Improving Impact of Self-Adaptation and Self-Management Research through Evaluation Methodology

Yuriy Brun

University of Washington

http://www.cs.washington.edu/homes/brun
Statements We Take for Granted

- Self-adaptation and self-management (SASM) are the future.
- SASM research will soon impact everyday software.
- Our work matters to people other than us.
My Goals

- Reasoning why SASM research is important.

- Quantitative analysis of impact SASM research has on premier software engineering venues.

- Increasing that impact through evaluation methodology.
Emergence of New Venues

- **SEAMS 2007**: 2006–2007
- **SaSo**: 2007–
- **Schloss Dagstuhl**: 2008, 2010
Industrial Commitment

- Software complexity crisis [IBM01] and autonomic computing [KC03]
- Education: 2 courses at GATech, OSU, and NCSU [KM10]
- NSF Center for Autonomic Computing ($35K annually)
  others involved: UF, UA, Rutgers, BAE Systems, Ball Aerospace, Intel, Microsoft, Northrop-Grumman, NEC, Raytheon, Xerox, Citrix, Imaginestics, and ISCA Technologies

- Large-Scale Internet Computing Initiative [Pre07]
- Over $5 million to study cloud computing [Pre09]

- 2010 appropriations includes adaptive systems technology [Jon09]
“Aspects of autonomic computing have ingrained themselves into products across IBM’s entire product line. Rather than becoming a product of its own, autonomic computing improves the quality of existing, and newly developed products.”

-Brent A. Miller [KM10], lead architect, IBM
SASM Research in Premier SE Venues

Surveyed seven instances of premier conferences and two years of journals:

- FSE 2008, ESEC/FSE 2009
- ASE 2008, 2009
- TSE 2008–2009
- TOSEM 2008–2009
## Survey Results

<table>
<thead>
<tr>
<th>Venue</th>
<th>SASM papers</th>
<th>Total</th>
<th>Ratio</th>
<th>SASM papers</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICSE 2010</td>
<td>0</td>
<td>52</td>
<td>0.0%</td>
<td>—</td>
</tr>
<tr>
<td>ESEC/FSE 2009</td>
<td>2</td>
<td>37</td>
<td>5.4%</td>
<td>[CCG+09, DPT09]</td>
</tr>
<tr>
<td>ASE 2009</td>
<td>2</td>
<td>38</td>
<td>5.3%</td>
<td>[BJSS09, WM09]</td>
</tr>
<tr>
<td>ICSE 2009</td>
<td>5</td>
<td>50</td>
<td>10.0%</td>
<td>[CK09, CMP09, EGMT09]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>[MBNJ09, WNGF09]</td>
</tr>
<tr>
<td>TSE 2009</td>
<td>0</td>
<td>50</td>
<td>0.0%</td>
<td>—</td>
</tr>
<tr>
<td>TOSEM 2009</td>
<td>1</td>
<td>13</td>
<td>7.7%</td>
<td>[MZ09]</td>
</tr>
<tr>
<td>FSE 2008</td>
<td>2</td>
<td>31</td>
<td>6.5%</td>
<td>[RSE08, SRWE08]</td>
</tr>
<tr>
<td>ASE 2008</td>
<td>1</td>
<td>34</td>
<td>2.9%</td>
<td>[ZSD08]</td>
</tr>
<tr>
<td>ICSE 2008</td>
<td>1</td>
<td>55</td>
<td>1.8%</td>
<td>[SBP08]</td>
</tr>
<tr>
<td>TSE 2008</td>
<td>4</td>
<td>51</td>
<td>7.8%</td>
<td>[CPS08, CC08, DY08, HOM08]</td>
</tr>
<tr>
<td>TOSEM 2008</td>
<td>0</td>
<td>20</td>
<td>0.0%</td>
<td>—</td>
</tr>
<tr>
<td>Conference Total</td>
<td>13</td>
<td>297</td>
<td>4.4%</td>
<td></td>
</tr>
<tr>
<td>Journal Total</td>
<td>5</td>
<td>134</td>
<td>3.7%</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>18</td>
<td>431</td>
<td>4.2%</td>
<td></td>
</tr>
</tbody>
</table>
The survey revealed a gap between the expected impact and the apparent impact of self-adaptation and self-management research.
Potentials Reasons for the Impact Gap

- Maturity of the field.

- Emergence of new venues outside of software engineering.

- Relevance to other fields:
  - artificial intelligence
  - pervasive and ubiquitous computing
  - middleware infrastructure
  - multiagent
  - distributed robotics

Yuriy Brun (brun@cs.washington.edu)
Why Care about the Gap?

Software engineering can benefit from self-adaptation and self-management research being properly evaluated, reviewed, and published in its venues.
Many SASM papers involve evaluation methodologies far different from those of SE research.
Evaluation Differences

- Comparison

- Analysis
Evaluation Differences

- Comparison
  - Current technologies cannot handle dynamic environments.

- Analysis
Evaluation Differences

- **Comparison**
  - Current technologies cannot handle dynamic environments.
  - Certain aspects are crucial to dynamic environments.

- **Analysis**
Evaluation Differences

- **Comparison**
  - Current technologies cannot handle dynamic environments.
  - Certain aspects are crucial to dynamic environments.
  - Comparison to naïve SASM solutions.

- **Analysis**
Evaluation Differences

• Comparison
  - Current technologies cannot handle dynamic environments.
  - Certain aspects are crucial to dynamic environments.
  - Comparison to naïve SASM solutions.
  - Comparison to state-of-the-art SASM solutions.

• Analysis
Evaluation Differences

- **Comparison**
  - Current technologies cannot handle dynamic environments.
  - Certain aspects are crucial to dynamic environments.
  - Comparison to naïve SASM solutions.
  - Comparison to state-of-the-art SASM solutions.

- **Analysis**
  - Lack of a formal theoretical complexity analysis.
Case Study

Tile architecture: a distributed software system for solving computational problems inspired by nature’s self-assembly.
Distribute a computation so that each machine deals with an insignificant portion of the computation, and no small group of machines knows too much or has too much power.
Prototype

1. Build a prototype system.
## Prototype

1. Build a prototype system.
2. Run it.
Prototype

1. Build a prototype system.
2. Run it.
Prototype

1. Build a prototype system.
2. Run it.
3. No comparison of any kind!
Comparison

1. Compare running the system on $n$, $10n$, $100n$, $1000n$ machines.
Comparison

1. Compare running the system on $n$, $10n$, $100n$, $1000n$ machines.
2. Show scaling trends and great improvement.
Comparison

1. Compare running the system on $n$, $10n$, $100n$, $1000n$ machines.
2. Show scaling trends and great improvement.
Comparison

1. Compare running the system on $n$, $10n$, $100n$, $1000n$ machines.
2. Show scaling trends and great improvement.
3. Self-comparison is fundamentally flawed.
Comparison with State-of-the-Art

1. Theoretical analysis of privacy and security.
Comparison with State-of-the-Art

1. Theoretical analysis of privacy and security.
2. Compare to related work on distributed computation.
Comparison with State-of-the-Art

1. Theoretical analysis of privacy and security.
2. Compare to related work on distributed computation.
3.
Comparison with State-of-the-Art

1. Theoretical analysis of privacy and security.
2. Compare to related work on distributed computation.

- Without an analysis of frequency of faults, the reviewer has no way of assessing the benefits.
- Ideas published in a single paper may not be the state-of-the-art.
Analyze the Competition

1. Combine obvious existing solutions.
Analyze the Competition

1. Combine obvious existing solutions.
2. Analyze effects of environment dynamics on existing solutions.
Analyze the Competition

1. Combine obvious existing solutions.
2. Analyze effects of environment dynamics on existing solutions.
3. If your experiments cannot be repeated by independent scientists, you don't get to wear the scientist hat!

Yuriy Brun (brun@cs.washington.edu)
Analyze the Competition

1. Combine obvious existing solutions.
2. Analyze effects of environment dynamics on existing solutions.
3. If your experiments cannot be repeated by independent scientists, you don’t get to wear the scientist hat!
Yuriy Brun (brun@cs.washington.edu)
Impact gap: We 💖 SASM research, but it represents only 4.2% of the premier SE papers (none at ICSE 2010).

Evaluation methodology affects comparing SASM and SE research.

Careful evaluation can increase the impact of SASM research.


Siu-Nam Chuang and Alvin T.S. Chan.
Dynamic QoS adaptation for mobile middleware.

Valeria Cardellini, Emiliano Casalicchio, Vincenzo Grassi, Francesco Lo Presti, and Raffaela Mirandola.
QoS-driven runtime adaptation of service oriented architectures.

Radu Calinescu and Marta Kwiatkowska.
Using quantitative analysis to implement autonomic IT systems.


Shlomi Dolev and Reuven Yagel.  
Towards self-stabilizing operating systems.  

Ilenia Epifani, Carlo Ghezzi, Raffaela Mirandola, and Giordano Tamburrelli.  
Model evolution by run-time parameter adaptation.  

William G.J. Halfond, Alex Orso, and Pete Manolios.  
WASP: Protecting web applications using positive tainting and syntax-aware evaluation.  

IBM.  
Autonomic computing manifesto.

Richard M. Jones.
Fiscal year 2010 National Science Foundation appropriation.

Jeffrey O. Kephart and David M. Chess.
The vision of autonomic computing.

Paul Kontogiorgis and Brent A. Miller.
Interview on the state of and IBM’s contribution to autonomic computing.
Personal communication, January 2010.

Brice Morin, Olivier Barais, Gregory Nain, and Jean-Marc Jézéquel.
Taming dynamically adaptive systems using models and aspects.


Franco Raimondi, James Skene, and Wolfgang Emmerich.
Efficient online monitoring of web-service SLAs. 

Sylvain Sicard, Fabienne Boyer, and Noel De Palma. 
Using components for architecture-based management: the self-repair case. 

Michele Sama, David S. Rosenblum, Zhimin Wang, and Sebastian Elbaum. 
Model-based fault detection in context-aware adaptive applications. 

Yiqiao Wang and John Mylopoulos.
Self-repair through reconfiguration: A requirements engineering approach.


Westley Weimer, ThanhVu Nguyen, Claire Le Goues, and Stephanie Forrest.
Automatically finding patches using genetic programming.

Andrea Zisman, George Spanoudakis, and James Dooley.
A framework for dynamic service discovery.