Towards Business Driven Autonomic Service-Oriented Computing

Vladimir Tosic (NICTA, UWO, and UNSW)
Yan/Jenny Liu (NICTA and U. of Sydney)
Liming Zhu (NICTA and UNSW)

Introduction and Motivation

- **WS-Policy4MASC** is a policy-based language for run-time XML Web service management
  - **Powerful** support for specification of **business values** and **strategies** for maximization of business values
  - **Adaptive Server Framework (ASF)** is an autonomic middleware platform for adaptive applications
    - **Policy specification** is relatively simple/limited
  - **Adding** WS-Policy4MASC support to ASF
    - **Broadens applicability** of WS-Policy4MASC and improves autonomic computing support
    - Provides ASF **powerful policy specification and support for business-driven management**

About NICTA

- **Australian** center of excellence in **use-inspired ICT** research
- Established 2002, now 5 labs in 4 cities
- **NICTA** members (governments and universities):
  - **NICTA partners**:
    - 422 research staff, 296 postgraduate students
    - 4 research themes (e.g., managing complexity) & 6 business areas (e.g., software infrastructure)

Outline

- Introduction and motivation
- **WS-Policy4MASC**
  - Adaptive Server Framework (ASF)
  - Adding WS-Policy4MASC support to ASF
  - Conclusions and future work

WS-Policy4MASC – Background

- Execution of IT systems should be **aligned** with **business** objectives, values, and operations
  - IT improvements need not increase business value
- Our focus: **Web services** and their compositions
  - Distributed components using XML, SOAP, WSDL
- Competitive advantage: specifying and processing **business value and business strategy** information
  - Run-time specification: **WS-Policy4MASC**
  - Run-time management: MASC middleware
  - Design-time: UML profiles for WS-Policy4MASC
  - Run-time selection: UDDI extensions (ongoing)

WS-Policy4MASC

- **Policy** – high-level operation & management goals and/or rules (e.g., for security, performance, ...)
- **WS-Policy** is a general, W3C standardized policy specification framework for Web services
  - Leaves details for extensions (e.g., WS-Security)
- **WS-Policy4MASC** adds **new policy assertions and details** necessary for run-time management
  - Feasibility demonstrated by using it in the MASC middleware (see Erradi et al. at ICWS 2007)
  - Expressiveness, effectiveness, and usefulness evaluated on 2 case studies (1 simple, 1 medium)
Utility Policy Assertions

- **Diverse business values** assigned to conditions
  - E.g., if (*) was met, requester pays provider $2; otherwise provider pays requester $1
  - WS-Policy4MASC advantage is in modeling not only financial business values (prices, penalties), but also many others (e.g., customer satisfaction)
- **Business values classified along several dimensions**
  - Benefits vs. costs, agreed vs. possible, tangible vs. intangible, absolute vs. relative
- **Combinations determine business value types**
  - E.g., absolute intangible possible benefits (models the major aspects of customer satisfaction)

Meta-Policy Assertions

- Specify which **policy assertions are conflicting** and which **policy conflict resolution strategy** to use
  - **Policy conflict**: 2 or more policy assertions are triggered, but only 1 can be executed (e.g., skip activity vs. replace activity)
- WS-Policy4MASC advantage is in modeling policy conflict resolution **strategies that maximize sums of various business value types** associated with consequences of choosing 1 of the policies
  - Model diverse business strategies (e.g., exceptional customer satisfaction)
  - MASC implements corresponding algorithms

Policy Conflict Resolution

- Strategies are also **classified along several dimensions**, depending on business value classes
  - E.g., ‘tangible-only’ vs. ‘intangible-only’ vs. ‘tangible+intangible’
- When difference between summary business values for conflicting policies is smaller than some threshold, **tiebreaking rules** can be specified
  - E.g., ‘tangible+intangible’ instead of ‘tangible-only’
- **Combinations determine strategies**
  - E.g., ‘intangible-only agreed+possible benefits+ costs with tiebreaking tangible+intangible’ (models maximization of customer satisfaction)

MASC Middleware Architecture

- **Supported only by MASC middleware, so limited in practice** by MASC characteristics:
  - Current applications only for Web services
  - MASC prototype is .NET-centered
  - MASC has autonomic computing features, but many improvements in this direction are possible
- **On the other hand**, WS-Policy4MASC (in principle):
  - Can be easily applied to other domains
    - Is independent of the implementation platform
    - Is intended for business-driven management with minimal human intervention (autonomic)
Adaptive Server Framework (ASF)

- Developing adaptive systems is difficult
  - Adaptive logic is determined by application’s requirements for adaptation (application-specific)
  - Developers need appropriate support
- ASF provides basic components and services for developing customized adaptive applications
  - Application-specific and platform-specific adaptability
  - Separation between implementation of adaptive behavior from server application business logic
  - Layered architecture (supports legacy systems)
  - Standards based instrumentation and management
  - Low overhead and extensibility

ASF Architecture Principles

- Invocation ASF Analysers
  - ASF service layer
  - Standard-Standard -based Management Layer (JMX/ARM/WSDM)

Case Study: Adaptive Image Server

- Simulating TPC-W image servers
  - Based on server workload and network speed, adaptively return images at different levels of resolution and quality to both meet client requirements and optimize performance under peak load

Use of Interception

- ASF modules (components) are deployed as interceptors
  - 'plug-and-play' on multiple platforms
    - Net (ASF.Net and WCF)
    - Axis 2 SOAP engine
    - JBoss App Server
    - Main ESB

Derived Control Loops

- Embedded on an artificial performance model
Case Study Implementation

- Case study implemented using ASF and deployed on JBoss Server 4.0.1 with JRE 1.5.
- Two control loops with eight main customized adaptive components:
  - ImageScaleEngine
  - BandwidthSensor
  - ImageScaleMonitor
  - ImageScaleAnalyzer
  - ImageScaleEffector
  - CPUSensor
  - ConfigurationMonitor
  - ConfigurationEffector

Evaluation

<table>
<thead>
<tr>
<th>ImageSize</th>
<th>Non-adaptive TPS</th>
<th>Adaptive TPS</th>
<th>% Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small (10KB)</td>
<td>54.25</td>
<td>80.12</td>
<td>46.48%</td>
</tr>
<tr>
<td>Small/Medium (10~100KB)</td>
<td>5.78</td>
<td>8.85</td>
<td>54.84%</td>
</tr>
<tr>
<td>Medium (100~200KB)</td>
<td>1.84</td>
<td>3.10</td>
<td>66.70%</td>
</tr>
<tr>
<td>Large (500KB~2MB)</td>
<td>0.25</td>
<td>0.31</td>
<td>24%</td>
</tr>
</tbody>
</table>

Some ASF Limitations

- Current specification of policies in ASF is limited
  - Design goal was simplicity
  - Many details (and special cases) about management goals and actions cannot be expressed
- No support for business-driven management
  - Current focus is on maximization of performance
  - Conflicting policies selected based on priorities
- But: complements WS-Policy4MASC limitations
  - Has broader applicability (not only Web services)
  - Platform independent (J2EE, .NET)
  - Stronger support for autonomic computing

Adding WS-Policy4MASC to ASF

- Goal: leverage compatibilities/synergies to better support business-driven autonomic management
  - For service-oriented (but also other) systems
- Work in progress!
- WS-Policy4MASC is extended with application server configuration actions (easy to do)
- ASF is extended with modules for:
  - policy storage (reused from MASC – possible due to loose coupling in both MASC and ASF)
  - policy conflict resolution (reused from MASC)
  - policy application (new modules are needed)

Conclusions and Future Work

- Change is inevitable, adaptability (with minimal human intervention) is needed! => Autonomism
- Execution of IT systems should be aligned with business objectives, values, and operations
- ASF is a middleware framework to make building application-specific adaptivity simpler
- Adding WS-Policy4MASC to ASF is a step towards business-driven autonomic computing
- The immediate future work is implementing the new ASF modules to support WS-Policy4MASC policy application (execution)

Discussion

- Questions?
- Criticisms?
- Insights?
- Contact information:
  - vladat@server.computer.org (please, start Subject line with “Your Research” or “IGI Book”)

7/21/2009
**Call for Book Chapters**

- "Information Technology Aligned with Business Objectives and Values: Integrating Software Engineering, System Management, and Governance", edited by Vladimir Tosic
  - Value-based software engineering (VBSE)
  - Business-driven IT management (BDIM)
  - IT governance
  - Integration of results from these 3 areas
- Chapter proposal deadline: March 31, 2008
- Chapter submission deadline: April 30, 2008

**Use of WS-Policy4MASC in MASC**

- Feasibility of WS-Policy4MASC demonstrated by implementing MASC in-memory policy repository, algorithms for policy decision, and other modules
  - Used ‘XML schema to classes’ generator for policy repository classes
  - Examined expressiveness, effectiveness, and usefulness of WS-Policy4MASC on 2 case studies (weather report, stock trading)
  - To adapt a Web service composition at run-time, only change WS-Policy4MASC policies

**Integrating Design- and Run-Time**

- Linking all life-cycle stages with bi-directional information transfer, e.g., for impact analysis
- New closed-control loops for adaptation
  1. **Engineering** IT systems during design-time to maximally support run-time management
  2. **Managing** IT systems during run-time to align with and leverage business value information captured in augmented design-time models
  3. **Feeding back** monitored run-time information about business value compliance and various changes to improve/update design-time models

**UML Profiles for WS-Policy4MASC**

- An approach to VBSE+BDIM integration
  1. **Engineering**: Specification of business value (and other non-functional information) in UML models
     - E.g., using Eclipse plug-ins
  2. **Management**: Generation of WS-Policy4MASC policies from UML models (semi-automatically)
     - E.g., using XSLT on XMI representations of UML
  3. **Feedback**: Showing run-time management information (e.g., measurements) in UML models (cf. dashboards) and using it for various analyses
     - E.g., using Eclipse plug-ins

**UML Profile for Policy Assertions**

- An approach to VBSE+BDIM integration

**Example Use of This UML Profile**

- Part of a case study used to evaluate our UML profiles
- Ongoing work: Tool support for showing run-time management information and corresponding analyses
Why Adaptive Middleware?

- Separation of Concern
- Business logic and are loosely-coupled/independent
- Easy to reconfigure and incorporate new adaptation
- Policy-driven behavior dynamically adapts application to environmental changes
- Adaptive middleware provides programming models and services

Adding Adaptation to Software

Scientific Challenges

- The creation of adaptation capabilities presents the following scientific challenges:
  - the invention of predictive models
    - to be solved to produce optimal parameter settings for controlling various aspects of application configuration and behavior.
  - the invention of new extensible software architectures
    - to provide flexible application server monitoring and feedback mechanisms, with minimal intrusion.
  - the invention of mechanisms to enable dynamic monitoring and adaption policies
    - reflection mechanisms
    - policies can be specified declaratively by the application designer.

ASF Architecture Overview

ASF Components and Interfaces

1. Introduction to Adaptive Middleware
2. Scientific Challenges
3. ASF Architecture Overview
4. ASF Components and Interfaces
5. Conclusion and Future Work
Observations

Example Application of ASF

- Enabling Adaptive Web services for Mobile Applications
  - Rendering images based on mobile application constraints: connection speed & screen resolution

Layered Architecture of ASF

- Control Flow
- Data Flow