Runtime Software Adaptation: Framework, Approaches, Styles

Peyman Oreizy
Launch21

Nenad Medvidovic
USC

Richard N. Taylor
UC Irvine
What? Our paper got the MIP award?!
What? Our paper got the MIP award?!

- 1st thought: Someone read our paper!
What? Our paper got the MIP award?!

- 1st thought: Someone read our paper!
- 2nd thought: Wow, lots of citations!
  - Original paper: 315
  - Follow-on journal paper: 375
  - 690 papers over ~9 yrs ≈ 1.5pppw
What? Our paper got the MIP award?!

- 1st thought: Someone read our paper!
- 2nd thought: Wow, lots of citations!
  - Original paper: 315
  - Follow-on journal paper: 375
  - 690 papers over ~9 yrs ≈ 1.5 pppw
- 3rd thought: Could one person be responsible for all of them?
Change during runtime?

- Critical systems require “continuous availability”
- Power grid, financial systems, ...
- Increasingly important in everyday systems
Hi,

I'm the Bloglines Plumber. Bloglines is down for a little fixer upper. We will be back shortly. Bloglines will be all better when I'm done with it.

Thanks,
The Bloglines Plumber
Server Error

The server encountered an error and could not complete your request.

If the problem persists, please mail error@google.com and mention this error message and the query that caused it.
Server Error

We're sorry, but Gmail is temporarily unavailable. We're currently working to fix the problem -- please try logging in to your account in a few minutes.
New software is available for your computer.

Installing this software may take some time. If you're not ready to install now, you can choose Software Update from the Apple menu later.

Install Name Version Size
--- --- ---
[
Mac OS X Update 10.5.1 39.8 MB

The 10.5.1 Update is recommended for all users running Mac OS X Leopard and includes general operating system fixes that enhance the stability, compatibility and security of your Mac.

For detailed information on this update, please visit this website:
For detailed information on security updates, please visit this website:

Note: Use of this software is subject to the original Software License Agreement(s) that accompanied the software being updated. A list of Apple SLAs may be found here: http://www.apple.com/legal/sla/.

Restart will be required.
How did we get here?
How did we get here?

- Serendipity
How did we get here?

• Serendipity
• Key insights:
  • connectors
  • explicit arch-model fielded with the system and used to govern change
  • architectural style
State of the Practice

- redundant and fault-tolerant hardware
- “hot pluggable” drives and memory
- system virtualization (ala VMware and Xen)
- binary code patching
- programming language facilities for dynamic loading, linking, and patching of code
- software designed for fault tolerance (architectural styles and patterns)
State of the Practice

• Each approach has its place
• No one approach encompasses the others
• Clear benefits to enacting change at multiple levels of abstraction
• Need a framework for comparing and combining approaches
Towards a Unifying Framework
Towards a Unifying Framework

All approaches:

1. Use a “model” to highlight some system details while hiding others

2. Grapple with 5 aspects of runtime change:
   a. evolve behavior
   b. evolve state
   c. adjust execution context
   d. asynchronous change
   e. probe running system
A Look Back

• What has happened in the past decade?
• Dynamic adaptation models
• Research projects
• Open-source and commercial systems
• Conferences, symposia, and workshops
Dynamic Adaptation Models I

- Prior to our ICSE 1998 paper
- Style-based models: CHAM, graph-grammars
- ADL-based models: Darwin, Dynamic Wright, Rapide
- Did not gain wide adoption
- Lack of system-level facilities
- Constrained notion of dynamism
Dynamic Adaptation Models II

- Subsequent to our ICSE 1998 paper
- “Figure 8” model: system adaptation driven by architecture
Dynamic Adaptation Models III

- Rainbow: similar to “Figure 8”
- Self-managed systems: dynamic plan generation
Research Projects

- Aura: QoS-driven system reconfiguration
- MobiPads: QoS optimization via dynamic reconfiguration
- Siena: Client-, server-, and network-level dynamism
- Grid computing: Dynamic addition and removal of computing resources
Commercial Solutions

- Koala: predefined dynamic adaptations via options
- Skype
  - Promotion/demotion of nodes
- P2P-based adaptations
- MapReduce: automatic data rerouting from failed to live nodes
Conferences/Symposia

- Dynamism as primary focus
- Dynamism as a means or by-product
- Dynamism in flagship SE conferences
Dynamism as Primary Focus

- ICAC
- ACW
- CHIAACS
- SEAMS
- WOSS
- DEAS
- WADS
- IWPSE
- Dagstuhl SESAS
Dynamism as By-Product

- MobiCom
- PerCom
- CD
- Middleware
Dynamism in SE Conferences

• What happened to dynamism at:
  • ICSE
  • FSE
  • ASE

• What about software architecture venues:
  • WICS
  • ECSA
  • QoS
  • CBSE
Promising Directions

- A simple message: if you want or need adaptable applications you can either:
  - Make no constraints on developers
    - ... and then work like crazy to try to obtain adaptation
  - Constrain development to make adaptation easier and predictable
- This should not be news: the message is styles
How Do You Make Adaptation Easier?

• Make the elements subject to change identifiable
• Make interaction controllable
• Provide for management of state
Lots of Successful Examples

- Pipe-and-filter
- Dynamic pipe-and-filter: Weaves
- Event-based systems: Field & pub-sub
- Event-based components and connectors: C2
- REST
<table>
<thead>
<tr>
<th>Arch Style</th>
<th>Update Behavior</th>
<th>Update State</th>
<th>Update exec context</th>
<th>Asynchrony of change</th>
<th>Impl. probes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pub-Sub</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Weaves</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>C2</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>REST</td>
<td>✔</td>
<td>Data-State externalized</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>CREST</td>
<td>✔</td>
<td>All computation state externalized</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>
Where Has Software Engineering Been WRT the Development of Effective Styles?

Not just for adaptivity, but other qualities too
UPOZORENJE! ISPUSTA SE VODA IZ BAZENA

ATTENTION! THE WATER IN THE SWIMMING POOL IS BEING CHANGED
A Call to Action

• A science of design
• A science of realization
• A science of dynamic adaptation
• A science of domain-specific software engineering
• (Discovery-based research)