Keynote Speech III

- Automated Software Transplantation
  Mark Harman (University College London, UK)
Automated Software Transplantation

QRS 2016
Talk by Mark Harman
PhD work by Alexandru Marginean

Collaborators
Earl Barr, Yue Jia, Justyna Petke
CREST, University College London
COWs

CREST Open Workshop

Roughly one per month

Discussion based

Recorded and archived
http://crest.cs.ucl.ac.uk/cow/

---

Search Based Software Engineering - SBSE

Mark Harman, UCL CREST
Roughly one per month

Discussion based

Recorded and archived http://crest

Search Based Software Engineering - SBSE
Cross-disciplinary Modeling – the Good, the Bad, and the Ugly

QRS 2016

August 2 2016, TU Wien

Gerti Kappel & Team
Business Informatics Group
Institute of Software Technology and Interactive Systems
Vienna University of Technology
Favoritenstraße 9-11/188-3, 1040 Vienna, Austria
phone: +43 (1) 58801-18804 (secretary), fax: +43 (1) 58801-18896
office@big.tuwien.ac.at, www.big.tuwien.ac.at
Cross-disciplinary Modeling –
the Good, the Bad, and the Ugly
QRS 2016
August 2 2016, TU Wien

Gerti Kappel & Team
Business Informatics Group
Institute of Software Technology and Interactive Systems
Vienna University of Technology
 Favoritenstraße 9-11/186-3 1040 Vienna, Austria
phone: +43 (1) 58801-18804 (secretary), fax: +43 (1) 58801-18896
office@big.tuwien.ac.at, www.big.tuwien.ac.at
Think Ecosystems:
People, Services/Processes, Things

Diverse users with complex networked dependencies and intrinsic adaptive behavior – has:

1. Robustness mechanisms: achieving stability in the presence of disruption

2. Measures of health: diversity, population trends, other key indicators
Integration Bugs

- A fundamental nature of integration bugs is in the architectural mismatch, where programmers make incorrect assumptions about various integrated components in software applications.
- When creating unit tests for methods of a given component, a programmer often does not have enough information on how these methods will be used by other programmers who will integrate this component with their components.

\[ S \rightarrow C_m \rightarrow S' \rightarrow C_k \rightarrow S'' \]

Unit test \( U_{C_m} \)  Unit test \( U_{C_k} \)
When creating unit tests for methods of a given component, a programmer often does not have enough information on how these methods will be used by other programmers who will integrate this component with other components.

\[ S \rightarrow C_m \rightarrow S' \rightarrow C_k \rightarrow S'' \]

- Unit test $U_{Cm}$
- Unit test $U_{Ck}$
Mission-specific software engineering details - Takeways

- Hardware target evolves during the development
  - COTS: test platform supplied by the vendor
  - Emulator (EM): platform with flight hardware peripherals are added as they are received from suppliers
  - FlatSat: platform for performing functional / interface tests, system-level tests and procedure validation
  - Flight hardware: the real deal
Mission-specific software engineering details - Takeways

- Hardware images evolve during the development.
- CORPS and platforms supplied by the vendor.
- Inherently, CORPS patterns with digital hardware peripherals are added as they are received from suppliers.
- Fast-track platform for performing functional / interface tests, system-level tests, and system-level validation.
- Flight hardware - the real deal.
Ensemble Formula

- An ensemble formula is a combination of solo formulas that improves the stability.

\[
susp_{ensemble}(s) = \sum_{x=1}^{n} n\text{susp}_x(s) \times w_x(s)
\]

coverage variables of entity \( s \)

normalized suspiciousness score
Motivation

- The committing and the branching as an important DVCS feature, have not been previously investigated together to find related file change sets.

- Although there are also CVCS that support committing and branching, the fact the developers in Git work locally and commit their changes on the remote server, motivates us to investigate how these factors influence the relatedness of change sets.
BAYESIAN FAULT LOCALIZATION USING DOLDA

Bug report ABC:
Customer: XXX
Site: YYY
Priority: Zzzz
Heading: It crashed!
Observation: During...

Probability distribution over classes:
Classes are the components of the system
Or
The design teams in the organization
Bug report ABC:
Customer: XXX
Site: YYY
Priority: Zzzzz
Helping: It crashed!
Configuration: During…

Probability distribution over classes:
Classes are the components of the system
Or
The design teams in the organization
Specifying and controlling elasticity of human-based services

What if we need to invoke a human?

#predictive maintenance analyzing chiller measurement
#SYBL_ServiceUnitLevel
Mon1 MONITORING accuracy = Quality.Acuracy
Cons1 CONSTRAINT accuracy < 0.7
Str1 STRATEGY CASE Violated(Cons1):
Notify(Incident.DEFAULT, ServiceUnitType.HBS)
Background

- Test case prioritization
  - Given: $T$, a selected test suite; $PT$, the set of permutations of $T$; $f$, a function from $PT$ to real numbers
  - Problem: Find a $T'' \in PT$ such that $(\forall T''') (T'' \in (PT \setminus \{T''\}) [f(T') > f(T''')]$

- APFD
  - To measure the weighted average of the percentage of faults detected during the execution of the test suite

\[
APFD = 1 - \frac{TF_1 + TF_2 + \ldots + TF_m}{mn} + \frac{1}{2n}
\]
### Table II
**The Summary of Subject Programs.**

<table>
<thead>
<tr>
<th>Program</th>
<th>Versions</th>
<th>LoC</th>
<th>Test</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>print_tokens</code></td>
<td>15</td>
<td>570/726</td>
<td>4115/4130</td>
<td>Lexical analyzer</td>
</tr>
<tr>
<td><code>replace</code></td>
<td>27</td>
<td>564</td>
<td>5542</td>
<td>Pattern recognition</td>
</tr>
<tr>
<td><code>schedule</code></td>
<td>16</td>
<td>374/412</td>
<td>2650/2710</td>
<td>Priority scheduler</td>
</tr>
<tr>
<td><code>tcas</code></td>
<td>29</td>
<td>173</td>
<td>1608</td>
<td>Altitude separation</td>
</tr>
<tr>
<td><code>tot_info</code></td>
<td>18</td>
<td>565</td>
<td>1052</td>
<td>Info. measure</td>
</tr>
<tr>
<td><code>space</code></td>
<td>35</td>
<td>6199</td>
<td>4333</td>
<td>ADL interpreter</td>
</tr>
<tr>
<td><code>flex</code></td>
<td>53</td>
<td>10459</td>
<td>567</td>
<td>Lexical analyzer</td>
</tr>
<tr>
<td><code>grep</code></td>
<td>29</td>
<td>14427</td>
<td>370</td>
<td>Pattern match</td>
</tr>
<tr>
<td><code>sed</code></td>
<td>29</td>
<td>14427</td>
<td>370</td>
<td>Stream editor</td>
</tr>
<tr>
<td><strong>NUDT</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Experiment</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Trustful analysis result of Reliability based testing

At the review and analysis of test cases, test procedures and test result:

- Test cases exist for each requirement
- The criteria of normal and robustness test are satisfied for each requirement;
- The objectives for structure coverage of source code is achieved:
  - Requirement Coverage 100%
  - Statement coverage 100%
  - Decision coverage 100%
  - MC/DC coverage 100%

Test is enough from every points of view.
Test procedure DATA is enough for Reliability Assess.
Reliability Trustful
What about A&T?

- Revision of safety standard in ISO 10218 is in progress.
- A&T Techniques need to address these:

**Characteristics**
- Non-deterministic
- Ever-changing environment
- Intense interaction
- Continuous behavior

**Challenges**
- Significance of the results
- Error masking
- Oracle Problem
- State space explosion
Deception Strategy Model

- Security Risk
  - Increases
  - Acknowledge Risk
- Goal
  - Risk
  - Triggers
  - Security Action
  - Effectiveness measured by
- Channel
  - Monitored by
- Deception Strategy
  - Deception Tactic
  - Deception Story
  - Deception Technique
- Decay
  - Simulation
  - Dissimulation
- Bias
- Perceived
  - Uses
  - Exploits
  - Exploits
Department of Computer Engineering

IEC/PAS 62814
Edition 1.0 2012-12

PUBLICLY AVAILABLE SPECIFICATION
PRE-STANDARD

Fevzi Belli,
Univ. Paderborn, Germany
and
Izmir Institute of Technology (IYTE) Izmir, Turkey

Dependability of software products containing reusable components – Guidance for functionality and tests

1. Introduction
2. Software Reuse Technology
3. Model-Based Reuse
4. Testing and Reusability
5. Conclusions
Background - STeC

- The Internet of Things (IoT) and Cyber-Physical Systems (CPS) are new trends of real-time systems in the area of information technology. The consistency between time and location is a basic feature of IoT and CPS.
  - Ad Hoc Wireless Network
  - IoV (Internet of Vehicle)
  - ITS (Intelligent Transportation System)
- Spatial-temporal consistency
  - An agent behaves at the required location at the required time and finishes the required tasks within the required time.
- STeC, proposed by Chen in 2012, to specify real-time systems
  - Spatial-temporal consistence language
- Focus on the Equivalence of three refinement semantics for STeC
  - Algebraic, denotational and operational refinements
Outlier Detection:
An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism” [Hawkins 1980]

Outlier Definition?
- Noise objects in clustering
- Top-n outliers
- Low probability to be generated

located at the border of the data space

Detection Method?
- Deviation-based Approaches
- Distance-based Approaches
- Density-based Approaches
- Statistical Tests
- Depth-based Approaches