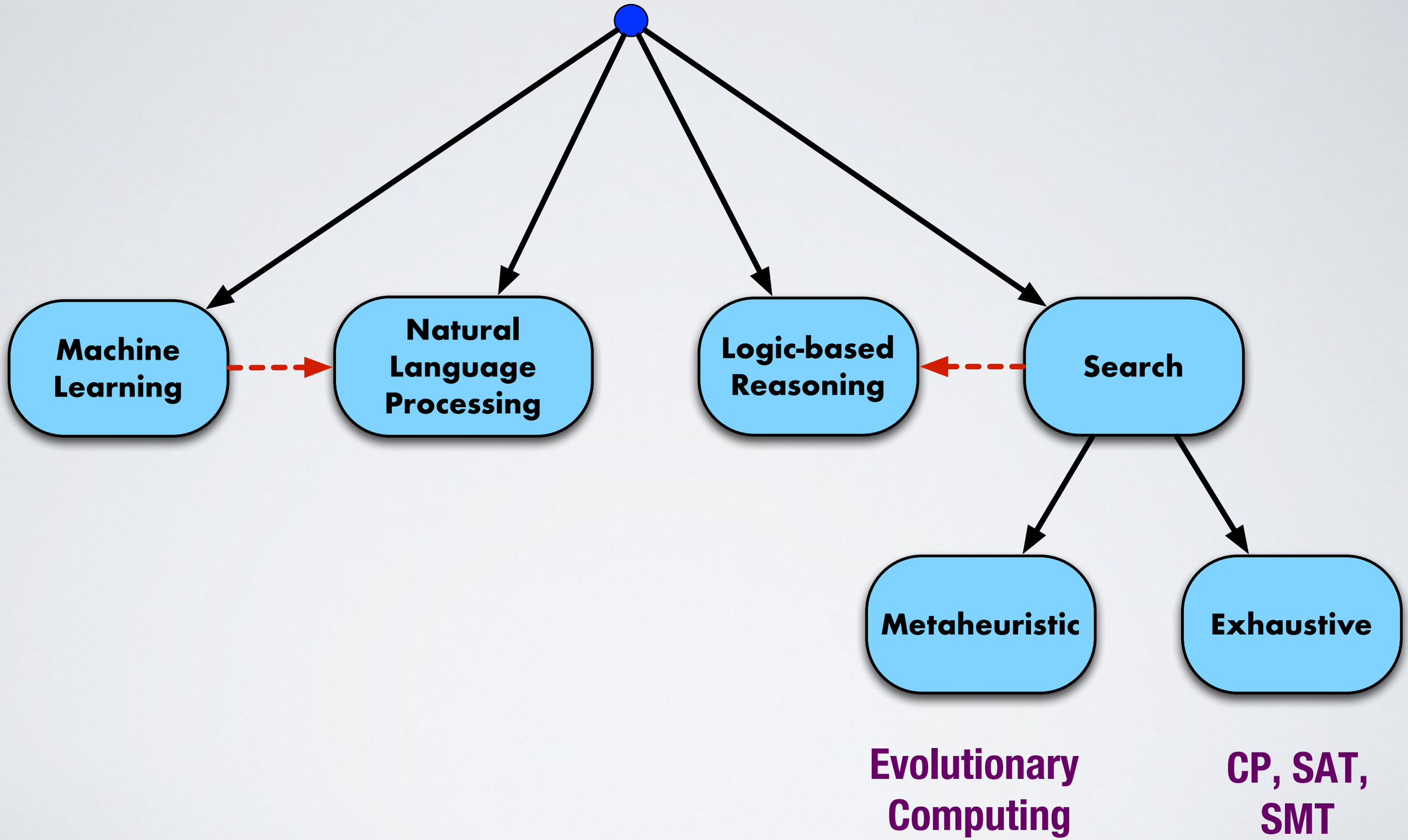
AI in SE: A 25-year Journey

Lionel Briand

PROMISE 2020

AI in SE


Machine Learning
Natural Language Processing
Logic-based Reasoning
Search
Metaheuristic
Exhaustive
Evolutionary Computing
CP, SAT, SMT

1991

Predicting faults in flight dynamics software

Colleagues and friends:
“Machine learning?
Why would you want to apply this?
This is not serious.”

Acknowledgments

- Shiva Nejati
- Fabrizio Pastore
- Mehrdad Sabetzadeh
- Chetan Arora
- Sallam Abualhaija
- Raja Ben Abdessalem
- Chunhui Wang
- Seung Shin
- Donghwan Shin
- Reza Matinnejad
- Annibale Panichella
- Hazem Fahmy
- Erik Arisholm
- Stefano Di Alesio
- Tao Yue
- Shaukat Ali
- Ghanem Soltana
- Fitash Ul Haq
- Khouloud Gaaloul
- Yvan Labiche
- Marwa Shousha
- Amin Sleimi
- Marcello Cecci
- …

Objectives

- Report on many years of experience about leveraging AI on **industrial research SE projects.**
- **Personal experience**, not a survey.
- Partial presentation (very much so).
- Focus on **real problems**, real solutions, in real contexts.
- **Example projects** and lessons learned.

Tim Menzies, SE for AI for SE, https://www.youtube.com/watch?v=7dJ5Oc9cblg

< ~2000

Making software development predictable

Context

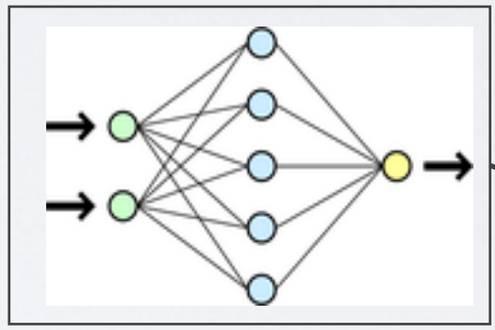
- Software development data repositories were few.
- Data available to researchers was scarce and hard to use.
- Research focus on resource and defect prediction.
- Hundreds of research papers.

Evolving Telecom System

For each type of CR involving this class
- Number of CRs
- Lines of code added and deleted in this class
- Number of CPs involving this class
- Total number of files changed in CRs
- Total number of tests failed in CRs
- Total number of developers involved in CRs
- Total number of past CRs of the developers

Class complexity
- Class size
- Coupling
- Cohesion
- …

Class change and fault history:
For each type of CR involving this class
And for the past three releases:
- number of CRs (n-1, n-2, n-3)



Neural Network
C4.5
Logistic regression, etc.

Highly fault-prone

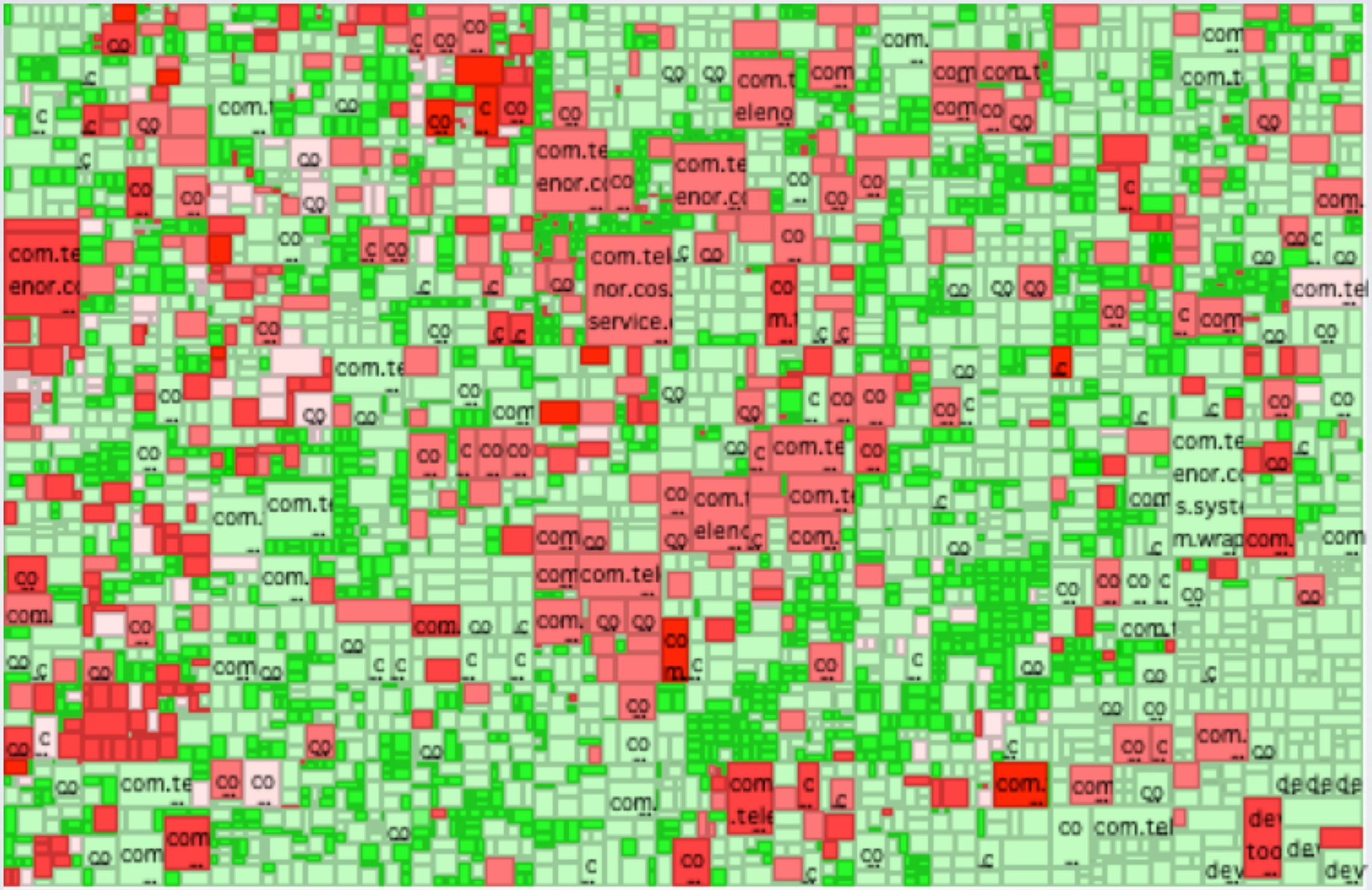
1.0

0.0

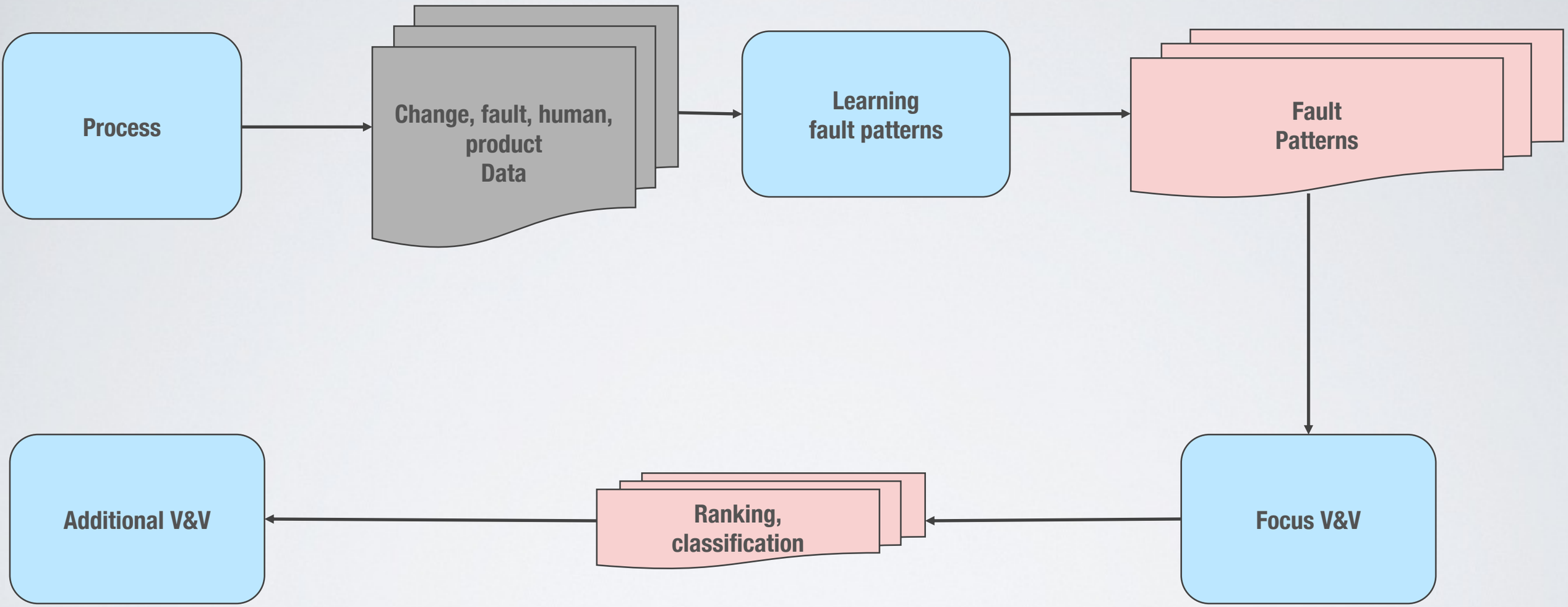
Not fault-prone

Arisholm et al. Journal of Systems and Software, 2010

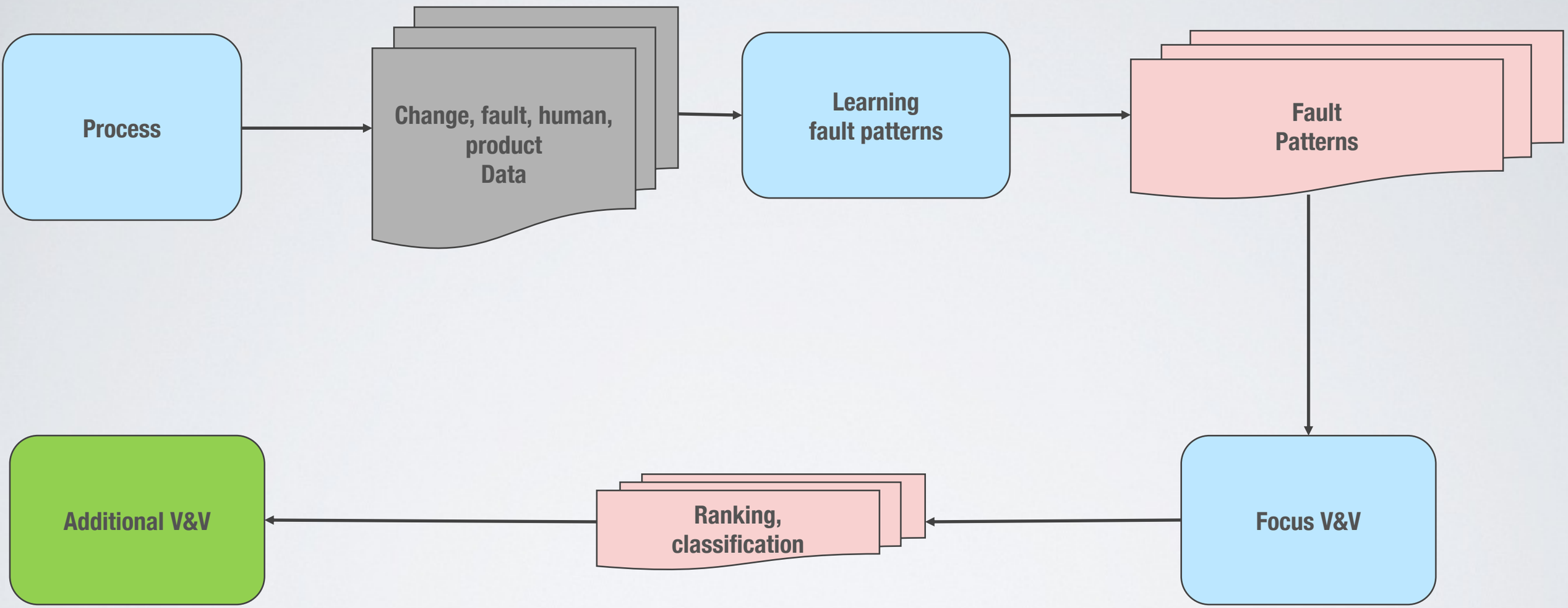
Tree Maps – Class Level



SVV Cost-Effectiveness



Actionable?



Mapping predictions to V&V practices is not easy

ML for Prediction: Benefits

- Some machine learning techniques, such as random forests, tend to be more accurate than classical statistical techniques, e.g., based on regression.

- More flexible and robust (less assumptions), less prone to overfitting, etc.

- As larger amounts of data became increasingly available, their application became more widespread.

- Mining Software Repositories

T. Menzies, “Machine Learning: A Tutorial”

ML for Prediction: Challenges

- Building and maintaining a corporate prediction system.
- We are predicting moving targets as development practices evolve quickly.
- Lots of papers on how to build prediction models.
- Very few papers on how to effectively use such prediction models, their benefits, etc.
- How to use them in a cost-effective way is far from obvious.

> ~2000

The rise of Search-Based SE (SBSE)

Why SBSE?

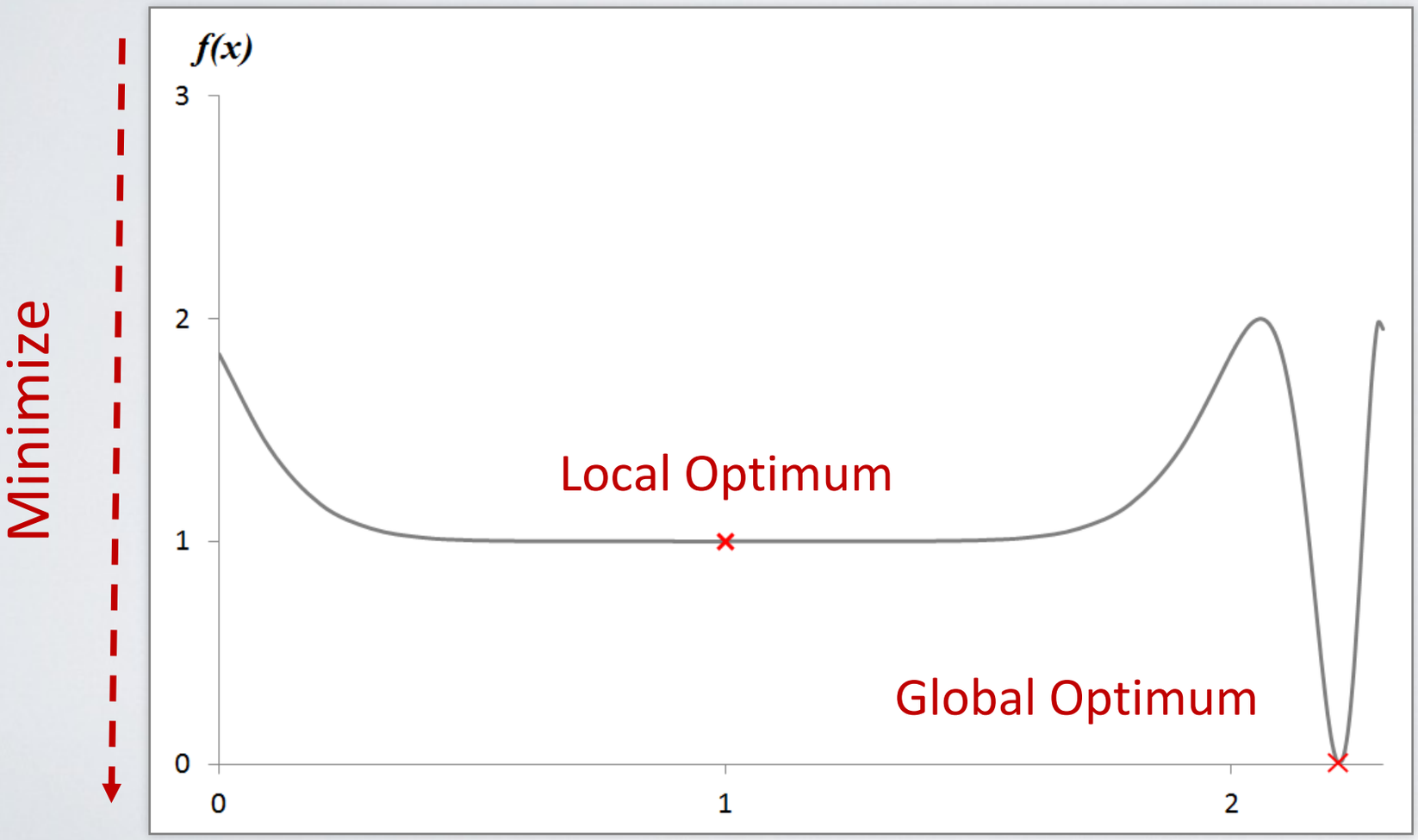
- After decades of research, there were no scalable, practical solutions for many automation problems.
- The community realized that many automation problems could be re-expressed as search problems.
- Stochastic optimization, Meta-heuristic search.
- Increasing realization that Search-Based Software Engineering has a much wider potential and is a research topic in itself.

Harman and Jones, “Search-Based Software Engineering”, 2001

Optimization

Find a value x^* which minimises (or maximises) the objective/fitness function f over a search space X:

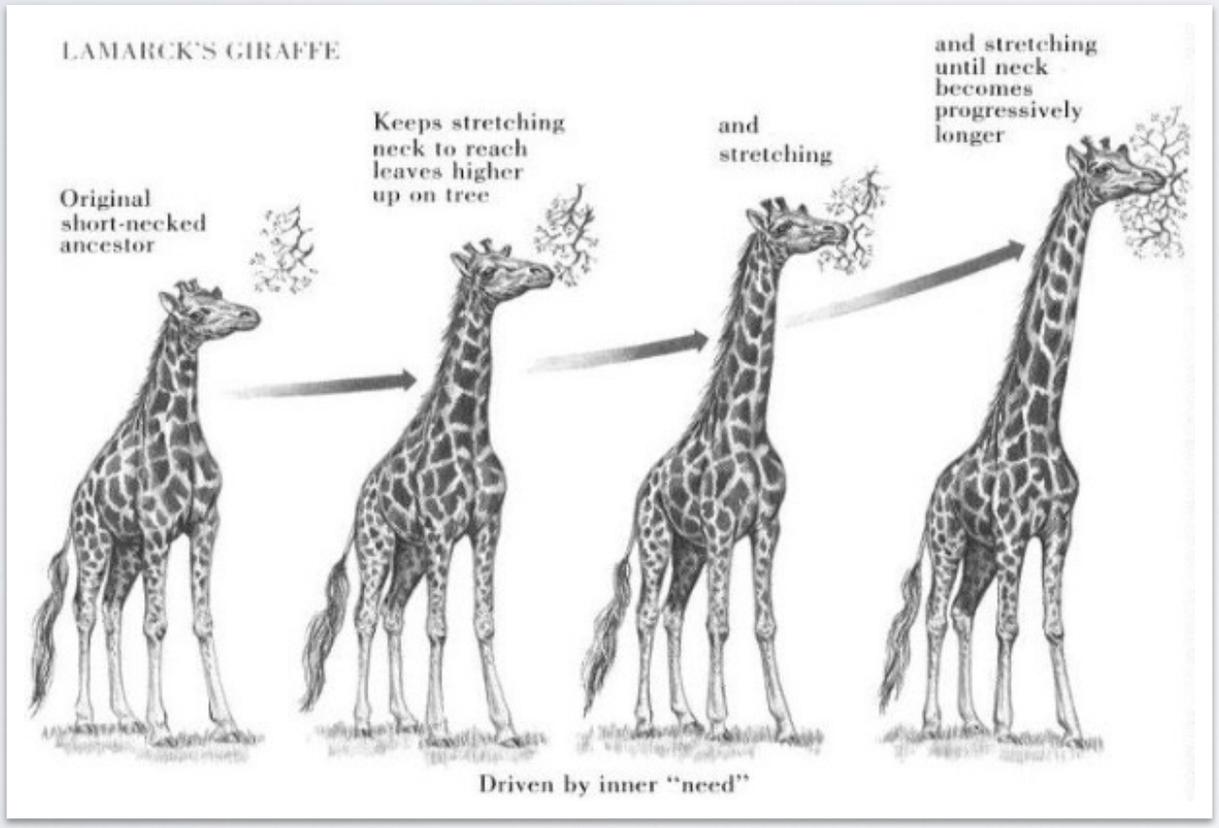
$$\forall x \in X : f(x^*) \leqslant f(x)$$



- No closed-form analytical description
- Black-box optimization
- Meta-heuristic search

Genetic Algorithms (GAs)

Genetic Algorithm: Population-based, search algorithm inspired be evolution theory

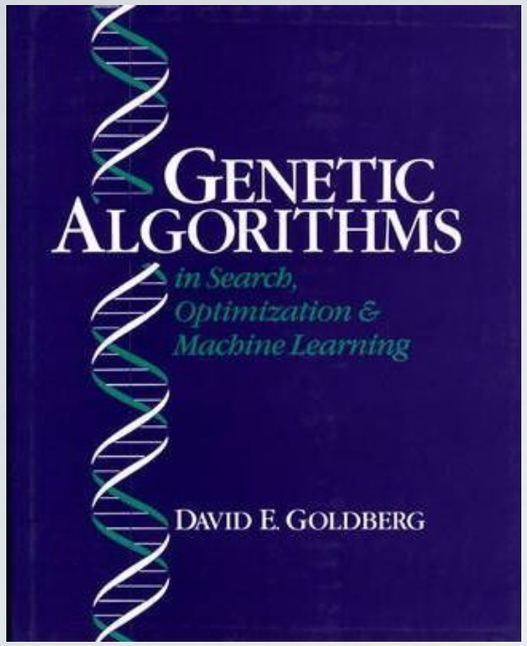


Natural selection: Individuals that best fit the natural environment survive

Reproduction: surviving individuals generate offsprings (next generation)

Mutation: offsprings inherits properties of their parents with some mutations

Iteration: generation after generation the new offspring fit better the environment than their parents

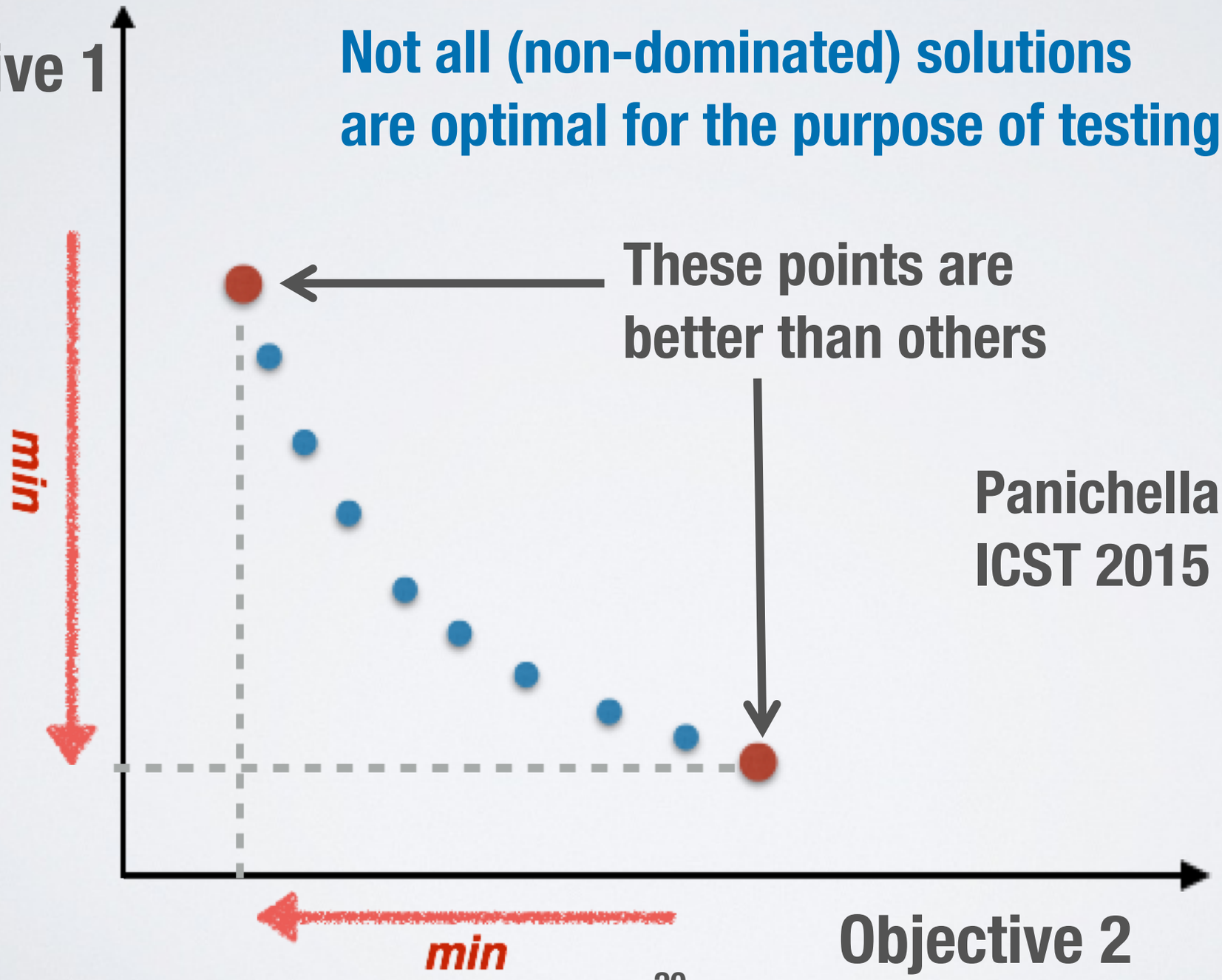


~1995

The first time I read about genetic algorithms, meta-heuristic search …

My first reaction:
“You must be kidding me”

MOSA: Many-Objective Search-based Test Generation



Not all (non-dominated) solutions are optimal for the purpose of testing

These points are better than others

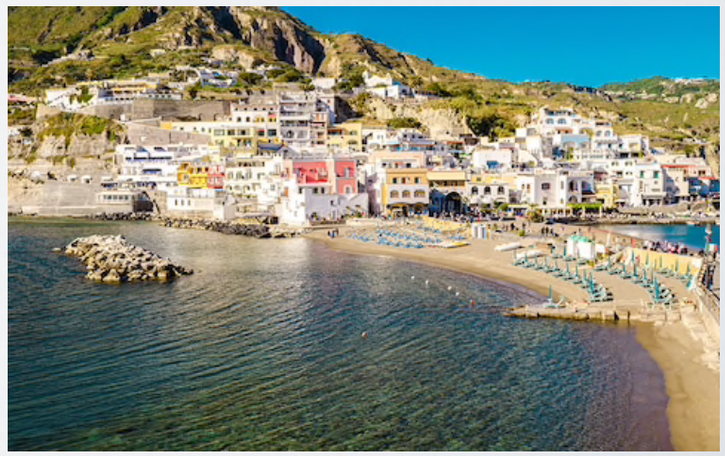
Panichella et. al.
ICST 2015

SBSE Applications

- **Many applications turned out to be promising**: Requirements prioritization, refactoring, test automation, program repair, etc.

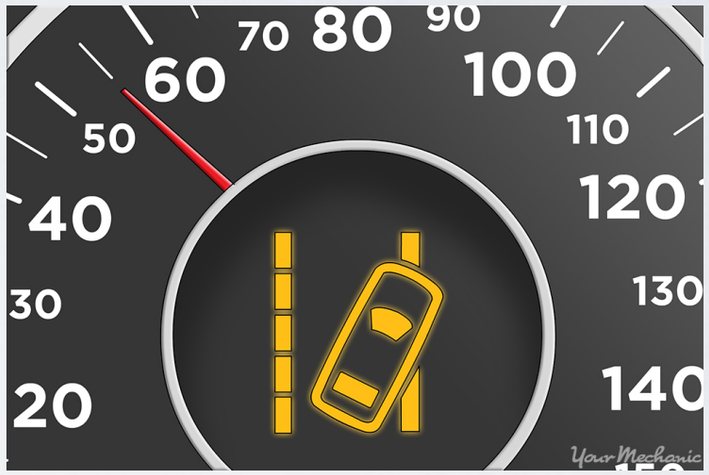
- **My first SBSE paper:** "Using genetic algorithms and coupling measures to devise optimal integration test orders." SEKE'02

- Many articles since then …



Advanced Driver Assistance Systems (ADAS)

Automated Emergency Braking (AEB)



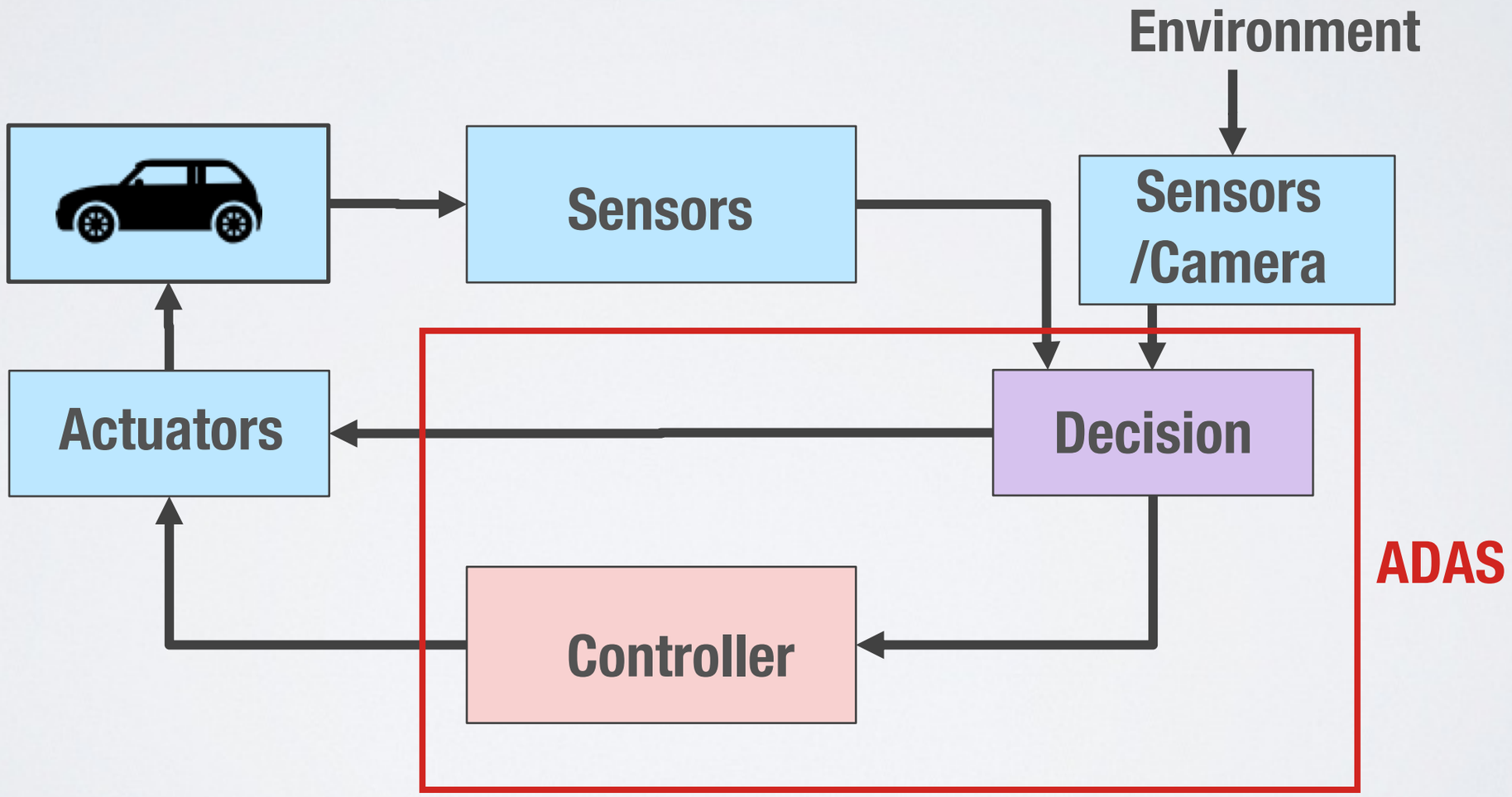
Lane Departure Warning (LDW)

Pedestrian Protection (PP)

Traffic Sign Recognition (TSR)

Advanced Driver Assistance Systems (ADAS)

Decisions are made over time based on sensor data



Automotive Environment

- **Highly varied environments**, e.g., road topology, weather, building and pedestrians …
- **Huge number of possible scenarios**, e.g., determined by trajectories of pedestrians and cars
- ADAS play an increasingly **critical role** in modern vehicles
- Systems must comply with **functional safety standards**, e.g., ISO 26262
- **A challenge for testing**

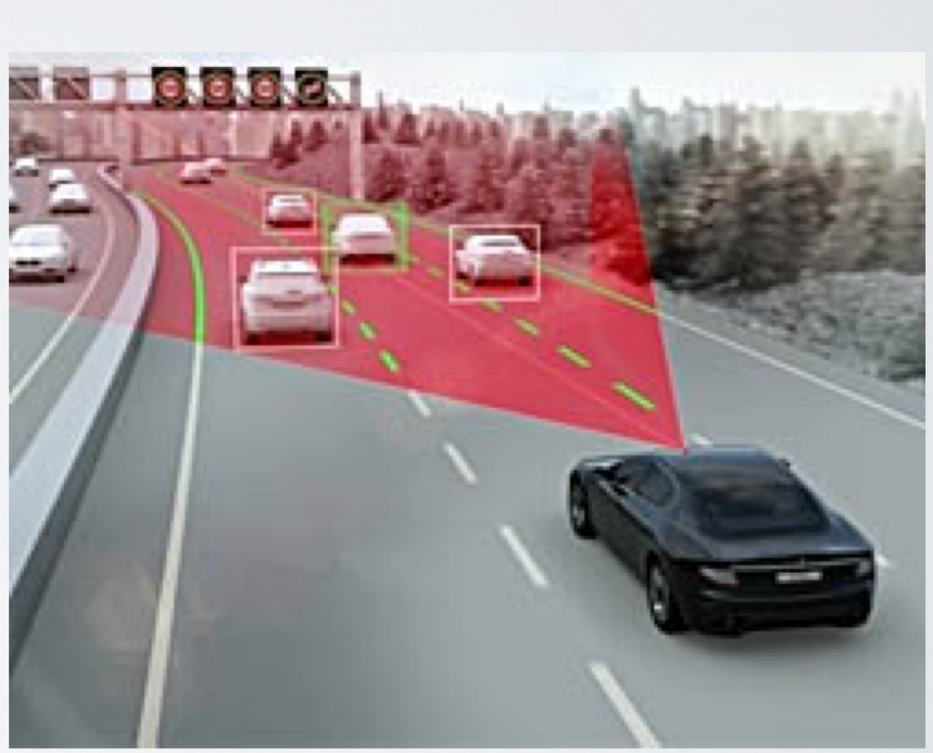
Testing ADAS

- Developing an automated testing technique for ADAS

- To help engineers efficiently and effectively explore the complex scenario space of ADAS

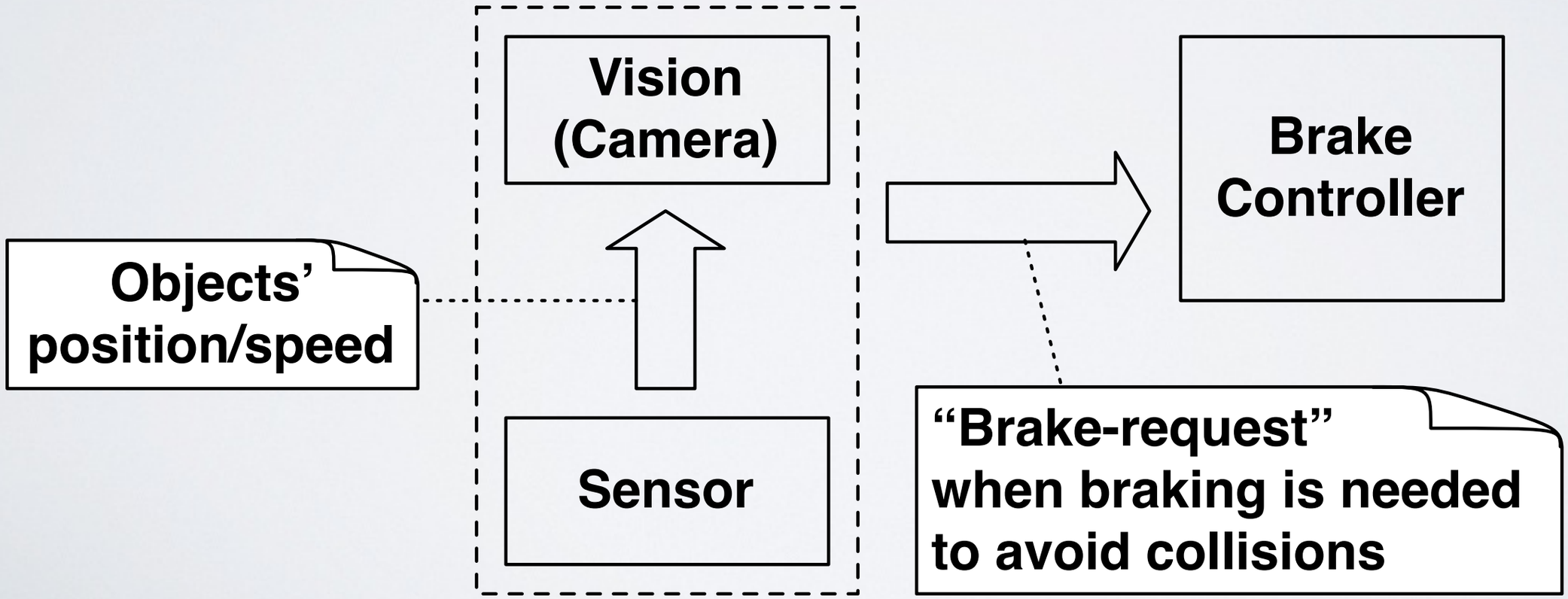
- To identify critical, failure-revealing test scenarios

- Characterization of input conditions that lead to most critical scenarios, e.g., safety violations



Automated Emergency Braking System (AEB)

Decision making



Example Critical Situation

- **“AEB properly detects a pedestrian in front of the car with a high degree of certainty and applies braking, but an accident still happens where the car hits the pedestrian with a relatively high speed”**

Testing ADAS

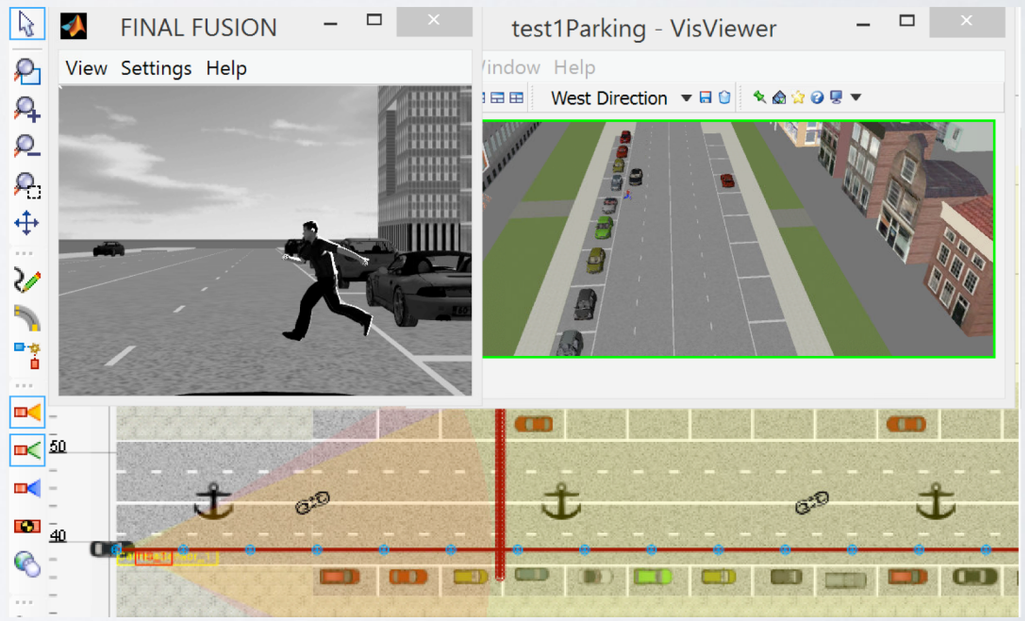
On-road testing

Time-consuming

Expensive

Unsafe

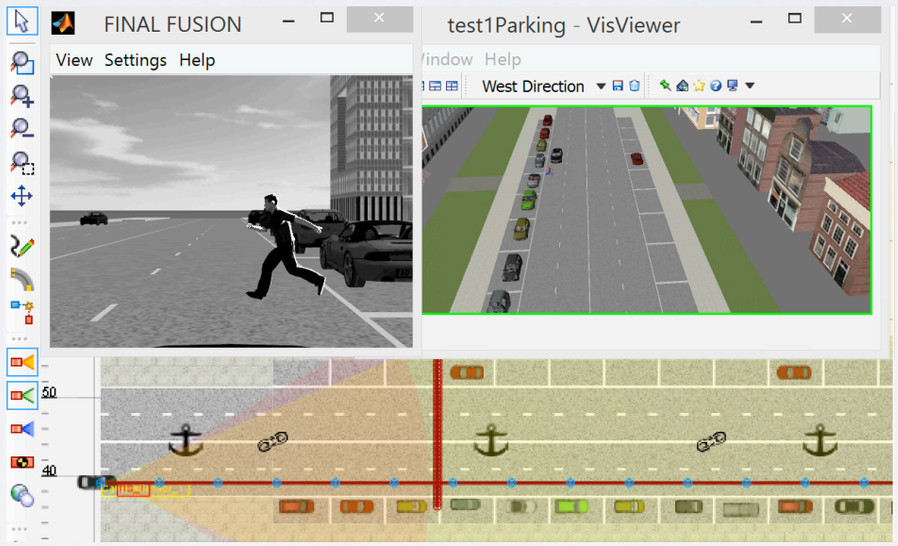
Simulation-based (model) testing



A simulator based on physical/mathematical models

Testing via Physics-based Simulation

Simulator (Matlab/Simulink)



- Physical plant (vehicle / sensors / actuators)
- Other cars
- Pedestrians
- Environment (weather / roads / traffic signs)

Model (Matlab/Simulink)

ADAS (SUT)

Test input

Test output

time-stamped output

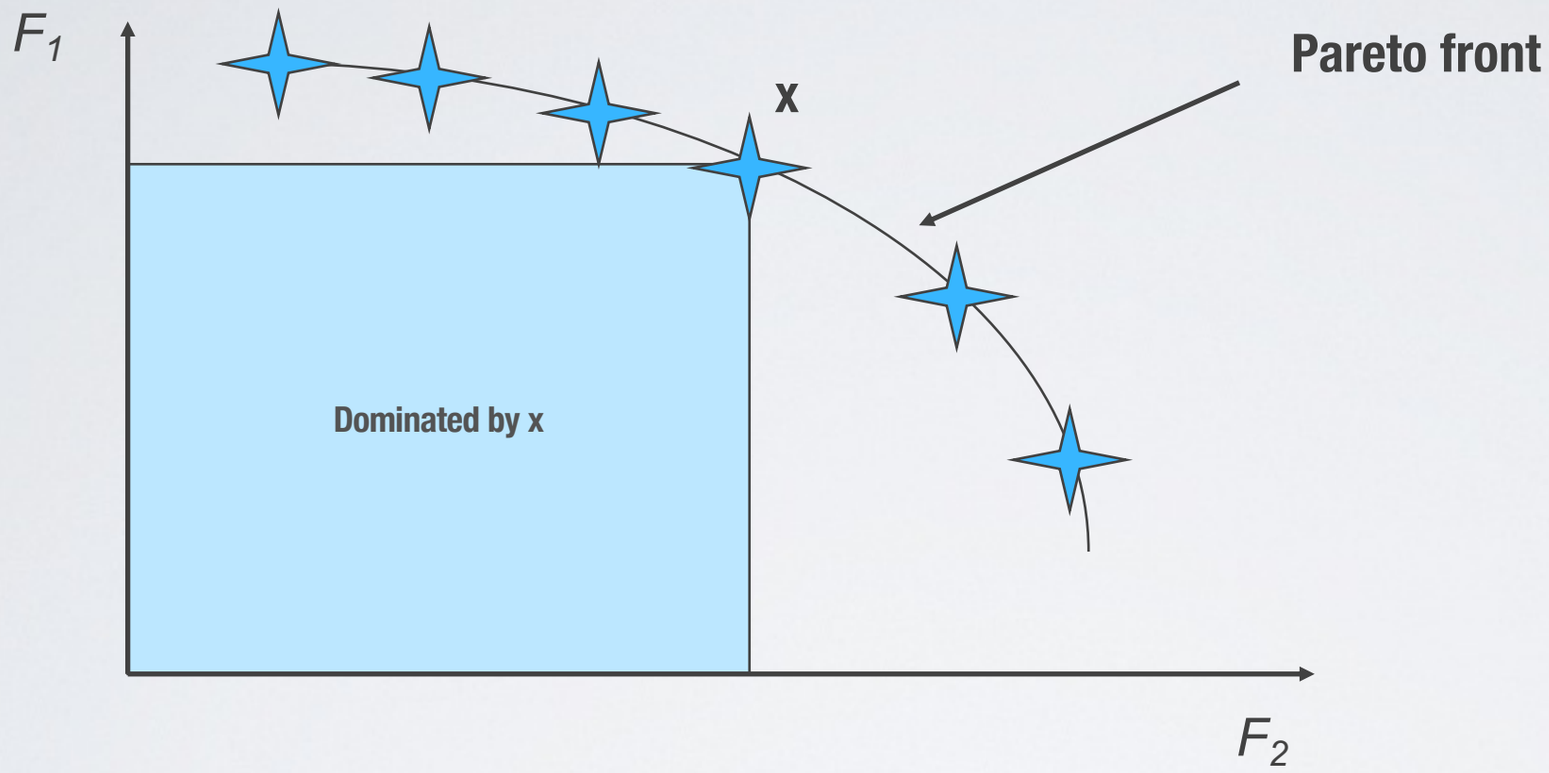
ADAS Testing Challenges

- Test input space is multidimensional, large, and complex.
- Explaining failures and fault localization are difficult.
- Execution of physics-based simulation models is computationally expensive.

Our Approach

- We use **multi-objective** search algorithm (NSGAII).
- **Objective Functions:**
  1. Minimum distance between the pedestrian and the field of view
  2. The car speed at the time of collision
  3. The probability that the object detected is a pedestrian
- We use **decision tree classification models** to speed up the search.
- Each search iteration **calls simulation** to compute objective functions.

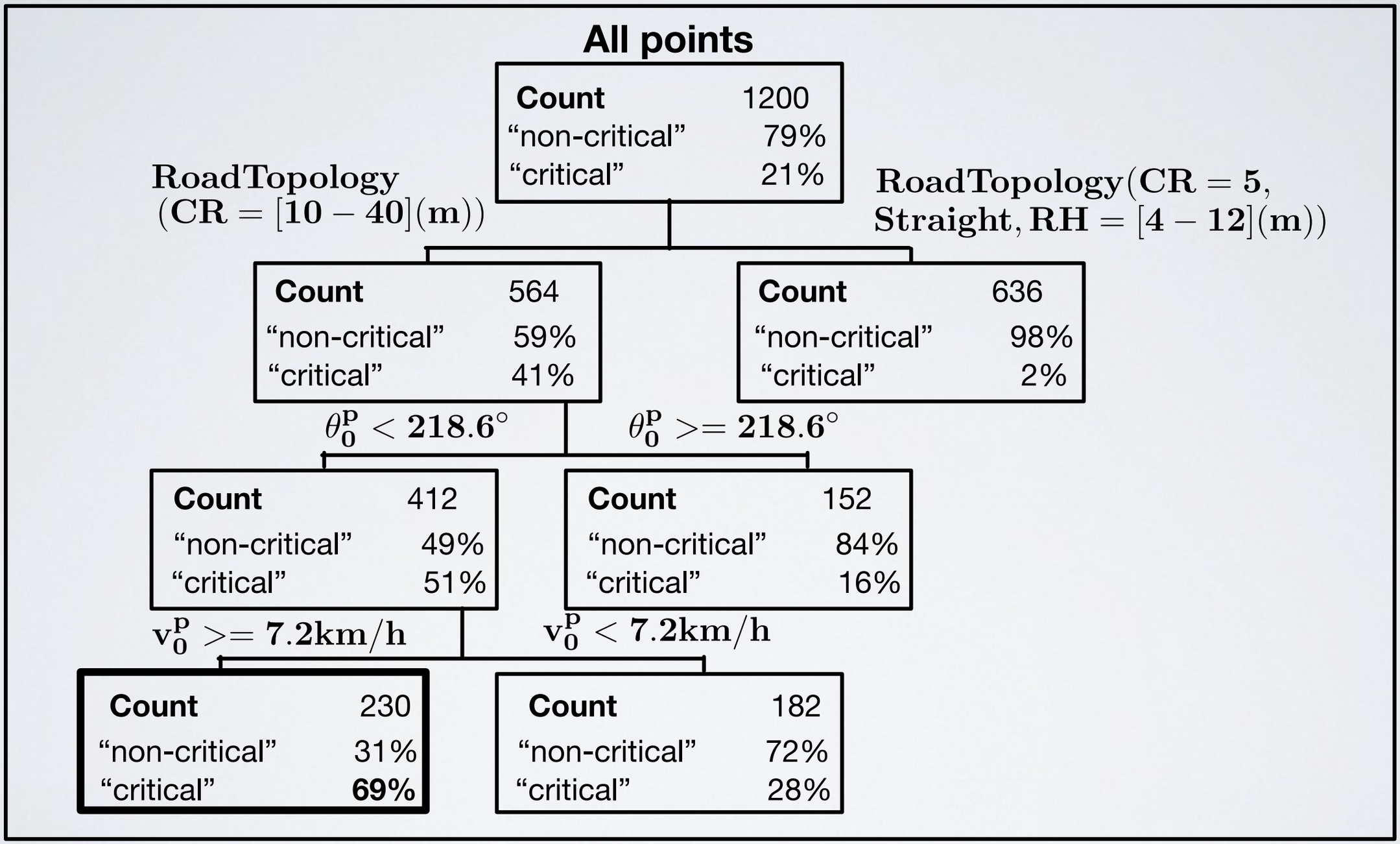
Multiple Objectives: Pareto Front



Individual A Pareto dominates individual B if A is at least as good as B in every objective and better than B in at least one objective.

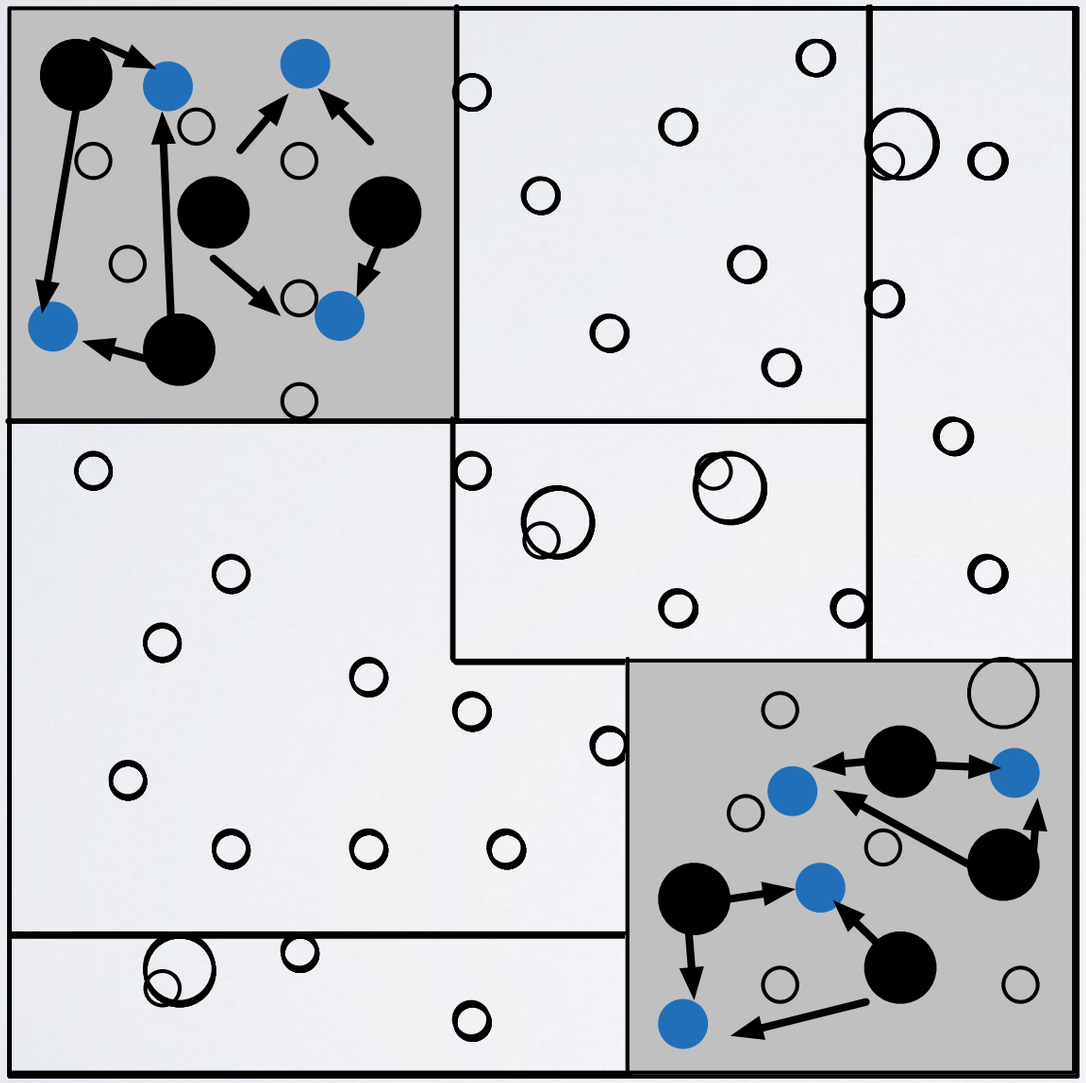
- A multi-objective optimization algorithm (e.g., NSGA II) must:
  - Guide the search towards the global Pareto-Optimal front.
  - Maintain solution diversity in the Pareto-Optimal front.

Decision Trees



Partition the input space into homogeneous regions

Genetic Evolution Guided by Classification



~~Initial input~~

~~Fitness computation~~

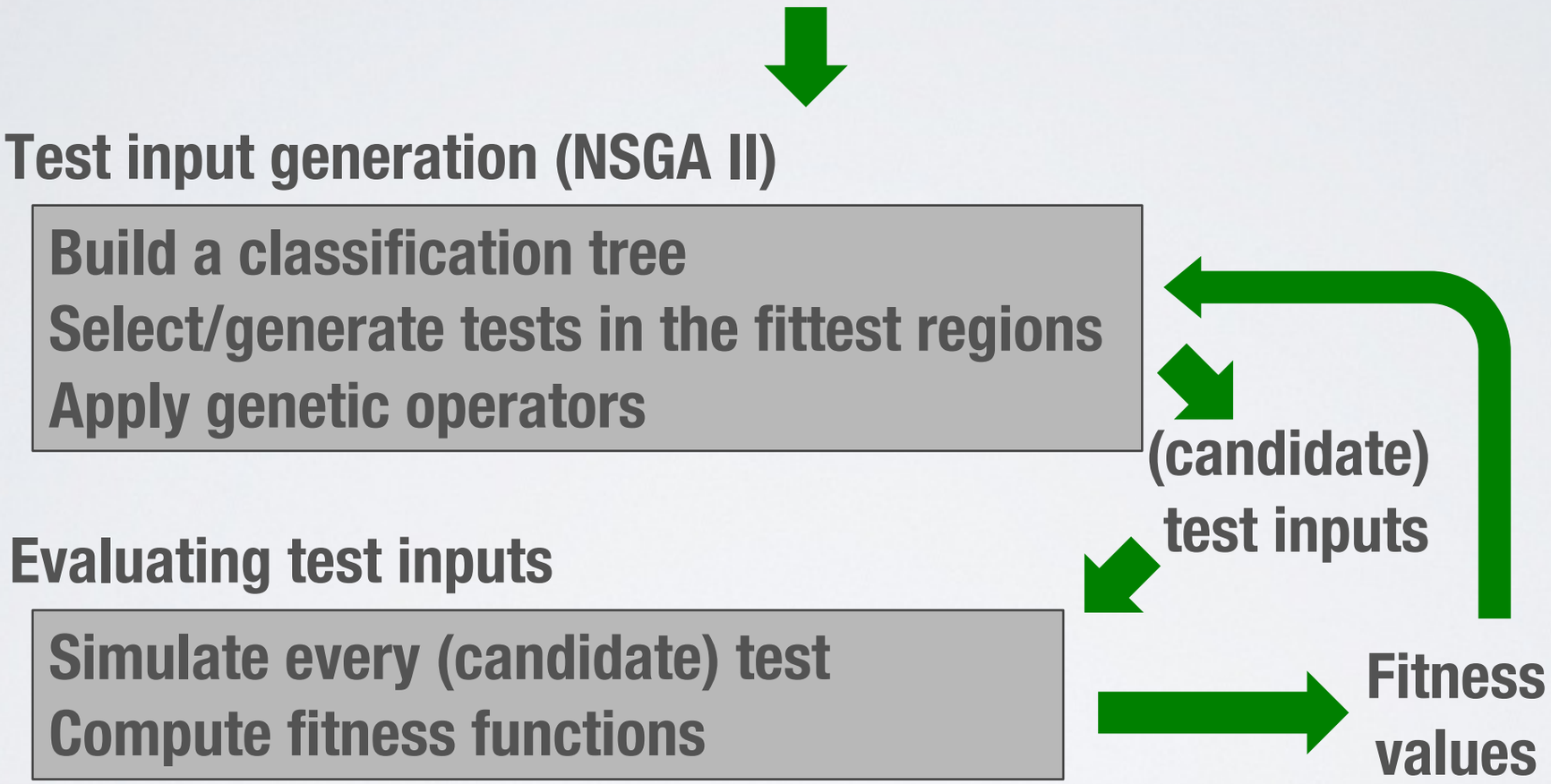
~~Classification~~

~~Selection~~

Breeding

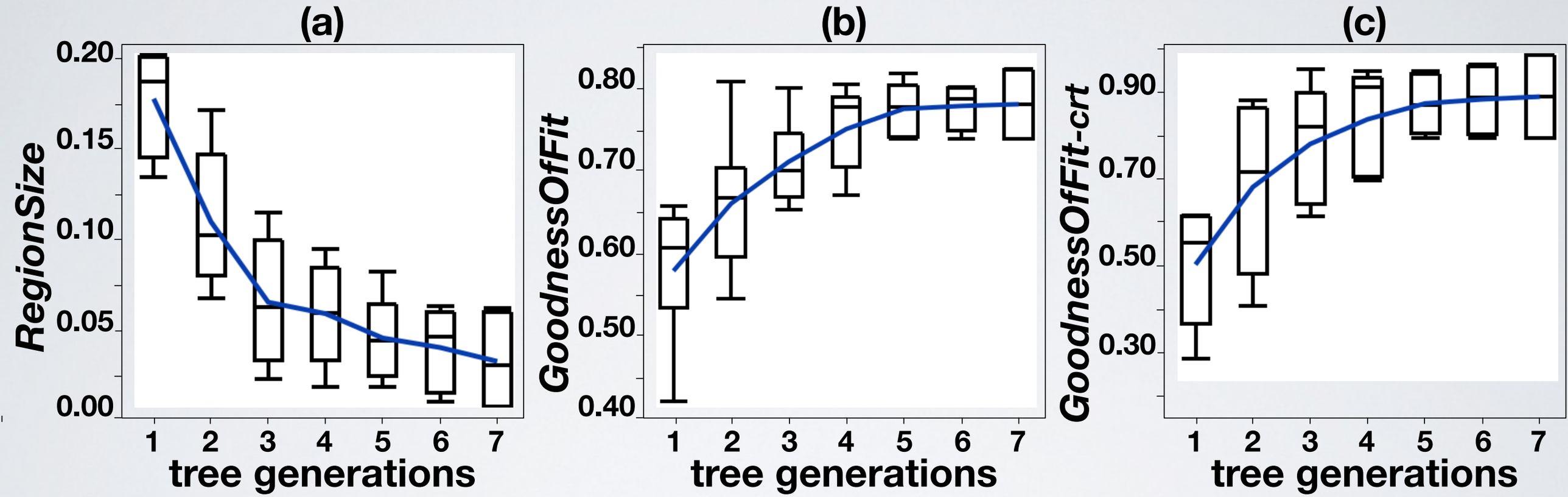
Search Guided by Classification

Input data ranges/dependencies + Simulator + Fitness functions



Test cases revealing worst case system behaviors +
A characterization of critical input regions

Generated Decision Trees



The generated critical regions consistently become smaller, more homogeneous and more precise over successive tree generations of NSGAII-DT

Engineers' feedback

- **The characterizations (decision trees) of the different critical regions can help with:**

(1) Debugging the system model

(2) Identifying possible hardware changes to increase ADAS safety

(3) Providing proper warnings to drivers

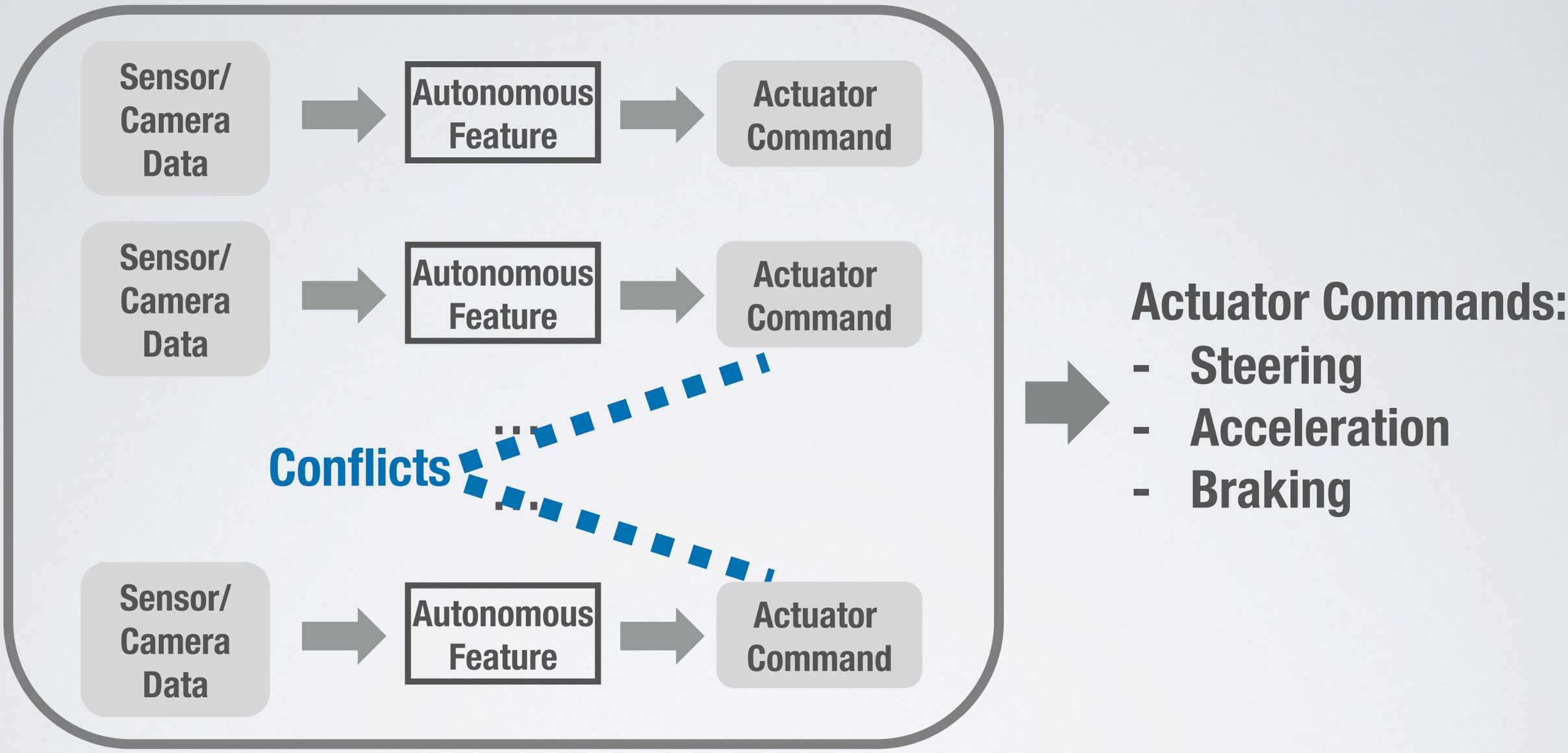
Integration of Autonomous Features

Automated Emergency Braking (AEB)

Traffic Sign Recognition (TSR)

Sensor/ Camera Data → Autonomous Feature → Actuator Command

- Steering
- Acceleration
- Braking



Undesired Feature Interactions

SBSE: Benefits

- Effective automation mechanism for a wide set of SE problems.
- Potentially high scalability: No exhaustive search.
- Can be effective under certain conditions: Search landscape, fitness computation, etc.
- Can be effectively parallelized.

SBSE: Challenges

- Performance can be an issue in large, high-dimensionality search spaces
- Computationally expensive fitness functions, e.g., simulations.
- Validation is experimental and computationally expensive.
- Many problems require dedicated, tailored search algorithms, e.g., many-objective search in testing.
- Devising the right search algorithm for a given problem requires expertise and experiments.
- Devising the right fitness functions is often a trade-off and is a trial and error process.

> ~2004

Increasingly Powerful Natural Language Processing

NLP

- Process and analyze large amounts of natural language data.
- Rule-based versus statistical NLP (based on machine learning).
- Preprocessing: Tokenizer, sentence splitter, POS tagger.
- Parsing: Constituency, dependency, semantic.
- NLP has made huge leaps forward.

Arnaoudova et al., “The Use of Text Retrieval and Natural Language Processing in Software Engineering”, ICSE’15

Why NLP?

- Significant documentation of many kinds in natural language …
- Example: Requirements
- are prominent throughout industry sectors, even safety-critical ones,
- are not fading away any time soon.

Why NLP?

- Check **well-formedness** of NL artifacts
- Extract useful **information** from NL artifacts
- Check **consistency and completeness** of NL artifacts
- Understand **relationship and dependencies** between NL artifacts

Experiences in Requirements Engineering

- Conformance of requirements with **templates**. (Arora et al.)
- **Impact analysis** of requirements changes (Arora et al., Nejati et al.)
- Identification and **demarcation** of requirements in large documents. (Abualhaija et al.)
- Requirements-driven system **testing**. (Wang et al.)

Traceability

- In many sectors, traceability between requirements and test cases is **required by standards**, customers, certifiers ...
- Requirements **change**, and therefore test cases as well.
- Huge traceability **matrices** are built and maintained manually.
- Academic work on **automatically matching** requirements and test cases is not close to be sufficiently accurate or practical.

Context

Automotive Embedded Systems

- Small but safety critical systems
- Traceability from requirements to system test cases (ISO 26262)
- Requirements act as a contract
- Many requirements changes, leading to negotiations

Problem

Automatically verify the compliance of software systems with their functional requirements in a cost-effective way

Objective

Support the Generation of System Test Cases from Requirements in Natural Language

Traceability is a by-product

Problem

Textual descriptions are often ambiguous, Incomplete, and not analyzable automatically

Compromise?

Stick to natural language but …

Restrict its usage so as to make it amenable to NLP for system testing purposes

Find the right balance

Restricted Use Case Specifications

- Use Case Modeling is widely used
- Restricted Use Case Modeling (RUCM)
- Experiments shows it yields better use cases
- Compliance is tool-supported (NLP)
- More analyzable with NLP

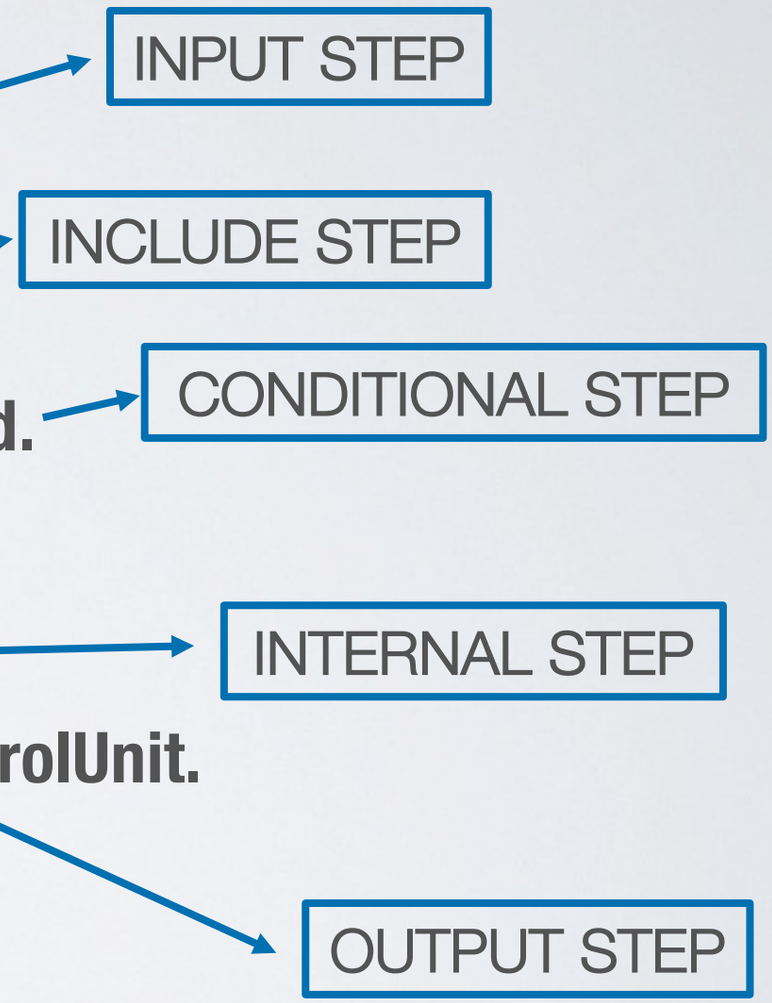
Yue et al. ACM TOSEM, 2013

RUCM Specifications Example

Precondition: The system has been initialized

Basic Flow

1. The SeatSensor **SENDS** the weight **TO** the system.
2. **INCLUDE USE CASE** Self Diagnosis.
3. The system **VALIDATES THAT** no error has been detected.
4. The system **VALIDATES THAT** the weight is above 20 Kg.
5. The system sets the occupancy status to adult.
6. The system **SENDS** the occupancy status **TO** AirbagControlUnit.



RUCM Specifications Example

Precondition: The system has been initialized

Basic Flow

1. The SeatSensor **SENDS** the weight **TO** the system.
2. **INCLUDE USE CASE** Self Diagnosis.
3. The system **VALIDATES THAT** no error has been detected.
4. The system **VALIDATES THAT** the weight is above 20 Kg.
5. The system sets the occupancy status to adult.
6. The system **SENDS** the occupancy status **TO** AirbagControlUnit.

INPUT STEP

INCLUDE STEP

CONDITIONAL STEP

INTERNAL STEP

OUTPUT STEP

Alternative Flow

RFS 4.

1. **IF** the weight is above 1 Kg **THEN**
2. The system sets the occupancy status to child.
3. **…**
4. **RESUME STEP 6.**

NLP for information extraction

Precondition: The system has been initialized

Basic Flow

1. The SeatSensor **SENDS** the weight **TO** the system.
2. **INCLUDE USE CASE** Self Diagnosis.
3. The system **VALIDATES THAT** no error has been detected.
4. The system **VALIDATES THAT** the weight is above 20 Kg.
5. The system sets the occupancy status to adult.
6. The system **SENDS** the occupancy status **TO** AirbagControlUnit.

DOMAIN ENTITY

Alternative Flow

RFS 4.

1. **IF** the weight is above 1 Kg **THEN**
2. The system sets the occupancy status to child.
3. …
4. **RESUME STEP 6.**

NLP for information extraction

Precondition: The system has been initialized

Basic Flow

1. The SeatSensor **SENDS** the weight **TO** the system.
2. **INCLUDE USE CASE** Self Diagnosis **CONSTRAINT**
3. The system **VALIDATES THAT** no error has been detected.
4. The system **VALIDATES THAT** the weight is above 20 Kg.
5. The system sets the occupancy status to adult.
6. The system **SENDS** the occupancy status **TO** AirbagControlUnit.

Alternative Flow

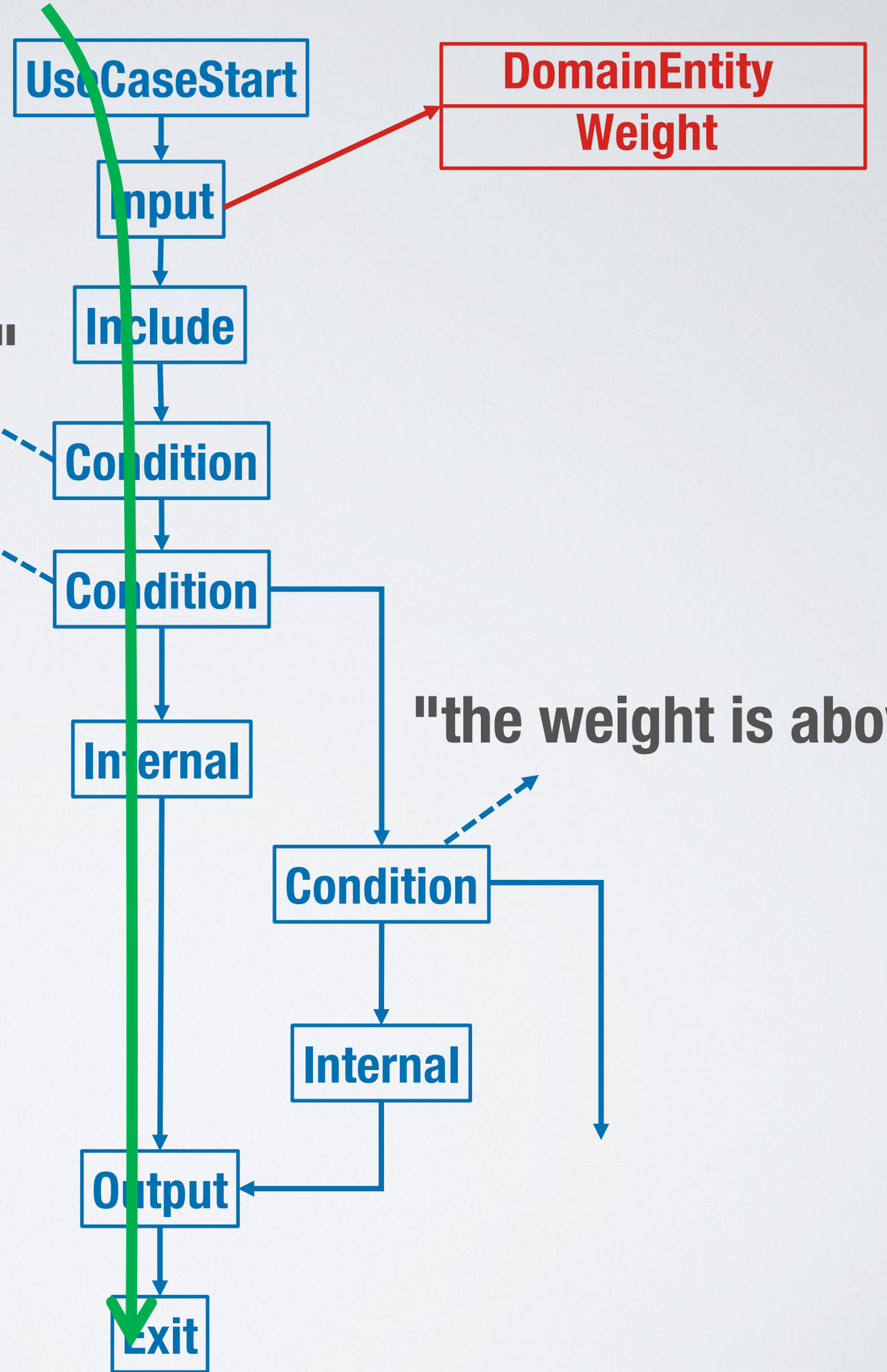
RFS 4.

1. **IF** the weight is above 1 Kg **THEN**
2. The system sets the occupancy status to child.
3. …
4. **RESUME STEP 6.**

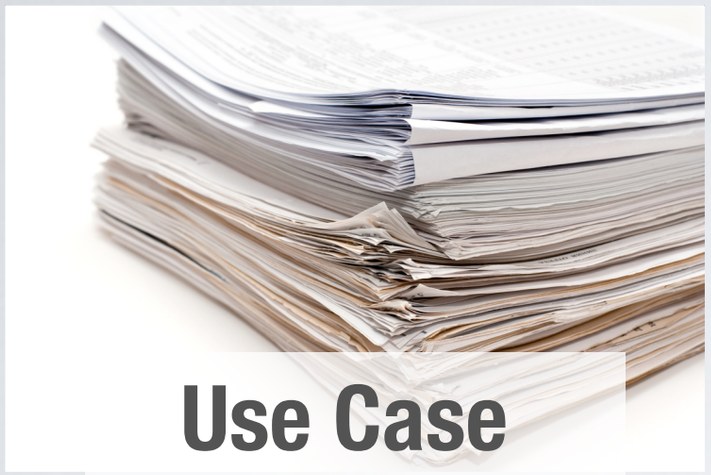
Test Model

"no error has been detected"

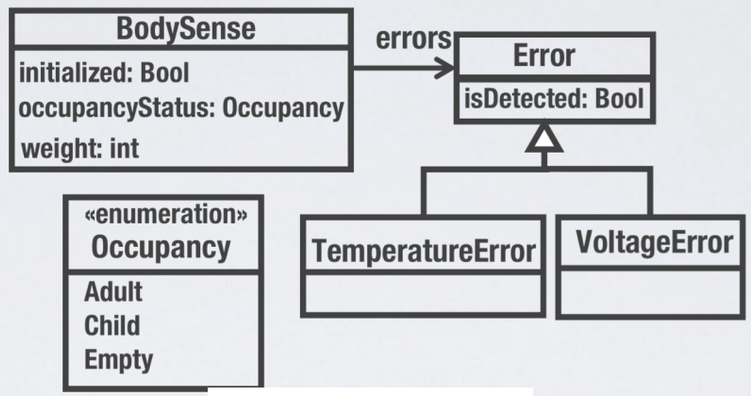
"the weight is above 20 Kg"



Automated Generation of System Test Cases for Embedded Systems from Requirements in NL



Use Case Specifications



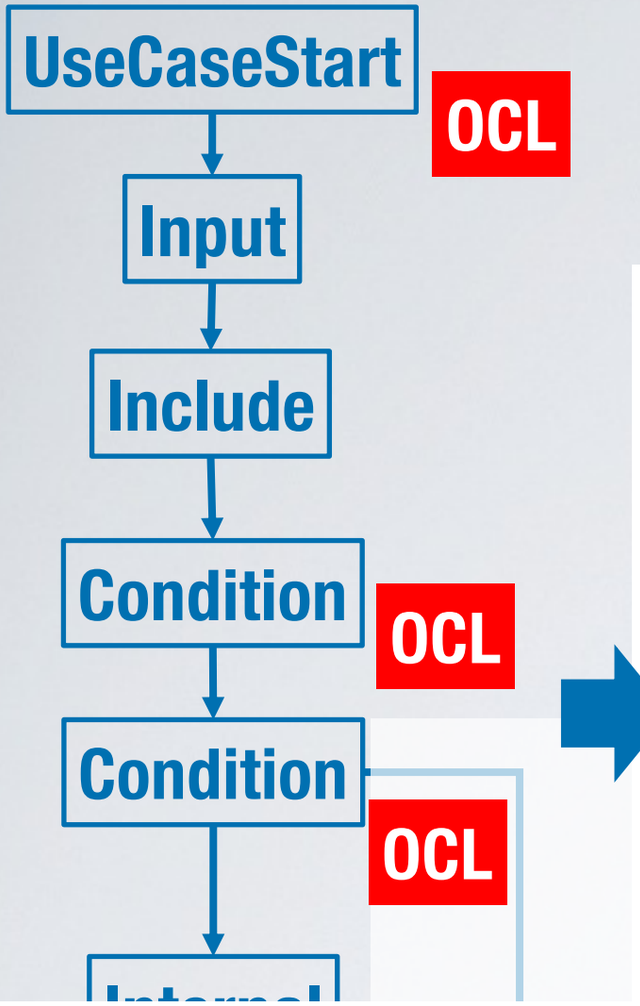
Domain Model

```
Error.allInstances()
->forAll( i | i.isDetected = false)
```

Constraints capturing the meaning of conditions

(Wang et al.) **UMTG** https://sntsvv.github.io/UMTG/

Executable Test Cases



Precondition: The system has been initialized.

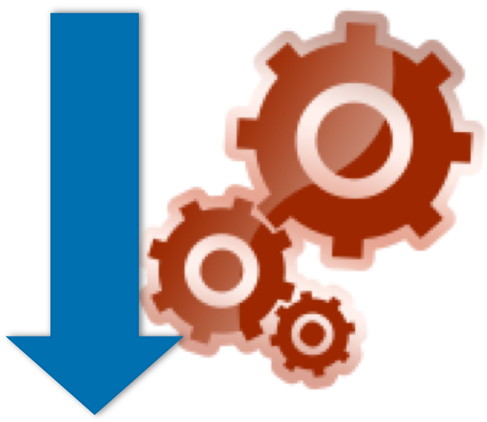
The SeatSensor **SENDS** the weight **TO** the system.

Path condition:

```
System.allInstances()->forAll( s | s.initialized = true )
AND System.allInstances()->forAll( s | s.initialized = true )
AND Error.allInstances()->forAll( e | e.isDetected = false)
AND System.allInstances()
      ->forAll( s | s.occupancyStatus = Occupancy::Adult )
```

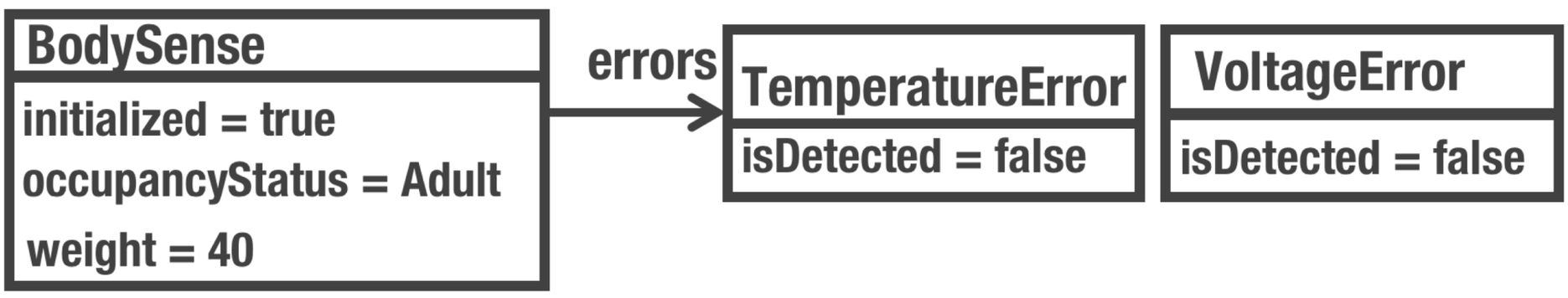
https://sites.google.com/view/hybridoclsolver/

Soltana et al.



Constraint Solving (PLEDGE)

Test inputs:

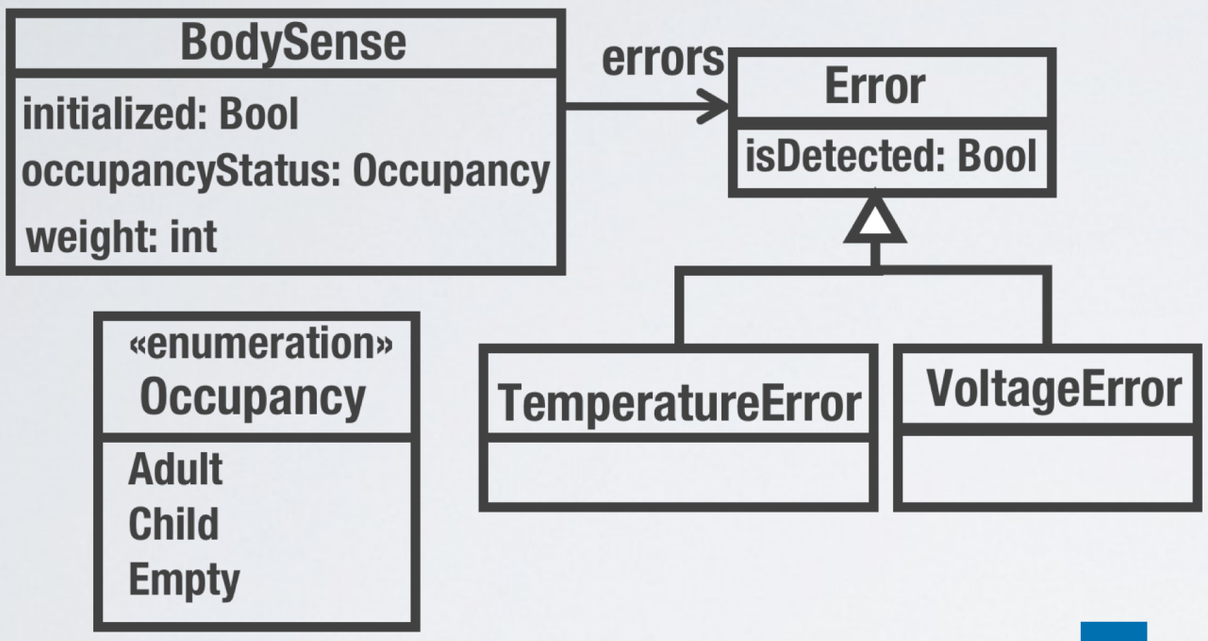


Challenge

Typically dozens of constraints

Engineers need help in defining constraints

Automatically Derive Formal Constraints



“The system VALIDATES THAT no error has been detected.”

(Wang et al.) → **OCLgen** — **Based on NLP**

→ **Error.allInstances()->forAll(i | i.isDetected = false)**

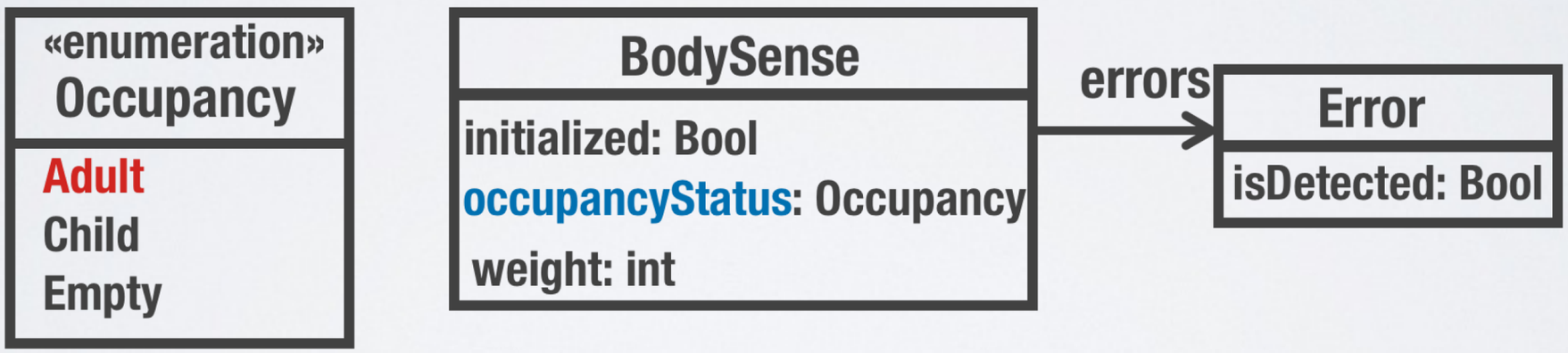
OCLgen Solution

1. determine the role of words in a sentence (SRL)

actor affected by the verb — final state

"The system sets the occupancy status to adult."

2. match words in the sentence with concepts in the domain model



3. generate the OCL constraint using a verb-specific transformation rule

```
BodySense.allInstances()
    ->forAll( i | i.occupancyStatus = Occupancy::Adult)
```

Semantic Role Labeling (SRL)

“The system sets the occupancy status to adult.”

A0	verb	A1	A2

A0: actor that performs an activity

A1: actor that is affected by the activity described in a sentence

A2: final state

Semantic Role Labeling (SRL)

Determine the role of words independently from passive voices, transitive/intransitive forms

"The system resets the counter."

A0 verb A1

"The counter was reset by the system."

A1 verb A0

"The counter has been reset."

A1 verb

Enable pattern-based text transformations:

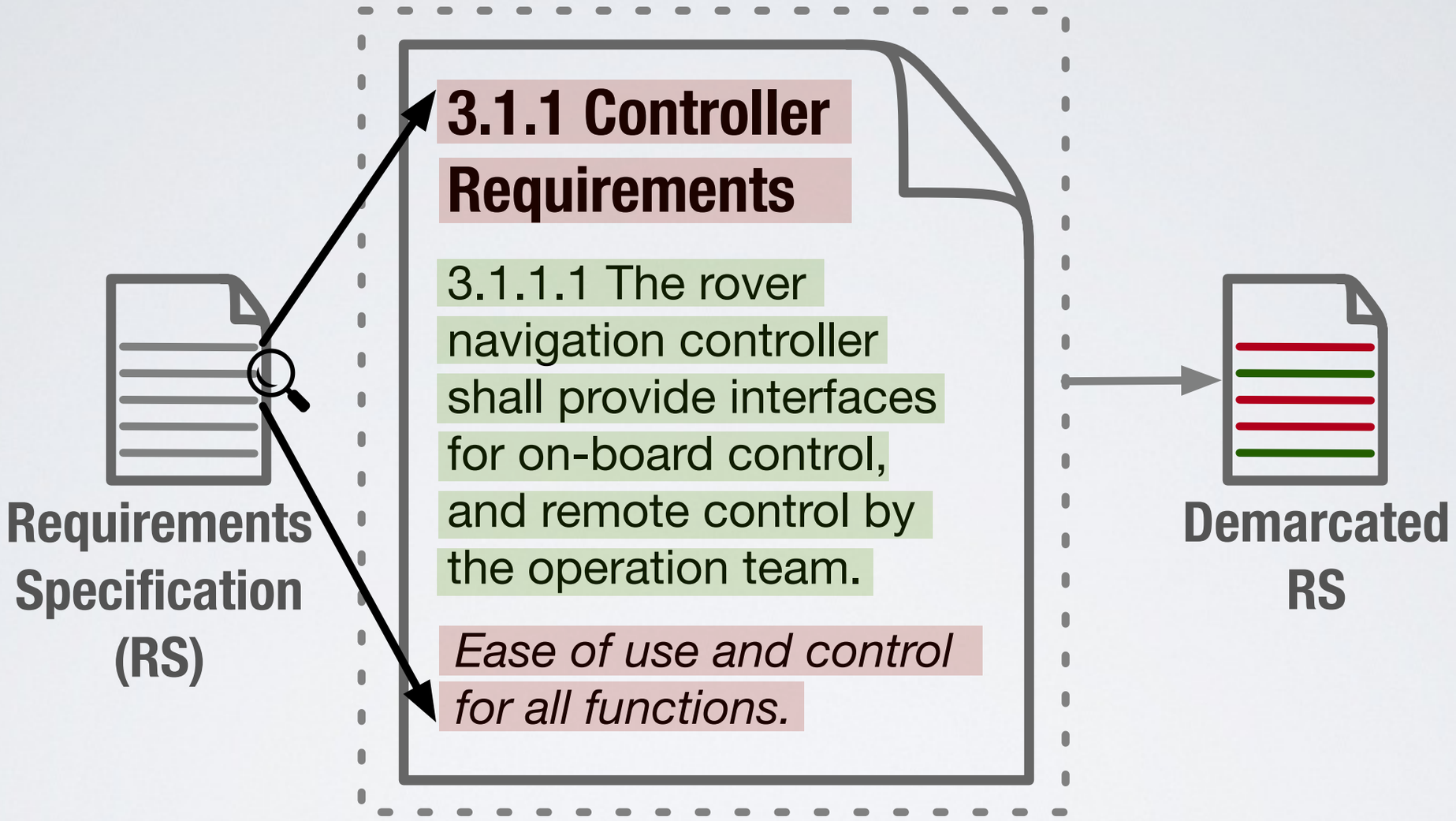
```
System.allInstances()->forAll( i | i.counter = 0 )
                                    <A1> = 0
```

NLP in SE: Summary

- **Increasingly powerful, many applications**
- **Wide variation** across domain practices and documents.
- Inherent **ambiguity and inconsistency** of natural language.
- Relevant data is usually **spread across artifacts.**
- Designing the **right NLP pipeline** in not easy.
- NLP components are **not fully accurate.**
- **Human** in the loop.

Combining Strengths

Requirements Demarcation



Requirements vs. information about project scope, comments, rationale …

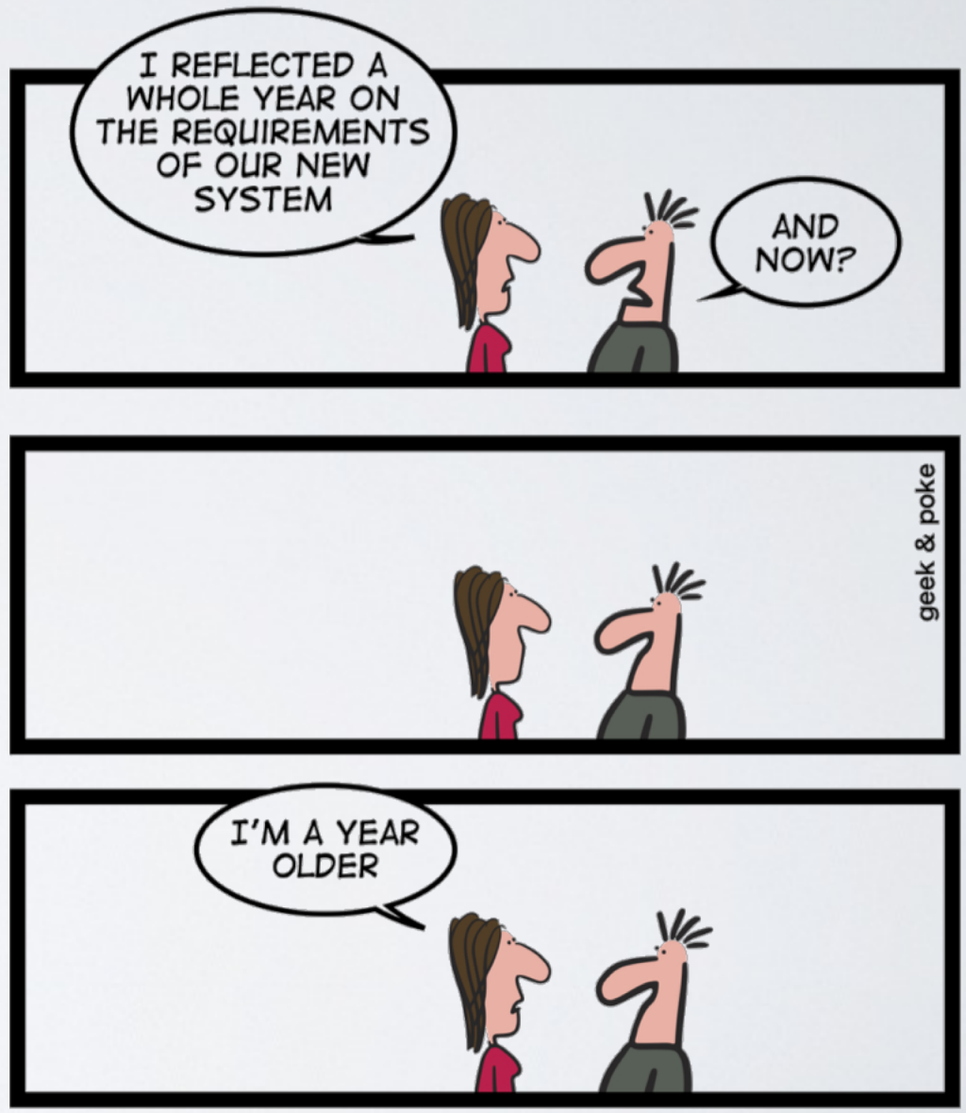
Why is demarcating requirements important?

Only requirements may be

- considered as **contractually binding**
- subject to **automated quality assurance tasks** like ambiguity detection, change impact analysis, etc.

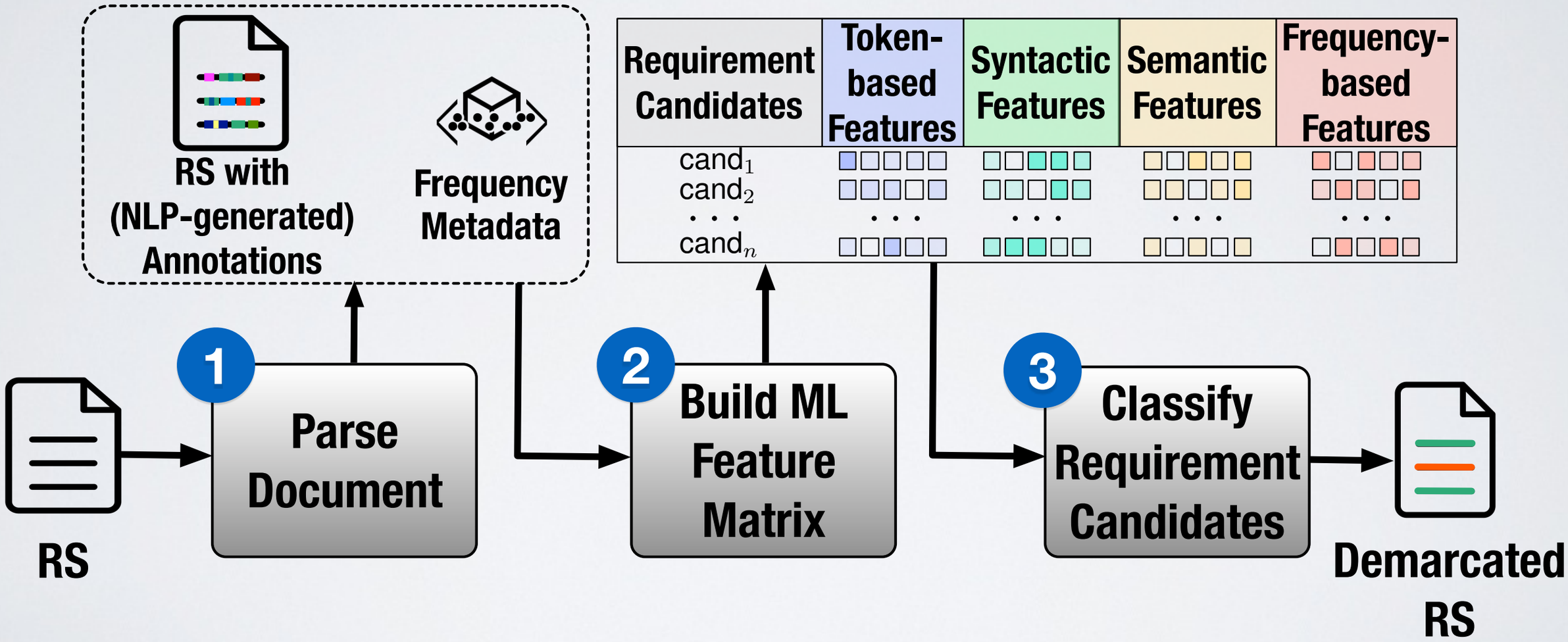
Requirements Demarcation in Practice

- **Requirements specifications are large** with 100s or 1000s of statements
- Requirements are **not clearly separated** from non-requirements
- Manual requirements demarcation is **time- and effort-consuming**



Overview

Natural language processing (NLP) and machine learning (ML)



Abualhaija et al., RE 2019

Other Projects

- **Security testing** of financial applications: Search+ML+String solver (Appelt et al., Jan et al.)
- **Satellite in-orbit testing** (HiL testing): Search+Simulation (Shin et al.)
- Solving complex **system-level constraints** for system testing: Search+SMT (Soltana et al.)
- **Recommender systems for security controls** in banking applications: NLP+ML (Bettaieb et al.)
- **Testing and debugging Simulink models**: Search+ML (Matinnejad et al., Liu et al.)

Conclusions

Metaheuristic Search

- Most SE automation problems can be re-expressed into search (stochastic optimization) problems.
- Versatile. Parallelizable. Many algorithms matching various situations.
- But not that easy to use and deploy for practitioners.
- **Issues:** Scalability issues, choice of search algorithm, massive experiments for validation

Machine Learning

- Helps guide search by learning interesting parts of the search space on the go.
- Complements NLP when no deterministic rules are possible to extract information and make decisions based on text.
- Helps understand actionable patterns in SE data, e.g., interpret test results and support debugging by understanding failure conditions.

NLP

- **NLP helps exploit information in natural language and pushes automation further.**
- **Extract useful features for ML and computing fitness functions.**

The Road Ahead

- AI plays a key role in automating many software engineering tasks and helping decision support
- Real solutions usually involve several techniques, combined to achieve the best trade-offs.
- Real solutions strike a balance in terms of scalability, practicality, applicability, and optimal results.
- Research in this field cannot be oblivious to context (e.g., domain): Working assumptions, desirable trade-offs …
- We need more multi-disciplinary research driven by (well-defined) problems in context.

Academia-Industry Gap

- Realism: Many academic papers address problems that are **unlikely to exist as defined**, e.g., working assumptions.
- Lack of awareness: On the other hand, many industrial problems (often long standing) are **insufficiently addressed by research**.
- **Context factors and working assumptions** have a huge impact on software engineering solutions.
- **Scalability and practicality** aspects are largely ignored by research – they are often considered as an afterthought

Articles

Twin-Win Model: A human-centered approach to research success

Ben Shneiderman[a,1]

[a]Department of Computer Science & Human-Computer Interaction Lab, University of Maryland, College Park, MD 20814

Edited by William Rouse, Stevens Institute of Technology, Hoboken, NJ, and accepted by Editorial Board Member Pablo G. Debenedetti May 4, 2018 (received for review February 15, 2018)

A 70-year-old simmering debate has erupted into vigorous battles over the most effective ways to conduct research. Well-established beliefs are being forcefully challenged by advocates of new research models. While there can be no final resolution to this battle, this paper offers the Twin-Win Model to guide teams of researchers, academic leaders, business managers, and government funding policymakers. The Twin-Win Model favors a problem-oriented approach to research, which encourages formation of teams to pursue the dual goals of breakthrough theories in published papers and validated solutions that are ready for widespread dissemination. The raised expectations of simultaneously pursuing foundational discoveries and powerful innovations are a step beyond traditional approaches that advocate basic research first. Evidence from citation analysis and researcher interviews suggests that simultaneous pursuit of both goals raises the chance of twin-win success.

research model | basic research | applied research | Twin-Win Model | human-centered research

The usually quiet world of academic research is being awakened by explosive battles over how to do research (1–4). The traditional linear model of research argued for curiosity-driven basic research in laboratories to acquire new knowledge. This may have been productive in the knowledge-poor early days of discovery, but now, in our knowledge-rich, information-overloaded world, new models are needed. Since collecting new knowledge has become so easy, researchers need to consider which forms of new knowledge would be most beneficial. Collecting the length of every rat's tail or the number of characters in every tweet would add to the store of knowledge. However, it seems clear that collecting the location of every rat to understand the spread of disease or the time stamp of every tweet to understand sleep patterns in different cities would be more helpful in raising further questions and useful in recommending constructive actions.

In short, some knowledge is likely to be more useful than others, because the knowledge relates to meaningful problems and may suggest constructive actions. Knowledge is tied to meaningful problems by way of a causal theory that permits intervention so as to contribute to improvements in human life or environmental preservation. Therefore, my claim is that research can become more productive if the pursuit of new knowledge is tied to actionable insights that can lead to societal benefits and sustainable conservation.

Leading organizations have identified key challenges, such as the 17 Sustainable Development Goals of the United Nations or the 14 Grand Challenges of the US National Academy of Engineering. Common challenges include healthcare delivery, clean air and water, smart cities, improved education, and energy availability.

While most academics support the idea of responding to these challenges by collecting new knowledge that is tied to actionable insights, too often researchers fail to structure their research plans in ways that are likely to lead to the dual successes of new knowledge and societal benefits. The Twin-Win Model (Fig. 1) favors a problem-oriented approach to research, which encourages formation of teams between academics and professionals to pursue the dual goals of breakthrough theories in published papers and validated solutions that are ready for widespread dissemination.

Background

The idea of bringing academic researchers in closer contact with professionals who face authentic problems has long been discussed as a way to achieve higher societal benefits. The famed American poet and philosopher Ralph Waldo Emerson spoke in 1837 about academics working more closely with farmers, business people, and government. Emerson called for academics to engage in the real world: "Action . . . is essential . . . Without it, thought can never ripen into truth." That encouragement remains valid today.

More than a century later, Vannevar Bush's (5) 1945 manifesto *Science: The Endless Frontier, a Report to the President on a Program for Postwar Scientific Research* sought to separate academic work from practical problems. He argued for a linear model, suggesting that basic research came first, which led to applied research and then commercial development. The linear model was vigorously opposed by Tom Allen (6) in the 1970s, Deborah Shapley and Rustum Roy (7) in the 1980s, and many others. An important contribution was Donald Stokes' (8) 1997 book *Pasteur's Quadrant: Basic Science and Technological Innovation*, which proposed a fresh strategy: "use-inspired basic research." His reference to Pasteur reminded readers about Pasteur's work on the problems of vintners and dairy farmers, which produced the twin-win of solutions to their problems and the germ theory of disease. Lewis Branscomb's (9) 2007 essay supported the idea that creativity and utility (basic and applied) research were happy partners. Steven Chu, Nobel Prize winner in physics and US Secretary of Energy, reinforced the need for a shift in research: "We seek solutions. We don't seek—dare I say this—just scientific papers anymore."

In the past few years, *The New ABCs of Research: Achieving Breakthrough Collaborations* (10), which advocated for "applied and basic combined (ABC)," was joined by Narayanamurti and Odumosu's (2) book on *Cycles of Invention and Discovery: Rethinking the Endless Frontier*. Dan Sarewitz (3) wrote a powerful essay on "Saving science," pushing for reform of science to increase its impact, while reducing the prevalence of results that could not be replicated. Sarewitz (3) stressed that "scientists must come out of the lab and into the real world." A similar call for emphasizing applications as the path to discoveries came

This paper results from the Arthur M. Sackler Colloquium of the National Academy of Sciences, "Modeling and Visualizing Science and Technology Developments," held December 4–5, 2017, at the Arnold and Mabel Beckman Center of the National Academies of Sciences and Engineering in Irvine, CA. The complete program and video recordings of most presentations are available on the NAS website at www.nasonline.org/modeling_and_visualizing.

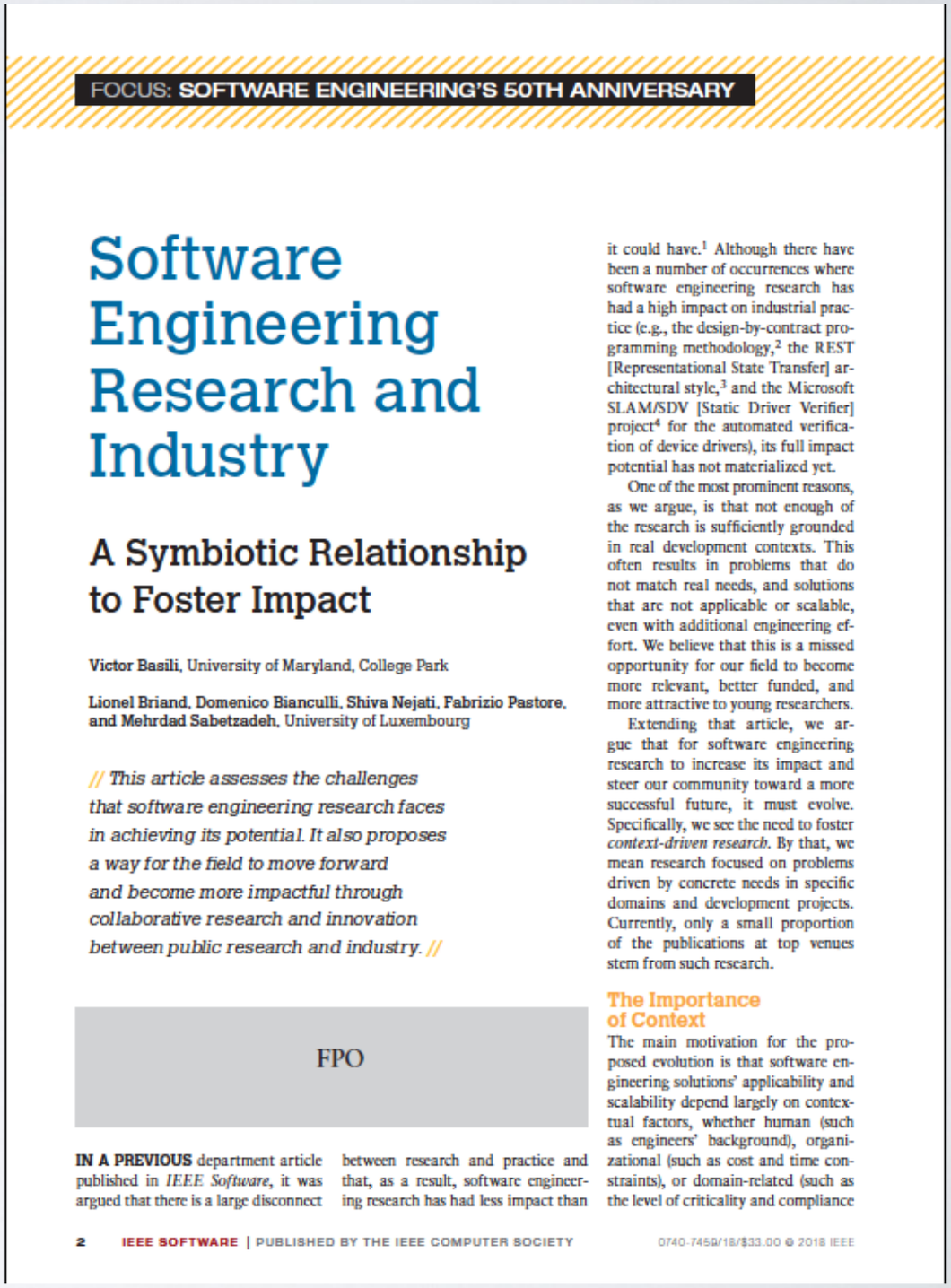
Author contributions: B.S. designed research, analyzed data, and wrote the paper.

The author declares no conflict of interest.

This article is a PNAS Direct Submission. W.R. is a guest editor invited by the Editorial Board.

[1]Email: ben@cs.umd.edu.

Published online December 10, 2018.



FOCUS: SOFTWARE ENGINEERING'S 50TH ANNIVERSARY

Software Engineering Research and Industry

A Symbiotic Relationship to Foster Impact

Victor Basili, University of Maryland, College Park

Lionel Briand, Domenico Bianculli, Shiva Nejati, Fabrizio Pastore, and Mehrdad Sabetzadeh, University of Luxembourg

// This article assesses the challenges that software engineering research faces in achieving its potential. It also proposes a way for the field to move forward and become more impactful through collaborative research and innovation between public research and industry. //

FPO

IN A PREVIOUS department article published in *IEEE Software*, it was argued that there is a large disconnect between research and practice and that, as a result, software engineering research has had less impact than it could have.[1] Although there have been a number of occurrences where software engineering research has had a high impact on industrial practice (e.g., the design-by-contract programming methodology,[2] the REST [Representational State Transfer] architectural style,[3] and the Microsoft SLAM/SDV [Static Driver Verifier] project[4] for the automated verification of device drivers), its full impact potential has not materialized yet.

One of the most prominent reasons, as we argue, is that not enough of the research is sufficiently grounded in real development contexts. This often results in problems that do not match real needs, and solutions that are not applicable or scalable, even with additional engineering effort. We believe that this is a missed opportunity for our field to become more relevant, better funded, and more attractive to young researchers.

Extending that article, we argue that for software engineering research to increase its impact and steer our community toward a more successful future, it must evolve. Specifically, we see the need to foster *context-driven research*. By that, we mean research focused on problems driven by concrete needs in specific domains and development projects. Currently, only a small proportion of the publications at top venues stem from such research.

The Importance of Context

The main motivation for the proposed evolution is that software engineering solutions' applicability and scalability depend largely on contextual factors, whether human (such as engineers' background), organizational (such as cost and time constraints), or domain-related (such as the level of criticality and compliance

 0740-7459/18/$33.00 © 2018 IEEE

AI in SE: A 25-year Journey

Lionel Briand

PROMISE 2020

References

Selected References

- Matinnejad et al., “MiL Testing of Highly Configurable Continuous Controllers: Scalable Search Using Surrogate Models”, ASE 2014
- Matinnejad et al., “Automated Test Suite Generation for Time-Continuous Simulink Models”, ICSE 2016.
- Matinnejad et al., “Test Generation and Test Prioritization for Simulink Models with Dynamic Behavior”, IEEE Transactions on Software Engineering, 2018
- Liu et al., “Effective Fault Localization of Automotive Simulink Models: Achieving the Trade-Off between Test Oracle Effort and Fault Localization Accuracy”, Empirical Software Engineering (Springer), 2019
- Liu et al., “Simulink Fault Localisation: An Iterative Statistical Debugging Approach”, Software Testing, Verification & Reliability (Wiley), 2016
- Ben Abdessalem et al., "Testing Vision-Based Control Systems Using Learnable Evolutionary Algorithms”, ICSE 2018
- Ben Abdessalem et al., "Testing Autonomous Cars for Feature Interaction Failures using Many-Objective Search”, ASE 2018
- Nejati et al., “Evaluating Model Testing and Model Checking for Finding Requirements Violations in Simulink Models”, ESEC/FSE 2019.

Selected References

- Shousha et al., "Using Genetic Algorithms for Early Schedulability Analysis and Stress Testing in Real-Time Systems", Journal of Genetic Programming and Evolvable Machines (Springer), 2006.

- Shousha et al., “A UML/MARTE Model Analysis Method for Uncovering Scenarios Leading to Starvation and Deadlocks in Concurrent Systems”, IEEE Transactions on Software Engineering, 2012.

- Nejati and Briand, “Identifying Optimal Trade-Offs between CPU Time Usage and Temporal Constraints Using Search”, ISSTA 2014

- Di Alesio et al. “Combining genetic algorithms and constraint programming to support stress testing of task deadlines”, ACM Transactions on Software Engineering and Methodology, 2015

- Arisholm et al., “A systematic and comprehensive investigation of methods to build and evaluate fault prediction models”, Journal of Systems and Software, 2010

- Yue et al., “Facilitating the transition from use case models to analysis models: Approach and experiments.”, ACM TOSEM, 2013

Selected References

- Wang et al., “Automatic generation of system test cases from use case specifications”, ISSTA 2015
- Wang et al., “Automatic Generation of Acceptance Test Cases from Use Case Specifications: an NLP-based Approach”, IEEE TSE (accepted), 2020
- Wang et al., “Oracles for Testing Software Timeliness with Uncertainty”, ACM TOSEM, 2019
- Shin et al., “Test case prioritization for acceptance testing of cyber-physical systems”, ISSTA 2018
- Shin et al.,“HITECS: A UML Profile and Analysis Framework for Hardware-in-the-Loop Testing of Cyber Physical Systems”, MODELS 2018
- Soltana et al., “Practical Model-driven Data Generation for System Testing”, ACM TOSEM, 2019
- Shin et al., “Dynamic Adaptive Network Configuration for IoT Systems: A Search-based Approach”, ACM/IEEE SEAMS, 2020

Selected References

- Jan et al., Automatic Generation of Tests to Exploit XML Injection Vulnerabilities in Web Applications. IEEE Transactions on Software Engineering (TSE), 2019

- Appelt et al., A Machine Learning-Driven Evolutionary Approach for Testing Web Application Firewalls. IEEE Transactions on Reliability (TR), 2018

- Appelt et al., Automatically Repairing Web Application Firewalls Based on Successful SQL Injection Attacks. IEEE International Symposium on Software Reliability Engineering (ISSRE 2017)

- Jan et al., Search-based Testing Approach for XML Injection Vulnerabilities in Web Applications. IEEE International Conference on Software Testing, Verification and validation (ICST 2017)

- Jan et al., Automated and Effective Testing of Web Services for XML Injection Attacks. International Symposium on Software Testing and Analysis (ISSTA 2016)

- Ceccato et al., SOFIA: An Automated Security Oracle for Black-Box Testing of SQL-Injection Vulnerabilities, IEEE/ACM International Conference on Automated Software Engineering (ASE 2016)

Selected References

- Arora et al. “Extracting Domain Models from Natural-Language Requirements: Approach and Industrial Evaluation”, MODELS 2016
- Arora et al. “Automated Extraction and Clustering of Requirements Glossary Terms”, IEEE TSE, 2017
- Arora et al., “An Active Learning Approach for Improving the Accuracy of Automated Domain Model Extraction”, ACM TOSEM, 2019
- Nejati et al., “Automated Change Impact Analysis between SysML Models of Requirements and Design”, FSE 2016
- Abualhaija et al., “A Machine-Learning Approach for Demarcating Requirements in Textual Specifications”, RE 2019
- Bettaieb et al., “Decision Support for Security-Control Identification Using Machine Learning”, REFSQ 2019
- Sleimi et al., “ A Query System for Extracting Requirements-related Information from Legal Texts”, RE 2019
- Sleimi et al., “Automated Extraction of Semantic Legal Metadata using Natural Language Processing”, RE 2018