Overview of presented papers

Tomáš Bureš

DISTRIBUTED SYSTEMS RESEARCH GROUP
http://nenya.ms.mff.cuni.cz

CHARLES UNIVERSITY PRAGUE
Faculty of Mathematics and Physics
The Workshop

- Held on September 5-6 in Lisbon, Portugal
- Co-located with ESEC/FSE’05
- Organized by
  - Elisabetta Di Nitto
    - Dipartimento di Elettronica e Informazione, Politecnico di Milano, Italy
  - Amy Murphy
    - Department of Informatics, University of Lugano, Switzerland
The Program – “Tracks”

- Adaptation in Middleware
- Publish/Subscribe Middleware and Models
- Security in Middleware
- Testing and Instrumentation
- Exploiting Components
- CORBA
- Connecting Middlewares

- 15 papers including our paper
A Middleware for Autonomic QoS Management based on Learning

- Patrice Vienne and Jean-Louis Sourrouille, PRISMa, INSA Lyon (France)

Use reinforcement learning to decide the QoS level of an application

- Application parted to tasks,
- Each task can run at different QoS levels
- After task is executed perceptions and monitors are collected
- Based on the monitors and perceptions the QoS level for the next action is decided
• QoS statistics
  ▪ Perception
    • relevant information to specify the system state (e.g., resource availability)
  ▪ Monitor
    • relevant information to assess the system ability to provide the required services, i.e., its utility (e.g., the served request proportion of a data server)

• Pros:
  ▪ system is self-adapting

• Cons:
  ▪ system is a bit unpredictable, the performance is degraded at the beginning
Adaptation in Middleware

• A Middleware-Independent Model and Language for Component Distribution
  - Hans Albrecht Schmid, Marco Pfeifer, Thorsten Schneider, *University of Applied Sciences Konstanz (Germany)*
  - Created a distributable component models on the top of CompJava
  - Distinguish
    - *non-remotable and remotable* component types
    - *local* and *distributed* component implementation
• Generate RMI proxies based on the component architecture description
• Optimized for performance
  ▪ proxies are used only if a link crosses the distribution boundary
• Striving for Versatility in Publish/Subscribe Infrastructures
  ▪ Roberto Silveira Silva Filho and David F. Redmiles, School of Informatics, UC Irvine (USA)

• Presents YANCEES Framework
  ▪ An experimental framework designed to provide versatility to publish/subscribe architectures
Figure 1. YANCEES general approach summary

Figure taken from: Silva Filho, R. S., Redmiles, D. F.: Striving for Versatility in Publish/Subscribe Infrastructures
Striving for Versatility in Publish/Subscribe Infrastructures

Figure 2. General view of the YANCEES framework.

Tomáš Bureš
DSRG Seminar 10/04/05

Figure taken from: Silva Filho, R. S., Redmiles, D. F.: Striving for Versatility in Publish/Subscribe Infrastructures
• Uncertain Event-Based Model for Egocentric Context Sensing
  - Mauro Caporuscio, Paola Inverardi, *University of L'Aquila (Italy)*

  - Filtering events using fuzzy logic formulas.
  - Quite formal, details can be found in the paper.
Secure Event Types in Content-Based, Multi-Domain Publish/Subscribe Systems

- Lauri Ilja Waltter Pesonen, Jean Bacon, *University of Cambridge, Computer Laboratory (UK)*

- They aim to prevent:
  - forged types, where a malicious party tries to disrupt the data flow by introducing a type definition to the system with the same name as another legitimate type
  - tampered types, where a malicious party changes an existing type definition
  - accidental name collisions, where two parties both introduce a new type to the system with the same name
Secure event types

1. Type Issuer’s Public Key: \( P_{ti} \)
2. User-Friendly Name: \( n \)
3. Version Number: \( v \)
4. Attributes: \( A \)
5. Delegation Certificates: \( D \)
6. Digital Signature: \( s \)
Using Parse Tree Validation to Prevent SQL Injection Attacks

- Gregory Buehrer, Bruce W. Weide, Paolo A.G. Sivilotti, The Ohio State University (USA)

Problem: Web-applications vulnerable to SQL-injection attacks.

When SQL statements constructed as
- q = “SELECT * FROM reports WHERE id=” + id;

Using dynamic queries helps, but is inconvenient
- q = “SELECT * FROM reports WHERE id=?”;
The solution:

- \( q = \text{SQLGuard.init}() + \text{"SELECT * FROM reports WHERE id="} + \text{SQLGuard.wrap(id)}; \)
- \text{SQLGuard.init} generates a unique identifier used to delimit the id’s
- \text{SQLGuard.wrap} wraps the id using the unique identifier
- They compare the parse trees with/without the unique identifiers. If they do not match, there is a threat of SQL injection attack.
• Experiences in Coverage Testing of a Java Middleware
  - Mehdi Kessis\textsuperscript{1}, Yves Ledru\textsuperscript{2}, Gerard Vandome\textsuperscript{3}
  - \textsuperscript{1} France Telecom R&D-ObjectWeb (France), \textsuperscript{2} LSR/IMG (France), \textsuperscript{3} Bull/ObjectWeb (France)
  - They took JonAS server as a case study and evaluated the code coverage of JonAS test suite.
Experiences in Coverage Testing of a Java Middleware

Figure taken from: Kessis, M., Ledru, Y., Vandome, G.: Experiences in Coverage Testing of a Java Middleware
• Conclusion:
  “Following our study we noticed that reaching 100% of code coverage in large scale applications is nearly impossible even with the simplest coverage metric (line coverage). In fact, in practice, some parts of code concern debug mode, exception, monitoring possibility, require supplementary test efforts. ... Generally, developers judge these parts not worth to test. Testing such pieces of code can increase radically the cost of the maintenance phase and it is not recommended to adopt it as an initial goal.”
• Portable, Efficient, and Accurate Sampling Profiling for Java-Based Middleware
  ▪ Walter Binder, EPFL (Switzerland)
  ▪ Use byte-code counting for profiling
  ▪ Do byte-code manipulation to augment the code to do byte-code instructions counting
  ▪ After a specific amount of byte-code instructions, the snapshot of call-stack is taken
  ▪ Reproducible for deterministic applications
  ▪ No need for special JVMs or system-specific code
Exploiting Components

• Dynamic Adapter Generation for Data Integration Middleware
  - Peng Gong\textsuperscript{1}, Ian Gorton\textsuperscript{2}, David Dagan Feng\textsuperscript{1}
  - \textsuperscript{1}School of Information Technologies, the University of Sydney (Australia), \textsuperscript{2}Empirical Software Engineering, National ICT (Australia)

  ▪ Tried to address uniform access to different data sources
    • database, FTP, ...
  ▪ Created an ontology assigning semantics to data source API
Exception Handling in Component Composition with the Support of Middleware

- Yaodong Feng, Gang Huang, Yali Zhu, Hong Mei, School of Electronics Engineering and Computer Science, Peking University (China)

Problem: Handling exceptions when composing components together.
• Process:
  1. Assemble application of components ignoring the exceptions
  2. Identify exceptions thrown
  3. Remove from the list the exceptions handled
  4. For exceptions not handled, employ strategies to the middleware
     ▪ e.g., replication, replacement, reboot, ...

    Exceptions typically cause other exceptions, so it is usually enough to handle the “core” exceptions
Optimizing Layered Middleware

- Omer Erdem Demir\(^1\), Eric Wohlstadter\(^2\), Stefan Tai\(^3\), Prem Devanbu\(^1\)
- \(^1\) UC Davis (USA), \(^2\) Univ. British Columbia (Canada), \(^3\) IBM Thomas J. Watson Research Center (USA)

- Allow parts of an application to be executed in lower layers of a middleware
  - request forwarding
  - simple application functionality
  - login rejection, etc.
Optimizing Layered Middleware

Figure taken from: Demir, O. E., Wohlstadter, E., Tai, S., Devanbu, P.: Optimizing Layered Middleware
• A Flexible and Extensible Object Middleware: CORBA and Beyond
  - Franz J. Hauck¹, Rudiger Kapitza², Hans P Reiser¹, Andreas I. Schmied¹
  - ¹ Distributed Systems Lab, University of Ulm (Germany), ² University of Erlangen (Germany)

  • Introduce *portable profile managers* to CORBA to allow for features such as
    * partitioning, fault-tolerance, QoS guarantees
    * simple interoperation with non-CORBA middleware platforms
The CORBA Connector Model

- Sylvain Robert¹, Ansgar Radermacher¹, Vincent Seignole¹, Sebastien Gerard², Virginie Watine¹, Francois Terrier¹
- ¹ CEA List (France), ² Bull/ObjectWeb (France)

- Extended CCM and OMG D&C of a notion of explicit connector
The CORBA Connector Model

• Changes to CCM
  ▪ omitted message sinks and sources
    • modeled by a special connector
  ▪ connectors parted to fragments
    • similar to our connector units
  ▪ connector homes for connector creation
  ▪ fixed and template connectors
    • template connectors need a code generator to generate code for a particular connector instance

• Changes to OMG D&C
  ▪ added a connector meta-element in the deployment artifacts
• Hybrid Service-Oriented Architectures: A Case Study in the Automotive Domain
  - Luciano Baresi, Carlo Ghezzi, Antonio Miele, Matteo Miraz, Andrea Naggi, Filippo Pacifici, Politecnico di Milano (Italy)

  - A case study of a JXTA ↔ Jini (JavaSpace) bridge they have developed
    - JXTA is a set of protocols that allow devices to communicate in a peer to peer way.
    - JavaSpace is a Linda-like tuple space.
Conclusion

• There was no single thread of the workshop

• From the discussions it was apparent that
  ▪ connecting different middlewares, integration of data sources, etc... is still a difficult problem
  ▪ people were not completely sure whether they need components
  ▪ services are an emerging trend