Dynamic weaving for aspect-oriented programming
a 10 year perspective

Andrei Popovici, Thomas Gross, Gustavo Alonso

Presentation in Modularity AOSD-2012, Potsdam
March 30th 2012
Contents

- Rationale for the 2002 paper
  - The dynamic weaving approach
  - The road ahead
### Motivation: Run-time Adaptation (1/2)

#### Changes

- **Many devices beyond conventional computers**
  - Wearable computing, sensor networks, etc.
  
  - Application integration: in automotive, enterprise architectures, etc.

- **Increasing mobility**
  - Applications do not know in advance in which context (car, home) they operate
  
  - Range of mobility significantly increases

- **Rapid changes in world economy – need for more agility**
  - Unpredictable and rapid changes to global production inputs (e.g., value chains, TTM)
  
  - Need for faster and better integration

#### Challenges

- Debugging a network of hundreds of sensors

- Developing software for hundreds of deployment configurations

- Make systems designed as black / grey / white boxes become more adaptive
The changes have rather amplified: why software needs to adapt?

Value stream

Probability steel price change

Before 2000
2000-2005
After 2005

Monthly price change in percent

Macro/regulatory

S&P 500 Volatility Index (1990-2011)

Old Normal
Average: 18.6
Stdev: 5.9

New Normal
Average: 22.0
Stdev: 9.4

Non-economic

Number of disruptive events

Global interconnectness

- Geophysical events
- Meteorological events
- Hydrologic catastrophes
- Climatic catastrophes

- S&P 500 Volatility Index (1990-2011)
- Average: 18.6
- Stdev: 5.9
- New Normal
- Average: 22.0
- Stdev: 9.4

- 2009 – Russia
- Shut off all gas supplies to Europe

- Mar 2011 – Japan
- Oil political crisis
- Supply disruptions in pulp cause prices to reach near all-time high

- Apr 2010 & May 2011
- Iceland
- Volcanic eruption stopped air traffic

- Apr 2010 – Greece
- Major economic crisis dramatically reduced the value of Euro

- Apr 2009 – Dubai
- Economic crisis hits the world economies and major currencies

- Oct 2008 – Lehman Brothers filed Ch. 11 starting the global economic crisis

- Mar 2009 – Chile
- FTA between Asian nations removes tariffs on 90% of goods

- Apr 2009 – South
- Chinese transportation capacity shortage leads to price spikes

- Sep 2008 – USA
- Lehman Brothers filed Ch. 11 starting the global economic crisis

- Feb 2010 – Haiti
- Earthquake shutdown all commercial activity

- Jan 2009 – Russia
- Government and banking system collapsed

- Jan 2009 – Chile
- Earthquake hit all major supply chains

- Jan 2009 – Iceland
- Volcanic eruption stopped air traffic

- Mar 2009 – Iceland
- Government and banking system collapsed

- Feb 2010 – Haiti
- Earthquake hit all major supply chains

- Mar 2010 – Chile
- FTA between Asian nations removes tariffs on 90% of goods

- Apr 2010 – Greece
- Major economic crisis dramatically reduced the value of Euro

- Apr 2010 & May 2011
- Iceland
- Volcanic eruption stopped air traffic

- Apr 2010 – Greece
- Major economic crisis dramatically reduced the value of Euro

- Apr 2009 – Dubai
- Economic crisis hits the world economies and major currencies

- Mar 2009 – Chile
- FTA between Asian nations removes tariffs on 90% of goods

- Apr 2009 – South
- Chinese transportation capacity shortage leads to price spikes

- Oct 2008 – Lehman Brothers filed Ch. 11 starting the global economic crisis

- Sep 2008 – USA
- Lehman Brothers filed Ch. 11 starting the global economic crisis

- May 2010 – Gulf of Mexico
- Major oil spill affected the ocean traffic and seafood industry

- Jun 2009 – Honduras
- Major political crisis affected most businesses

- May 2010 – Gulf of Mexico
- Major oil spill affected the ocean traffic and seafood industry

- Apr 2010 & May 2011
- Iceland
- Volcanic eruption stopped air traffic
Postponing adaptation not only a software topic: used by companies to be agile in responding to changes

<table>
<thead>
<tr>
<th>Concept and solution</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>▪ Products are <strong>differentiated as late as possible</strong> in the value chain to allow <strong>responsiveness to actual demand</strong>, rather than forecast demand</td>
<td>Dyeing is carried out last so that customer <strong>color preferences</strong> can be satisfied</td>
</tr>
<tr>
<td>▪ <strong>Modularization</strong> can be used to create standard platforms</td>
<td><strong>Segments customers</strong> to trade-off against configurability and short lead times</td>
</tr>
<tr>
<td>▪ <strong>Postponement</strong> can be used to allow late customization</td>
<td>Bottles are labeled according to live choice at games and delivered in &lt;10 minutes giving <strong>ultimate customization</strong></td>
</tr>
<tr>
<td>▪ <strong>Variants can be removed</strong> completely through the design of the product so that postponement is not needed</td>
<td><strong>Removes variants</strong> of language instructions by using pictographs</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Result</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>▪ Inventories are held close to the customer allowing <strong>fast response</strong> to changing demand</td>
<td></td>
</tr>
<tr>
<td>▪ By customizing later (e.g., specific packaging, language instructions), products are kept in a standard, light form for longer. <strong>Faster and more flexible transport</strong> options may be used</td>
<td></td>
</tr>
<tr>
<td>▪ Delaying final customization ensures <strong>volume and mix flexibility</strong>; the last differentiating steps occur after real demand is known</td>
<td></td>
</tr>
</tbody>
</table>
Adaptation is becoming more important as consumer electronics and ‘software as a product’ gain momentum.

### Consumer electronics are starting to converge with cars

<table>
<thead>
<tr>
<th>Feature</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>PNDs</td>
<td><img src="image" alt="PNDs" /></td>
</tr>
<tr>
<td>Satellite radio</td>
<td><img src="image" alt="Satellite radio" /></td>
</tr>
<tr>
<td>Mobile navigation</td>
<td><img src="image" alt="Mobile navigation" /></td>
</tr>
<tr>
<td>Night vision</td>
<td><img src="image" alt="Night vision" /></td>
</tr>
<tr>
<td>Embedded car navigation</td>
<td><img src="image" alt="Embedded car navigation" /></td>
</tr>
<tr>
<td>Head-up display</td>
<td><img src="image" alt="Head-up display" /></td>
</tr>
<tr>
<td>iPod connectivity</td>
<td><img src="image" alt="iPod connectivity" /></td>
</tr>
<tr>
<td>Hands-free telephony</td>
<td><img src="image" alt="Hands-free telephony" /></td>
</tr>
<tr>
<td>E-call/b-call(^1) such as OnStar/BMW assist</td>
<td><img src="image" alt="E-call/b-call" /></td>
</tr>
<tr>
<td>Connectivity to CE devices</td>
<td><img src="image" alt="Connectivity to CE devices" /></td>
</tr>
<tr>
<td>Rear-seat entertainment</td>
<td><img src="image" alt="Rear-seat entertainment" /></td>
</tr>
</tbody>
</table>

### OEMs are starting to offer software as a product in cars

- Upgradable, configurable vehicles
- Customized driver experiences
- Aftersales services (apps)
- New distribution channels
- New diagnosis and repair processes

**VW: "tank up functionality"**

**Android-based apps**

**Audi A2** “… personalize by downloading software activating features such as heated seats, customized navigation, or stiffer suspension”

---

1 Emergency call/breakdown call
### CHALLENGES

- **Debugging a network of hundreds of sensors**
- **Developing software for hundreds of deployment configurations**
- **Make systems designed as black / grey / white boxes become more adaptive**

### CHANGES

- **Many devices beyond conventional computers**
  - Wearable computing, sensor networks, etc.
  - Application integration: in automotive, enterprise architectures, etc.

- **Increasing mobility**
  - Applications do not know in advance in which context (car, home) they operate
  - Range of mobility significantly increases

- **Rapid changes in world economy – need for more agility**
  - Unpredictable and rapid changes to global production inputs (e.g., value chains, TTM)
  - Need for faster and better integration

---

Stop all of them, debug by hand?

Do not try this at home ...
Ideal adaptation platform

Adaptation paradigms at the time

- Limited ability to identify when to adapt
- The application is responsible for knowing how to adapt
- Limited ability to react

We needed a generic adaptation mechanism that …

- Supports run time adaptation
- Is independent of who drives the adaptation process (the application itself or the computing environment)
- Is independent of the adaptation and the information needed to adapt
- Is small enough to be useful in mobile settings
- Easily fits with existing developing tools (no new languages, no new developing techniques)
Contents

- Rationale of the 2002 paper
- The dynamic weaving approach
- The road ahead
PROSE had a dual scope

Research scope

PROSE as platform for adaptive middleware

Objective

- As late as possible “interception” mechanism of the code execution (key in middleware)
- High expressiveness of describing adaptations
- Ability to work with multiple devices at a time

PROSE as software engineering project

- System for adaptation based on aspect-orientation
- Dynamic (run-time) weaving
- High flexibility in describing and deploying aspects
In a nut-shell, the system had five key traits …

1. Application was not “aware” / prepared for aspects and advice execution
2. The first implementation of the JVM was using the debugger interface
3. The “aspect interface” was used to abstract from the actual interception mechanism in the JVM (and survived 3 later JVM implementations)
4. The aspect engine was doing all the work
   ▪ When aspects were executed OR when they were woven into the JVM
5. Aspects and advice were all described in Java
... and its design was ‘balanced’ from the start towards flexibility rather than performance

<table>
<thead>
<tr>
<th>Area</th>
<th>Score</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expressiveness</td>
<td>High</td>
<td>Ability to capture a significant set of joinpoints</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Less than compile-time</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Better than other dynamic systems</td>
</tr>
<tr>
<td>Flexibility of execution altering / interception</td>
<td>Low</td>
<td>“Late” decision on what advice to execute (hence highly flexible) …</td>
</tr>
<tr>
<td>Performance</td>
<td></td>
<td>… and inefficient compared to load-time / compile-time approaches</td>
</tr>
<tr>
<td>Ease of use (also in distributed &amp; mobile settings)</td>
<td></td>
<td>Later implementations improved performance but decreased flexibility / expressiveness</td>
</tr>
</tbody>
</table>

- Aspects / and pointcuts were all Java classes, hence they were “network-friendly” (serialiazeable, etc)
- Trying hard to resemble AspectJ syntactics of aspects. Writing and extending the “language” was quite straightforward
PROSE was designed as middleware for adaptation and, as such, has considered many problems that other systems were not designed to address.

- **Atomic weaving**: in distributed settings, it is not enough to adapt one application, one needs to adapt many in a single logical step.

- **Registration, activation, and filtering of advices**: PROSE allows to include an advice without activating it, activating it but filter its execution, and sophisticated forms of filtering (at the AOP engine level)

- **Advice querying, matching and similarity searches on pools of distributed JVMs**: for development and debugging in large networks of devices

- **Automatic leasing mechanism on extensions (aspects)**

- **Security model and security checks on aspects** (signatures, sand boxing, authorization mechanisms)

- **Complete API based design for added flexibility** (add your own AOP engine)

... but was this too powerful?
Contents

- Rationale of the 2002 paper
- The dynamic weaving approach
- The road ahead
Key idea in PROSE: at the time controversial, today commonplace

<table>
<thead>
<tr>
<th>At the time controversial …</th>
<th>…today commonplace</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Key idea in PROSE:</strong> Adaptation as a run time software/systems engineering technique</td>
<td>instrumentation</td>
</tr>
<tr>
<td>the rest followed by being consequent</td>
<td>verification</td>
</tr>
<tr>
<td></td>
<td>state migration</td>
</tr>
<tr>
<td></td>
<td>distribution</td>
</tr>
<tr>
<td></td>
<td>adaptation</td>
</tr>
<tr>
<td></td>
<td>updates</td>
</tr>
</tbody>
</table>

A recent example: Multiplicity computing Cadar, Pietzuch, and Wolf, FoSER 2010
Since then, PROSE used in …

| Databases          |  
|--------------------|---
| Angela Nicoara, Gustavo Alonso:  
Making Applications Persistent at Run-time. ICDE 2007 |

| Systems            |  
|--------------------|---
| Angela Nicoara, Gustavo Alonso, Timothy Roscoe:  
Controlled, systematic, and efficient code replacement for running java programs. EuroSys 2008 |

| Software Engineering |  
|----------------------|---
| Susanne Cech Previtali, Thomas R. Gross:  
Aspect-based dynamic software updating: a model and its empirical evaluation. AOSD 2011 |
**Design & compile time vs. run time**

**PROSE:**

<table>
<thead>
<tr>
<th>Dynamic = run time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aspect = adaptation / modification (not necessarily an orthogonal concern)</td>
</tr>
<tr>
<td>PROSE = specialized JVM</td>
</tr>
</tbody>
</table>

**TODAY:**

- Increasing complexity in the underlying hardware
  - Virtualization
  - Clusters
  - Cloud
  - Mobile phones

- Can we adapt the deployment of an application
- What should be the working unit?
OSGi Application (Service Perspective)

Test module:test.project.main

Service:LogService

Service:CalculationService

Test module: test.project.logging

Test module: test.project.calculation
R-OSGi

Peer A

Peer B

Peer C

OSGi Framework

LogService

CalculationService

LogServiceProxy

CalcServiceProxy

test.project.main

test.project.calculation

test.project.logging
Example: Universal Remote Control

R-OSGi Service UI
AlfredO: Flexible multi-tier architecture
Mouse Controller application
Cirrostratus

**virtual deployment**

- implicit requirements
- non-functional requirements

**physical deployment**
Conclusions

Run time software/systems engineering

(Dynamic) Aspect oriented programming a key technique

Still many open, interesting questions
- too powerful?
- too difficult?
- a systems technique?
- a developer technique?
- working units? => the module?

Great prospects for the future
- very many application scenarios