY OF GOTHENBURG, MÄLARDALEN UNIVERSITY SWEDEN, MALMÖ UNIVERSITY, LINKÖPING UNIVERSITY, SAAB, SCANIA, SIEMENS, TOYOTA MATERIAL HANDLING, WÄRTSILÄ, VOLVO GROUP, VOLVO, qamcom, JEPPESEN A BOEING COMPANY, ERICSSON, BOSCH, GRUNDFOS, CEVT, AXIS COMMUNICATIONS, ESAB, DEIF, zenseact, Tetra Pak

Software Center

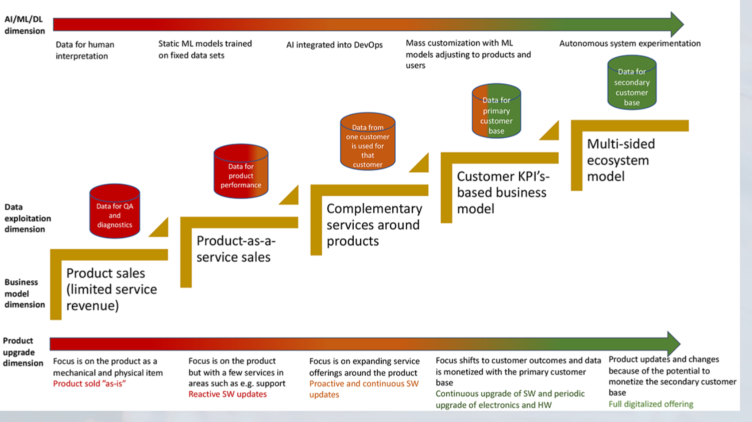
Strategic Ecosystem-Driven R&D Management

Research theme:
Customer Data- and Ecosystem-Driven Development

Project number: Software Center #9

Partners:
Saab AB
Siemens AB
Toyota Material Handling
Volvo Car Corporation
Volvo Truck Corporation
Ericsson AB
Jeppesen Systems AB
DEIF

Chalmers University of Technology
Malmö University



Summary

In this project, we study digitalization and digital transformation of the embedded systems industry and the many ways in which this impacts the business ecosystems in which companies operate. The goal of this project is to provide companies with strategic guidance for *how to transition from traditional companies towards digital companies*. This involves e.g., the transition towards more service-oriented revenue, continuous monetization of data and the adoption of new and innovative ways-of-working.

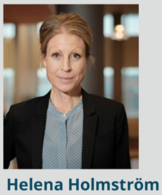
Digitalization and Digital Transformation

- Digitalization is transforming industry to an extent we have only seen the beginnings of
- With software, data and AI, companies seek to complement their existing physical offerings with new digital services
- To transition from transactional business models towards continuous (digital) business models is key to survive and stay competitive
- To increase service revenue, companies need to expand the scope of their business models and identify new revenue streams that go beyond their existing sales

Service monetization approaches

- Traditional product sales: Service opportunities and monetization is explored only after the customer has bought the product
- Edge customers: Service opportunities and monetization is explored in collaboration with "edge customers" (smaller customers that might be more open-minded and willing to experiment)
- New customers: Service opportunities and monetization is done outside the primary business and not with the main customer group(s)

Contact us:



Helena Holmström Olsson
helena.holmstrom.olsson@mau.se
+46-40-6657319

Jan Bosch
jan.bosch@chalmers.se
+46-31-7725716

www.software-center.se

Software Center

Previously: RE for Large-Scale Agile System Development

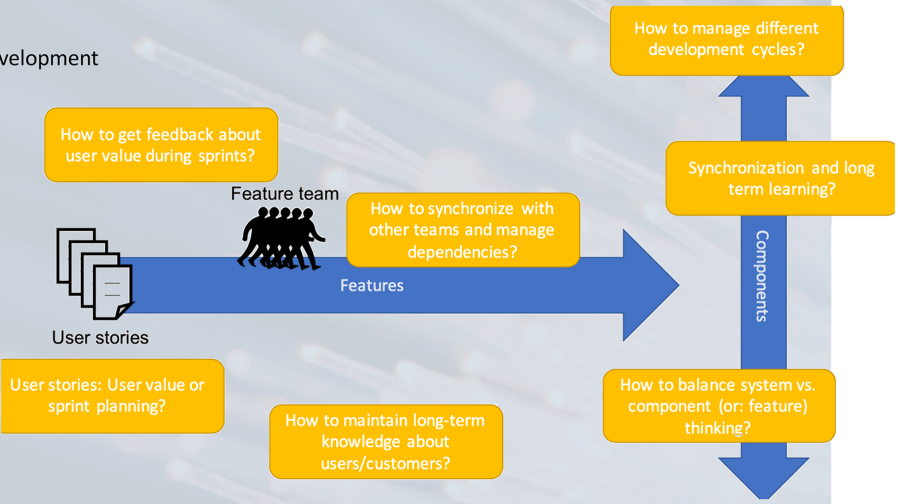
Engineering Knowledge Flows in Large-Scale Agile System Development

Project acronyme:
Flow (previously : RE4Agile / MaRK-C)

Research theme:
Customer Data- and Ecosystem-Driven Development

Project number:
Software Center #27

Partners:
Axis Communications AB
Ericsson AB
Grundfos Holding A/S
Saab AB
Siemens AB
Tetra Pak
Volvo Car Corporation
Volvo Truck Corporation
Zenseact
Chalmers University of Technology
University of Gothenburg



Summary

The Flow project (Engineering Knowledge Flows in Large-Scale Agile System Development, formerly known as RE for Large-Scale Agile System Development) provides new approaches to manage knowledge flows in large-scale agile system development. Traditional engineering disciplines and processes cannot keep up with the speed of agile development. Agile frameworks do not sufficiently cover important engineering knowledge. The Flow project specifically investigates challenges, best practices, methods, and tools for building, sharing, and maintaining engineering knowledge at scale. For this, we rely on 5 years of research on managing requirements knowledge in large-scale agile system development and extend it to related knowledge areas such as architecture, design, and release management knowledge.

Coordination Through Engineering Knowledge

Coordination through requirements, architectural decisions and other engineering knowledge: Improving coordination between methodological (agile) islands based on requirements-related boundary objects. This allows to:

- **Consider the needs of developers, product owners, system managers, architects, …**
- **Identify knowledge items**
- **Anticipate governance and shared maintenance**

Traceability in Large-Scale Agile System Engineering

Establish and foster collaborative traceability and construct traceability information models.

- **Cross-organizational view on how knowledge connects**
- **Design and Define Traceability Strategy**
- **Coordination needs and Traceability Structure as complementing mechanisms to define information models and knowledge flows**

Contact us:

Eric Knauss
Eric.Knauss@cse.gu.se
+46-31- 31-772 10 80

Jennifer Horkoff

Jan-Philipp Steghöfer

Hans-Martin Heyn

Jörg Holtmann

www.software-center.se

CHALMERS · UNIVERSITY OF GOTHENBURG · MÄLARDALEN UNIVERSITY SWEDEN · MALMÖ UNIVERSITY · LINKÖPING UNIVERSITY · SAAB · SCANIA · SIEMENS · TOYOTA MATERIAL HANDLING · WÄRTSILÄ · VOLVO · VOLVO · qamcom · JEPPESEN · ERICSSON · BOSCH · GRUNDFOS · CEVT · AXIS COMMUNICATIONS · ESAB · DEIF · zenseact · Tetra Pak

Software Center

Project #29

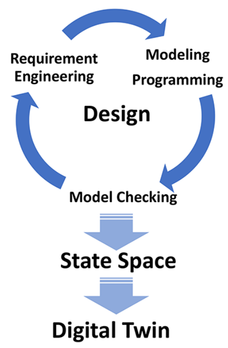
Using Digital Twin to Detect Cyber-Attacks On Cyber-Physical Systems

Monitoring Cyber-Physical Systems and Detecting Attacks

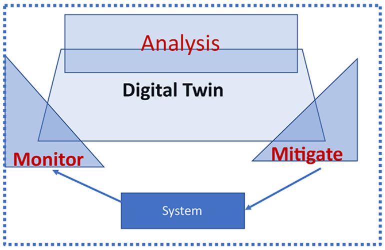
An IDS using a Digital Twin

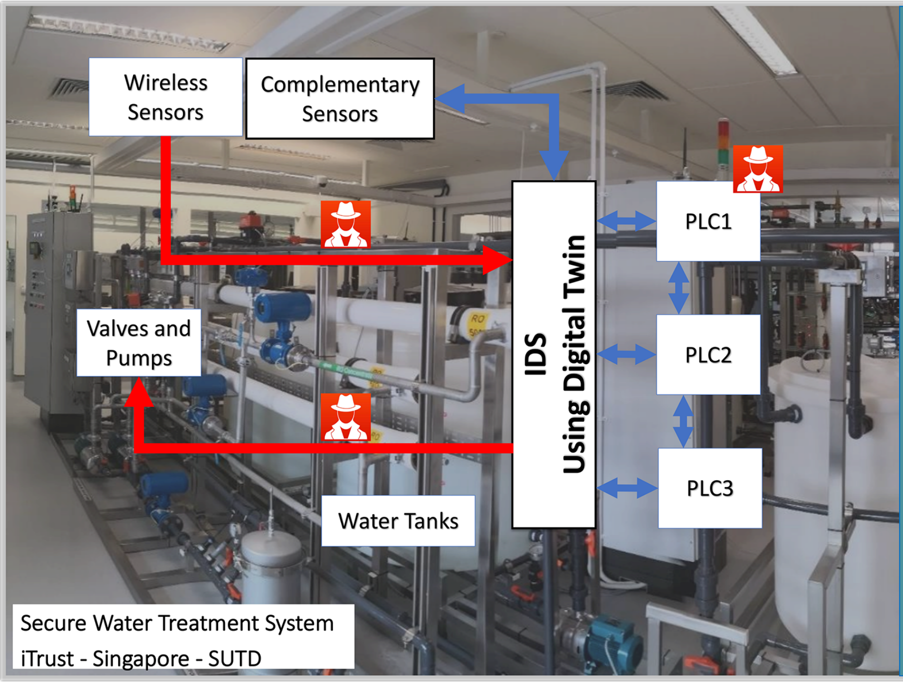
Intrusion Detection Systems (IDSs) are deployed in communication networks to detect attacks on the systems. IDSs are generally equipped with a set of rules to detect the attacks. The rules are written by safety and security engineers.

Digital Twin represents an image of a physical object.



Our IDS monitors input and output of the system and employs a Digital Twin to detect cyber-attacks.
The Digital Twin represents legitimate behavior of the system extracted at design time.
We build the digital twin based on an actor model of the system, and model checking.





Secure Water Treatment System
iTrust - Singapore - SUTD

IMPACT

- Eliminate the need to write policies for detecting attacks by safety/security engineers
- Provide concrete tractable reports after a successful detection.
- Low False-Positive Rate (FPR)

NOVELTY

- Employing a Digital Twin to detect cyber-attacks including coordinated attacks.
- The Digital Twin is created based on an executable model and formal methods.

CONTACT

MÄLARDALEN UNIVERSITY SWEDEN

Project contact: Marjan Sirjani
Mälarden University- Sweden
Email: marjan.sirjani@mdh.se

Afra 3.0

http://www.rebeca-lang.org/

Project #29 Partners

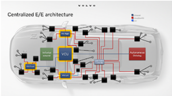
DEIF

VOLVO

Fereidoun Morad Marjan Sirjani Sara Abbaspour

Software Center
Project #29

Analysis of Timing Properties in System of Cyber-Physical Systems



MACMa: Modeling and Analyzing Event-driven Autonomous Systems

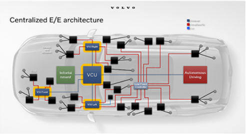
Timing Analysis

- End-to-end delay, Age, …
- Scheduling and Re-scheduling
- Safety Assurance and Robustness
- Availability

For SCPS such as:

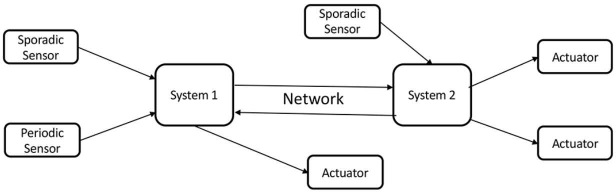
- Automotive Architecture
- Interoperable Medical Devices
- Factories

System of Cyber-Physical Systems (SCPS)

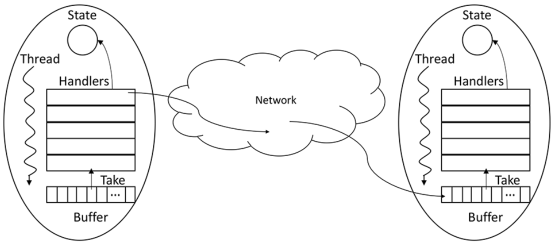


A generic view of a SCPS:

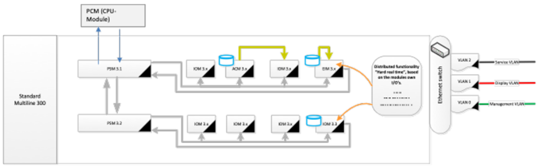
Multiple systems are communicating through different middleware. Each system handles different triggers from inputs and creates outputs (periodically or event-driven).



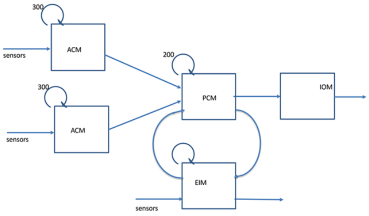
We use an Actor Model for various Timing Analysis applying Formal Methods.



DEIF Multi-line 300 Platform



Computing max/min end-to-end delays



Ref	PCM Offset	ACM1 Offset	ACM2 Offset	IOM1 Offset	IOM2 Offset	Min Delay	Max Delay	Avg Delay	Paths	States	Transitions
01	8000000	1000000	1000000	10	10	0	2000	1371	490	21066	30462
02	8000000	1000000	1000000	10	100	0	2000	1371	490	21066	30462
03	8000000	1000000	1000000	100	100	0	2000	1371	490	21066	30462
04	8000000	1000000	1500000	10	10	0	2000	1681	116	13543	13566
05	8000000	1000000	1500000	10	100	0	2000	1681	116	13543	13565
06	8000000	1000000	1500000	100	100	0	2000	1681	116	13543	13566
07	8000000	1000000	2000000	10	10	0	3000	1735	121	13045	13068
08	8000000	1000000	2000000	10	100	0	3000	1735	121	13045	13067
09	8000000	1000000	2000000	100	100	0	3000	1735	121	13045	13068
10	8000000	1500000	1500000	10	10	0	2000	1371	490	19563	28291
11	8000000	1500000	1500000	10	100	0	2000	1371	490	19563	28290
12	8000000	1500000	1500000	100	100	0	2000	1371	490	19563	28291
13	8000000	1500000	2000000	10	10	0	3000	1735	121	12544	12567
14	8000000	1500000	2000000	10	100	0	3000	1735	121	12544	12566
15	8000000	1500000	2000000	100	100	0	3000	1735	121	12544	12567
16	8000000	2000000	2000000	10	10	0	3000	2177	970	18069	26133
17	8000000	2000000	2000000	10	100	0	3000	2177	970	18069	26132
18	8000000	2000000	2000000	100	100	0	3000	2177	970	18069	26133
19	15000000	1000000	1000000	10	10	0	2000	1371	490	42063	60791
20	15000000	1000000	1000000	10	100	0	2000	1371	490	42063	60790
21	15000000	1000000	1000000	100	100	0	2000	1371	490	42063	60791
22	15000000	1000000	1500000	10	10	0	3000	1735	121	27544	27567
23	15000000	1000000	1500000	10	100	0	3000	1735	121	27544	27566
24	15000000	1000000	1500000	100	100	0	3000	1735	121	27544	27567
25	15000000	1000000	2000000	10	10	0	2000	1681	116	27043	27066
26	15000000	1000000	2000000	10	100	0	2000	1681	116	27043	27065
27	15000000	1000000	2000000	100	100	0	2000	1681	116	27043	27066
28	15000000	1500000	1500000	10	10	0	3000	2177	970	40569	58633
29	15000000	1500000	1500000	10	100	0	3000	2177	970	40569	58632
30	15000000	1500000	1500000	100	100	0	3000	2177	970	40569	58633
31	15000000	1500000	2000000	10	10	0	3000	1735	121	26545	26568
32	15000000	1500000	2000000	10	100	0	3000	1735	121	26545	26567
33	15000000	1500000	2000000	100	100	0	3000	1735	121	26545	26568
34	15000000	2000000	2000000	10	10	0	2000	1371	490	39066	56462
35	15000000	2000000	2000000	10	100	0	2000	1371	490	39066	56461
36	15000000	2000000	2000000	100	100	0	2000	1371	490	39066	56462

CONTACT

MÄLARDALEN UNIVERSITY SWEDEN

Project contact: Marjan Sirjani
Mälarden University- Sweden
Email: marjan.sirjani@mdh.se

Project #29 Partners

DEIF VOLVO VOLVO GROUP VOLVO

Marjan Sirjani (MDH), Jesper Graversen (DEIF), Tobias Christensen (DEIF), Bahman Pourvatan (MDH)

Afra 3.0

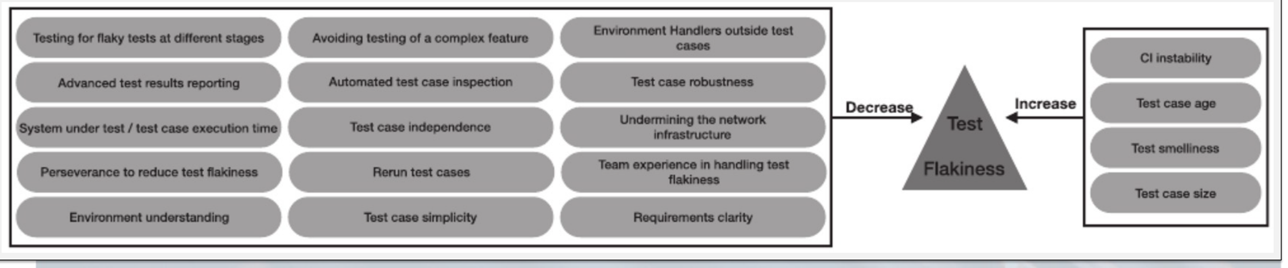
http://www.rebeca-lang.org/

Flaky Tests & Software Center: In a Nutshell

Software Center

4 Levels Strategy

Where to Start – level 0 ---- Capture Perceptions of Flaky Tests Among Team Members [1]



Few Flaky Tests – level 1 ---- Flaky Test Prediction [2]

```
@Test
public void testCodingEmptySrcBuffer() throws Exception {
 final WritableByteChannelMock channel = new ritableByteChannelMock(64);
 final SessionOutputBuffer outbuf = new SessionOutputBufferImpl(1024, 128);
 final BasicHttpTransportMetrics metrics = new BasicHttpTransportMetrics();
 final IdentityEncoder encoder = new IdentityEncoder(channel, outbuf, metrics);
 encoder.write(CodecTestUtils.wrap("stuff"));
 final ByteBuffer empty = ByteBuffer.allocate(100);
 empty.flip();
 encoder.write(empty);
 encoder.write(null);
 encoder.complete();
 outbuf.flush(channel);
 final String s = channel.dump(StandardCharsets.US_ASCII);
 Assert.assertTrue(encoder.isCompleted());
 Assert.assertEquals("stuff", s);
}
```

⇓

```
pty src buffer codec test utils standard charsets
channel assert equals encoder byte buffer empty test
coding empty assert allocate flush outbuf metrics
dump complete wrap write flip stuff completed
```

Table 3. Top 20 features by Information Gain.

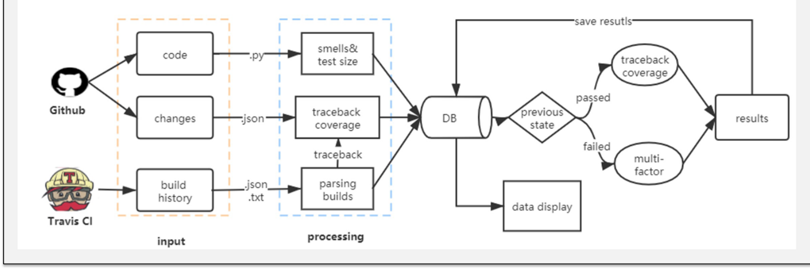
(a) Original study

Feature	Inf. Gain	Tests	Flaky tests	Non-Flaky tests
job	0.2053	528	524	4
table	0.1449	414	406	8
id	0.1419	584	522	62
action	0.1365	395	387	8
oozie	0.1359	274	274	0
services	0.1309	378	371	7
coord	0.1192	307	307	0
getid	0.1076	288	287	1
coordinator	0.1070	258	258	0
xml	0.1061	253	247	6
loc	0.0977	-	-	-
workflow	0.0913	207	207	0
getstatus	0.0884	248	246	2
throws_keyword	0.0873	10	3	7
record	0.0845	314	296	18
jpa	0.0780	207	207	0
jpaservice	0.0752	200	200	0
service	0.0733	434	367	67
wf	0.0721	192	192	0
coordinatorjob	0.0689	184	184	0

(b) Replication study

Feature	Inf. Gain	Tests	Flaky tests	Non-Flaky tests
job	0.1449	530	525	5
table	0.1029	414	406	8
id	0.1004	577	525	52
services	0.0977	378	371	7
action	0.0972	396	388	8
oozie	0.0942	346	346	0
loc	0.0879	-	-	-
coord	0.0826	307	307	0
xml	0.0752	356	341	15
getid	0.0746	288	287	1
coordinator	0.0741	278	278	0
get	0.0691	2194	1260	934
workflow	0.0633	240	240	0
throws_keyword	0.0615	2348	1327	1021
getstatus	0.0613	248	246	2
record	0.0596	314	296	18
service	0.0590	451	383	68
jpa	0.0541	207	207	0
jpaservice	0.0521	200	200	0
wf	0.0499	192	192	0

No Flaky Tests – level 2---- Multi-Factor Detection [3]



Multi-Factor approach

1. Number of test smells in the failed test cases (F1)
2. Whether failed test cases executed on latest code changes (F2)
3. Test cases history (how many times, the test cases failed due to unrelated code changes) (F3)
4. Test case size (perceived factors identified in our earlier study) (F4)

Above factors for decision making

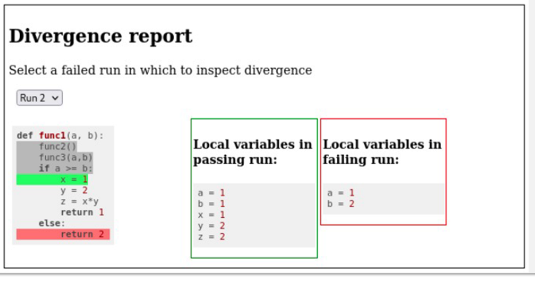
Machine Learning

Do these test cases have the characteristics similar to the flaky one?

Output

Graphs and CSV files

Many Flaky Tests – level 3---- Divergence Algorithem for Randomess in Flaky Tests



[1] Ahmad, A, Leifler, O, Sandahl, K. Empirical analysis of practitioners' perceptions of test flakiness factors. Softw Test Verif Reliab. 2021; 31:e1791. https://doi.org/10.1002/stvr.1791

[2] A. Ahmad, O. Leifler, and K. Sandahl, 'An evaluation of machine learning methods for predicting flaky tests', in Proceedings of the 8th international workshop on quantitative approaches to software quality co-located with 27th asia-pacific software engineering conference (APSEC 2020)

[3] Azeem Ahmad, Francisco Gomes de Oliveira Neto, Zhixiang Shi, Kristian Sandahl, Ola Leifler, 'A Multi-factor Approach for Flaky Test Detection and Automated Root Cause Analysis', presented at the 2021 28th Asia-Pacific Software Engineering Conference (APSEC).

Contact us:

Azeem Ahmad
Azeem.ahmad@liu.se

Kristian Sandahl
Kristian.sandahl@liu.se

CHALMERS · UNIVERSITY OF GOTHENBURG · MÄLARDALEN UNIVERSITY SWEDEN · MALMÖ UNIVERSITY · LINKÖPING UNIVERSITY · SAAB · SCANIA · SIEMENS · TOYOTA MATERIAL HANDLING · WÄRTSILÄ · VOLVO · qamcom · JEPPESEN · ERICSSON · BOSCH · GRUNDFOS · CEVT · AXIS COMMUNICATIONS · ESAB · DEIF · zenseact

Software Center

Managing Model Inconsistencies

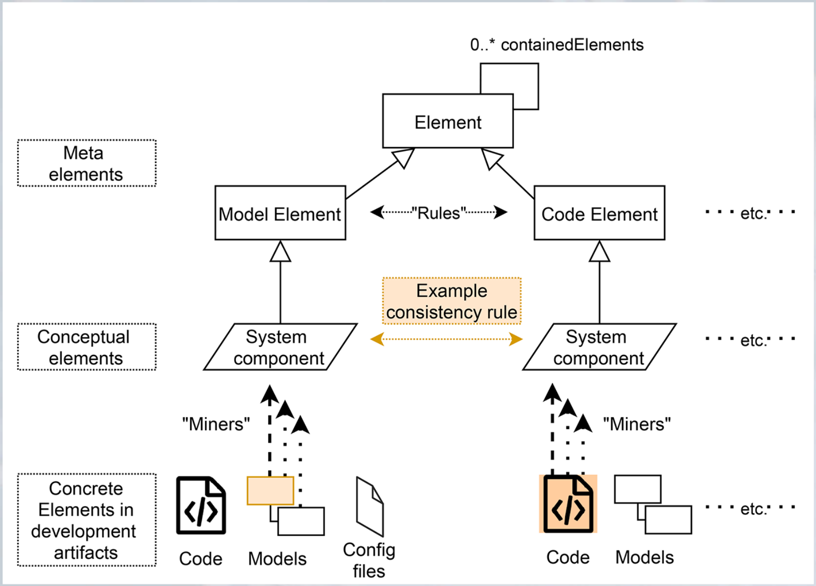
Project acronym: MMInc

Research theme:
Continuous Architecture

Project number: #35

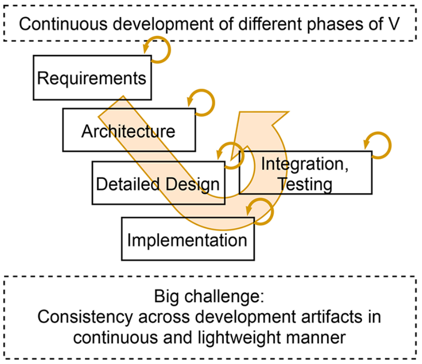
Partners:
Mälardalen University
Grundfos Holding A/S
Saab AB

Associated Partners:
Robert Bosch
Tetra Pak



Summary

When adopting short development cycles, one of the most challenging tasks is to establish and maintain consistency between diverse development artifacts, for example system models, detailed design models, code, requirements, tests, etc. Complete consistency across all artefacts is usually not required nor desirable, but being notified of inconsistencies to prevent their propagation to other development artefacts is important. **Our project aims to provide a framework for lightweight consistency checks across development artefacts to avoid blockages on manual review tasks and ultimately, to support continuous model-based developmen**t.



Consistency checking settings and solutions

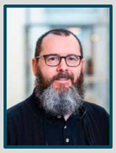
In our most recent study we looked at different industrial settings of model-based development (MBD). We derived four levels of adoption towards continuous MBD and three steps to migrate between the levels.

MBD adoption levels
3) Continuous MBD-ready
2) Linked artefacts, manual checks
1) Disjunct artefacts + semantics
0) Disjunct informal artefacts

Steps between levels
2=>3) Automated checks
1=>2) Bridge models
0=>1) Model with semantics

Contact us:


Robbert Jongeling


Jan Carlson


Federico Ciccozzi


Antonio Cicchetti

<firstname>.<lastname>@mdh.se

www.software-center.se

CHALMERS · UNIVERSITY OF GOTHENBURG · MÄLARDALEN UNIVERSITY SWEDEN · MALMÖ UNIVERSITY · LINKÖPING UNIVERSITY · SAAB · SCANIA · SIEMENS · TOYOTA MATERIAL HANDLING · WÄRTSILÄ · VOLVO · VOLVO · qamcom · JEPPESEN · ERICSSON · BOSCH · GRUNDFOS · CEVT · AXIS COMMUNICATIONS · ESAB · DEIF · zenseact · Tetra Pak

Automated Recovery of Data Pipelines

Aiswarya Raj M*, Jan Bosch and Helena Holmström Olsson
Department of Computer Science and Engineering, Chalmers University of Technology, Sweden
aiswarya@chalmers.se

Software Center

1. Introduction

- Data is the new currency and key to success
- Collecting high-quality data from multiple distributed sources requires much effort
- Data pipelines are implemented in order to increase the overall efficiency of data-flow from the source to the destination since it is automated and reduces the human involvement

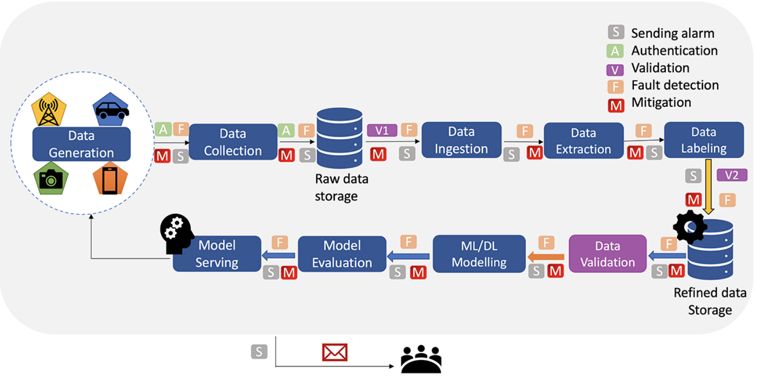
2. Motivation

- Data pipeline failure due to faults at different stages of data pipelines is a common challenge that eventually leads to significant performance degradation of data-intensive systems

3. Objective

- To describe the faults encountered by the practitioners during the development and maintenance of data pipelines and the possible mitigation strategies
- To validate the data pipeline model by implementing fault detection and mitigation in an industrial data pipeline.

4. Conceptual model of Data pipeline

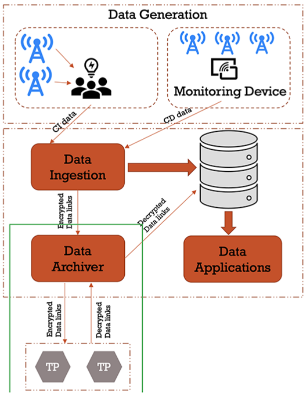


5. Typical faults and mitigation strategies

Stage	Faults	A1	A2	A3	B1	Mitigation Strategies	Sending Alarms			
							A1	A2	A3	B1
Data Generation	Data source failure	X	X	X	X	Set a proxy which never fails		X	X	X
	Inactive data source	X	X		X	Send notification to restart the source		X	X	X
Data Collection	Authentication failure	X	X		X	Functional user credentials	X	X	X	X
	Data sending job failure	X	X		X	Send notification about failure	X	X	X	X
	Unexpected data	X	X		X	Send email to flow guardian		X		X
Data Ingestion	Incompatible ingestion methods	X	X	X	X	Log the error, Define dedicated ingest modules		X	X	X
	Data Extraction faults	X	X		X	Conversion to acceptable format, Formalize the data, Define data extraction method for all data formats		X	X	X
	Change in data formats	X	X		X	Versioning mechanism	X	X	X	X
Data Storage	Insufficient storage	X	X		X	Alarm to the developer and then to support team	X	X	X	X
	Data duplication	X	X		X	Use of HDFS		X		X
	Infrastructure Failure					Sending alarm to IT support				
Data Processing	Transformation faults	X	X		X	Define lossless approaches				
	Unclear definitions and wrong interpretations	X	X	X	X	Contact SMEs				X
Data Sink	Human errors	X	X	X	X	Data validation				
	Schema errors	X	X		X	Define common schema and common language Write different parsers	X	X	X	X
	Dirty data	X	X	X	X	Statistical methods, Data imputation techniques	X	X	X	X

6. Automated Recovery

- Included 2 connector level components: Fault detection and Mitigation
- Inadequate bandwidth or insufficient resources
- Failed dump ids were 32,453 over 30 days(37% of total dump ids).
- Failed dumps are automatically resent as small batches along with the new dumps to the third parties for decryption



7. AI-Powered Fault Tolerance

Anomaly Detection

- Anomalies in data are detected in this phase.
- With the help of data analyst, we check if the anomaly is a fault
- If it is a fault, it will be logged by the data scientist and forms a dataset

Fault Identification

- Dataset from the previous phase will be used for training the Fault identification model in this phase
- Thus, in this phase the fault will be identified automatically. But mitigation strategy will be identified by the data scientist.
- Data scientist then updates the dataset with the fault and the corresponding mitigation strategy

Mitigation Strategy Recommendation

- Updated dataset will be used for training the model in this phase and thus the model will recommend mitigation strategy when the same fault appears again
- However, mitigation should be implemented by the data scientist

Automated Mitigation

- In the final phase, the system by itself implements the mitigation without the help of data scientist

9. References

[1] Aiswarya Raj, Jan Bosch, Helena Holmström Olsson, and Tian J Wang. Modelling data pipelines. In *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 13–20. IEEE, 2020.

[2] Aiswarya Raj, Jan Bosch, Helena Holmström Olsson, Tian J Wang, et al. Towards automated detection of data pipeline faults. In *2020 27th Asia-Pacific Software Engineering Conference (APSEC)*, pages 346–355. IEEE, 2020.

Towards Federated Learning

Hongyi Zhang, Jan Bosch and Helena Holmström Olsson
Department of Computer Science and Engineering, Chalmers University of Technology, Sweden
hongyiz@chalmers.se

Software Center

1. Introduction

- Nowadays, billions of devices constantly generate data.
- Data and ML/DL methods enables the companies to produce better products and smarter services.
- Federated Learning technique does not require centralized data for model training, which is suitable for edge learning scenarios where nodes have limited data.
- However, despite the fact that Federated Learning has significant benefits, companies struggle with integrating Federated Learning components into their systems.

2. Objective

- To identify the reasons why our case company considers Federated Learning as an applicable technique.
- To summarize the services that a complete Federated Learning system needs to support in industrial scenarios
- To identify the challenges that industries are attempting to solve when adopting and transitioning to Federated Learning.

3. Case Study

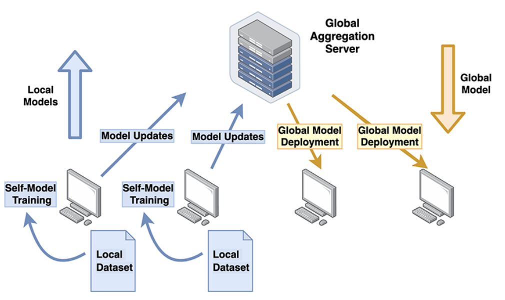
Three different use cases within one of the most well-known ICT (Information and Communication Technology) suppliers to service providers

- Data collection and analysis
- System architecture design and operation
- Machine learning project design, development and operation

4. Benefits

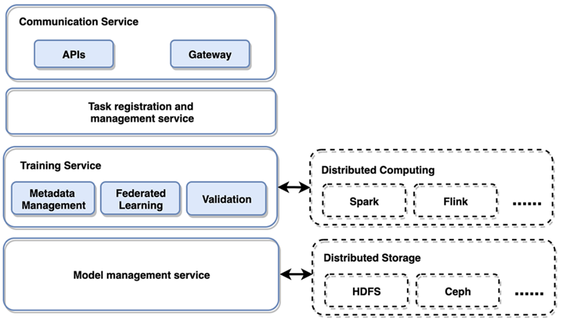
With Federated Learning:

- Data privacy-preserving
- Take advantage of the local computation
- Fast model deployment and evolution (Smarter models)



5. Required Services

- Communication services: A gateway service to handle service routing and expose the API interfaces
- Task registration and management service: Assure the service's high availability
- Training Service: Meta data management and learning monitoring
- Model management service: Store and manage trained model



6. Challenges

We derive the challenges for industries stepping towards reliable Federated Learning services.

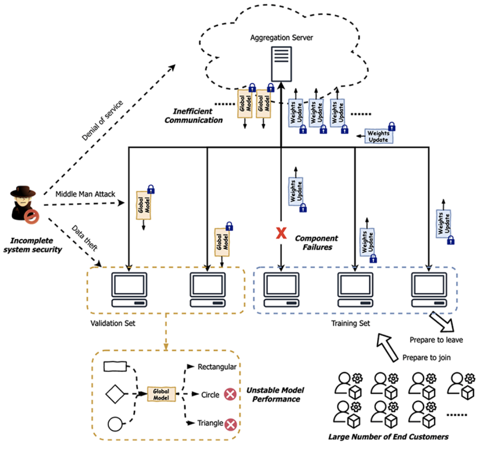
1 - **Components Failures:** Federated Learning system robustness and fault tolerance issues are significantly more prevalent than in traditional distributed system environments.

2 - **Inefficient Communication:** The problems of how to reduce the communication round in real industrial scenarios, how to efficiently utilize network resources while maintaining or even improving model prediction performance still need to be searched and verified.

3 - **Unstable Model Performance:** The problem of how to ensure the model can perform well on all the edge devices in Federated Learning systems are still tricky and need to be verified in different real-world application scenarios.

4 - **Large Number of End Customers:** The mechanism of how to handle device joining in and schedule device utilization still need to be researched and algorithms can be designed.

5 - **Incomplete System Security:** With the increasing attention and focus on AI-powered industrial solutions, security issues are more essential to the service provider as well as commercial Federated Learning systems.



7. References

[1] Hongyi Zhang, Anas Dakkak, David Issa Mattos, Jan Bosch, and Helena Holmström Olsson. Towards federated learning: A case study in the telecommunication domain. In *International Conference on Software Business*. Springer, 2021.

Software Center: Metrics project

Noise Handling For Improving Machine Learning-Based Test Case Selection

Software Center

Vision:

Testing in continuous integration (CI) is a cost-intensive process that requires heavy computational power and long execution time of tests at every CI cycle. We propose a machine learning (ML) based method that uses textual analysis of source code to reduce the size of test suites in CI.

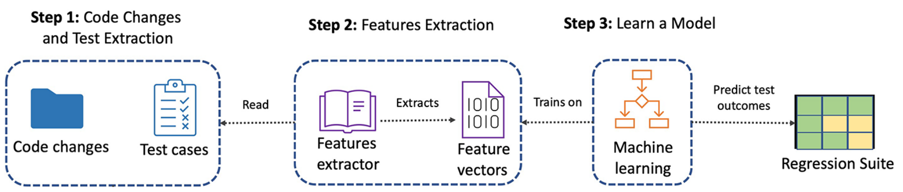
Our vision is to reduce the time required by CI systems to select and execute tests that reveal faults in new code integrations.

Industrial research partners:

- Ericsson
- Grundfos
- Axis

Approach: Method Using Bag of Words for Test Case Selection

- MeBoTS extracts code changes from the development repository.
- Uses Bag of Words for modelling dependencies between code changes and test case execution outcomes.
- Builds a classifier from BoW to predict which code lines will trigger a test case failure/pass.



Class Noise In Source Code Data

```
1   #include "pch.h"
2   #include <iostream>
3   using namespace std;
4
5   int main()
6   {
7       //lines 8  to 11 were added in the first commit
8       for (int i = 0; i < 10; i++)
9       {
10          std::cout << "double number: " << i*2;
11      }
12      //lines 13  to 16 were added in the second commit
13      for (int i = 0; i < 10; i++)
14      {
15          std::cout << "plus 1 number: " << i+1;
16      }
17  }
```

Feature vectors from bag-of-words | Test result

line	literal	int	main	include	{	}	quote	std	cout	for	class
1	1	0	0	1	0	0	2	0	0	0	0
2	1	0	0	1	0	0	0	0	0	0	0
8	0	1	0	0	0	0	0	0	0	1	0
9	0	0	0	0	1	0	0	0	0	0	0
10	1	0	0	0	0	0	0	1	1	0	0
11	0	0	0	0	0	1	0	0	0	0	0
12	1	0	0	0	0	0	0	0	0	0	0
13	0	1	0	0	0	0	0	0	0	1	1
14	0	0	0	0	1	0	0	0	0	0	1
17	0	0	0	0	0	1	0	0	0	0	1

Contradictory entries

Lines 8 and 13 are contradictory lines that were extracted from two revisions.

- In the first revision, the test case failed -> class is set to '0'.
- In the second revision, the test case passed -> class is set to '1'.

Noise handling Approach

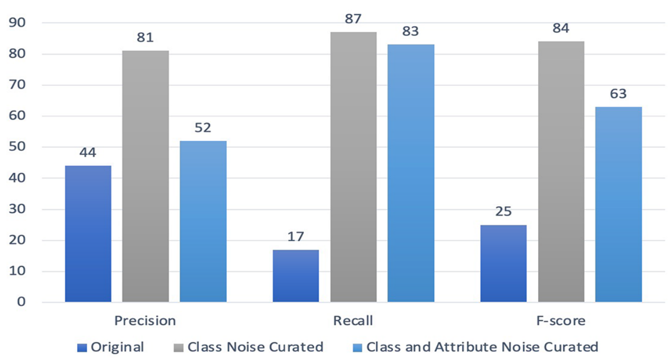
- Select syntactically unique lines.
- Relabel the class value of contradictory entries from 0 to 1.

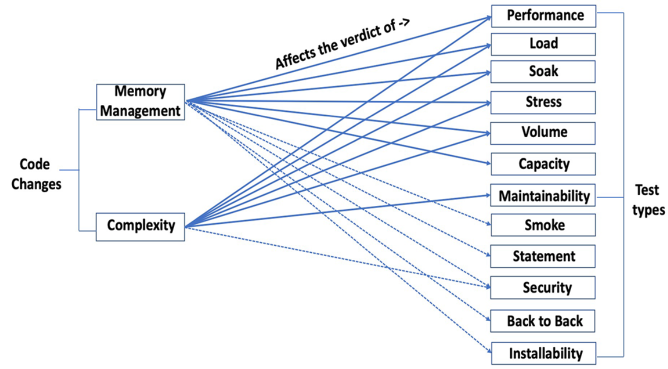
Attribute Noise In Source Code Data

```
10  int main()
11  {
12      if (x > y) {std::cout<< "x is greater than y" <<endl;}
13      if (y > z) {std::cout<< "y is greater than z" << endl;}
14      if (x > z) {std::cout<< "x is greater than z" << endl;}
15      if (x > y and y < z)  { std::cout << "x is greater than z" << endl; }
16  }
```

line	if	var	(	)	{	}	<	>	and	;	class
12	1	2	1	1	1	1	0	1	0	1	1
13	1	2	1	1	1	1	0	1	0	1	1
14	1	2	1	1	1	1	0	1	0	1	1
15	1	4	1	1	1	1	1	1	1	1	1

The attribute values in line 15 deviate from the values in lines 12, 13, and 14.





[Khaled.al-sabbagh, miroslaw.staron, regina.hebig]@gu.se
Department of Computer Science and Engineering
CHALMERS | UNIVERSITY OF GOTHENBURG

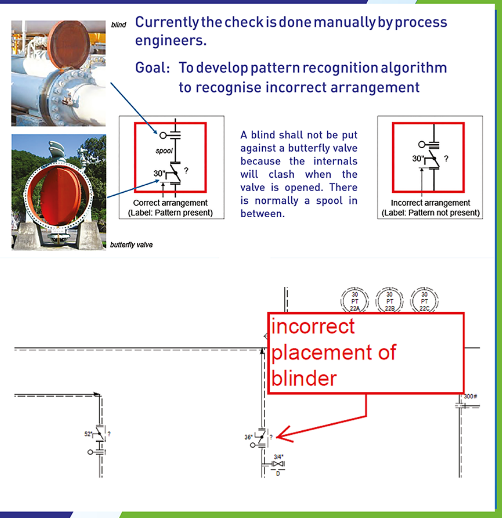
CHALLENGES IN DEVELOPING AND DEPLOYING AI IN THE ENGINEERING, PROCUREMENT AND CONSTRUCTION INDUSTRY

Software Center

MCDERMOTT

INTRODUCTION

- AI application in the engineering, procurement and construction (EPC) industry has not yet proven track records in large-scale projects since
- AI solutions for industrial applications became available only recently, and experience of deployment and lessons learned are still to be built up
- Shortage of in-depth studies that focus on experience from EPC companies with high safety standards like the petrochemical or energy sector



OBJECTIVE

What are the primary challenges the engineering and construction industry is experiencing when selecting, developing, and deploying ML/DL solutions?

INTRODUCTION TO THE AI PROJECTS DEVELOPED WITHIN EPC COMPANY

A. "Defect identification on the engineering drawings"

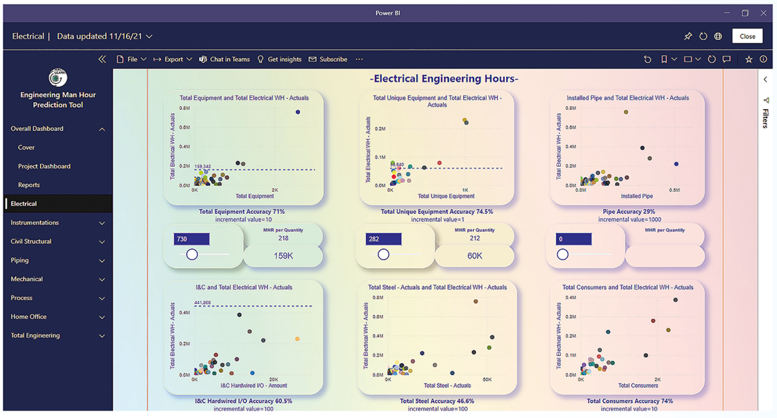
Can highlight the specific patterns on P&IDs (piping and instrumentation diagrams) identified as defects or, in other words, engineering mistakes.

Users: Process engineers might use the developed tool to review engineering drawings before issuing them to the construction team.

B. "Engineering hours budget prediction tool"

Utilizes historical data for future engineering hours budget prediction

Users: The developed engineering hours prediction tool is used by functional management as a reference when reviewing the hours' estimate made by discipline lead engineers during the proposal phase of the project.



Project Challenging Mapping

Category	Challenges	Use Case 1: *Engineering hours budget prediction tool*	Use Case 2: *Mistakes recognition on the engineering drawings*	Use Case 3: *Digitalization of PDF drawings into CAE (Computer Aided Engineering) compatible files*
Business challenges	Challenges in adequately estimate the business benefit	-	-	✓
	Challenges in identifying the success metrics for AI/ML proposed solutions	-	✓	✓
	Challenges in identifying the resources to be invested to develop and deploy the AI solution	✓	✓	✓
Architectural challenges	Model development challenges	-	✓	-
	High variation of graphical symbol representation	-	✓	✓
	Challenges in an integration into company software platforms	-	✓	✓
	Unpredictable Performance	-	✓	✓
	Legal constrains	-	-	✓
Process challenges (Data collection & interpretation)	Unregistered databases which are run in silos	✓	-	-
	Different practices of collecting data across company	✓	-	-
	No standard definition of harvested statistics and when those statistics shall be collected	✓	-	-
	Low priority activity	✓	-	-
	Poor definition of the collected statistics	✓	-	-
Organizational Challenges	Inadequate Cost Time Resources estimation	-	✓	✓
	Challenges in goals setting	✓	✓	✓

CONCLUSION

We described in detail the primary challenges which EPC industry is experiencing when selecting, developing, and deploying ML/DL solutions.

The findings show:

- Domain experts must gain hands-on experience managing AI projects to overcome business, development and organizational constraints
- Top management commitment is essential to support the start of AI project development and provide the resources to launch the fail-safe projects that are necessary before deploying AI tools on a large scale

Transforming Automotive Architecture with Assistance from AI

Software Center

Vision:

To train a deep-learning model that understands software design and eventually use it to assist software design tasks. The collaboration of AI and human experts on software design has the potential to accelerate software engineering by maintaining high-quality software architecture over time.

Current work:

Programming Language Understanding (PLU) models are typically trained on large corpora of source code. We ask the question – has a PLU model, which has seen millions of coding examples during training, implicitly picked up ideas of design? We test this by setting design-related challenges to a pre-trained model.

Metrics theme

Research partners:

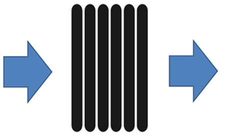
- CEVT
- Chalmers
- Volvo AB
- Volvo Cars

STEP 1: Training a PLU model for the C-language

```
// Get BatterySOC MMMM
input->batterySOC.status = SIG_NOK;
rteRet = Rte_Read_BatterySOC_rqst(&input->batterySOC.data);
if (rteRet == RTE_E_OK)
{
    rangeRet = validateSignal(input->batterySOC.data);
    // Check range
    MMMM (rangeRet == SIG_IN_RANGE)
    {
        // Everything OK, assign signal
        input->batterySOC.MMMM = SIG_OK;
    }
}
```

Masked language modeling

Multi-layer transformer



```
// Get BatterySOC signal
input->batterySOC.status = SIG_NOK;
rteRet = Rte_Read_BatterySOC_rqst(&input->batterySOC.data);
if (rteRet == RTE_E_OK)
{
    rangeRet = validateSignal(input->batterySOC.data);
    // Check range
    if (rangeRet == SIG_IN_RANGE)
    {
        // Everything OK, assign signal
        input->batterySOC.status = SIG_OK;
    }
}
```

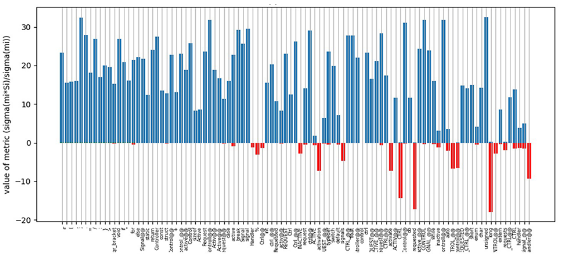
STEP 2: Testing/probing the model on AUTOSAR knowledge

; + ( ] struct case if switch return | signal handler control active

Less design sensitive | More design sensitive

Is the model finding these tokens easy to predict when masked?

And these difficult?



First results:

- A C-language PLU model pre-trained on ~100 million non-automotive C-files from GitHub seems to be quite comfortable in handling AUTOSAR-C code
- This is an encouraging sign that this PLU model understands key AUTOSAR design contexts, and can help solving design (and many more) related tasks

Ongoing work – we're currently testing if the PLU model understands AUTOSAR design relationships. Here again initial results are encouraging

dhasarathy.Parthasarathy@volvo.com
Volvo AB

CEVT VOLVO CHALMERS | UNIVERSITY OF GOTHENBURG

Architectural Design and Verification/Validation of Systems with Machine Learning Components

Project acronyme: DeVeLop

Research theme: Metrics

Partners:
Volvo Car Corporation
Chalmers University of Technology
University of Gothenburg

Contact us:

Vasilii Mosin: vasilii.mosin@volvocars.com

Darko Durisic: darko.durisic@volvocars.com

Miroslaw Staron: miroslaw.staron@chalmers.se

Project Background

Due to their stochastic nature, using machine learning (ML) components requires specific approaches related to the architectural design and verification/validation of the vehicle's electrical system. On the one side, suitable architecture containing both stochastic and traditional deterministic components shall be defined which assures safety, robustness, fault tolerance, and the exchange of data between ML components and the rest of the vehicle. On the other side, the nature of ML algorithms makes them difficult to test and analyze from a safety perspective, which requires implementation of specific verification and validation methods.

Semantic Anomaly Detection (RQ3)

Deep learning (DL) algorithms solving perception tasks in the automotive industry can fail when applied on data which is significantly different from the training data. In order to minimize the risks related to these failures, anomaly detection methods can be used, in particular, anomaly detection approaches based on autoencoders (Fig. 1).

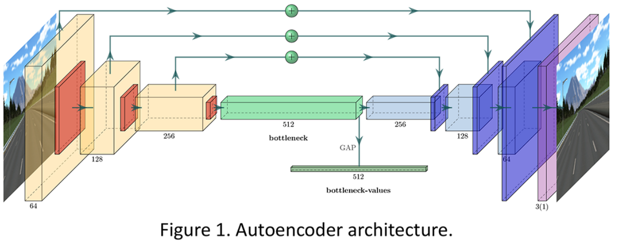


Figure 1. Autoencoder architecture.

When an autoencoder is trained using only normal data, the reconstruction error calculated as the difference between the output in the input data can be used as anomaly score. In this study, we answer the following RQ: How capable are autoencoders in detecting semantic anomalies in highway driving scenarios? Fig. 2 presents examples of the input, output, and reconstruction error of the autoencoder. The results obtained in this study show that the autoencoders are capable of detecting semantic anomalies to some degree.

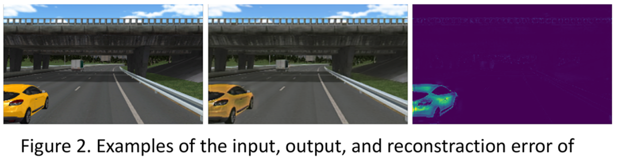


Figure 2. Examples of the input, output, and reconstraction error of the autoencoder.

General Research Questions

RQ1: How do we make a sustainable architectural design of an automotive system as a combination of individual ML software components and traditional safety critical components?
RQ2: How to assure quality of the architectural design of future car architectures developed in an agile way, where ML and stochastic calculations are part of the safety-critical functions?
RQ3: What are reliable measures of model certainty that can be used as quality indicators by systems architects and how do we infer them from the data and model?

Test Input Prioritization (RQ3)

The goal of test input prioritization is to prioritize test inputs that are more likely to reveal the errors of a ML algorithm faster during the testing. It can also reduce the testing cost by eliminating the need of labelling of unnecessary test inputs. In this study, we compare white-box, data-box, and black-box input prioritization techniques in terms of their effectiveness and efficiency (RQ: How effective and efficient are surprise adequacy, autoencoder-based, and similarity-based input prioritization approaches for testing DL algorithms?). Fig. 3 demonstrates the measured effectiveness of different approaches as the area under the curve representing the cumulative number of errors growing during the testing against the test inputs in the prioritized order. The results show that the different techniques are more and less effective and efficient in different cases.

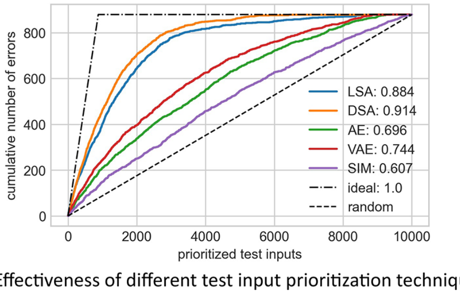


Figure 3. Effectiveness of different test input prioritization techniques.

Architectural Patterns for ML (RQ1)

The goal of this case study is to document experiences of which of the emerging ML architectural patterns are used in modern cars already, which are planned for the future and how they are used. We study the software architecture of a modern car product line based on existing patterns and workshops with software architects (RQ: How applicable are the architectural patterns for ML in vehicle software architecture in practice?). The results show that many ML-specific patterns are used or planned to be used in the near future (Fig. 4).

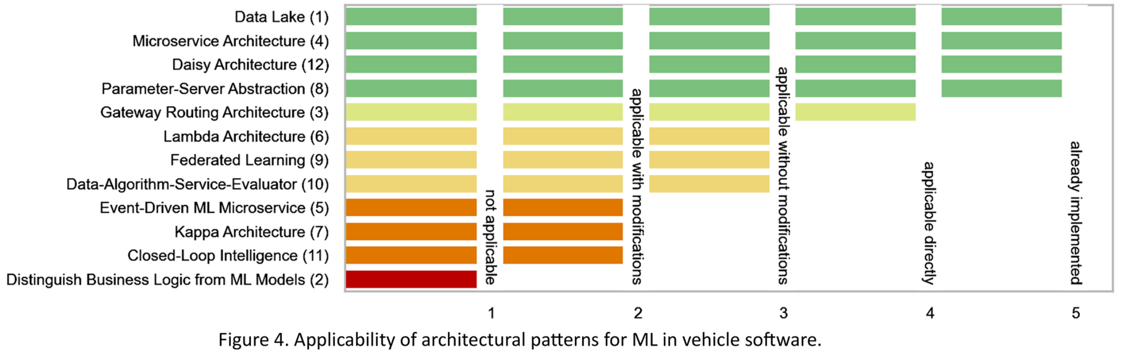


Figure 4. Applicability of architectural patterns for ML in vehicle software.