Peking
University
Application
Server

PKUAS
Middleware for Component based Systems

Gang Huang
School of Electronics Engineering and Computer Science
Peking University, Beijing, 100871, China

Oct. 27, 2005

Peking
University

Agenda

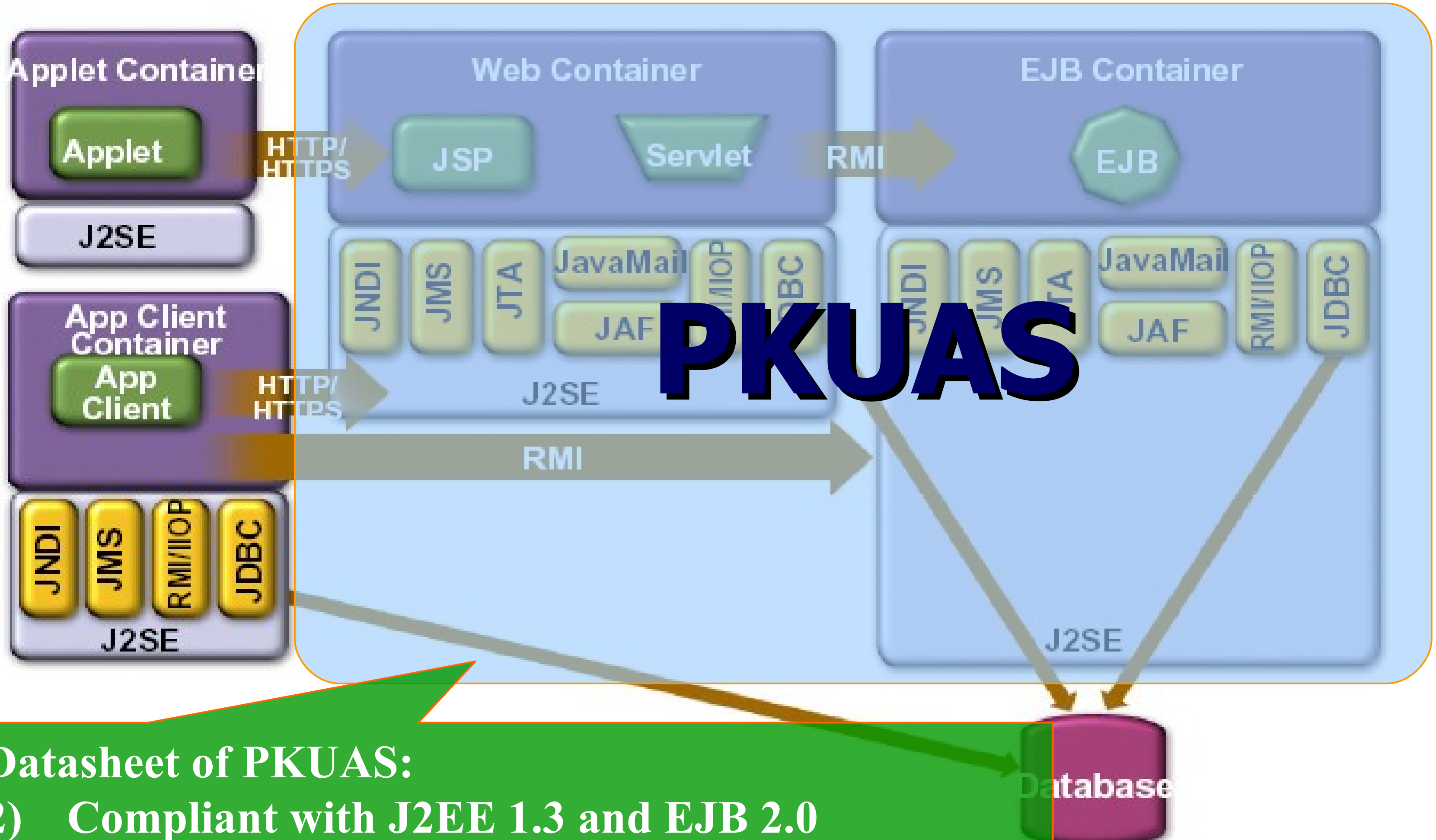
- **Overview of PKUAS**
  - Motivation
  - Implementation
  - Applications
- **Research Roadmap**
  - Customizability and exte
  - Architecture based reflect
  - Self-adaptation
- **Other Work**
  - Collaboration with Objec Innovation Transfer)
  - Possible collaborating poi

We have multiple relative independent teams for research and engineering about middleware:
→22 people for PKUAS engineering and application (open source)
→10 people for PKUAS research
→7 people for agent@PKUAS
→11 people for software architecture and component model

SI@PKU definitely emphasizes on research and this talk also discusses research topics other than open sources

Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

PKUAS: A J2EE Application Server

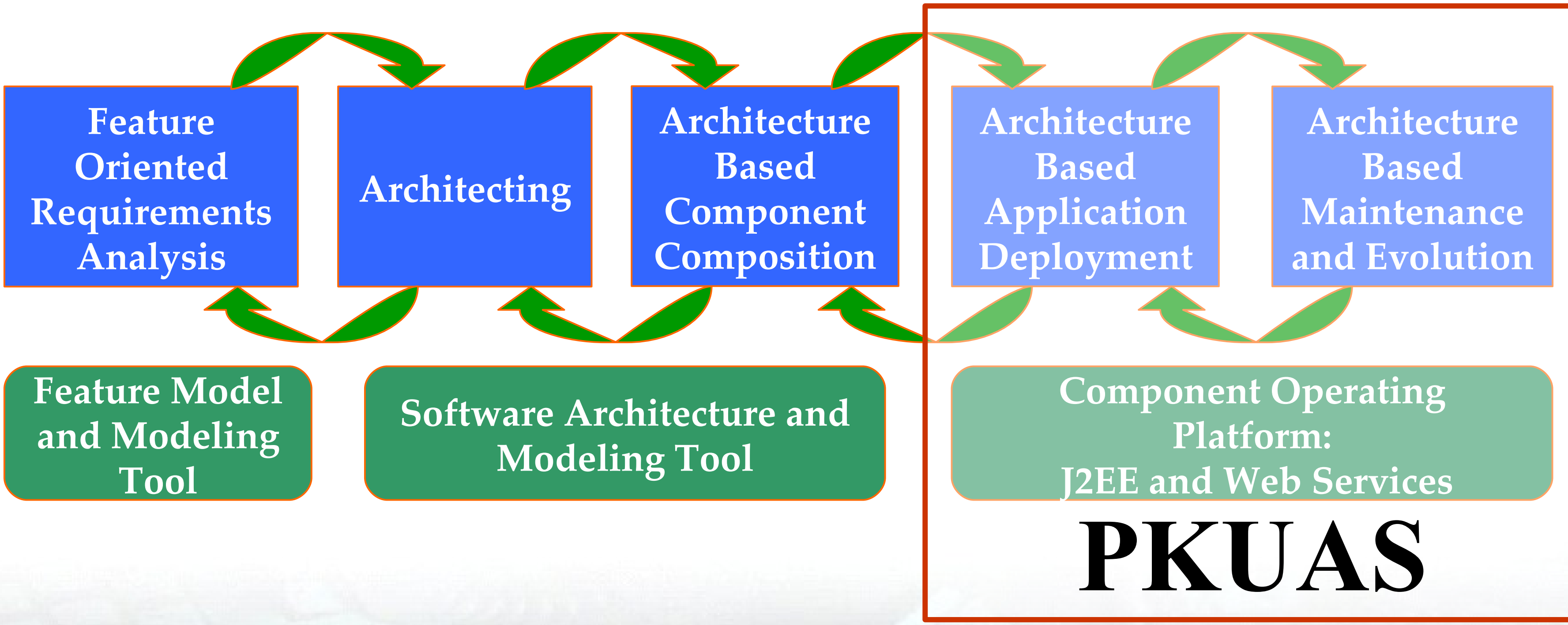


Datasheet of PKUAS:

2) Compliant with J2EE 1.3 and EJB 2.0
3) Commercial and practical applications
4) Distinguished and innovative features
5) Sponsored by 863, 973, NSFC …
6) 3/2001, 8/2002, 12/2003, 20/2004, 30/~ developers

PKUAS: Academic Value

- **Infrastructure for software engineering research**
  - ABC (Architecture Based Component Composition)
- **Test-bed for distributed computing research**
  - Configurable, reflective, self-adaptive middleware

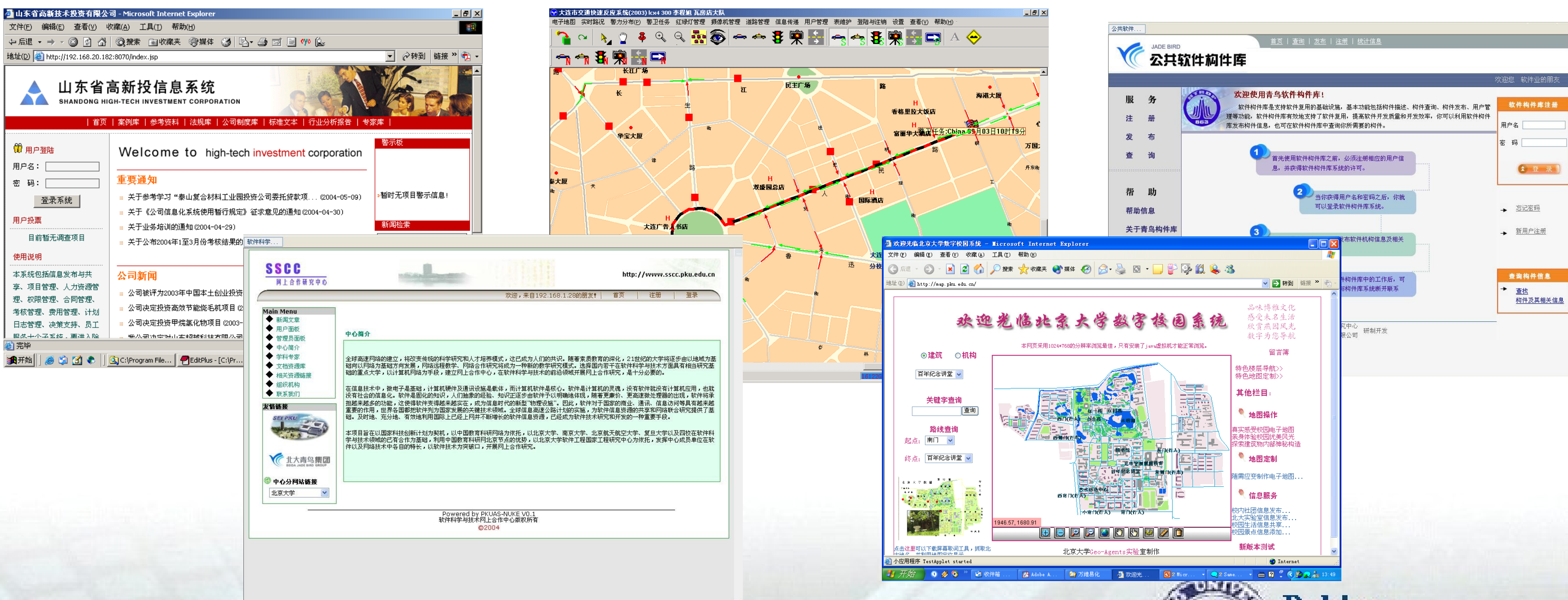


Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

Peking University

PKUAS: Industrial Value

- **Technical support for Chinese software industry**
  - Commercial applications
    - Finance, transportation, education, government …
  - A part of Orientware
    - Chinese Middleware Suite (Issued by Chinese High-Tech Development Program)



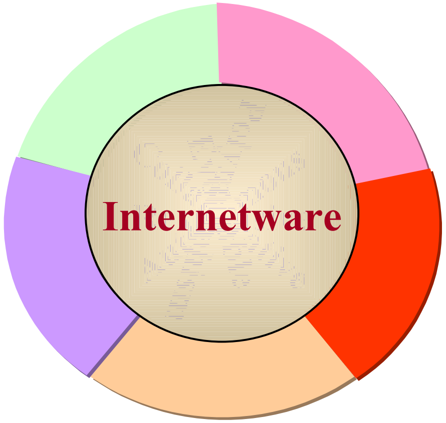
Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

Peking University

PKUAS: Research Roadmap

Usability

Self-Optimization Self-Organization And Self-Healing — **2005~**



Deployment and Reflection based on Architecture — **2003~2005**

Customizability and Extensibility by Componentization — **2001~2003**

Manageability

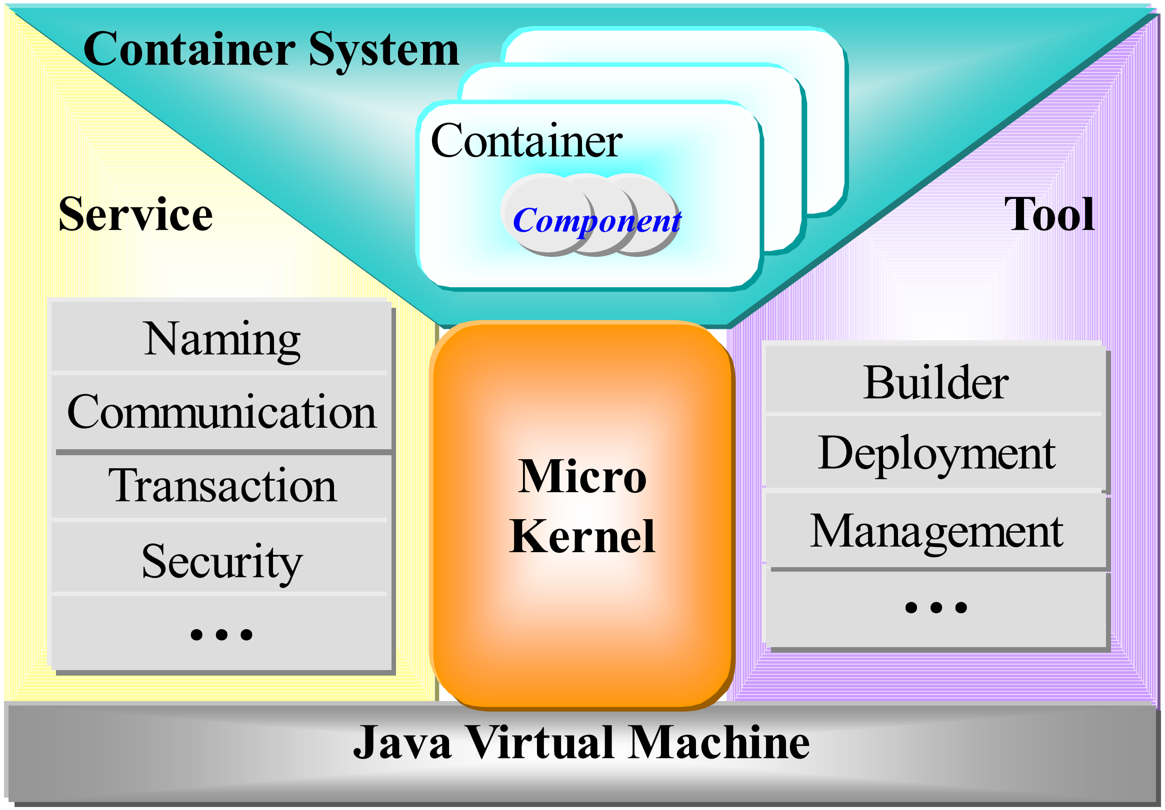
Improve the usability and manageability for coping with rapid and continuous changes in the open and dynamic Internet

Response to change *Time to market* *Return of investment* *Cutting edge*

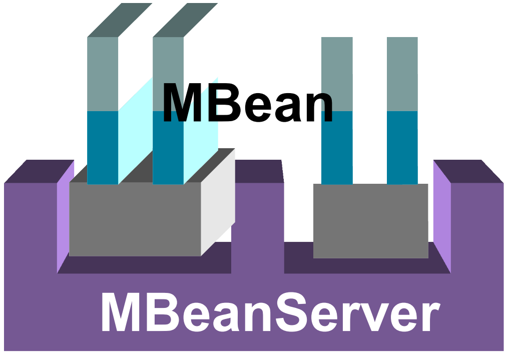
Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

Peking University

Componentized Architecture



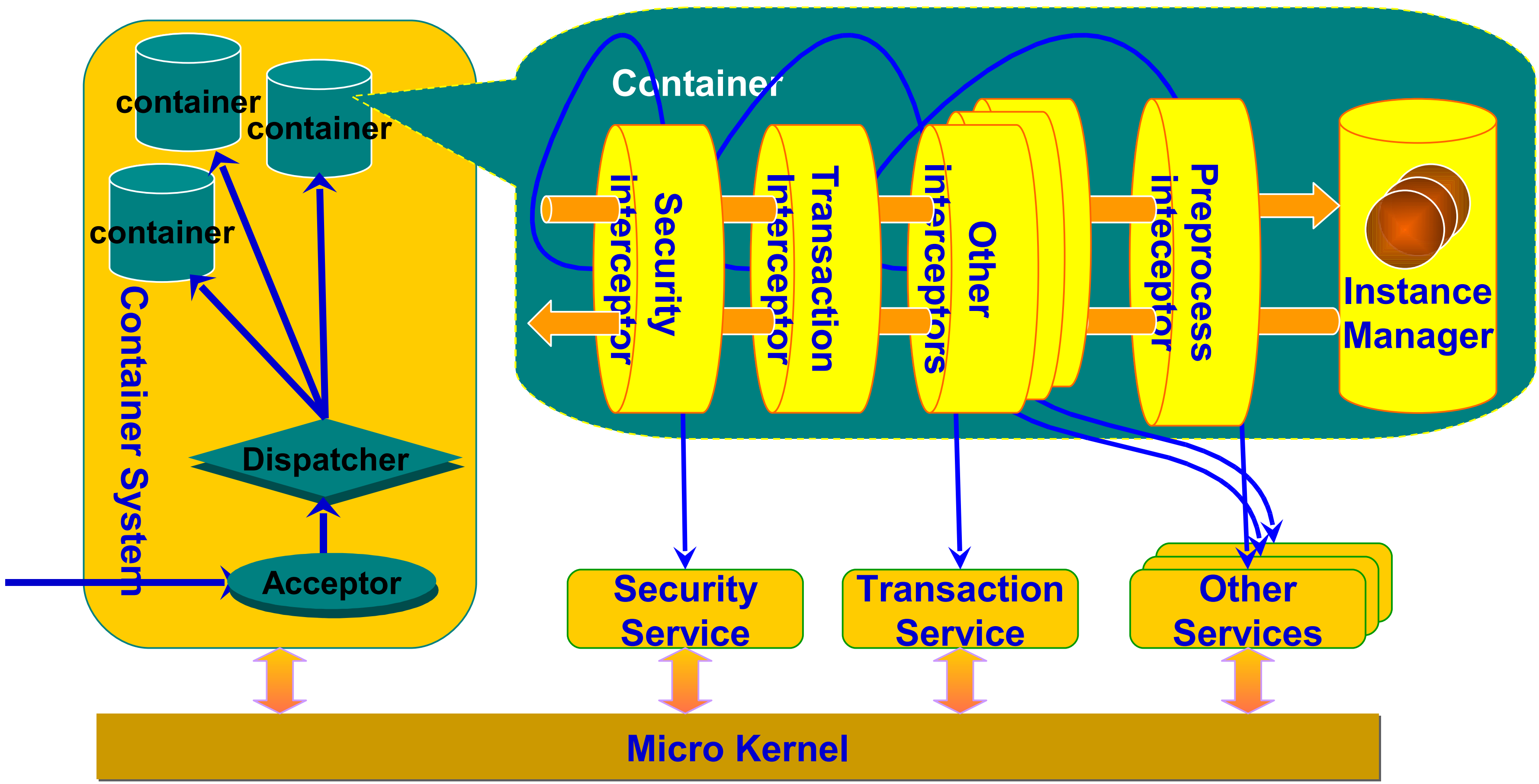
Investigate
→TAO/ZEN
→J2EE RI
→JBoss
→OpenEJB



- Componentized PKUAS
  - PKUAS not only supports component based systems (J2EE), but also implement itself as a component based system
  - Employ JMX: the standard management in Java

It seems compliant with exo-kernel ☺

Containers and Services



One container holds one EJB vs. One container holds all EJBs

→One-One helps to fine-grained management

→Application interference problem

Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

Peking University

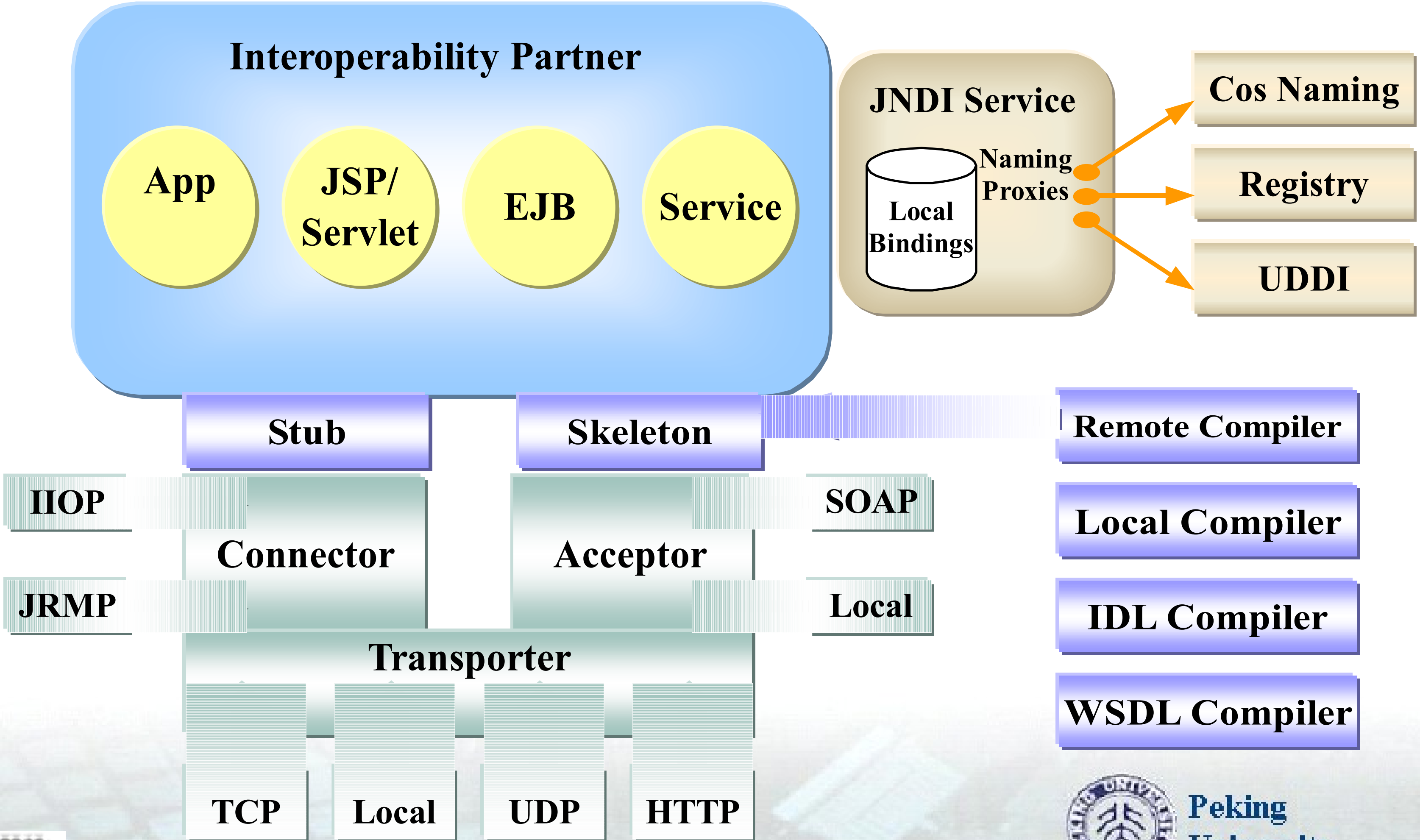
Interoperability

On Demand of Clients

Similar to CAROL@JOnAS with some differences

→One PKUAS client can use more than one protocol at the same time

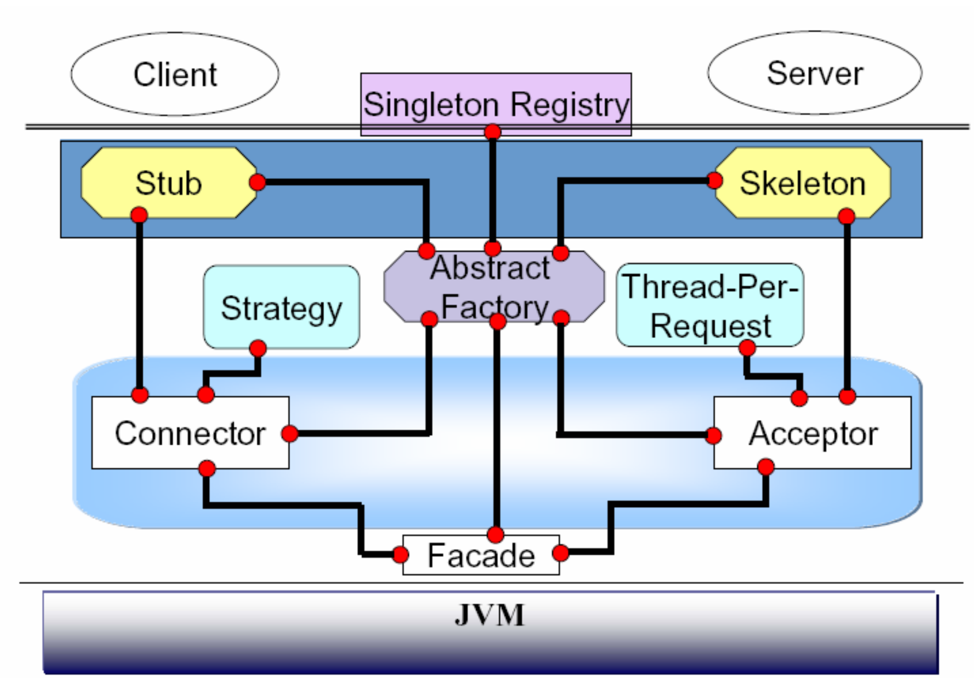
→CAROL is bound with RMI while PKUAS supports more (like EJBLocal)

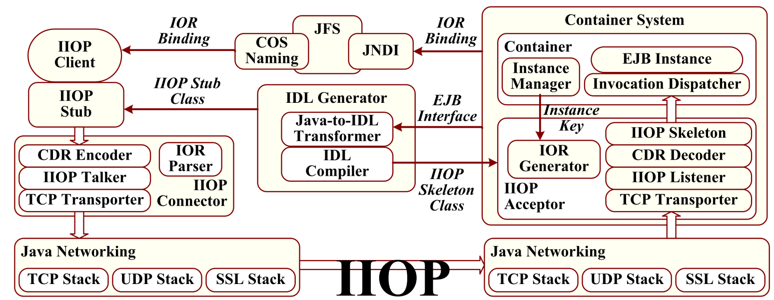


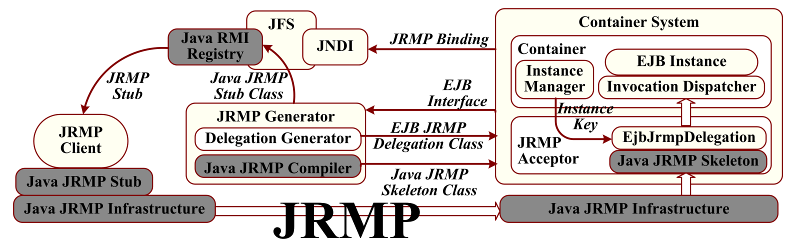
Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

Interoperability

Pattern Oriented Framework







- Patterns in PKUAS Interoperability
  - Cope with changes (of time and space)
  - Easy to understand and reuse
  - Similar to Douglas Schmidt's patterns in TAO with some differences
    - Façade for pluggable transport protocols
    - Singleton Registry for multiple interoperability protocols

Fractal also emphasizes on pattern oriented framework

Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

Peking University

Publications on Componentization

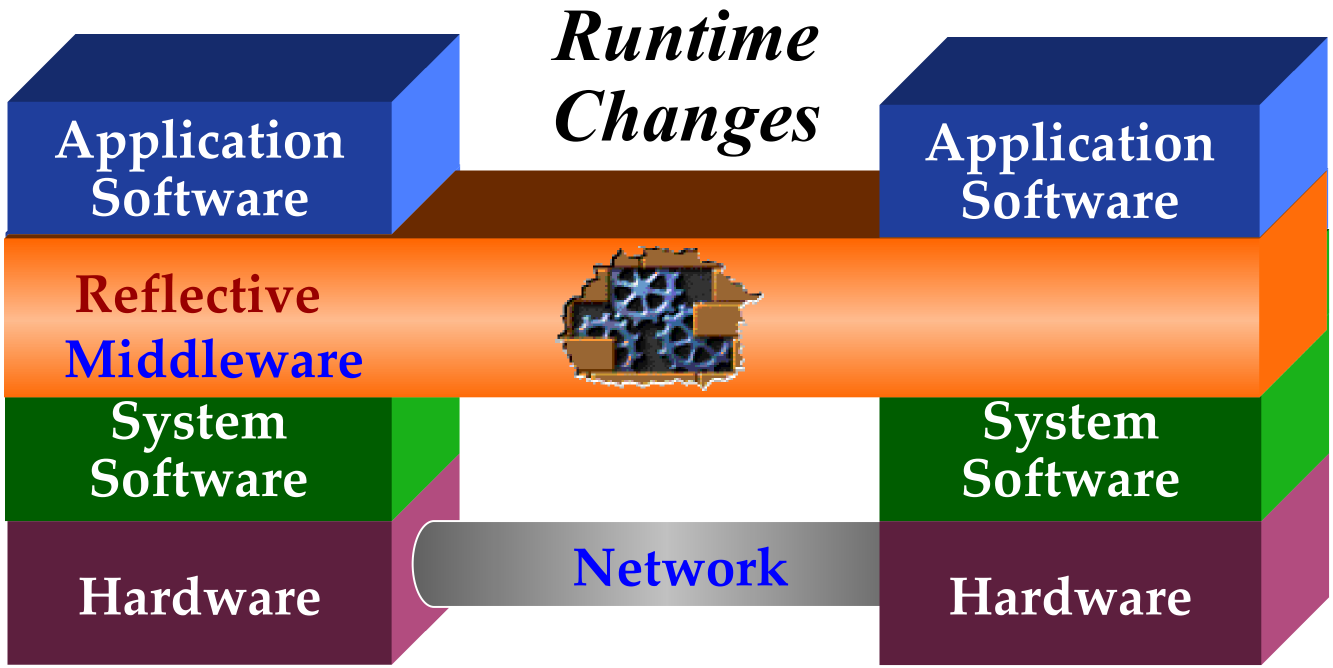
English Papers

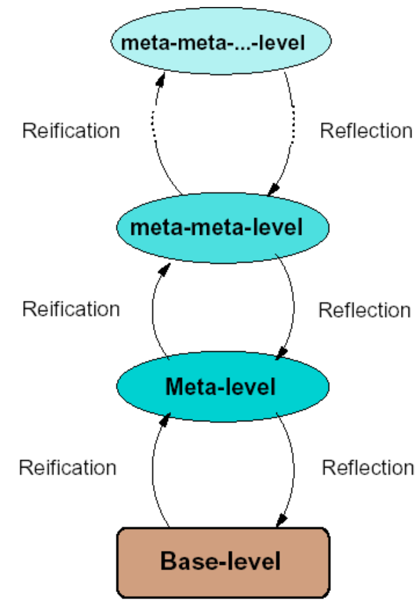
1. An application server to support online evolution. International Conference on Software Maintenance, 2002, pp. 131 – 140
2. A Systematic Approach to Composing Heterogeneous Components. Chinese Journal of Electronics, Vol. 12, No. 4, 2003, pp. 499-505.
3. Microkernel Architecture: Making Application Servers Open to Change, Chinese Journal of Electronics, Vol.14, No.3, July 2005, pp443-448

Others

- 8 Chinese Papers
- 3 Chinese Patents

From Customization to Reflection





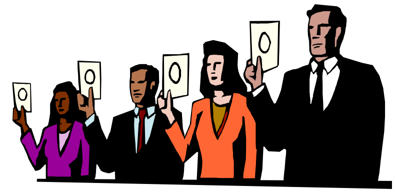
- Customizable PKUAS
  - Only supports changes pre runtime, but the open and dynamic Internet makes runtime changes frequent, rapid and continuous
- Reflective Middleware
  - Open up the internal implementation

Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

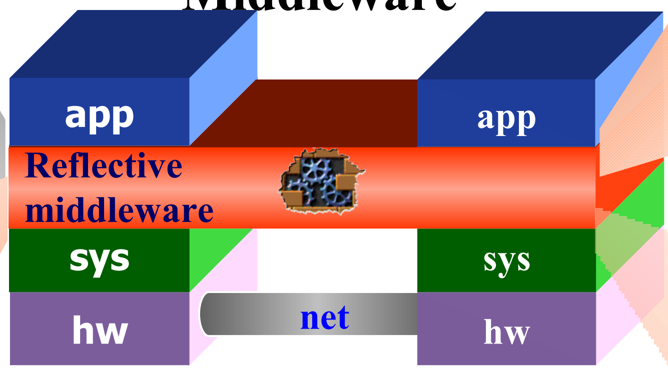
Peking University

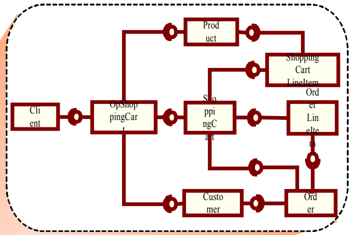
Limitations of Current Reflective Middleware

Irrelative fragments

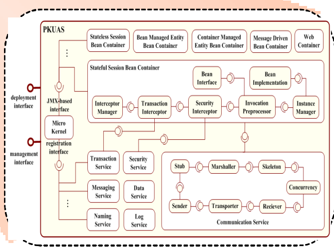


Users of Middleware





Application SA



Platform SA

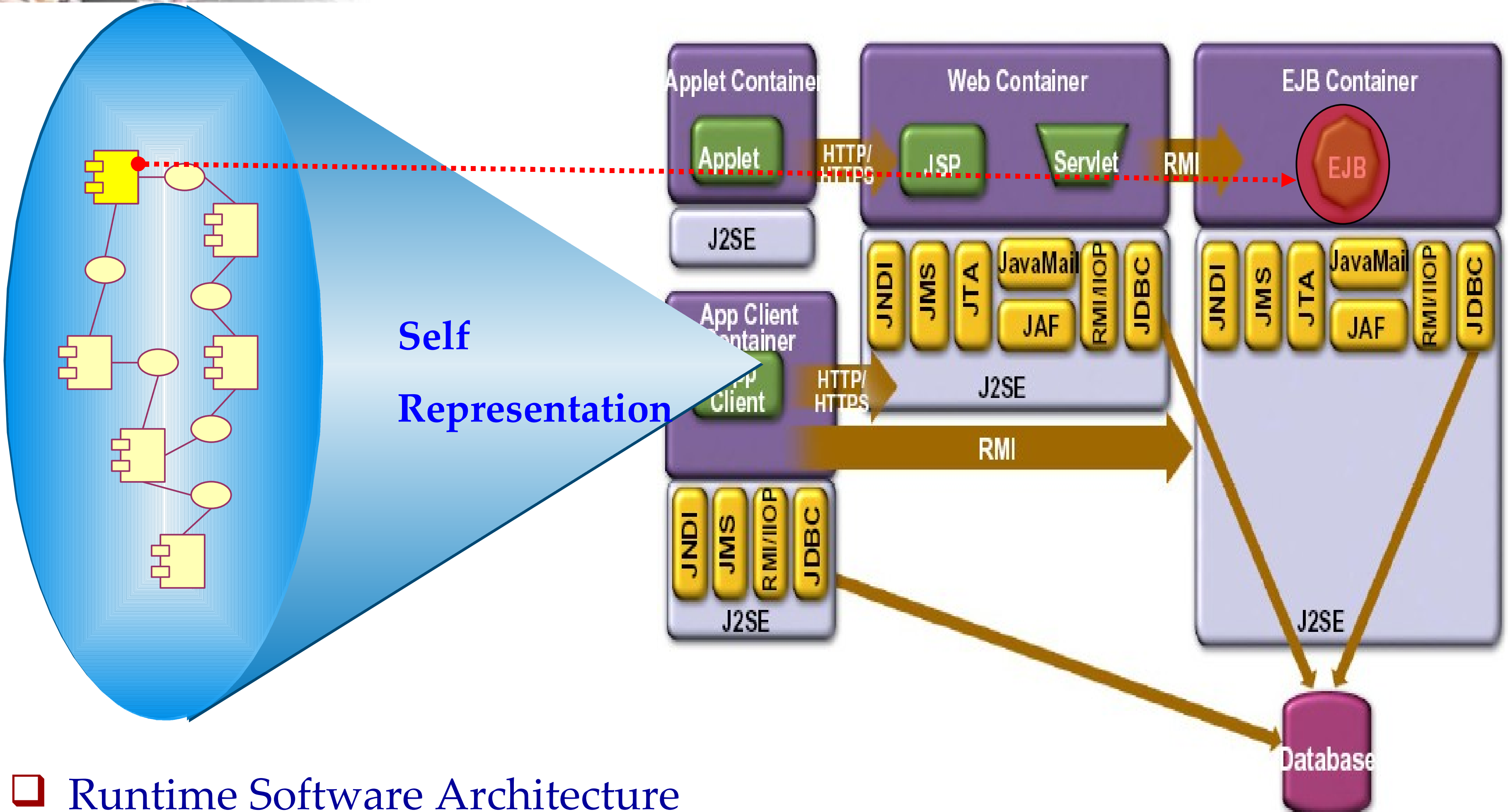
Introducing Software Architecture

Uniform, understandable, easy-to-use

Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

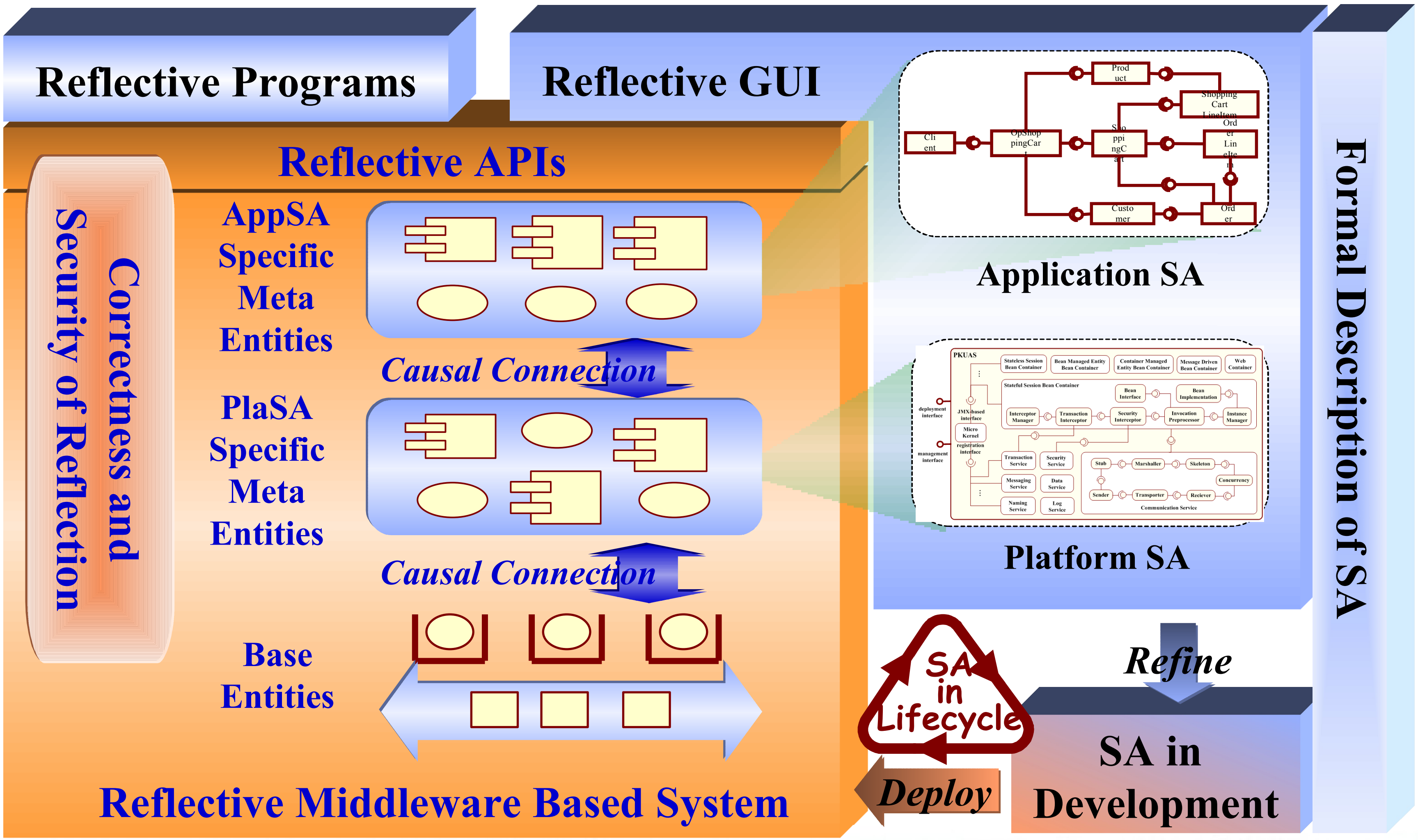
Peking University

Architecture based Reflection



- Runtime Software Architecture
  - A model representing a runtime system as a set of architectural elements which are causally connected with the internal states and behaviors of the runtime system

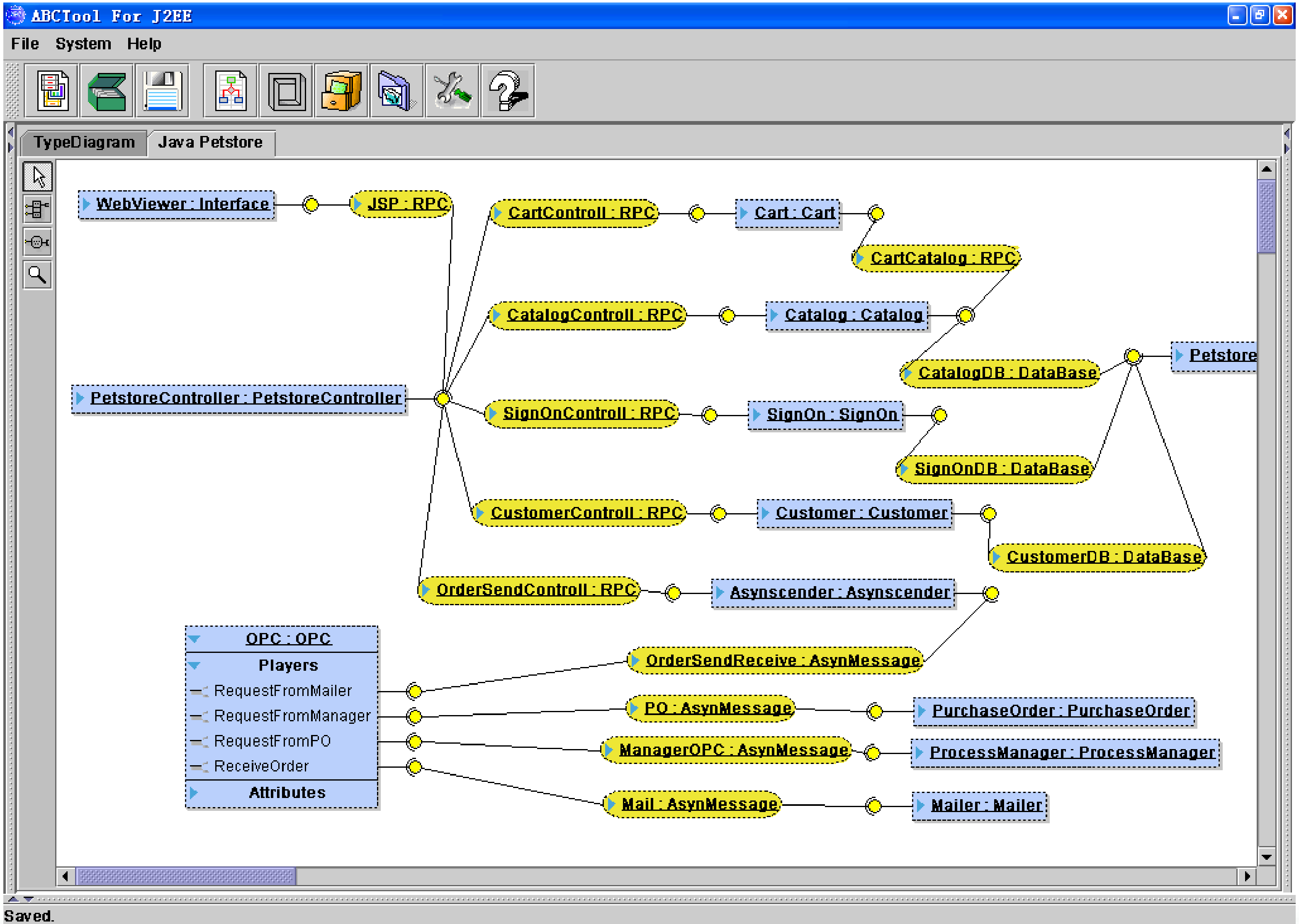
Architecture based Reflective Framework



Fractal: reflective component model
(PKUAS is reflective component framework)

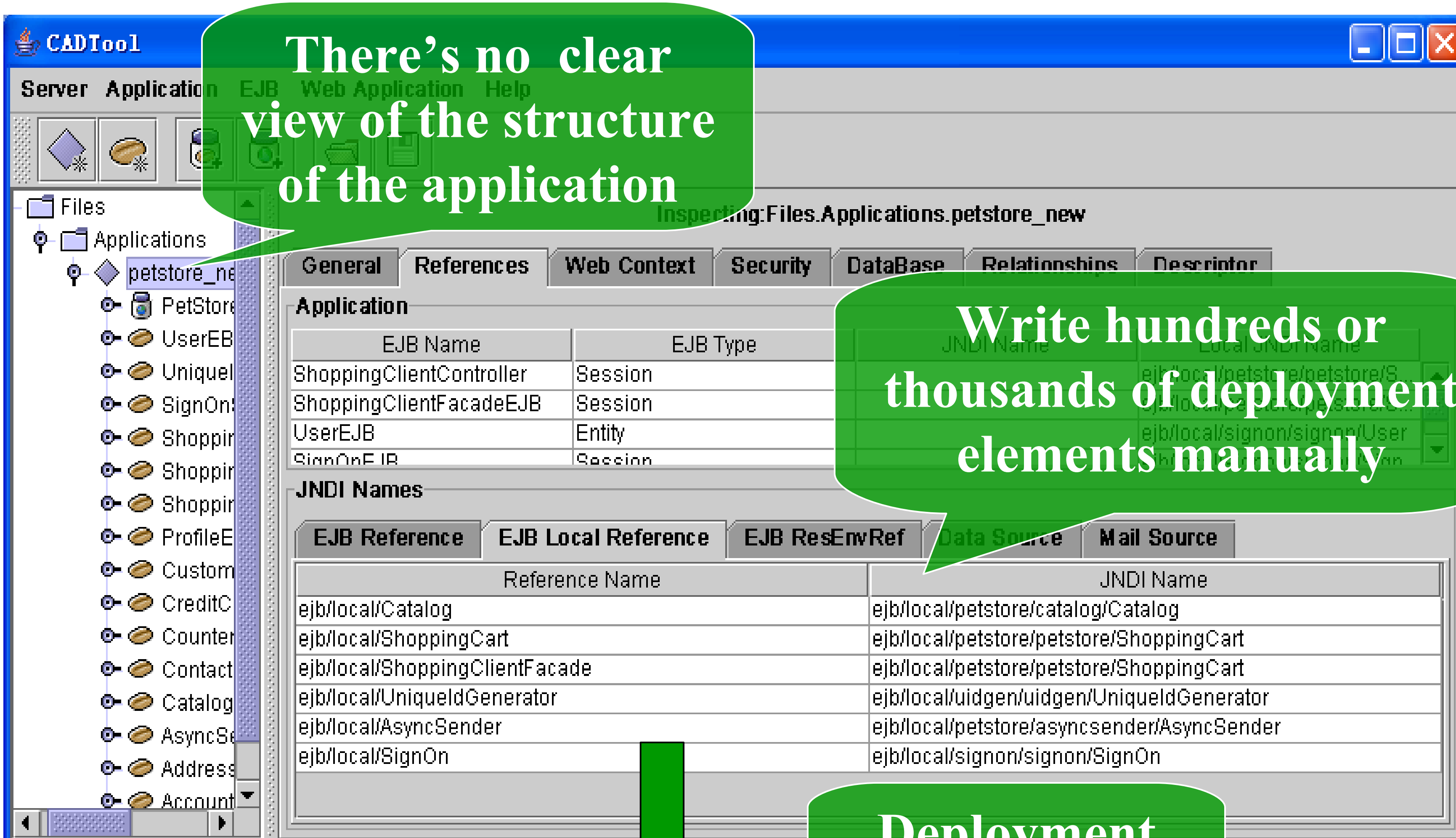
Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

Architecture Modeling and Managing Tool



Ideally, the architecting tool (ABCTool) can be reused for architecture based management (PKUAS)

J2EE Deployment Tool



There's no clear view of the structure of the application

Write hundreds or thousands of deployment elements manually

Deployment node by node

Visualization of architecture models in the development

Automatic calculation of deployment factors

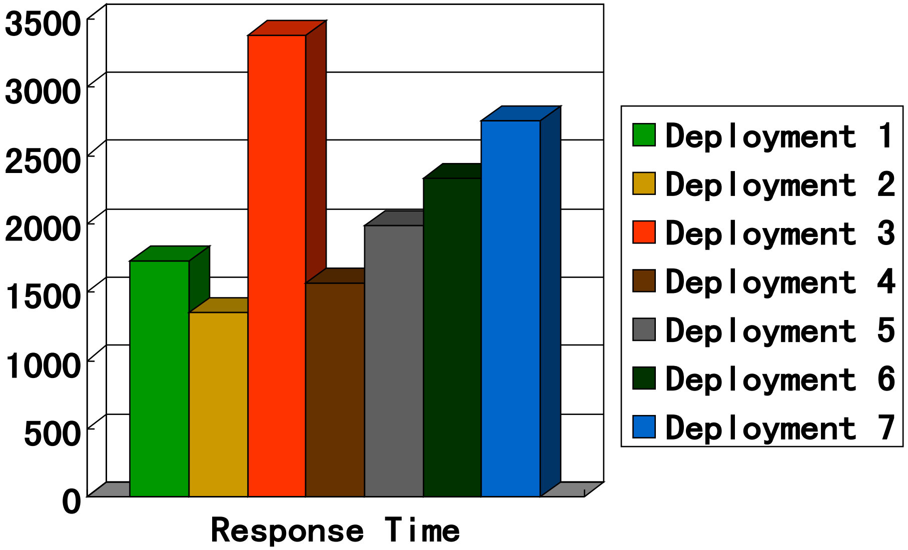
Drag-and-drop deployment of components

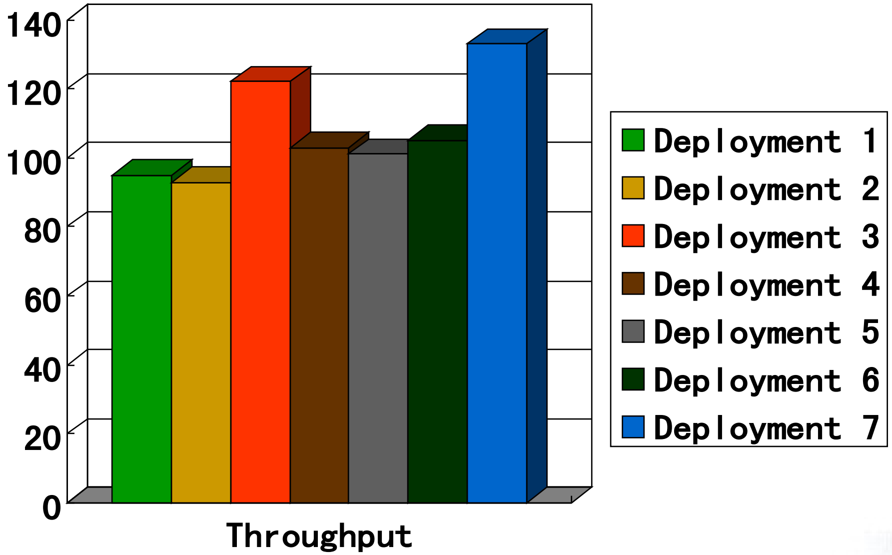
Visualization of servers and their capabilities

Fractal: software deployment and configuration management

Case Study: RUBiS

- **RUBiS**
  - An eBay-like bidding system with a set of Servlets and 17 session EJBs
  - Different deployments on 3 servers have different response times and throughputs





RUBiS is a popular sample for ObjectWeb ☺

Publications on Reflection and Deployment

English Papers

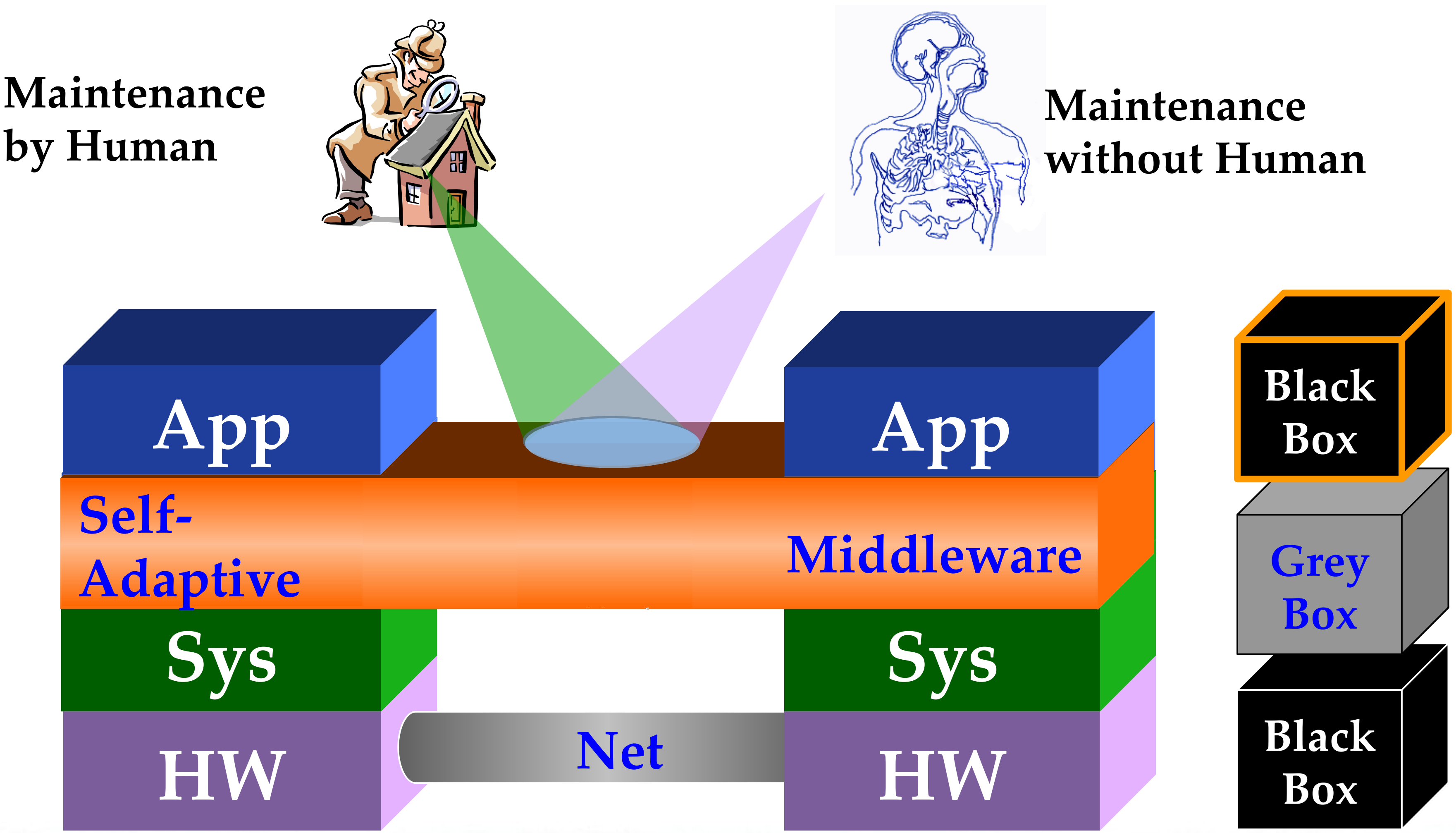
1. Towards Software Architecture at Runtime. ACM SIGSOFT Software Engineering Notes, Vol.28, No.2, March 2003.
2. Runtime Software Architecture based Software Online Evolution. In Proceedings of 27th Annual International Computer Software and Applications Conference (COMPSAC), 2003, Dallas, Texas, US, pp.230-235.
3. PKUAS: An Architecture-based Reflective Component Operating Platform, invited paper, 10th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS), Suzhou, China, 26-28 May 2004, pp 163-169.
4. Runtime Software Architecture Based On Reflective Middleware. Science in China, Series F, 2004, Vol.47, No.5, 555-576.
5. Architecture Model based J2EE Application Deployment for Dynamic Commerce. In Proceedings of IEEE International Confere Business (CEC04-EAST), Beijing, China, S
6. Architecture based Deployment of Large-S Principles. 8th International SIGSOFT Sy (CBSE), 2005, LNCS 3489, Springer, pp. 12
7. Towards a Unified Formal Model for Supporting Mechanisms of Dynamic Component Update, The fifth joint meeting of the European Software Engineering Conference and ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC-FSE'05), Lisbon, Portugal, September 5-9, 2005, pp. 80-89.
8. Dynamic Recovery and Manipulation of Software Architecture of Component-based Systems. Accepted for publication, International Journal of Automated Software Engineering, Springer, Vol. 13 No. 2, April 2006.
9. SOAR: Towards Dependable Service-Oriented Architecture via Reflective Middleware, Accepted for publication, International Journal of Simulation and Process Modelling, InderScience Publishers.

Formalization of reflective middleware from the perspective of software architecture

Others

- 5 Chinese Papers; 2 Chinese Patents
- Member of Expert Group of JSR 262: Web Service Connector for JMX

From Reflective To Self-Adaptive



Self-Adaptation

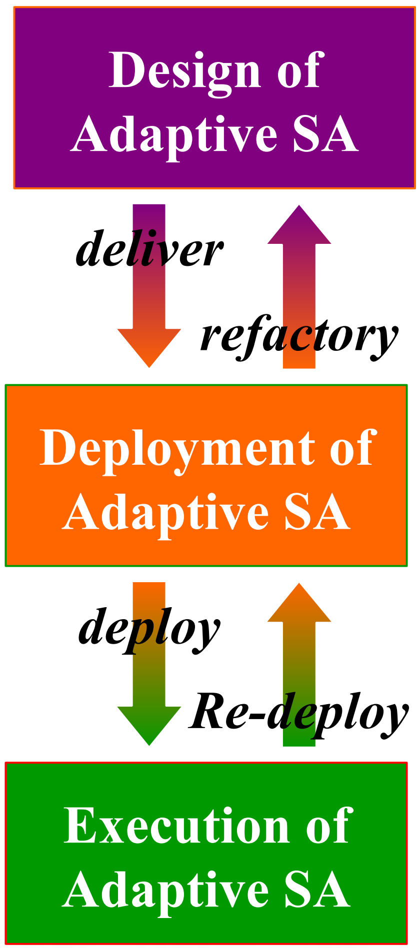
- **Configurable/reflective PKUAS**
  - Only address how to change
  - Cannot deal with why, when and what to change
    - These problems are critical to put PKUAS into practice

- **Self-adaptive PKUAS**
  - Can adapt itself for coping with rapid and continuous changes without stopping the whole system
  - The key is how to get the knowledge and make the decisions of why, when and what to change
  - Three approaches are investigated
    - Engineering approach
    - Empirical approach
    - Agent approach

Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn
Peking University

Engineering Approach

SASA



Philosophy

- Designers can predict some runtime changes and plan correct adaptations

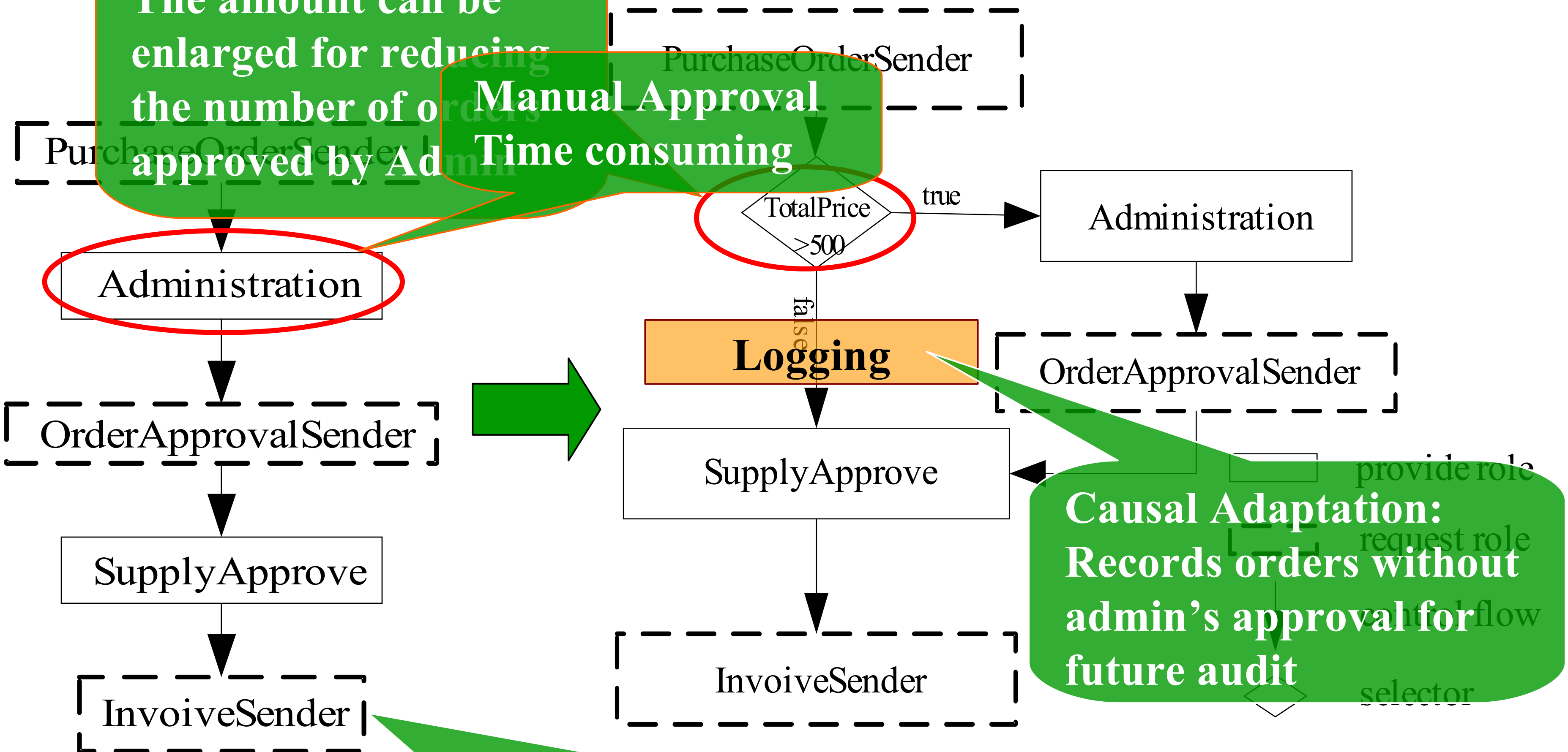
Self-Adaptive Software Architecture (SASA)

- When to change
  - Classical SA analysis methods, e.g., ATAM and SAAM
- What to change
  - Dynamic SA, e.g., Taylor@UCI and Garlan@CMU
- How to change
  - Runtime SA by reflective middleware or reflective component

Fractal: dynamic SA

The amount can be enlarged for reducing the number of orders approved by Admin

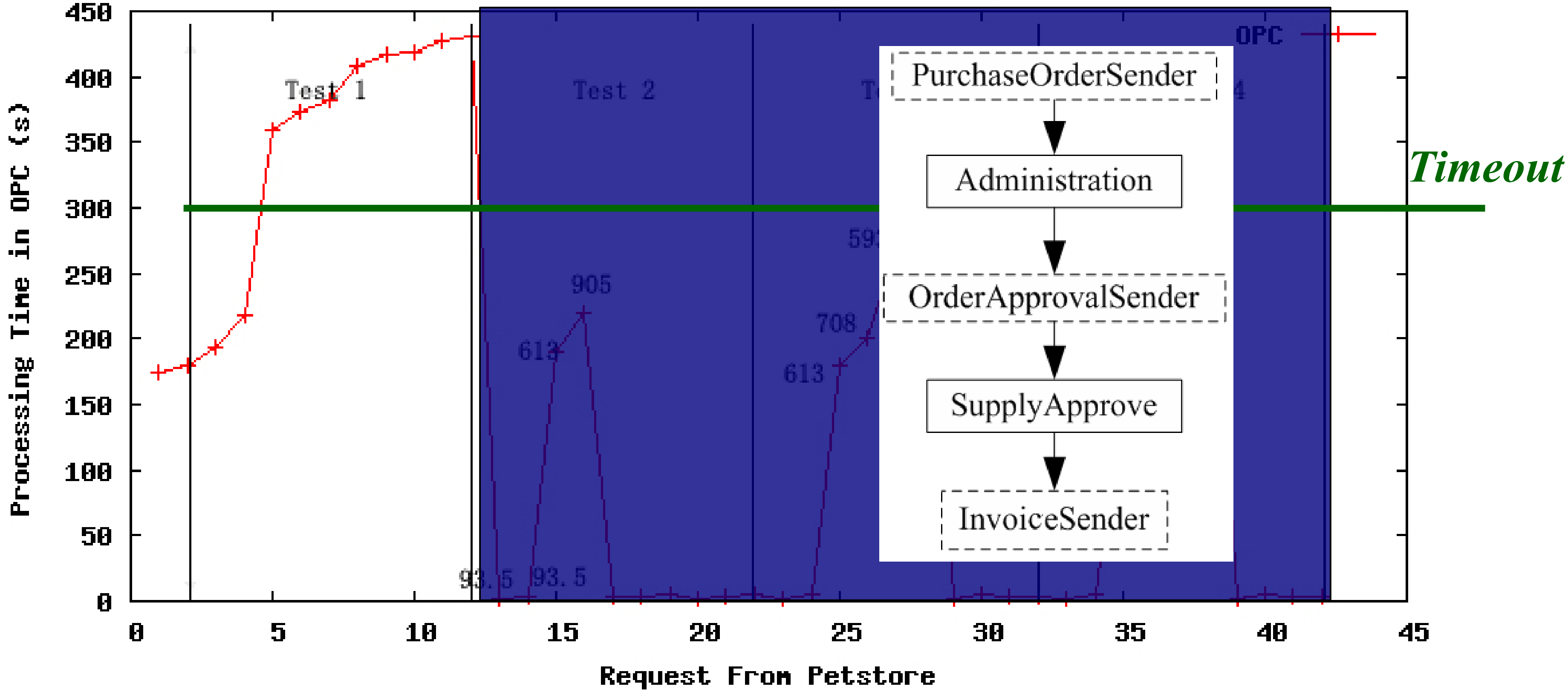
Manual Approval
Time consuming



Causal Adaptation:
Records orders without admin's approval for future audit

Scenario 2:
the OPC can handle more than 10 orders concurrently but the administrator cannot do it by hand.

SASA Case: JPS Test 1

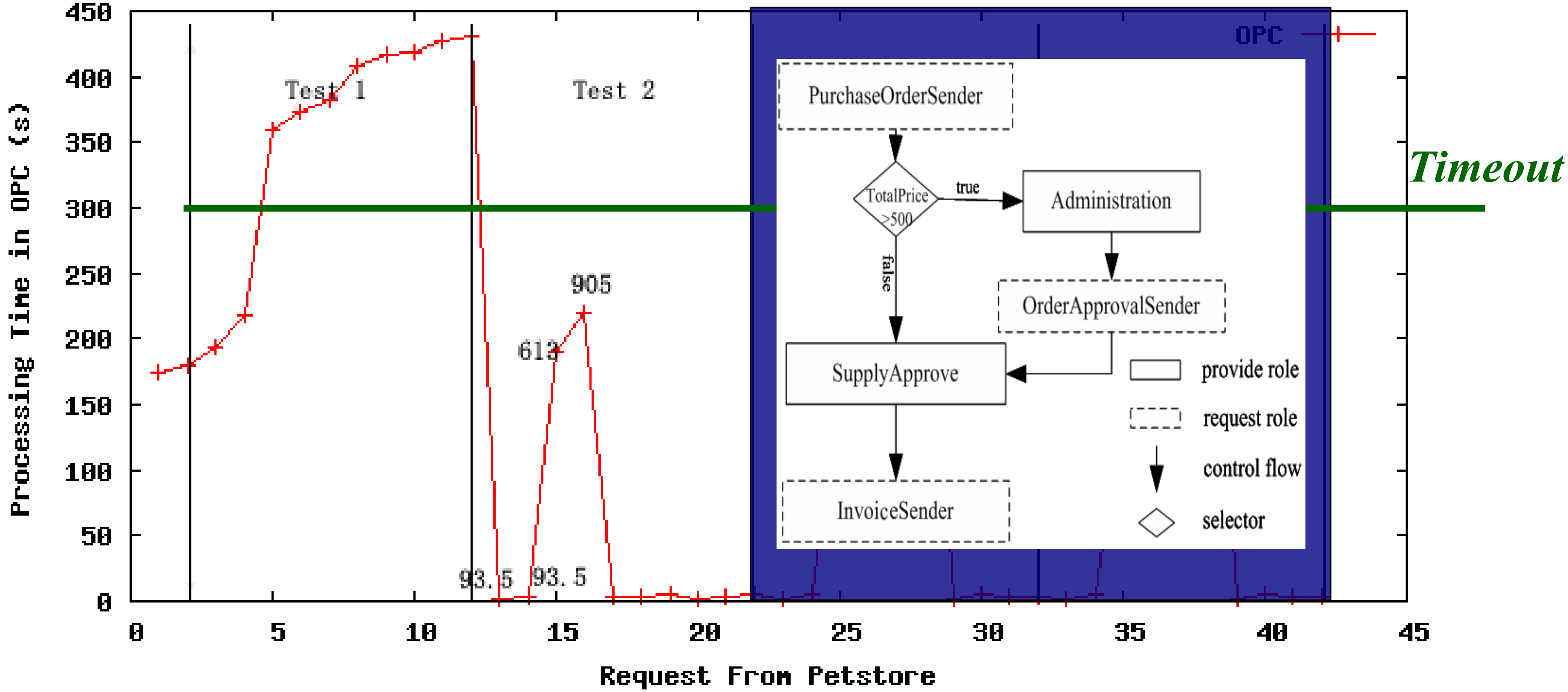


- **Every order has to be checked by admin**
  - **10 orders in 50 seconds; all orders are checked by admin**
  - **8 orders are timeout; the flow has to be adapted**

Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

Peking University

SASA Case: JPS Test 2

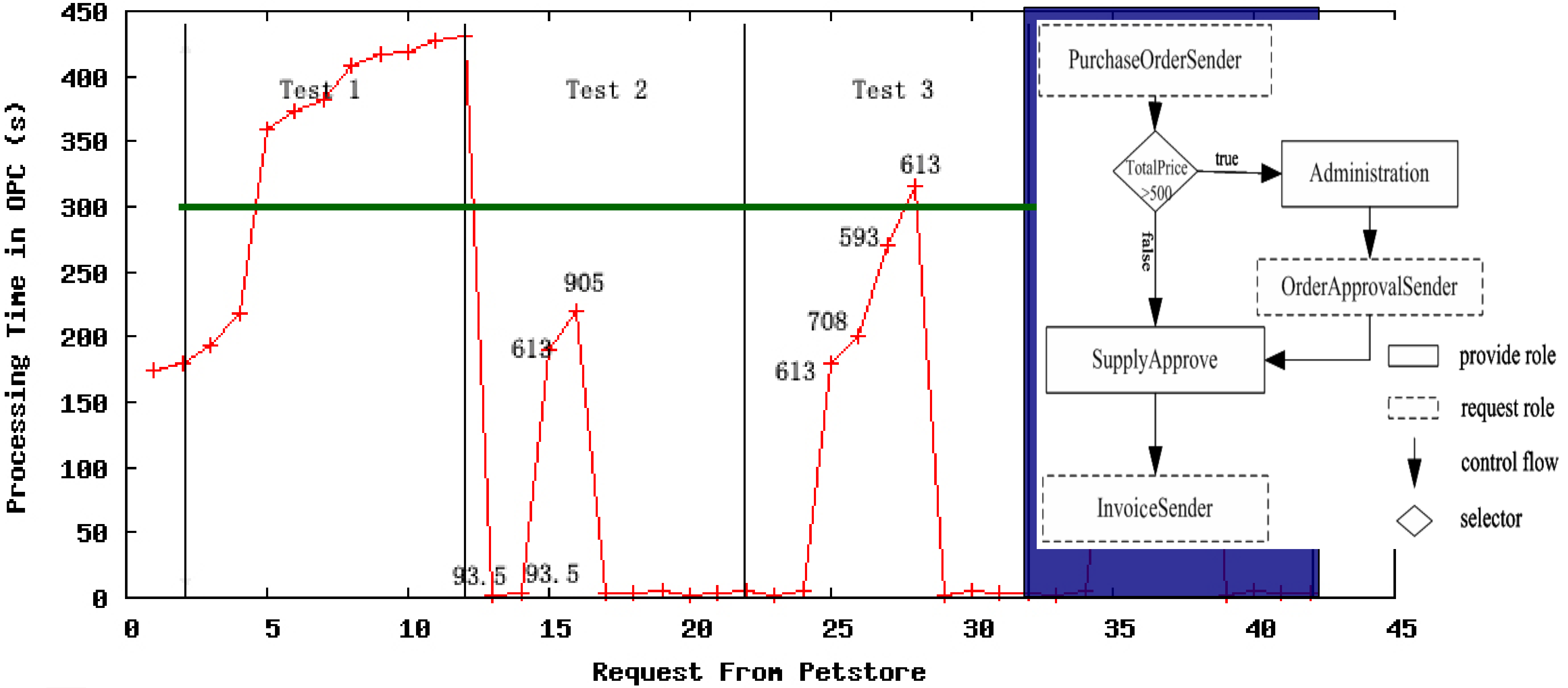


- **Only orders more than 500 dollars have to be checked by admin**
  - **10 orders in 50 seconds; 2 orders are checked by admin**
  - **None order is timeout; Adaptation is valid**

Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

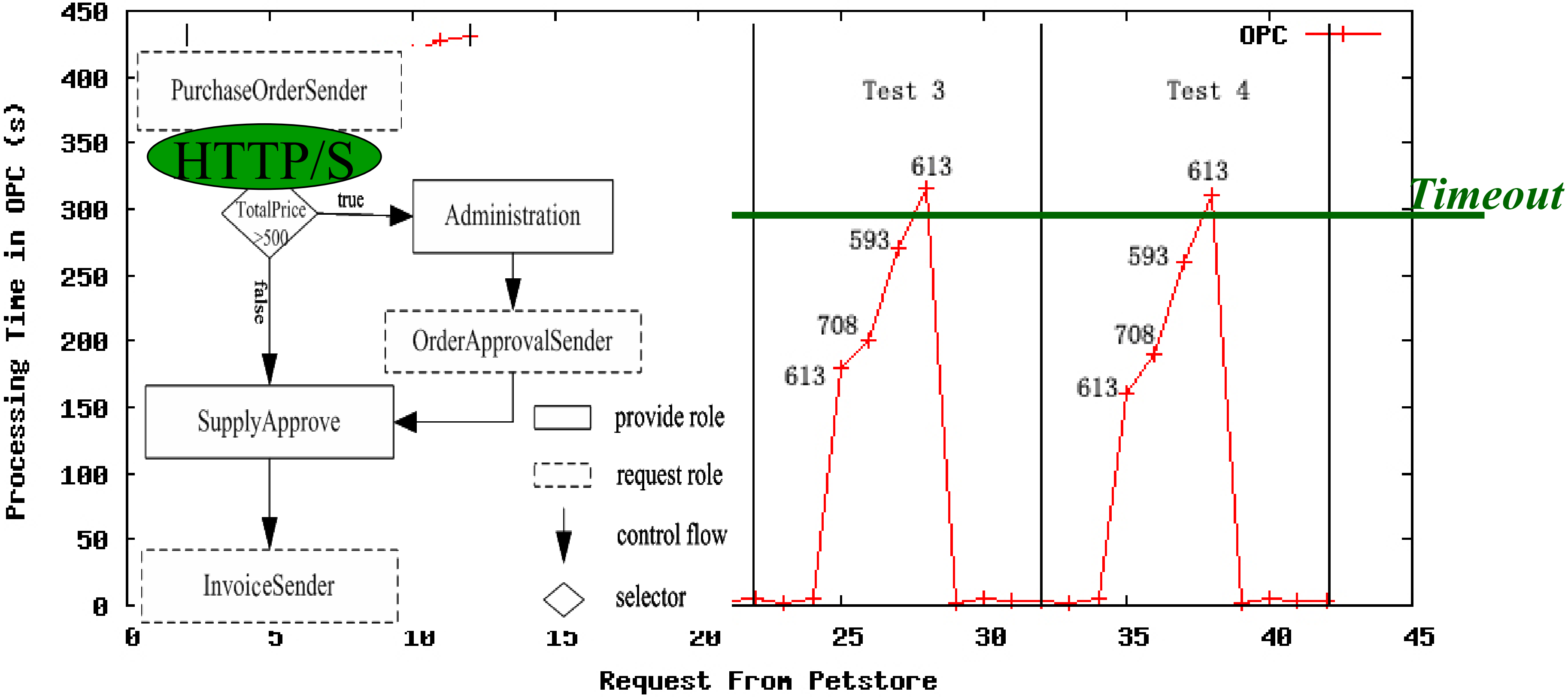
Peking University

SASA Case: JPS Test 3



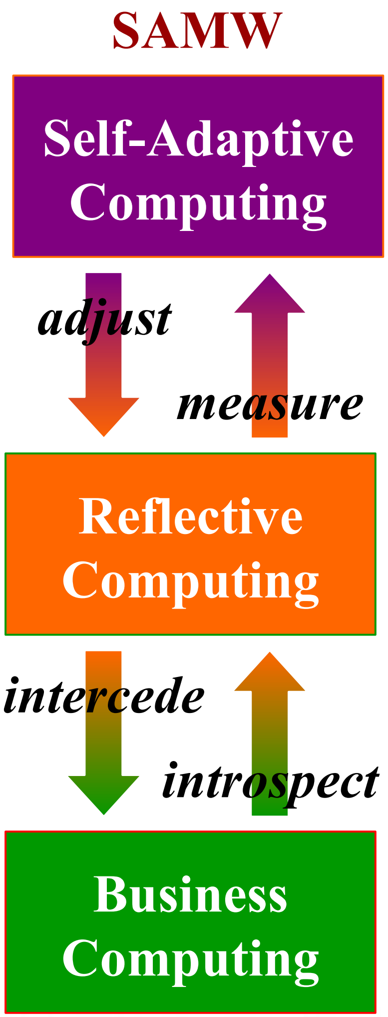
- Only orders more than 500 dollars have to be checked by admin
  - 10 orders in 50 seconds; 4 orders are checked by admin
  - 1 order is timeout; Adaptation cannot handle every condition

SASA Case: JPS Test 4



- **HTTPS is replaced by HTTP for decreasing the response time**
  - **10 orders in 50 seconds, 4 orders are checked by admin**
  - **1 order is timeout; Some adaptation is invalid at all**

Empirical Approach



❑ Philosophy

- **Service providers and domain experts can predict some runtime changes and implement application-independent adaptations based on their experiences and skills**

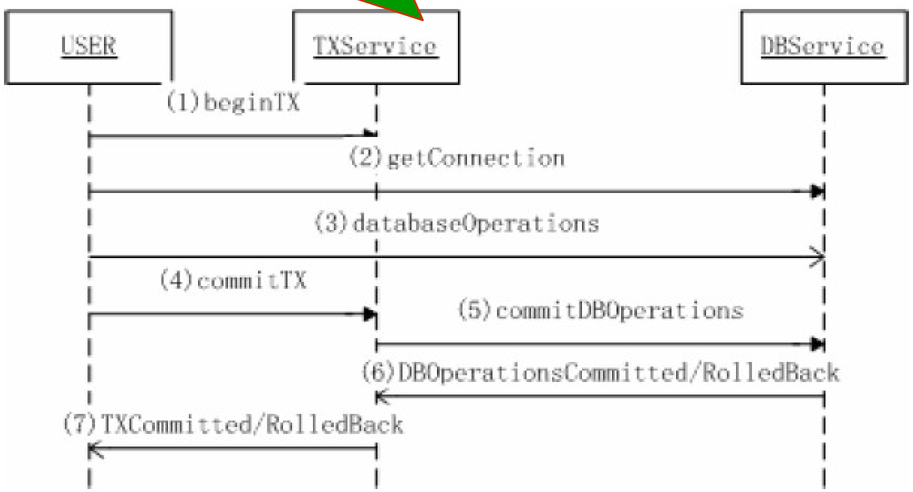
❑ Self-Adaptive Middleware (SAMW)

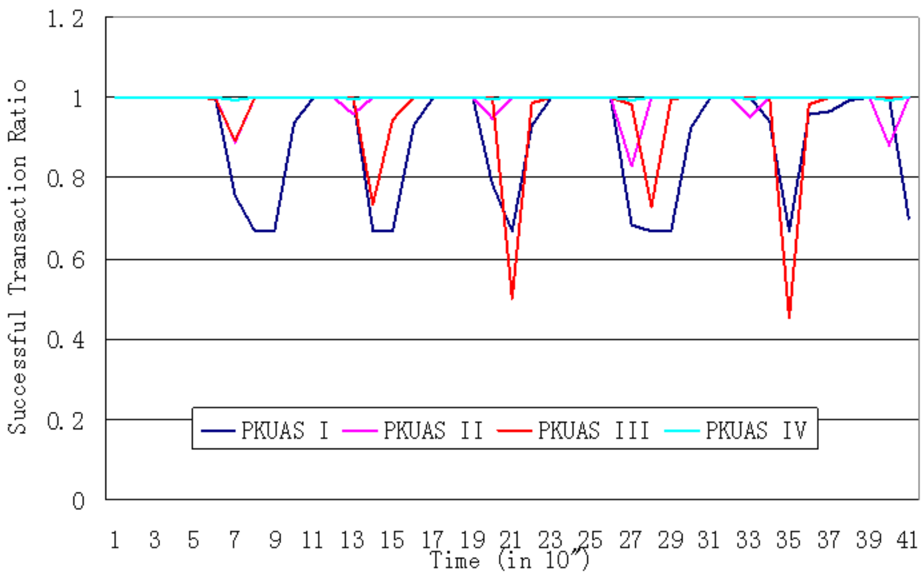
- **Encapsulate all mechanisms related to the adaptations as middleware services**
  - **Self-configuring**
  - **Self-optimizing**
  - **Self-healing**
  - **Self-protecting**
- **From reflective middleware to self-adaptive middleware**

Fractal: (architecture based) autonomic system management

SAMW Case: Self-Healing of Correlated Faults

DB failures cause TX failures

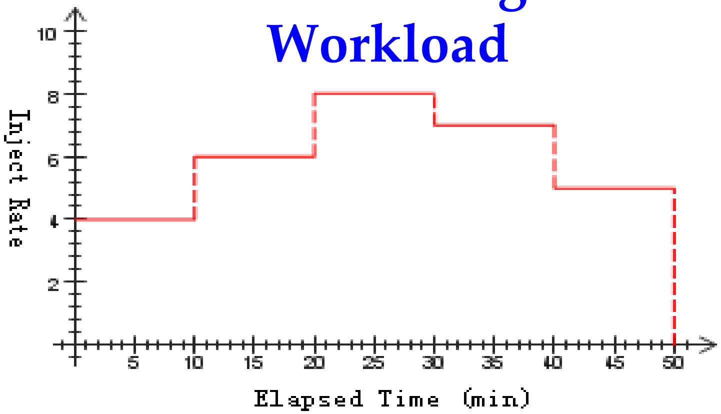




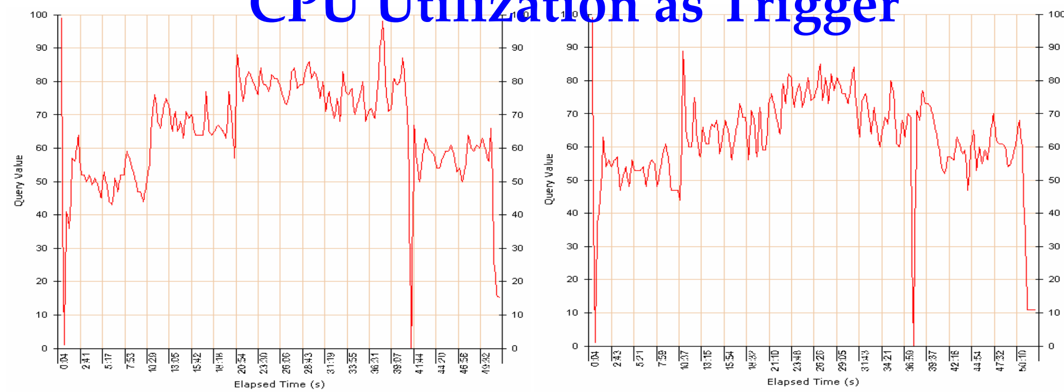
	No-correlated No-cache	No-correlated Cache	Correlated No-cache	Correlated Cache
Failed	3756	218	121	9
Fired	42351	35336	32072	34677
Availability	91.13%	99.38%	99.62%	99.97%

SAMW Case: Management of Autonomic Computations

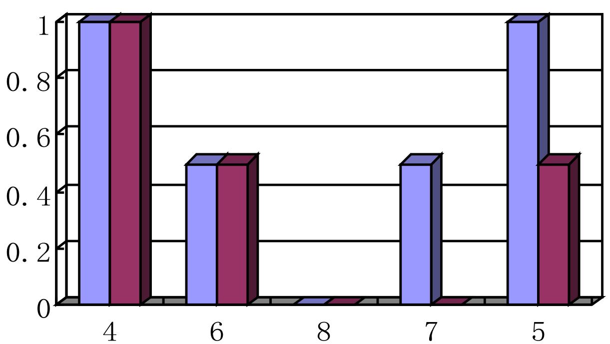
Fluctuating Workload



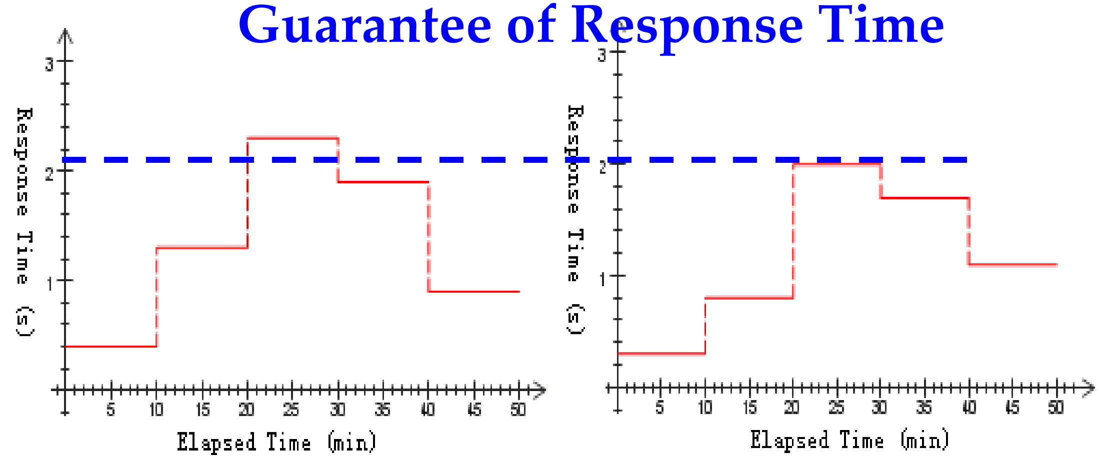
CPU Utilization as Trigger



Execution of AC



Guarantee of Response Time



Makes all response times acceptable

Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

Peking University

Publications on Self-Adaptation

- **English Papers**
  1. Towards Autonomic Computing Middleware via Reflection. In Proceedings of 28th Annual International Computer Software and Applications Conference (COMPSAC), Hongkong, China, September 28-30, 2004, pp.122-127.
  2. Quality Attribute Scenario Based Architectural Modeling for Self-Adaptation Supported by Architecture-based Reflective Middleware, In Proceedings of Asia Pacific Software Engineering Conference (APSEC 2004), Busan, Korea, November 30 - December 3, 2004, pp. 2-9.
  3. Towards Self-Healing Systems via Dependable Architecture and Reflective Middleware, invited paper, IEEE International Workshop on Object Oriented Real-time and Dependable Systems (WORDS), 2005, Arizona, USA, pp. 337-346.
  4. The Coordinated Recovery of Data Service and Transaction Service in J2EE, In Proceedings of 29th Annual International Computer Software and Applications Conference (COMPSAC05), Edinburgh, Scotland, July 2005, pp. 485-490.
  5. Feature Interactions Induced by Data Dependencies among Entity Components, 8th International Conference on Feature Interactions in Telecommunications and Software Systems (ICFI05), 28th June to 30th June, 2005, Leicester, UK, pp. 252-269.
  6. Feature Interaction Problems in Middleware Services, 8th International Conference on Feature Interactions in Telecommunications and Software Systems (ICFI05), 28th June to 30th June, 2005, Leicester, UK, pp. 313-319.
  7. Exception Handling in Component Composition with the Support of Middleware. Fifth International Workshop on Software Engineering and Middleware (SEM 2005), co-located with ESEC-FSE'05, Lisbon, Portugal, September 5-6, 2005, ACM Press, pp.90-97.
  8. Coordinated Recovery of Middleware Services: The Framework and Case Studies, invited for a special issue, The Computer Journal, Oxford Press.

Agent Approach

Autonomous Components

- Are such components whose autonomous capability is enhanced by agent technologies
- Continually evaluate and modify their behaviors to meet changing demands
- Collect new information when system is running
- Use rules to guide decision making at runtime

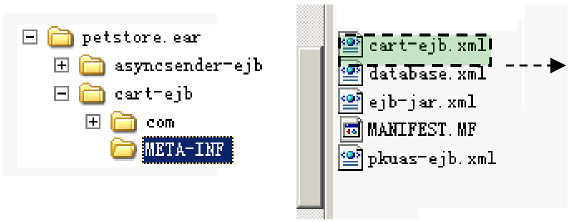
The Autonomization of Component

- Define adaptation rules and related knowledge;
- Weave rules with target component using PKUAS user-defined interceptor;
- Collect and translate runtime information into knowledge representation to activate the rules and influence component behavior.

Fractal is a reflective component model
We try to support an autonomous/self-adaptive component model

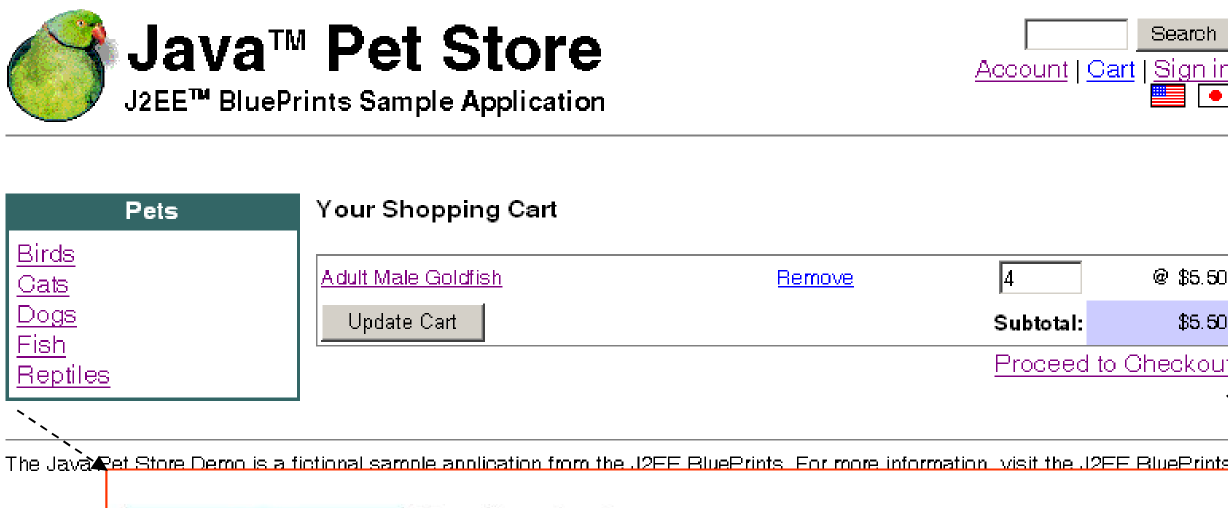
Case Study: AutoJPS

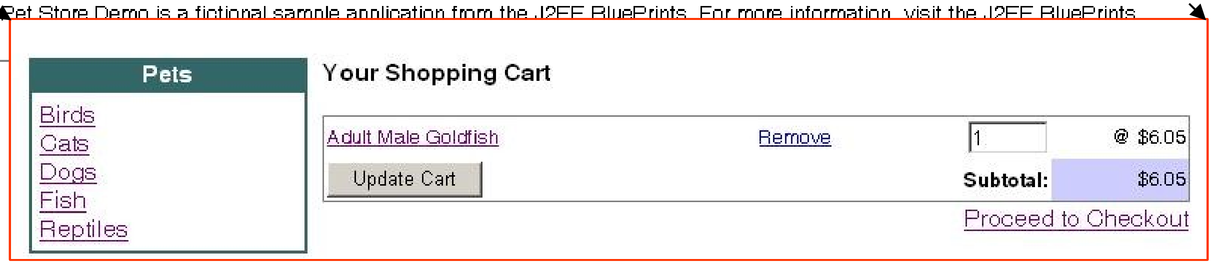
- Raise an item's price if it is a best-seller;



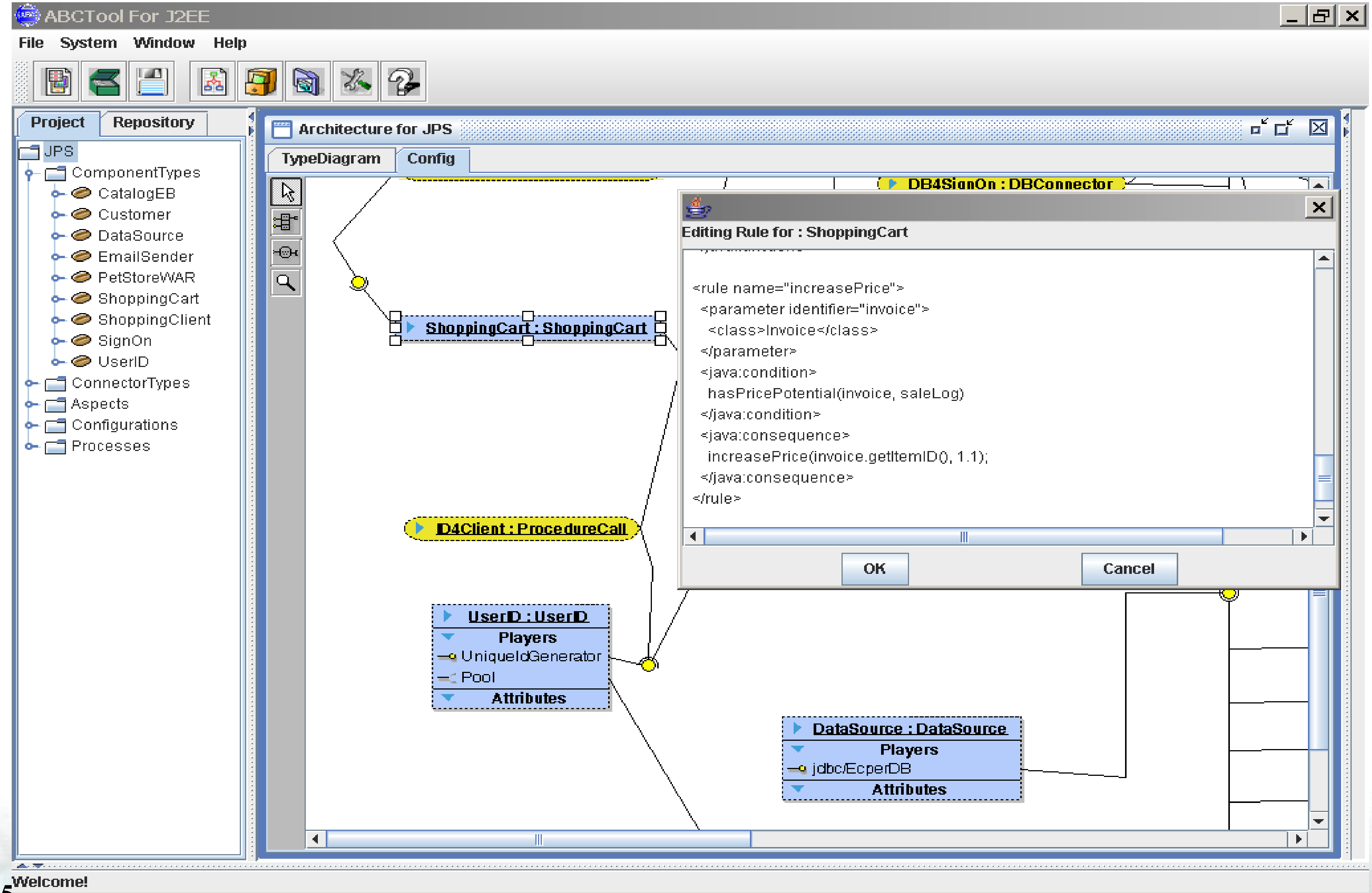
PKUAS will detect the rule file and load the interceptor automatically

Modify the price if an item's total sales amount has exceeded a certain number in a given time interval.

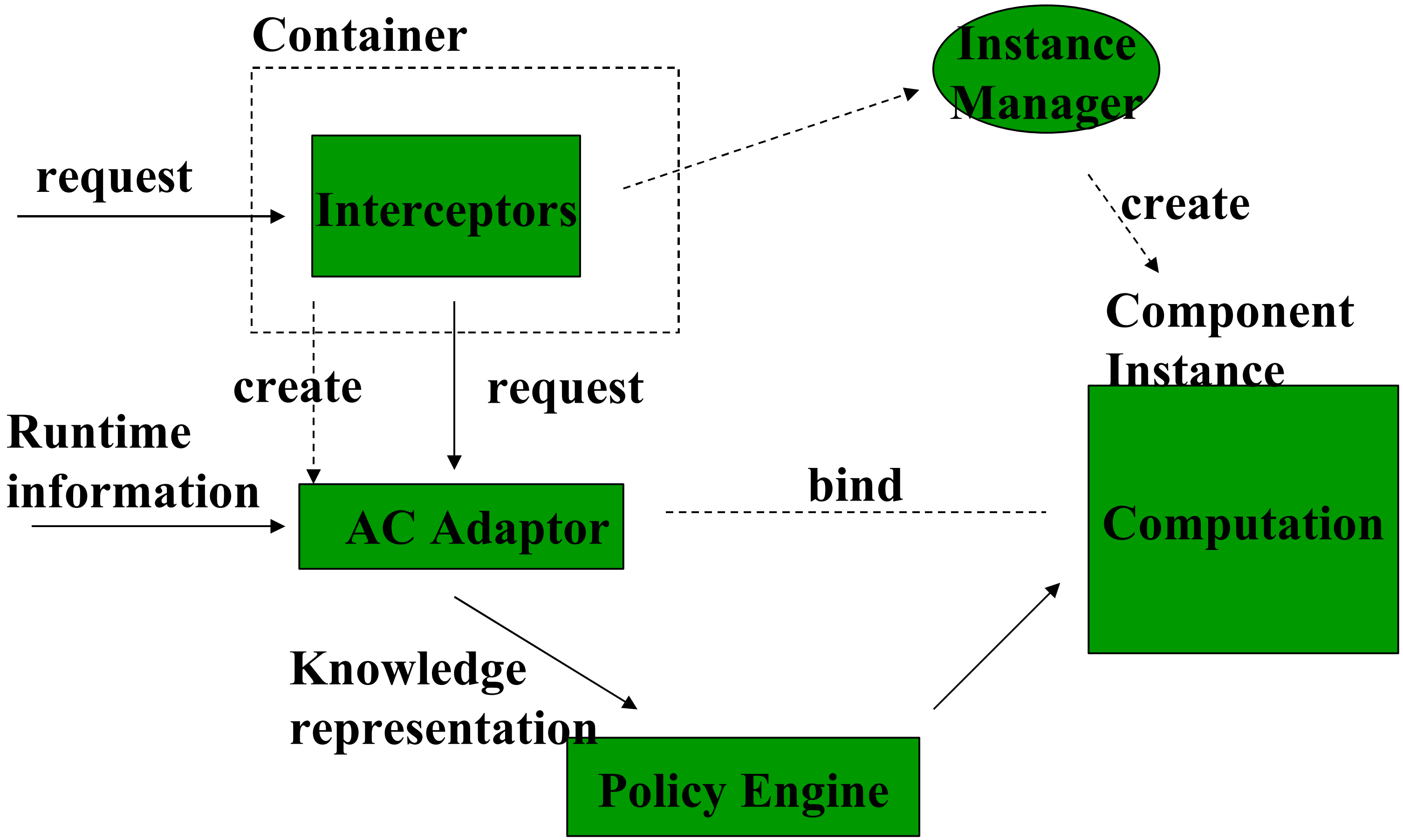




Rule Design in Software Architecture



Rule Engine in PKUAS



From Fractal perspective:
→membrane: rule-based control; →content: the same
→We do the similar mergence with Julia, i.e., encapsulate all things except *Content* into J2EE application server

Publications on Agent

English Papers

1. Eliminating Architectural Mismatches by Adopting an Agent-based Approach. 15th IEEE International Conference on Tools with Artificial Intelligence, 2003.11. an extension version is invited for a special issue on Software Quality Journal, 2005
2. Formal Framework for Adaptive Multi-Agent Systems. IEEE/WIC International Conference on Intelligent Agent Technology. 2003
3. Automated Adaptations to Dynamic Software Architectures by Using Autonomous Agents. Engineering Applications of Artificial Intelligence, Vol.17, No.7, 2004, pp. 749-770.
4. Organizational Models and Interaction Patterns for use in the Analysis and Design of Multi-Agent Systems. Web Intelligence and Agent Systems, No.1, 2005.
5. Dynamic Architectural Connectors in Cooperative Software Systems. 10th IEEE International Conference on the Engineering of Complex Computer Systems (ICECCS2005), Shanghai, China, 16-20 June 2005, pp. 477-486.

Conclusion

- **PKUAS (J2EE-Compliant Application Server)**
  - Promising academic and industrial values
  - Commercial and practical applications
  - Distinguished and innovative features
    - Customizability and extensibility
    - Architecture based reflection and deployment
    - Self-adaptation (main direction in the future)
  - More than 40 papers, 5 Chinese patents, 1 Java Specification
- **Other Work**
  - Dynamic Aspect Weaving
  - Trustworthy Component
  - Service Oriented Architecture with Reflection
- **How to Collaborate with ObjectWeb/JonAS ?!**
  - FIT: a Framework for Innovation Transfer based on Dynamic Aspect Oriented Programming

Motivation

- How to transfer innovative features from one middleware product to another one
  - The key is how to separate the implementation codes from the basic implementation
  - The codes for innovative features are usually scattered and tangled

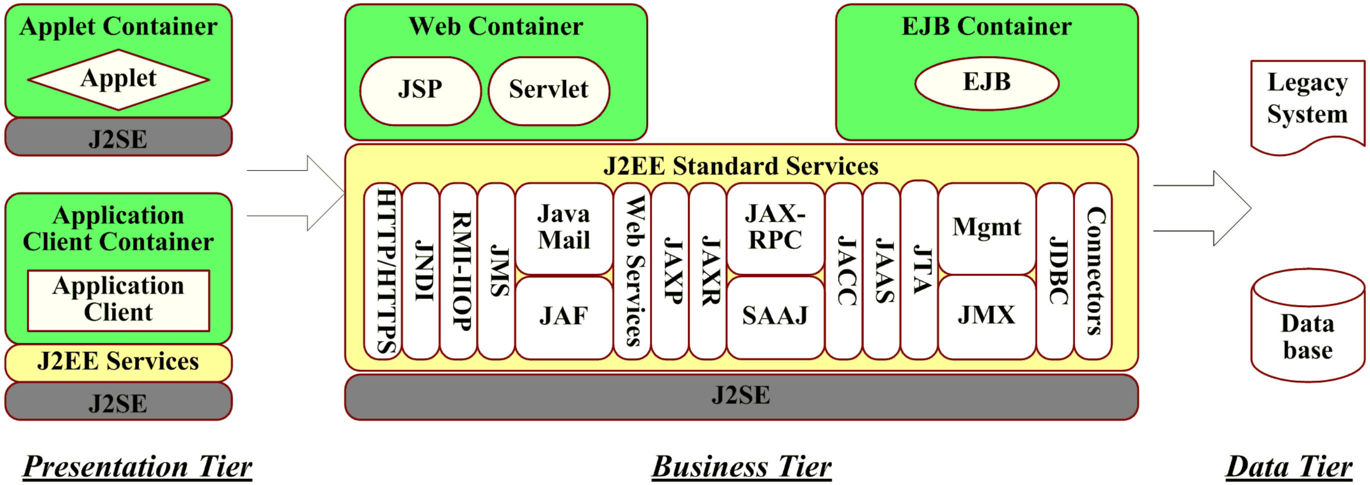
Can we implement the innovative features as ASPECT

ASPIRE | PISA | RSA | •••

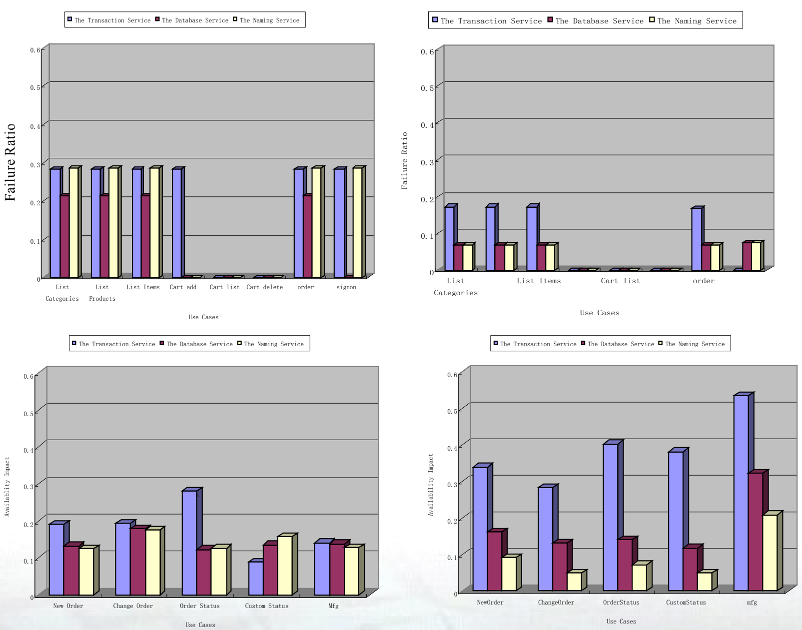
Dynamic AOP (Javassist)

PKUAS | JonAS | JBOSS

ASPIRE: Motivation



Application
SPecific
Improvement of
REliability



Phenomenon:

→Different middleware service has different impact on the reliability of the whole system and different cost and risk for fault tolerance.

→For a given middleware service in a given application

Is the service impact the reliability?

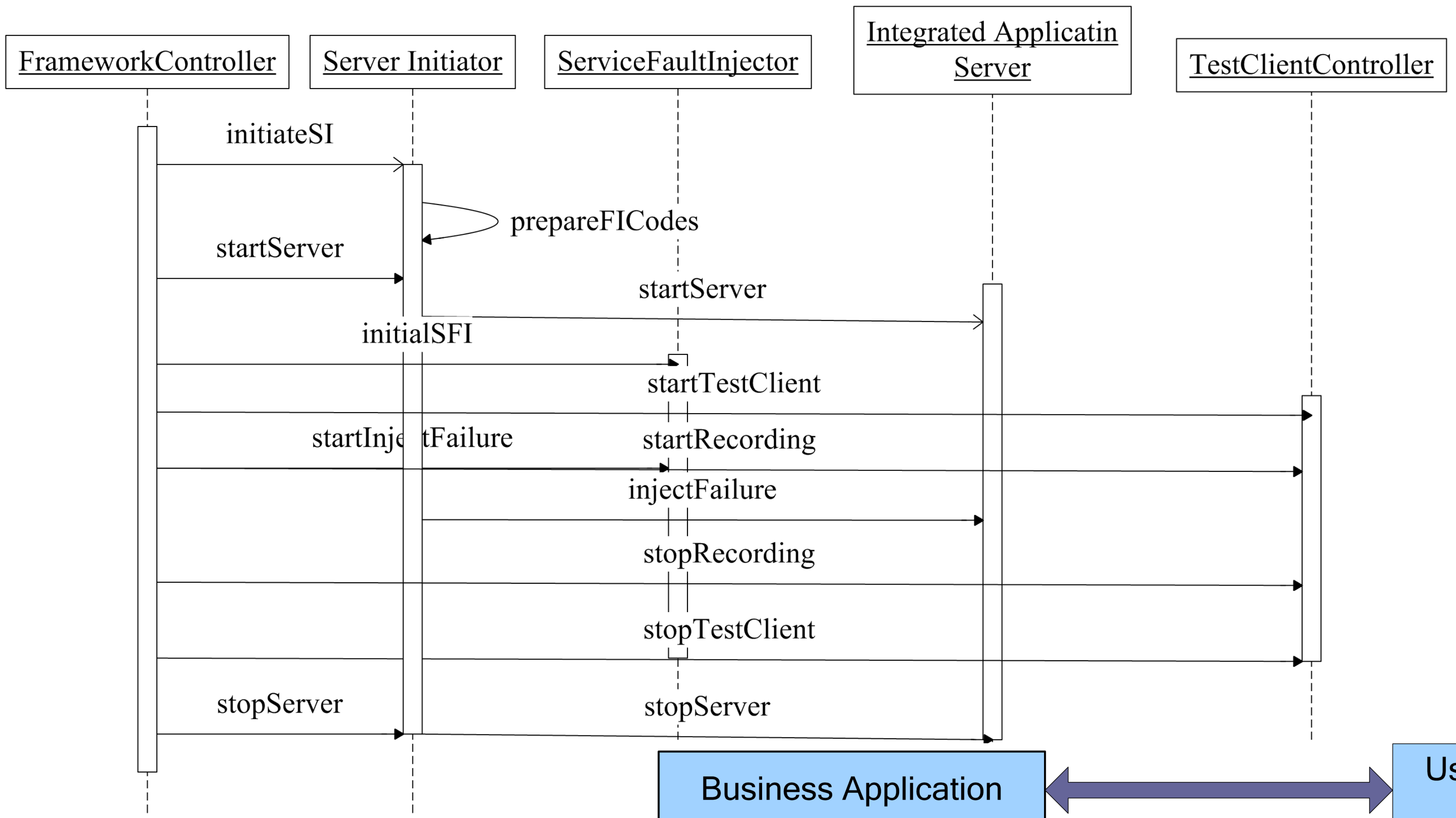
How much is the impact?

Is it worthy to become more reliable?

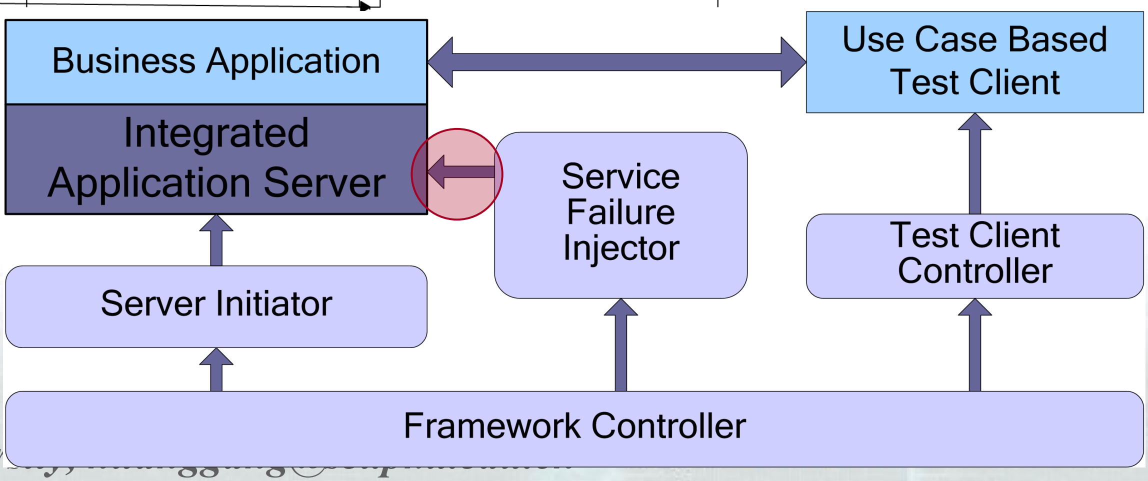
Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

Peking University

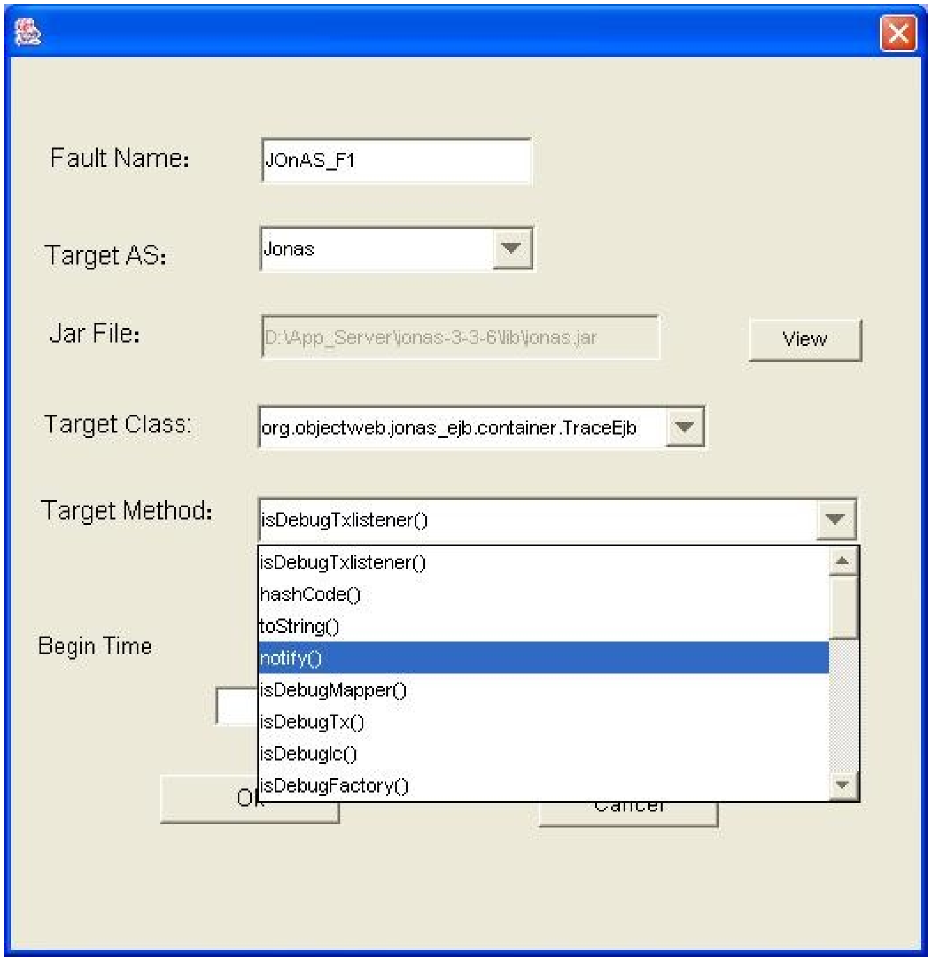
ASPIRE: Architecture



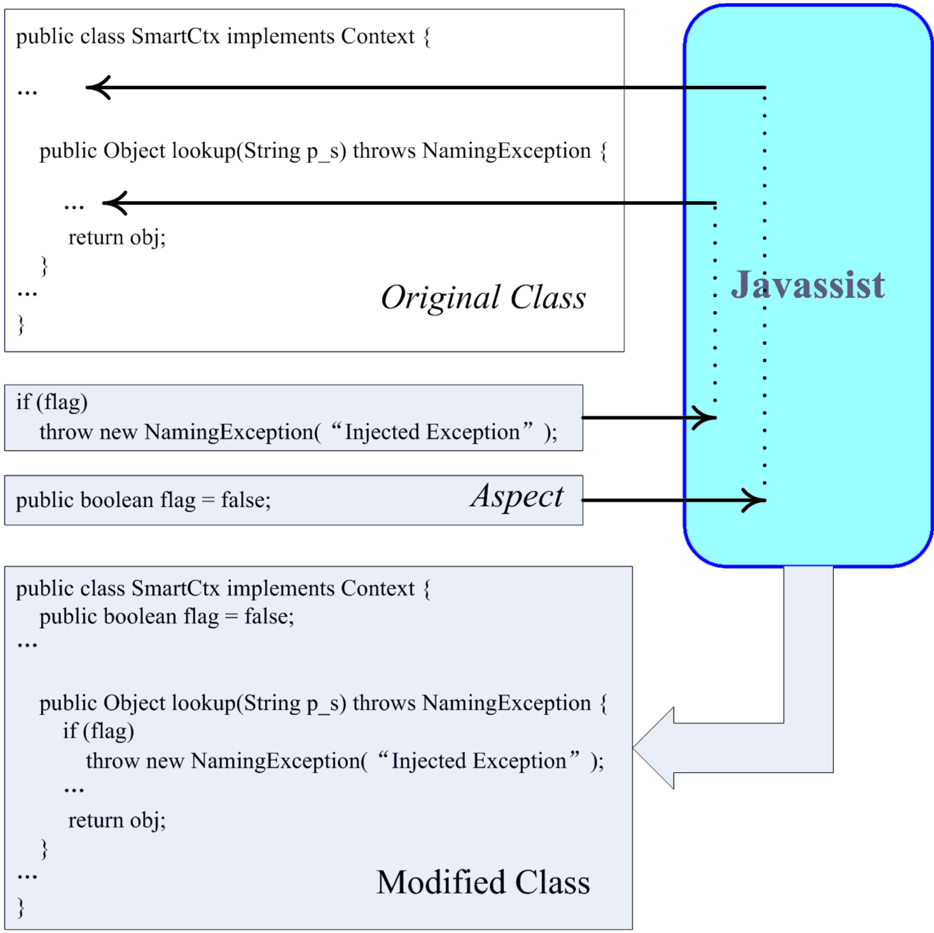
1) **Service exceptions as faults**
2) **Stimulated by fault injectors**
3) **Modification of middleware is done by dynamic AOP**



ASPIRE: Prototype



GUI of ASPIRE



Dynamic Aspect Weaving in ASPIRE

Peking University

Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

Potential Collaboration Points

Component Model

- Fractal is a reflective component model
- PKUAS is a reflective component framework
  - PKUAS tries to support autonomous component model
- Fractal components can be incarnated by PKUAS ?

Component Based Development Method

- Different philosophy?
  - Development from Scratch vs. Development by Reuse
- Fractal has an extensible ADL with a set of generators for programming languages
- ABC has ABC/ADL and a bridge to UML/MDA (JBOO)
- Leverage Fractal/ADL formalization and ABC/ADL composition

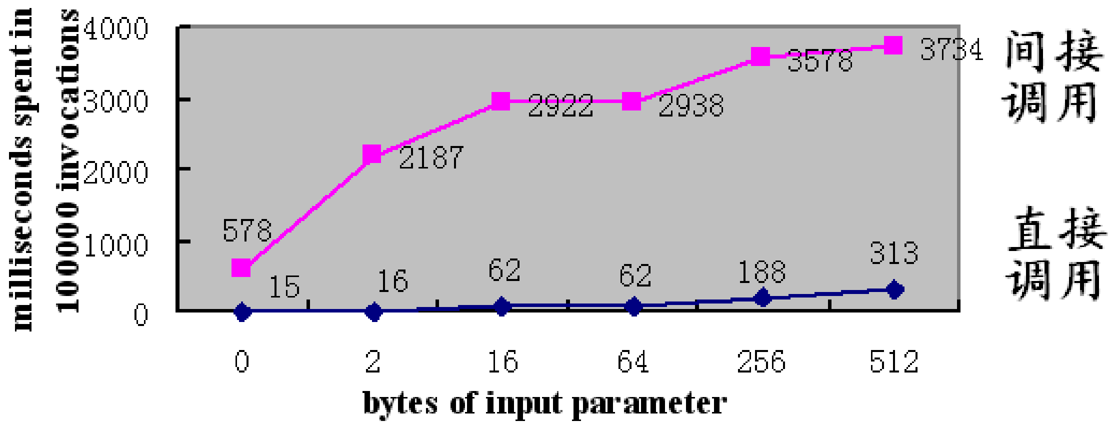
Autonomic System Management

- Both employ software architectures (DSA)
- PKUAS focuses on RSA and design of DSA
- Fractal's emphasis?


Thanks

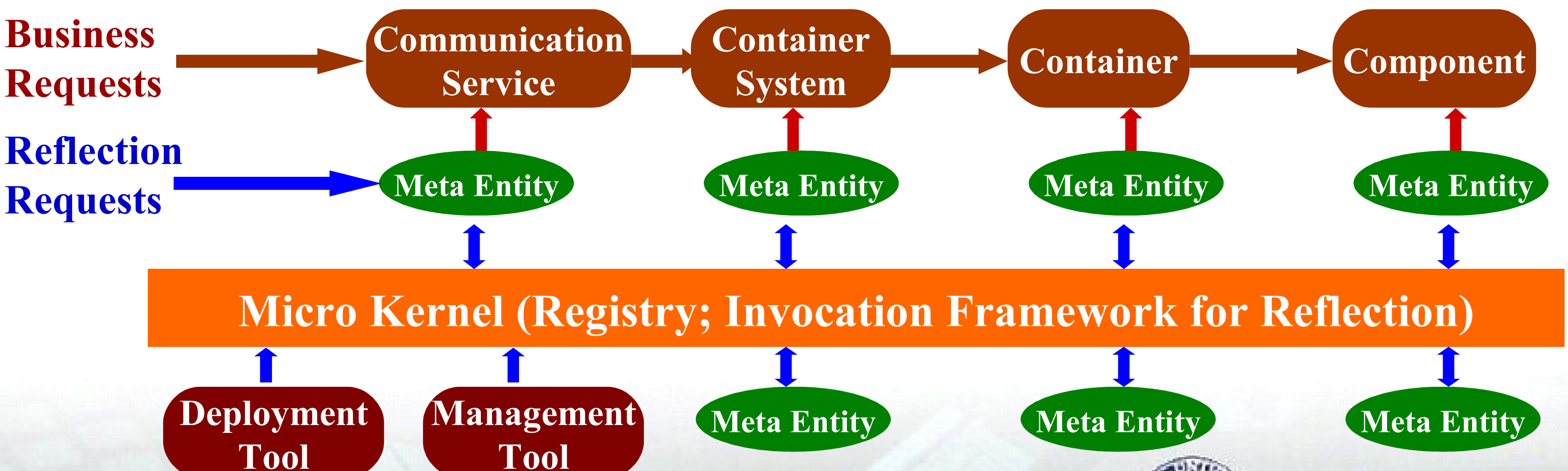
Componentization of PKUAS

Performance Optimization

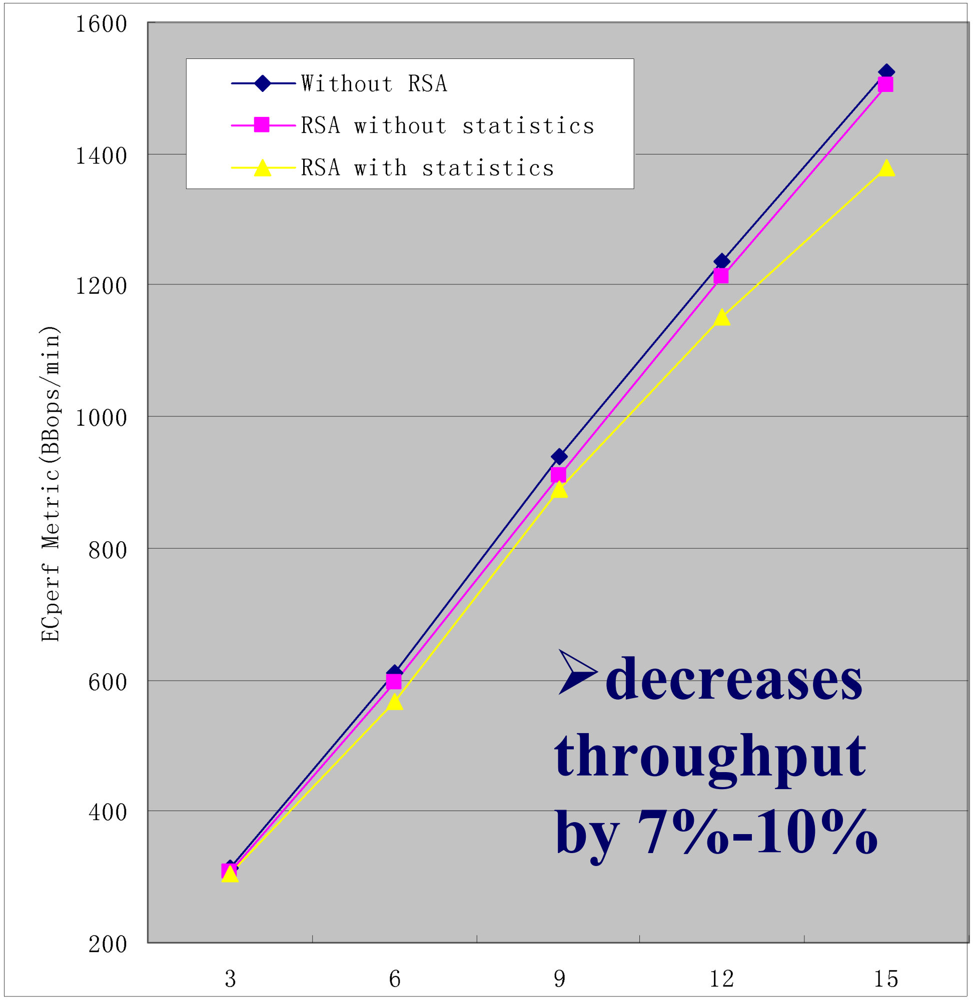


JMX Mediated Invocation :
- →Lower performance
- →For reflection requests

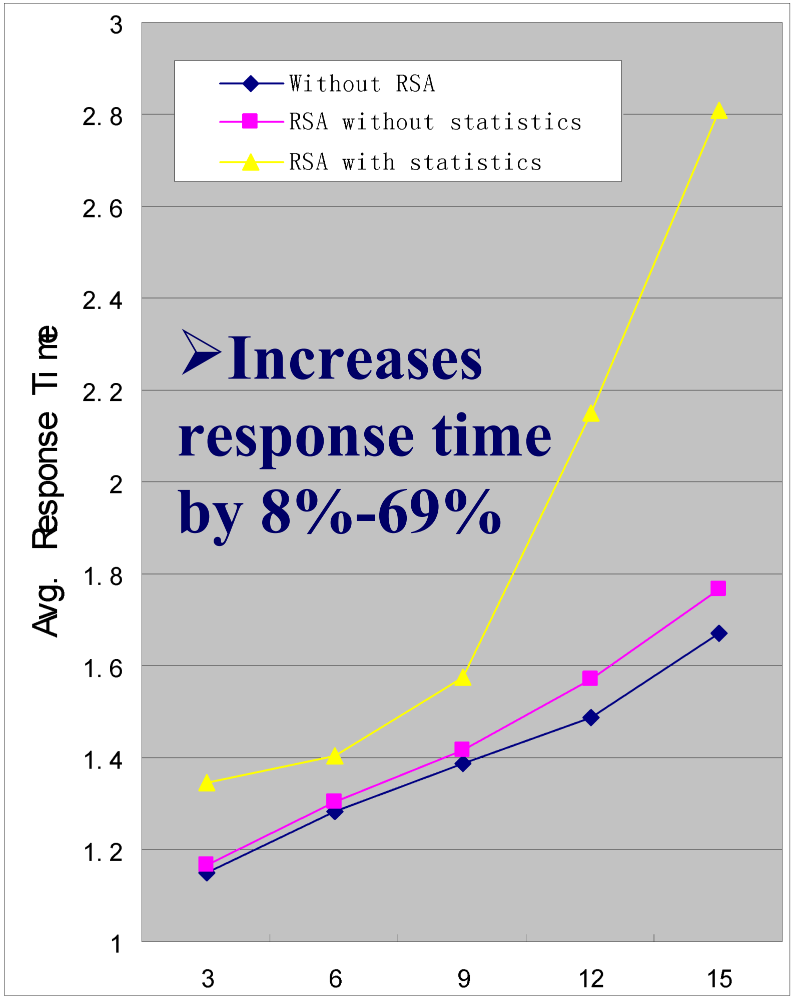
Direct Invocation:
- →Higher performance
- →For business requests



Performance Impact in ECperf



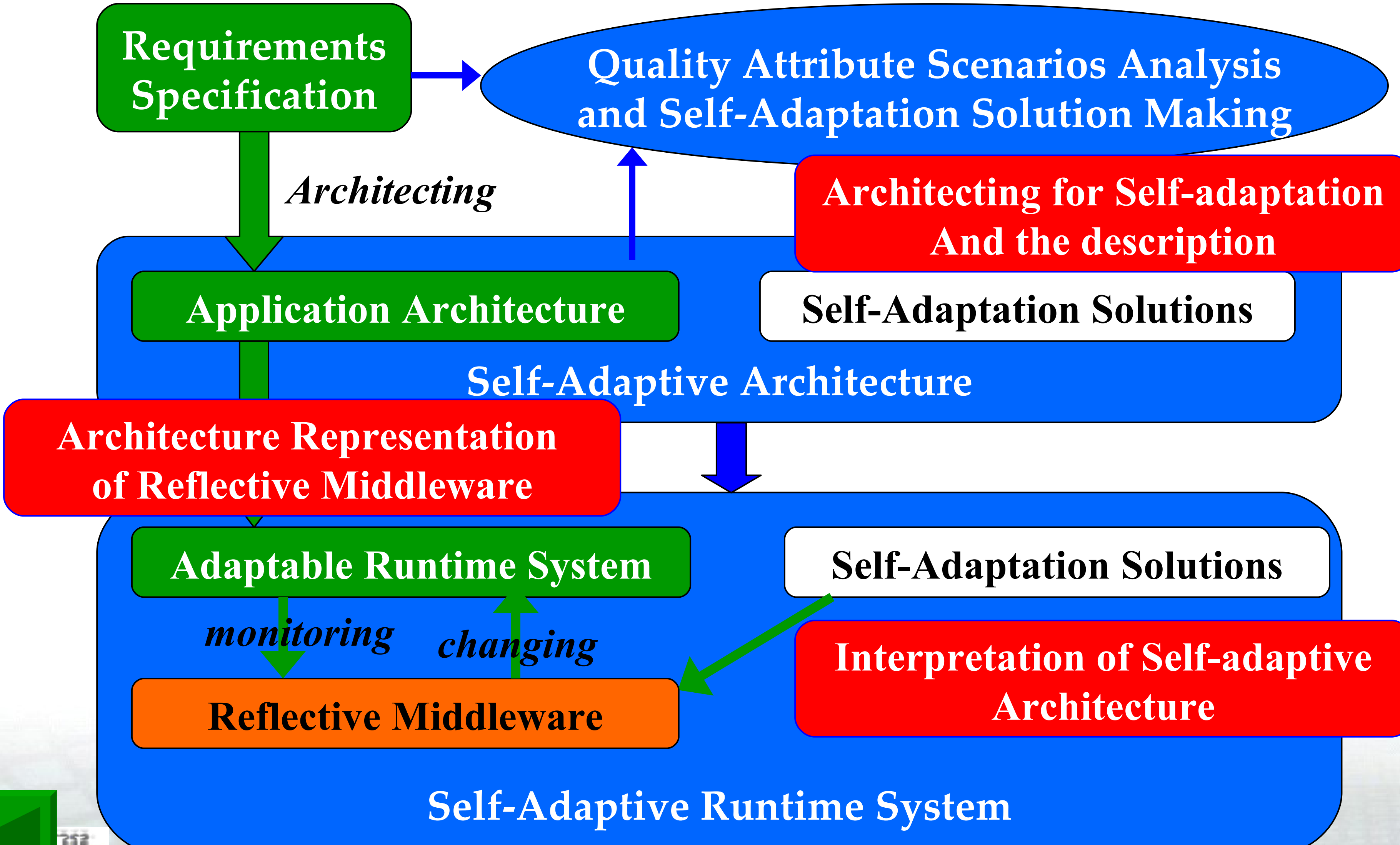
(a) RSA impact on throughput



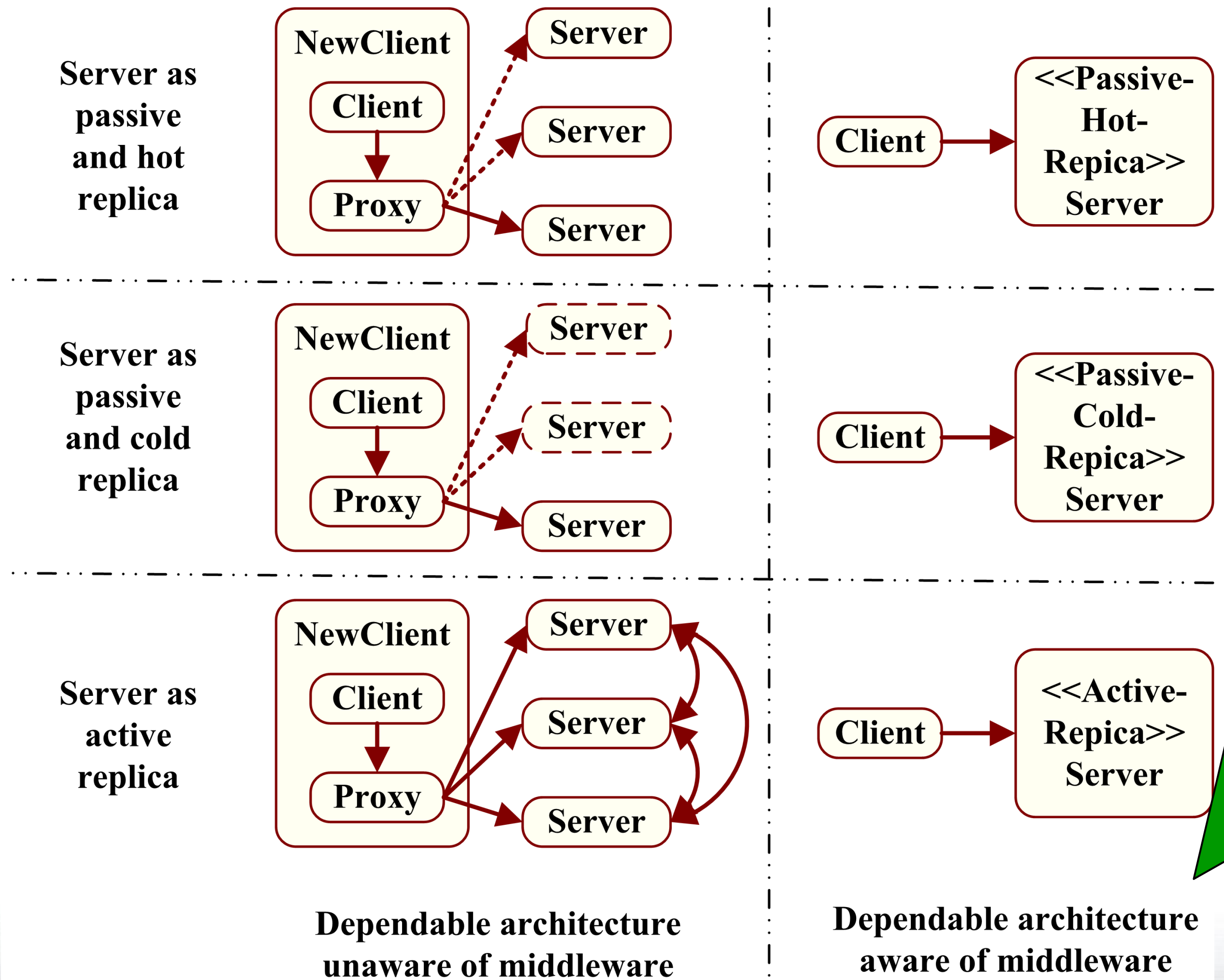
(b) RSA impact on response time

Dr. Gang Huang, Peking University, huanggang@sei.pku.edu.cn

Middleware Specific Architecting



Dependable Software Architecture



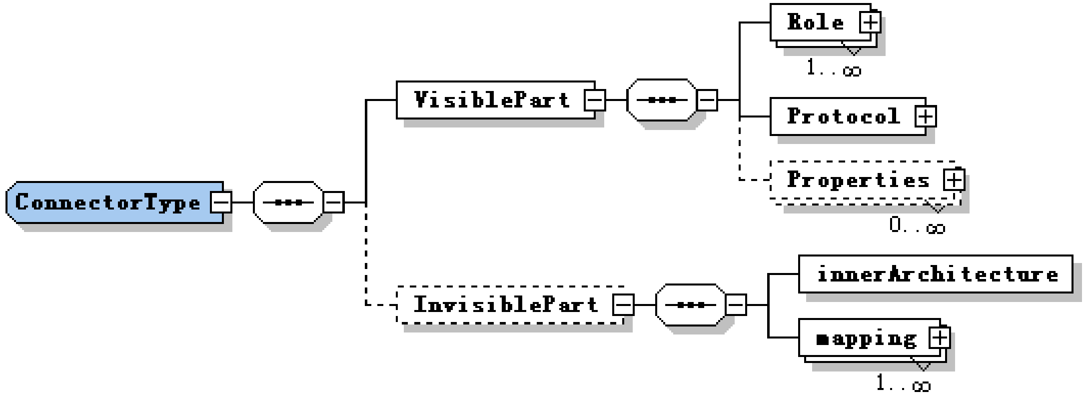
At Architecting →Predicting runtime failures and suggesting the prevention or recovery plan

At Runtime →interpreting the design decisions

Self-Healing !

Three Level of Connector

	Connection (structure)	Coordination (behavior)	Context (non-functional quality)
diversity of interoperability protocol	+		
enhanced interoperability protocols	+		+
Pub/Sub among multiple components	+		
choreography between two components		+	
execution flow among multiple components		+	

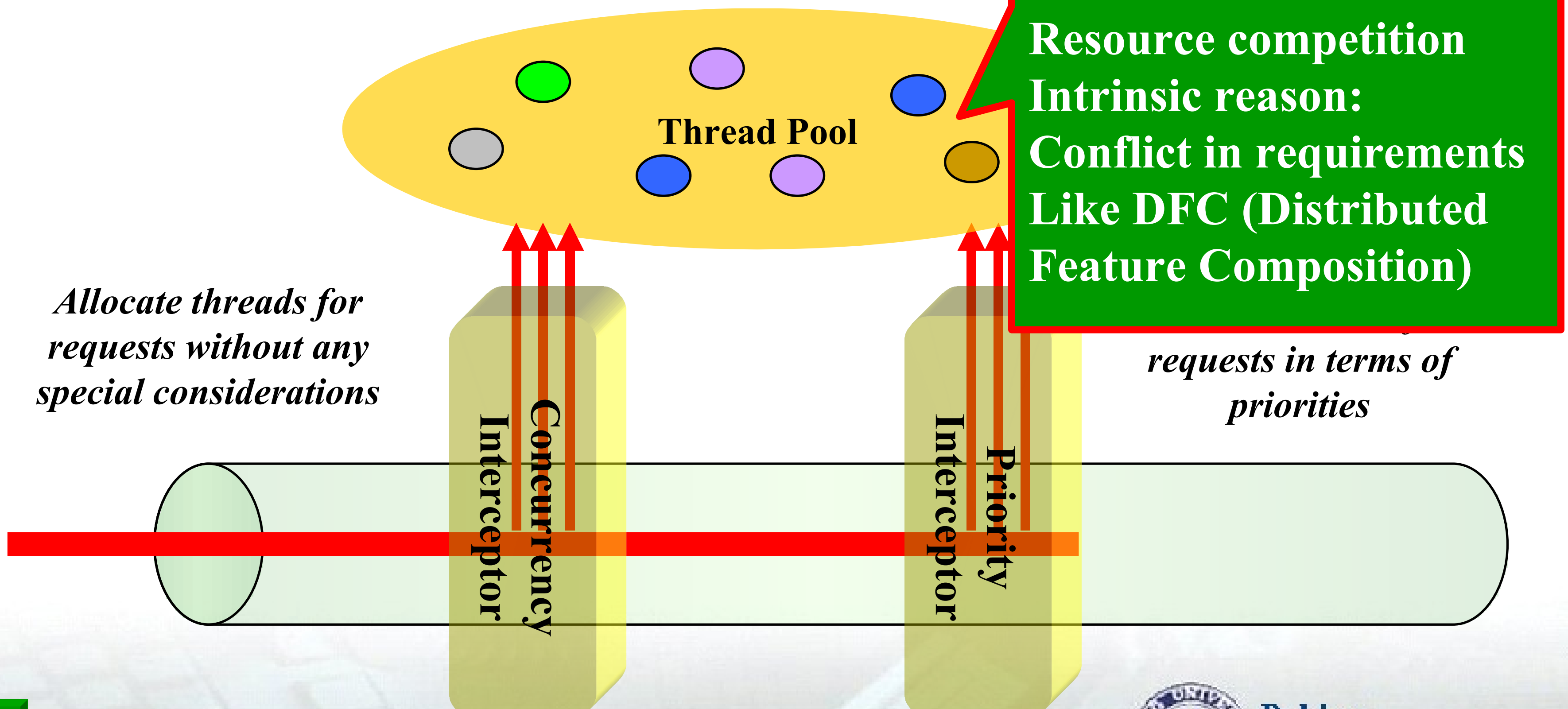


XML definition in ABC/ADL

Feature Interaction: a Case

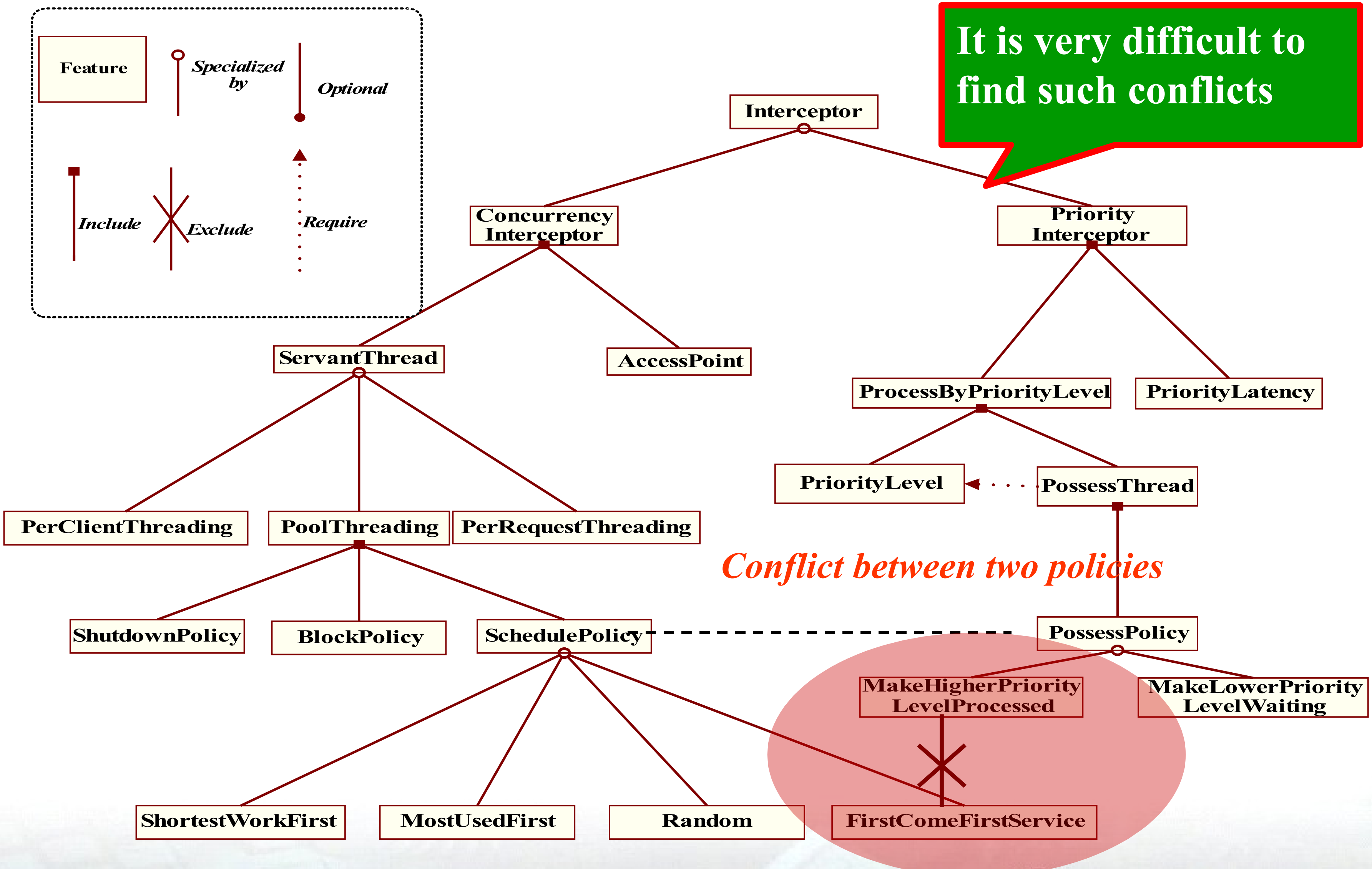
In this configuration

- If a request with high priority is coming, it can not be processed by CI until some threads are released. Then, PI cannot do any thing until CI allocates a thread to a request. Therefore, the function of PI is invalid



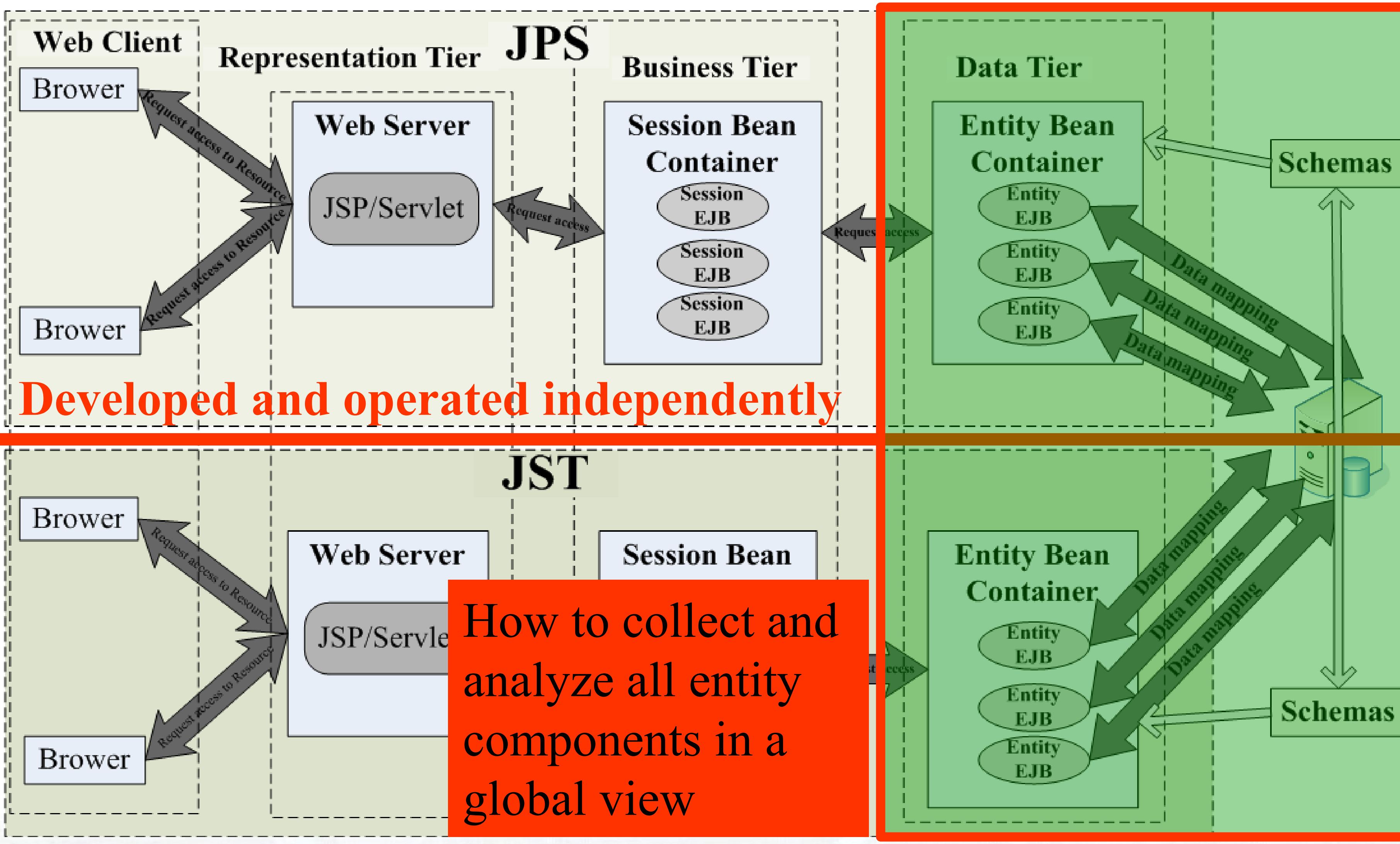
Interceptors

It is very difficult to find such conflicts



Peking University

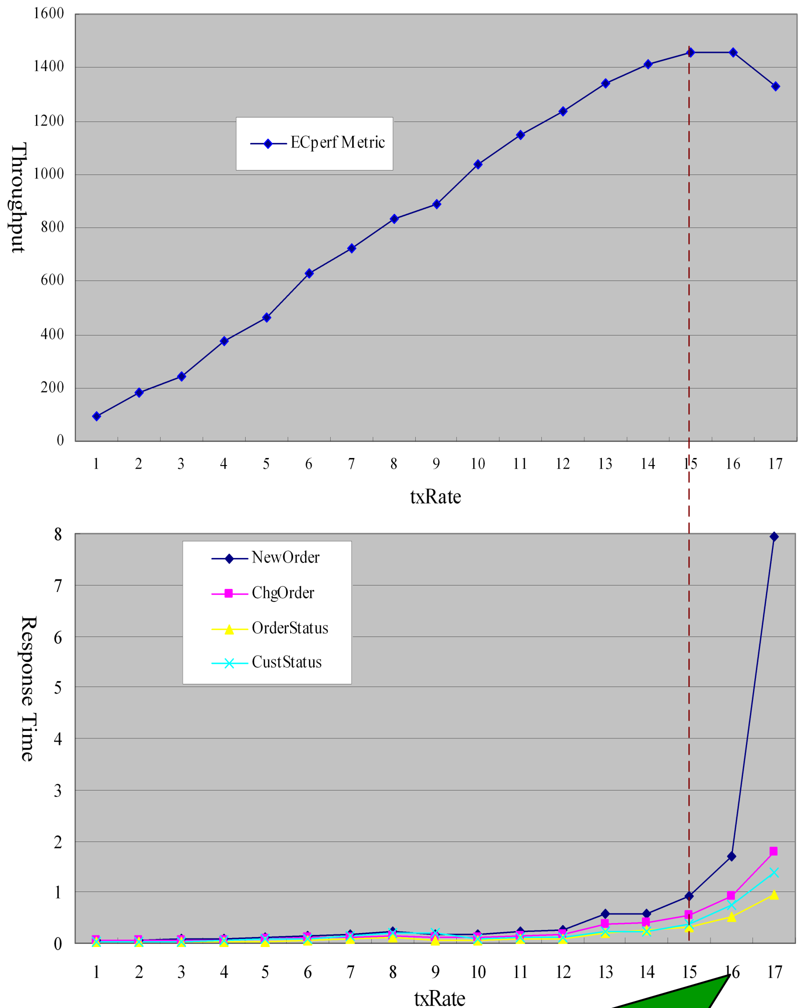
FI in Entity Components

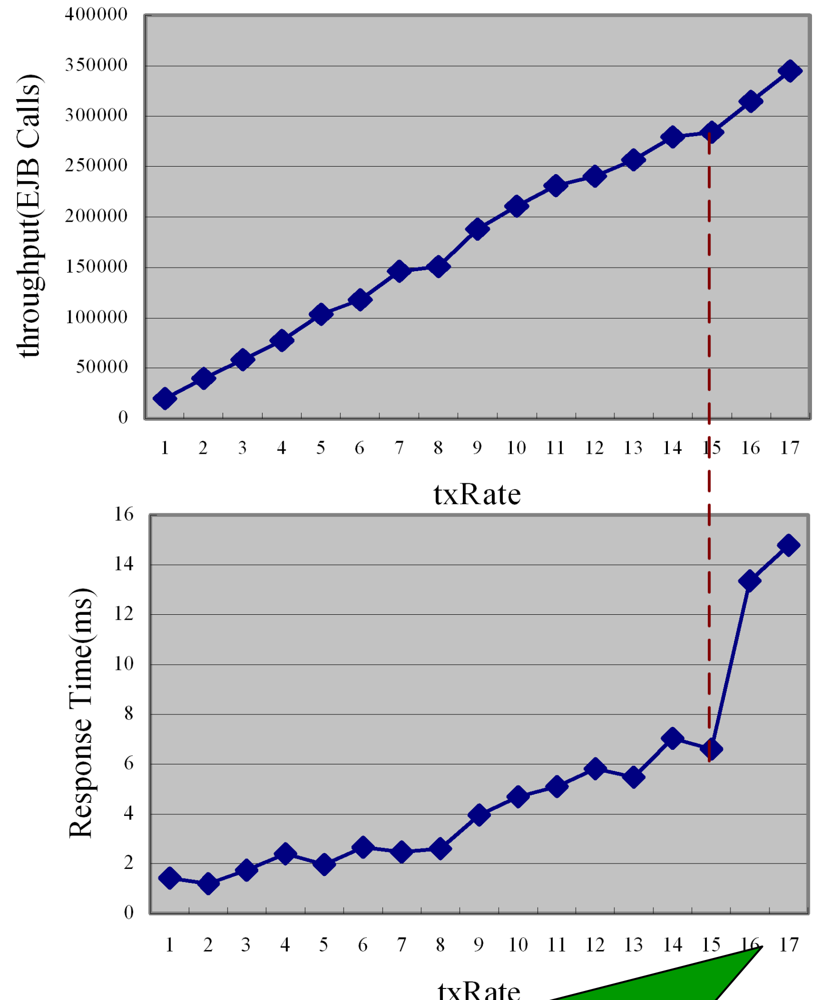


Developed and operated independently

How to collect and analyze all entity components in a global view

Auto Testing of ECPerf

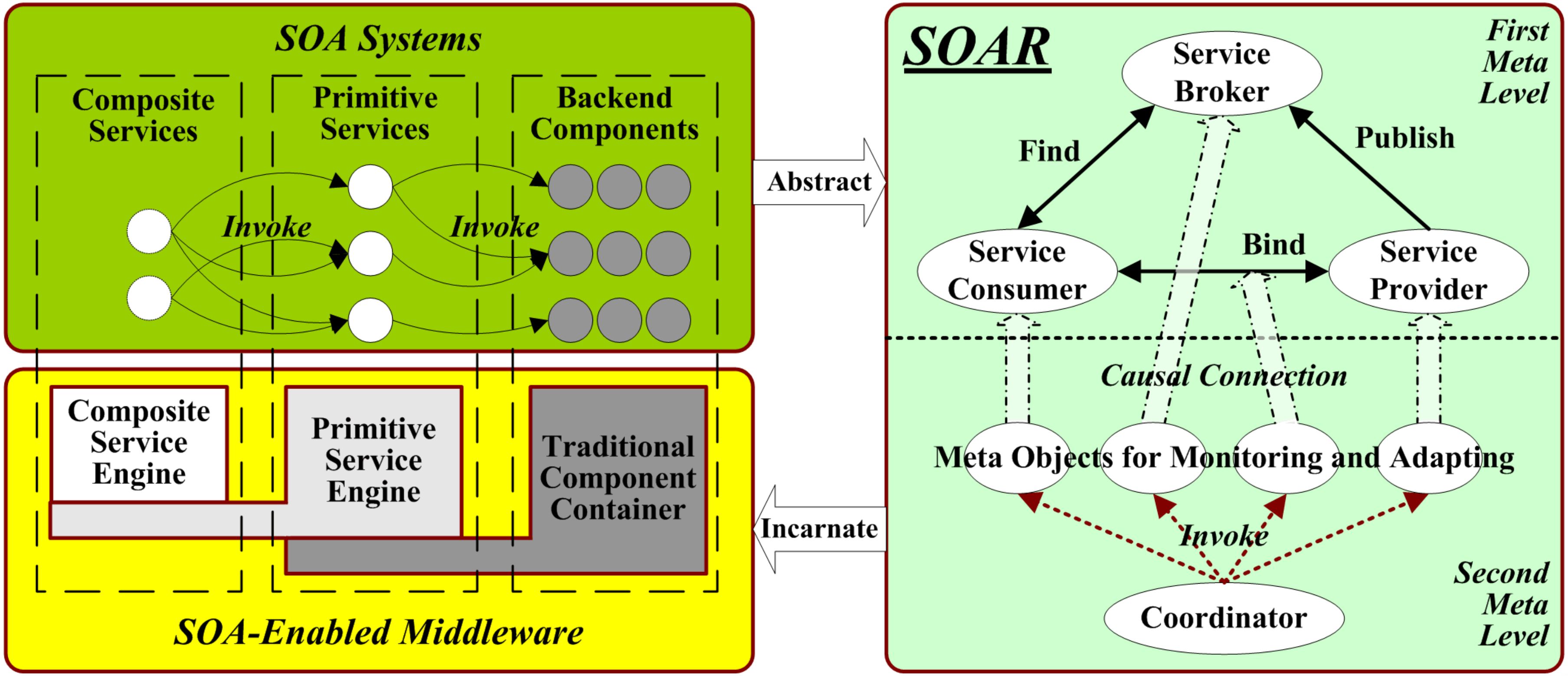




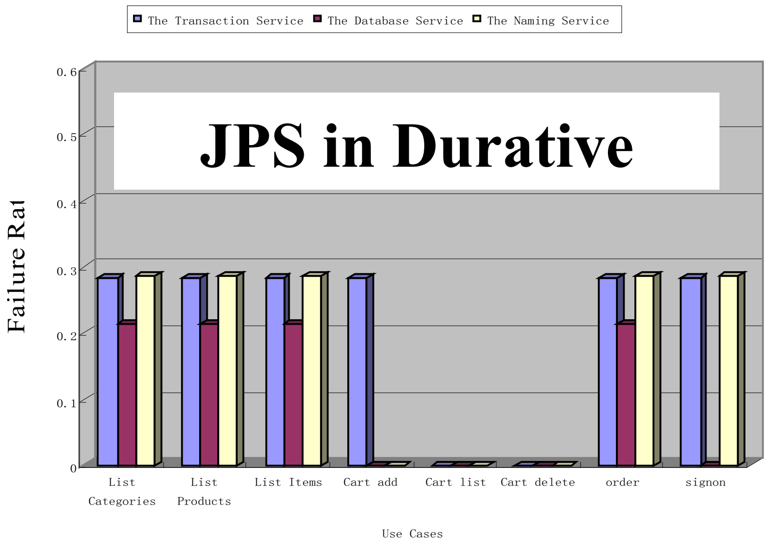
Test by hand:
30 x 3 x 17 = 1,530 minutes

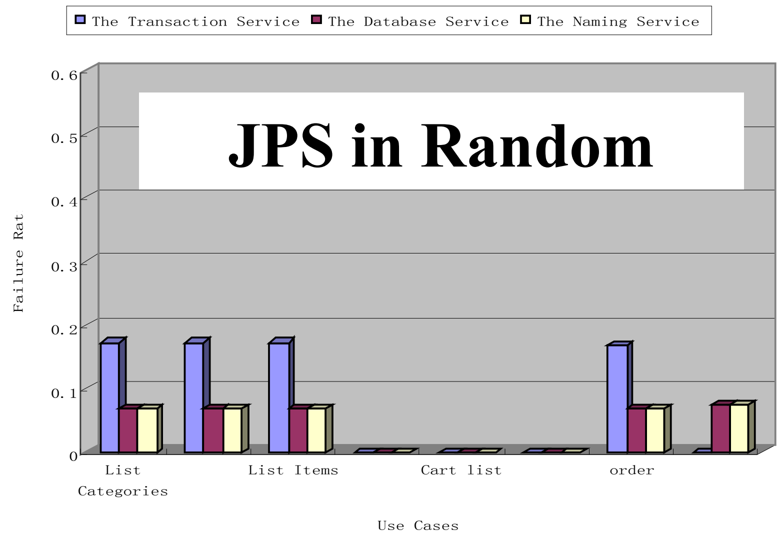
Test by self-optimization:
15 x 18 + 30 x 1 = 300 minutes

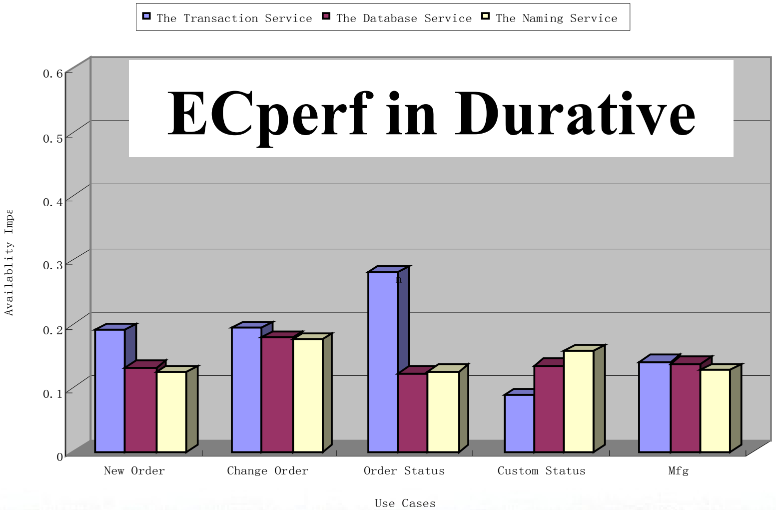
Service Oriented Architecture with Reflection

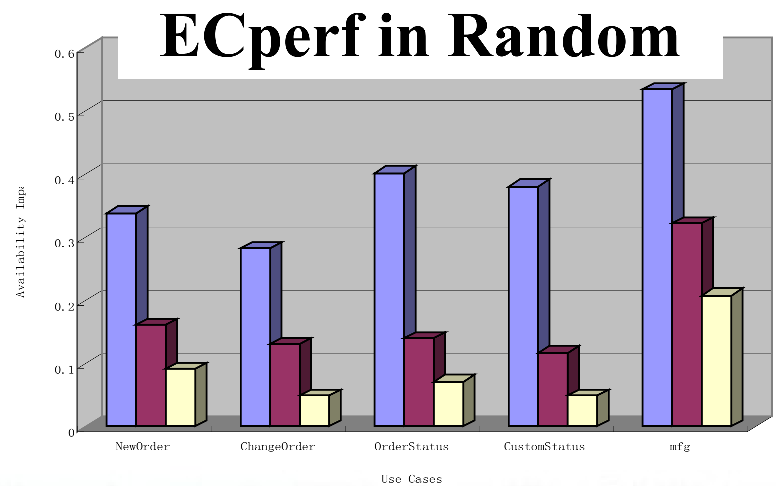


JARME





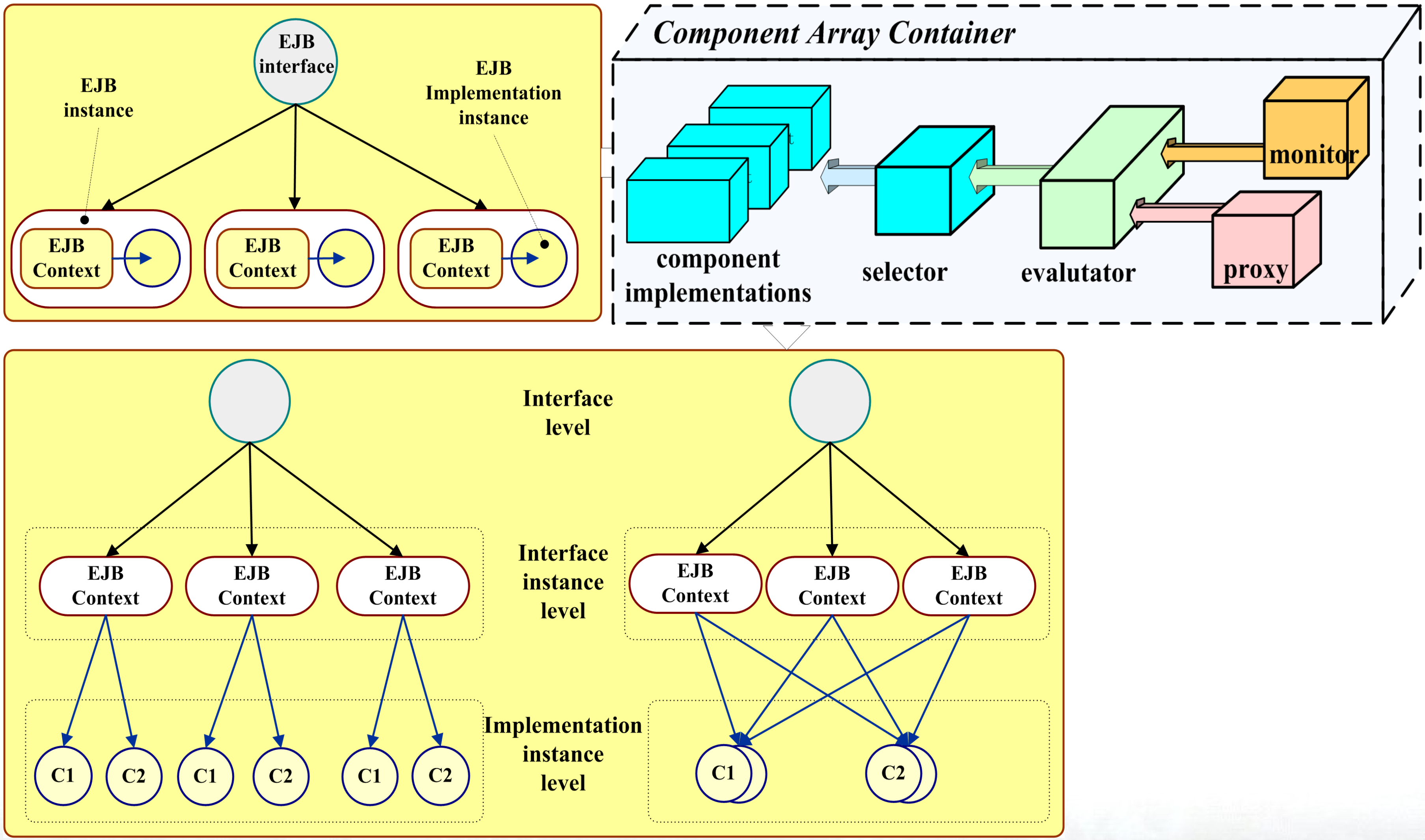




Transaction Service
Database Service
Naming Service

Peking University

Component Array



- "exo-kernel" of middleware architecture
  - 4 layers
    - Components (reflective)
    - Architectural frameworks (patterns, service elements)
    - System services
    - Personalities (standards, e.g., J2EE, CORBA, OGSI)