PV226 Lasaris Seminar

Towards Antifragile Critical Infrastructure Systems (Introduction)

Hind Bangui, Barbora Buhnova and Bruno Rossi*

*** brossi@mail.muni.cz**

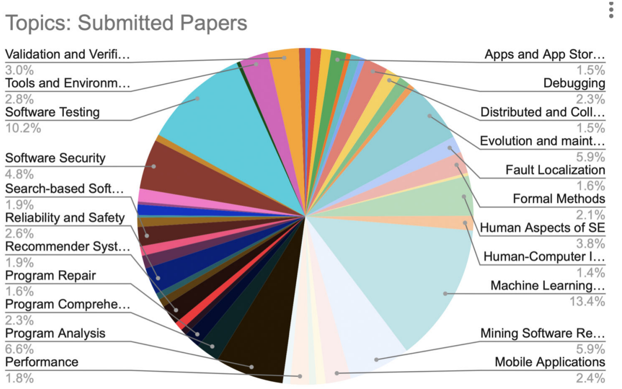
Department of Computer Systems and Communications,
Lasaris (Lab of Software Architectures and Information Systems)
Masaryk University, Brno

lasaris
www.lasaris.cz

(: Off-topic first :)

Where is Software Engineering heading?

ICSE 2022 submissions results from Andreas Zeller



https://docs.google.com/spreadsheets/d/1ENiA2w9wHrawZSXYUXxodiNFQvzZeMp_pMhccl68WJE/edit#gid=0

Andreas Zeller
@AndreasZeller
Follow

Software researcher at @CISPA. I work on @FuzzingBook, @Debugging_Book, and more. Testing, debugging, analyzing, and protecting software for a better world.

Top 10 Topics – Accepted

Topics	# Submitted Papers	# Accepted Papers	Acceptance Rate
Software Fairness	5	4	80,00%
Software Economics	4	2	50,00%
Program Synthesis	11	5	45,45%
Variability and Product Lines	11	5	45,45%
Configuration Management	16	7	43,75%
Green and Sustainable Technologies	5	2	40,00%
Fault Localization	28	11	39,29%
Software Ecosystems	18	7	38,89%
Release Engineering and DevOps	11	4	36,36%
API Design and Evolution	17	6	35,29%

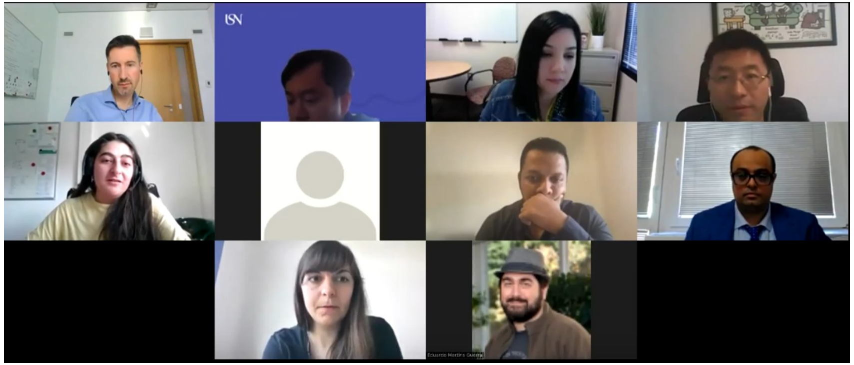
Top 10 Topics – Rejected

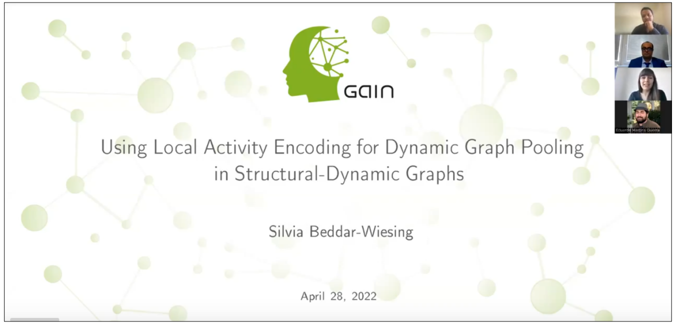
Topics	# Submitted Papers	# Accepted Papers	Acceptance Rate
<none>	9	0	0,00%
Modeling and Model-Driven Engineering	16	1	6,25%
Agile Methods and Software Processes	18	2	11,11%
Software Architecture and Design	16	2	12,50%
Requirements Engineering	22	3	13,64%
Embedded/Cyber-Physical Systems	19	3	15,79%
Parallel, Distributed, and Concurrent Sys	12	2	16,67%
Software Visualization	6	1	16,67%
Software Reuse	22	4	18,18%
Ethics in Software Engineering	11	2	18,18%

Top 10 Topics – Submitted

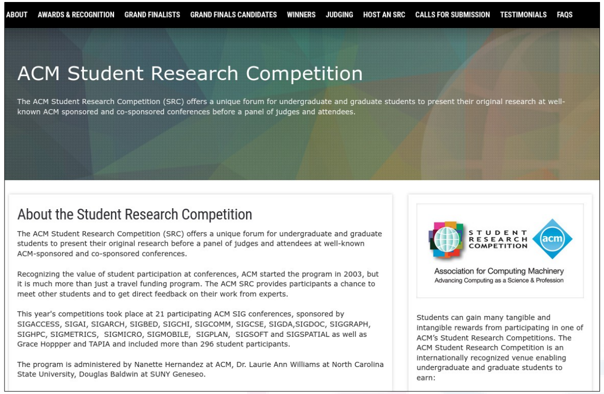
Topics	# Submitted Papers	# Accepted Papers	Acceptance Rate
Machine Learning with and for SE	237	74	31,22%
Software Testing	181	47	25,97%
Program Analysis	117	35	29,91%
Evolution and maintenance	105	31	29,52%
Mining Software Repositories	105	23	21,90%
Software Security	85	25	29,41%
Human Aspects of SE	68	20	29,41%
Validation and Verification	53	15	28,30%
Tools and Environments	49	12	24,49%
Reliability and Safety	46	15	32,61%

Student Research Competition at SAC’22





I suggest interested students to have a look at the ACM SRC page → https://src.acm.org



(: Back to the topic :)

Where is Software Engineering heading?

Ian Sommerville
@IanSommerville

I have been thinking about the how well we have done in software engineering research. Too long for a thread so I've written about the underwhelming impact of software engineering research.

iansommerville.com/technology/res…

@BNuseibeh @lionel_c_briand @profserious

11:13 AM · Apr 6, 2022 · TweetDeck

42 Retweets **10** Quote Tweets **105** Likes

https://iansommerville.com/technology/research-impact

The Underwhelming Impact of Software Engineering Research (April 2022)

This article was prompted by responses to a tweet I wrote in response to an analysis of the research area of papers submitted to the 2022 ICSE conference, the flagship conference for software engineering research. These led me to reflect on software engineering research in general. I'm sorry to say that I think that we, as a community, have not really delivered very much that's substantive in over 40 years of research.

Fundamentally, I think there are 3 related root causes of this situation:

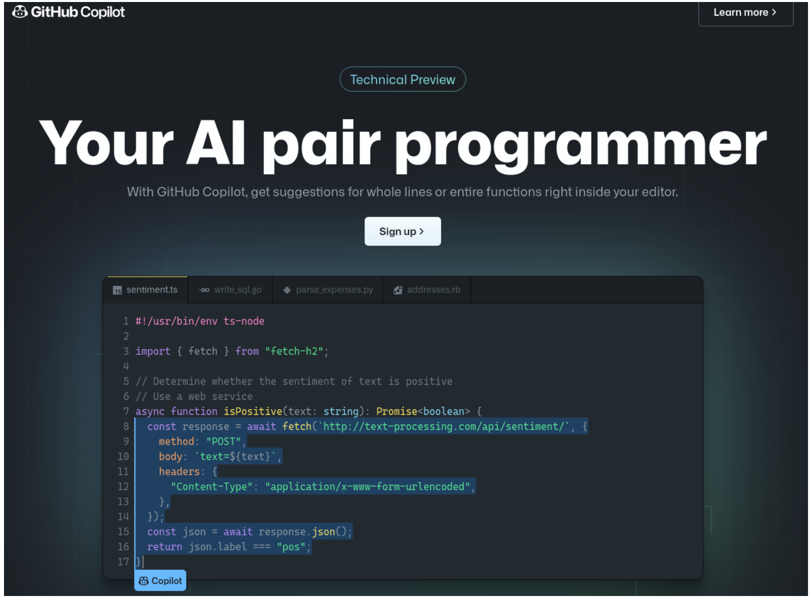
- Short-termism.
- Reductionist thinking
- Competition rather than cooperation

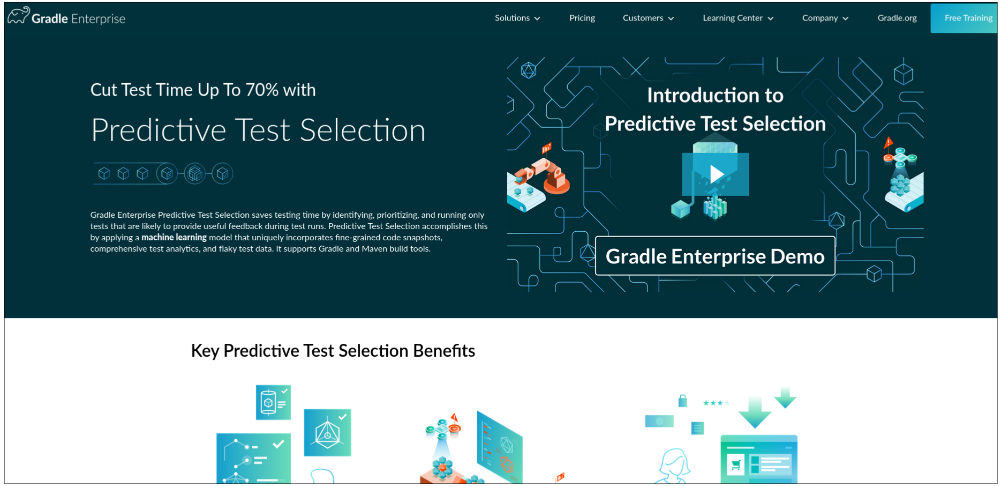
Short-termism is now endemic in software engineering (and indeed in all computer science) research. This is partly a consequence of the 'publish or perish' culture that started in the 1970s and that is now endemic in most countries. Researchers have to keep writing papers to remain credible - and who can blame them? There is a focus on short-term projects, publication of interim results, an unwillingness to invest in producing robust and reusable demonstrator systems and a tendency to jump on whatever bandwagon is fashionable at a particular moment in time (e.g. formal methods in the 1980s, machine learning now).

This short-termism has the direct consequence that much software engineering research is lacking ambition. Researchers who are mindful of their success metrics are reluctant to tackle long-term difficult projects. In the UK, in the early 2000s, there was a proposal to identify 'Grand Challenges in Computing Science'. Some interesting projects, such as a verifying compiler, were proposed - none came to fruition.

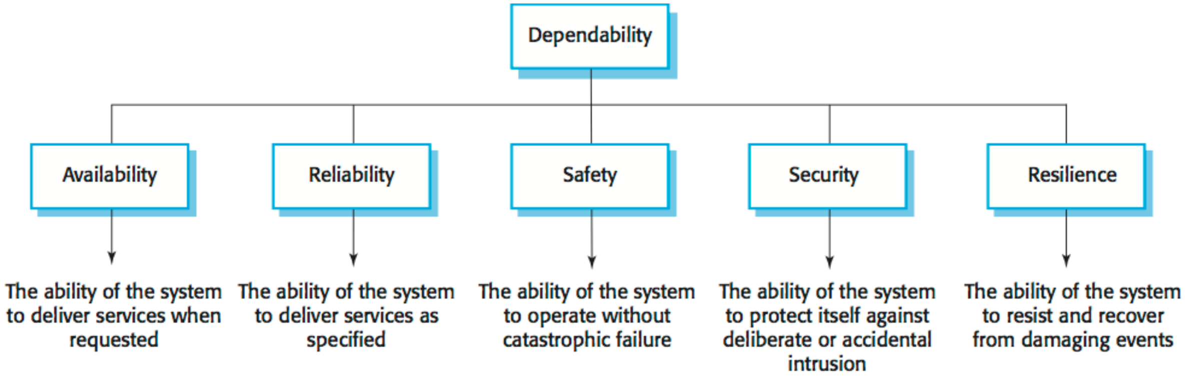
Short-termism has been exacerbated by the policy of many funding agencies, such as the European Commission, that academic research should be collaborative with industry. Industry, quite understandably, does not see its role as a long-term research funder and focuses on short or, at best, medium term research. Unfortunately, for both good and bad reasons, this research has a low priority for many companies. My experience over 30 years is that it is often under-funded, cancelled at short notice, and inadequately staffed.

Where is Software Engineering heading?





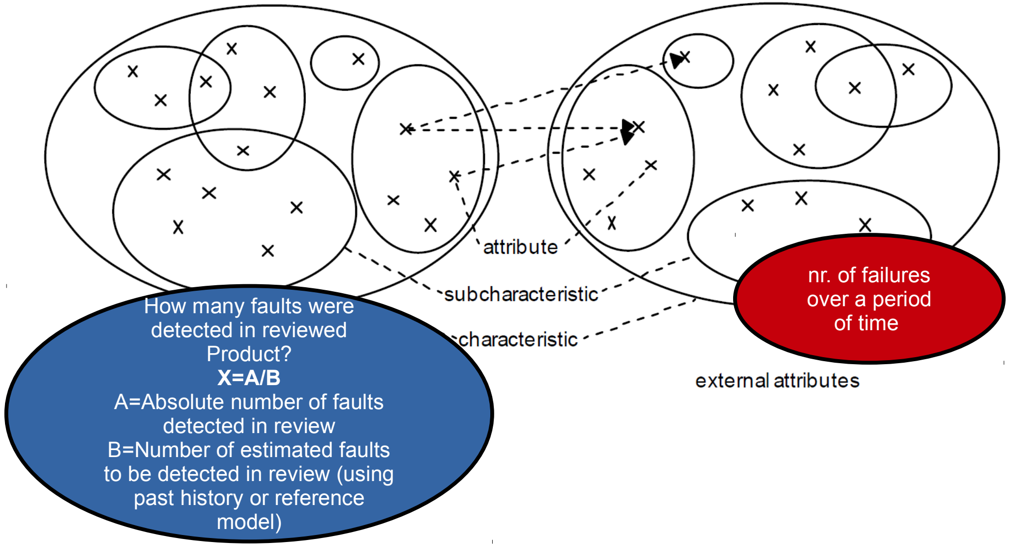
The Traditional View of Software Reliability & Resilience



The traditional view from I. Sommerville

- **Reliability** is the probability that a system will work as designed
- **Resiliency** can be described as the ability to a system to self-heal after damage, failure, load, or attack
- Some assumptions in SE:
  - **Faults** in Software / Hardware **might lead to failures**
  - We can try to **predict** and **take countermeasures** based on the analysis of past history
  - All models are based on **monothonic behaviour** (i.e., the fact that there are no **concept drifts**)
  - We can **adapt systems** based on our models of failure detection / location

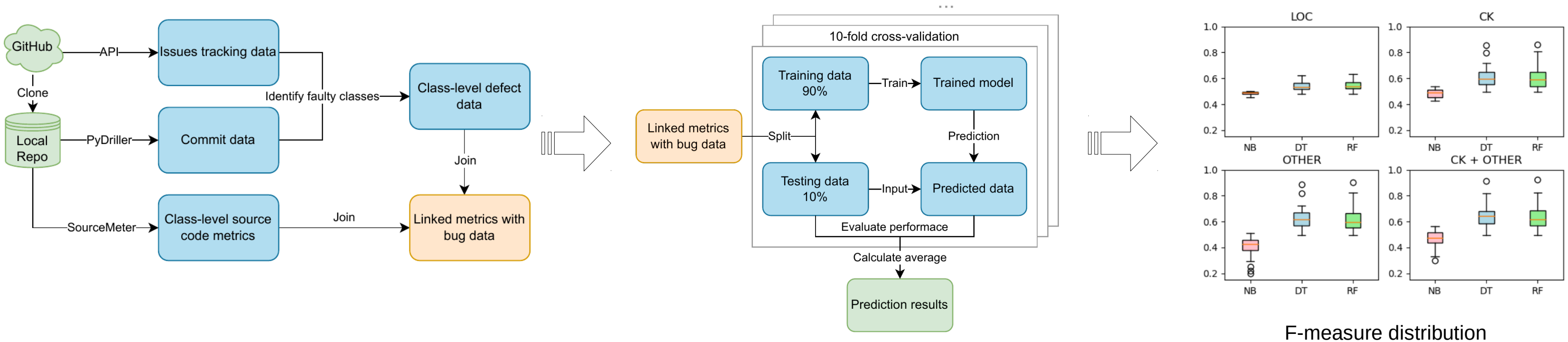
The Traditional View of Software Reliability & Resilience



- We do not know or cannot search through the whole space of failures
- We build models and use proxies (as faults) to estimate the failures and adapt systems **ex-post**

Defects Prediction as proxies

- We can have prediction models telling us about the prediction of defects in code



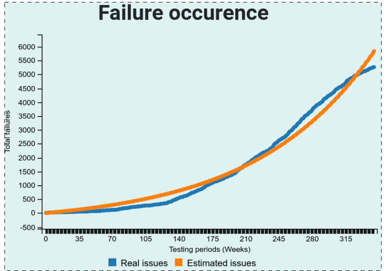
F-measure distribution

- It is assumed that the **more defects → the more the failures**
- look into code and improve to avoid future failures (for e.g., to see which modules require more attention)
- We need to develop/refactor, redeploy, etc… This is an **old** view of how software systems are built

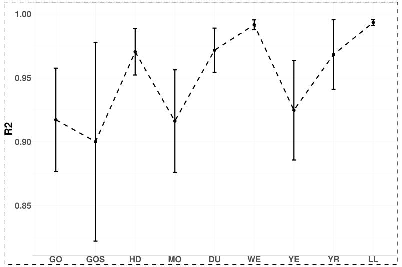
→ Software Metrics for Software Defects Prediction, Master's Thesis, Dominik Arne Rebro – supervisor: Bruno Rossi to be defended at FI MU 2022.

Software Reliability Growth Models (SRGM)

- We can try to fit the cumulative failure data curve to see which models could be better in giving us an estimate of our failures - clearly impossible to get *one-fits-all* models



Fitting all models for a project



RQ2 - Overview of the GoF (R^2) for the project categories and individual models (top-3 models highlighted).

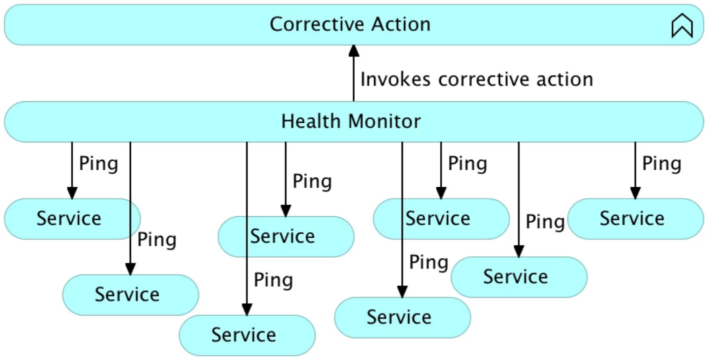
Model	C1		C2		C3		C4		C5		C6		C7		C8	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
DU	0.989	0.008	0.982	0.016	0.850	0.269	0.976	0.027	0.982	0.016	0.976	0.020	0.952	0.076	0.960	0.080
GO	0.949	0.056	0.971	0.018	0.766	0.283	0.952	0.061	0.958	0.040	0.970	0.027	0.934	0.050	0.945	0.079
GOS	0.842	0.331	0.949	0.114	0.774	0.255	0.845	0.259	0.872	0.256	0.987	0.008	0.969	0.026	0.980	0.017
HD	0.970	0.043	0.977	0.021	0.988	0.011	0.968	0.072	0.966	0.033	0.982	0.028	0.954	0.041	0.988	0.008
LL	0.996	0.002	0.993	0.009	0.871	0.277	0.993	0.007	0.989	0.009	0.994	0.009	0.984	0.009	0.994	0.055
MO	0.947	0.058	0.970	0.018	0.757	0.301	0.947	0.059	0.957	0.040	0.967	0.026	0.910	0.103	0.952	0.021
WE	0.995	0.003	0.993	0.009	0.865	0.274	0.993	0.008	0.987	0.011	0.994	0.008	0.958	0.077	0.993	0.008
YE	0.950	0.056	0.971	0.018	0.762	0.286	0.952	0.061	0.958	0.040	0.970	0.027	0.935	0.050	0.969	0.028
YR	0.985	0.017	0.986	0.014	0.921	0.165	0.984	0.018	0.975	0.030	0.987	0.009	0.970	0.023	0.985	0.012

→ Rossi, B., Russo, B., & Succi, G. (2010). Modelling failures occurrences of open source software with reliability growth. In IFIP International Conference on Open Source Systems (pp. 268-280). Springer, Berlin, Heidelberg
→ Chren, S., Micko, R., Buhnova, B., & Rossi, B. (2019). STRAIT: a tool for automated software reliability growth analysis. In 2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR). IEEE.
→ Radoslav Mičko, Software Reliability Growth Models for Open Source Software. Master Thesis FI MU, 2022.
→ Chren, S., Micko, R., Buhnova, B., & Rossi, B. Applicability of Software Reliability Growth Models to Open Source Software, to appear.

Self-healing Systems

- Modern systems of systems **embrace** failures
- Have monitoring capabilities and can **self-adapt** to emerging situations
- Can take **action to restart services** / processes → e.g., the circuit breaker pattern

  Examples are Microservices



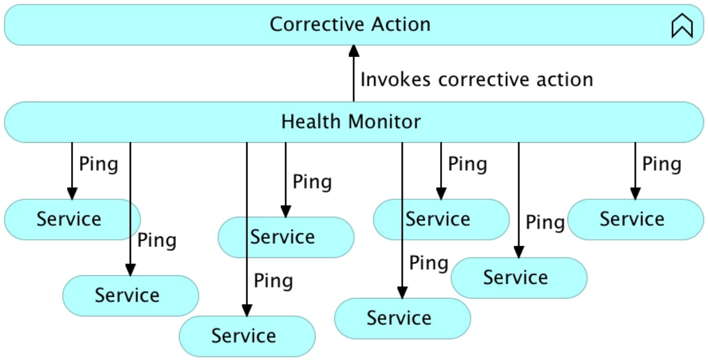
https://www.javacodegeeks.com/2016/01/self-healing-systems.html

→ Štefanko, M., Chaloupka, O., Rossi, B., van Sinderen, M., & Maciaszek, L. (2019). The Saga pattern in a reactive microservices environment. In Proc. 14th Int. Conf. Softw. Technologies (ICSOFT 2019) (pp. 483-490). Prague, Czech Republic: SciTePress.
→ Zezulka, M., Chaloupka, O., & Rossi, B. (2021). Integrating Distributed Tracing into the Narayana Transaction Manager. In COMPLEXIS (pp. 55-62).

Self-healing Systems

- Modern systems of systems **embrace** failures
- Have monitoring capabilities and can **self-adapt** to emerging situations
- Can take **action to restart services** / processes → e.g., the circuit breaker pattern

  Examples are Microservices



https://www.javacodegeeks.com/2016/01/self-healing-systems.html

What we are missing is the capability of systems to **learn** when self-adapting to failure

Learning from failures, take countermeasures, and self-adapt

This can be part of System-of-Systems modelling of "**emerging behaviour**"

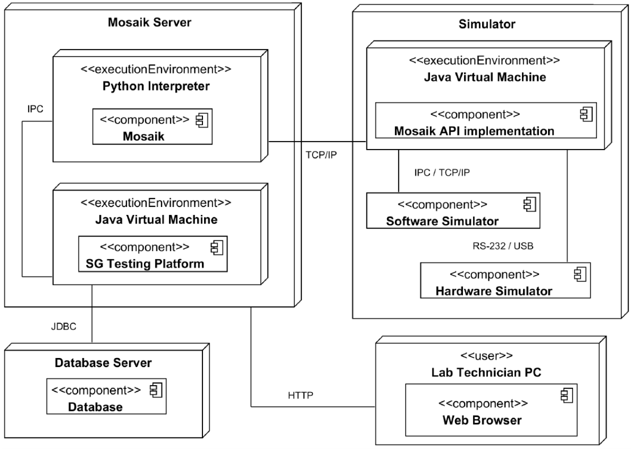
→ Štefanko, M., Chaloupka, O., Rossi, B., van Sinderen, M., & Maciaszek, L. (2019). The Saga pattern in a reactive microservices environment. In Proc. 14th Int. Conf. Softw. Technologies (ICSOFT 2019) (pp. 483-490). Prague, Czech Republic: SciTePress.

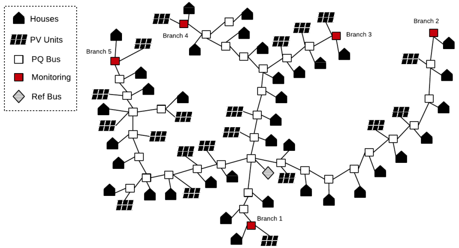
→ Zezulka, M., Chaloupka, O., & Rossi, B. (2021). Integrating Distributed Tracing into the Narayana Transaction Manager. In COMPLEXIS (pp. 55-62).

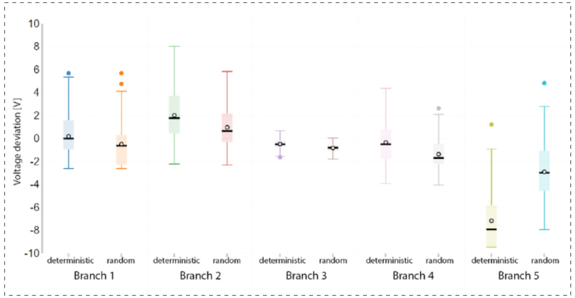
Using Simulations to learn expected behaviour (1/2)

- In previous work we created a **testing management platform** for Smart Grids based on the **Mosaik** framework for **co-simulations**
- We extended Mosaik with the disconnect method to remove edges from the dataflow graph and the entity graph → A simple way to simulate node failures

Smart Grids Testing Processes



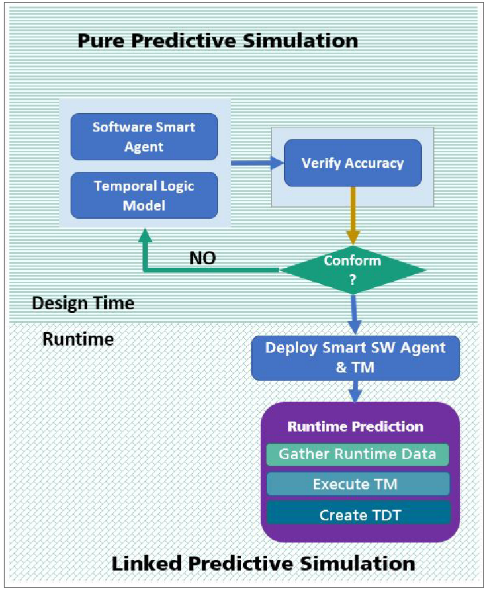




→ Mihal, P., Schvarcbacher, M., Rossi, B., & Pitner, T. (2022). Smart grids co-simulations: Survey & research directions. Sustainable Computing: Informatics and Systems,.
→ Schvarcbacher, M., Hrabovská, K., Rossi, B., & Pitner, T. (2018). Smart grid testing management platform (sgtmp). Applied Sciences, 8(11), 2278.
→ Gryga, L., & Rossi, B. (2021). Co-simulation of Smart Grids: Dynamically Changing Topologies in Failure Scenarios. In COMPLEXIS (pp. 63-69).

Using Simulations to learn expected behaviour (2/2)

- What about **comparing** results from **simulations** and **"real runs"** to determine expected behaviours?
- Systems can learn from running the system and simulation → AI can help in determing what could be the best course of action
- Simulation → failure vs reality→ failure?



TM = Temporal Model
TDT = Temporal Digital Twin

Needs definition of **what is an anomaly** as well

Can be done at **design time** or at **runtime (in real-time)**

→ Cioroaica, E., Blanco, J., Rossi, B. Timing Model for Predictive Simulation, conference paper, under revision.
→ Hind, B., Buhnova, B., Rossi, B.. Shifting Towards Antifragile Critical Infrastructure Systems. In Proceedings of the 7th International Conference on Internet of Things, Big Data and Security (IoTBDS 2022).

Maybe we need to move forward from the concept of resilience…

Maybe we need to move forward from the concept of resilience…

This is where the next talk starts

Maybe we need to move forward
from the concept of resilience…

This is where the next talk starts

Hind Bangui will have all the answers :)