Towards Context-based Autonomic Services

Schahram Dustdar

Distributed Systems Group
Institute of Information Systems
Vienna University of Technology
Austria

http://www.infosys.tuwien.ac.at/Staff/sd
www.VitaLab.tuwien.ac.at
Who we are

- 30 + people, several spin-offs

- Currently 12 active research projects
  - 6 EU funded projects (IST, ITEA, @LIS, eContent)
  - 6 National funded projects (FWF, FIT-IT)

- Innovation 2006
  - **JBoss** Innovation award for BPM/services integration
  - **Microsoft** Imagine Cup winners (Austria) and finalists New Delhi
  - **Web service Challenge** 2nd places at SCC 2006 and CEC 2006
  - **E-Governance Best Practice Award** by the EU
Foundations of Service-oriented Systems

- Service Registries
  - UDDI is not enough. How to discover, publish, and select services? How about transient service providers?

- Dynamic composition (and dynamic binding)
  - QoS-enhanced dynamic service composition and process rewriting techniques
  - Context-based composition

- Run-time monitoring

- Recovery strategies
  - Availability, Reliability, Dependability
  - Self-* properties of services

- Model-driven service composition
  - Process-driven Service-oriented Architectures
Vienna Service Infrastructure

- Dedicated 10 Blade servers with each 2 GB RAM and 2 Xeon 3.2Ghz CPUs, 1 TB storage

- Monitoring, Dynamic Composition, Orchestration, Choreography of services

- Open Testbed
Outline

Understanding today’s complex information systems
- Assumptions
- Complexity, Interaction, Autonomy
- Software- and Teamwork Evolution
- Emerging Team Forms

Key concepts
- Collaborative Activities & Web services
- Autonomic Services
- Mining
- Context Tunneling

- Conclusion
VITALAB Assumptions

- **Devices**
  Smaller/Faster/Cheaper/Always on:
  - Performance, Communications, Integrative

- **Communications and Coordination**
  Pervasive “Anytime-Anywhere“ infrastructures and Mobile Computing/Communications
  - Efficient usage of resources

- **Open dynamic ecosystems**
  - Autonomous systems
  - People and software services being integrated into evolving “solutions“ – often they fulfill critical societal missions
Complexity, Interaction, Autonomy

- Heterogeneous systems increasingly connected
  - Integration becomes more complex

- Software – and Hardware-Architects cannot plan for all potential interactions upfront
  - Increased interaction dynamics of systems (people and software services)

- Autistic software vs. Autonomic software

- Monitoring and Management of Internet-scale infrastructures becomes paramount
  - Autonomic & Services Computing including e.g.,:
    - Self-Healing
    - Self-Configuring
    - Self-Optimizing
    - Self-Protecting
Software Evolution - some lessons learned

- Requirements cannot be fully gathered upfront
- Requirements cannot be frozen
- Requirements intrinsically decentralized, complete control and pre-plan illusory
- When changed, impact whole product/process
- Evolution is intrinsic to software
  - it is NOT a “post-delivery” nuisance
Teamwork Evolution – some lessons learned

Team Configuration

- Flexible
- Stable

Time span
- Long-lived
- Short-lived

- Classic Teams
- Nomadic Teams
- Nimble Teams
- Virtual Teams
Where are the sources of change?

- Changes originate in the interaction with the physical environment (run-time)

- Implied by pervasive/ubiquitous computing requirements
  - mobility and context awareness
  - ambient intelligence and disappearing computers
    - external world changes unpredictably
      - because of context changes
      - because new active objects are encountered
The "SoC approach"

- Support **active objects** providing service, such as
  - taggable objects (e.g., RFID)
  - artifacts
  - sensors and sensor networks

- Ability to deal with **context changes and unanticipated events and changes**
  - self-* behaviors:
    - e.g., self-adapting, self-organizing

- Challenges: Dependencies between parts of systems are no longer fixed and predetermined
  - Human team activities and team forms
  - Software services
  - Interactions between teams and services
Example use case - Context Used

- **Get Users**
  - Based on structure in activities (e.g., activity owner)
  - Based on patterns (frequently active users in particular activity)

- **Get Location**
  - User location information from past activities that were held in similar context (e.g., all meeting locations related to WP2 meetings)

- **Get Documents**
  - All documents that have been created in similar activity (e.g., sub-activity at same level)

- **Context Tunneling**
Understanding today’s complex information systems

- Assumptions
- Complexity, Interaction, Autonomy
- Software- and Teamwork Evolution
- Emerging Team Forms

Key concepts

- **Collaborative Activities & Web services**
  - How can process knowledge be utilized in ad-hoc collaboration?
  - Who is the expert who provides the service?

- Autonomic Services
- Mining
- Context Tunneling

- Conclusion
Activities & Services

a1… basic activity: Atomic building blocks for composite activities

a2, a3 …complex activities: composed of basic and complex activities. Can be used as container for sub-activities, annotations, context, services used, constraints.

- Activities may be long-running
- Error handling and rollback
  - Undo
  - Compensation
Use Case: Work on Share Document

- Metadata part of document
- Metadata -> interactions
- Used for workflow mining and pattern detection
Humans can publish themselves as services, e.g.:
- Review service
- Consulting service (consultant pattern)

These services are integrated into activities, e.g.:
- "send for review"
- "get expert opinion"

New opportunities for (human) interaction pattern
discovery through improved semantics
Activity-based Discovery

- Service lookup and selection influenced by activity context (e.g., status pending)

- Activities status halted until fitting service available
  - Example: person arriving at particular location (raid use case)

- Active Registry
  - Holds context information
Activities and Web services

Reuse activities and design complex activity

Create new activity (template) and map to service(s)

Complex activity

Description and Service Mapping

XML description of possible activities

Register Service

Add new service

Service B

Human Service Provider
Understanding today’s complex information systems
- Assumptions
- Complexity, Interaction, Autonomy
- Software- and Teamwork Evolution
- Emerging Team Forms

Key concepts
- Collaborative Activities & Web services
- Autonomic Services
  - How does context impact service execution?
- Mining
- Context Tunneling

- Conclusion
Autonomic Services

- **Monitoring** (Observations): logging, sensors, and event hooks
- **Analysis**: mining and event processing
- **Context** information: user (=consumer) context & service execution context
- **Reasoning & Prediction**: context models, rules patterns, event correlation
- **Planning & Execution**: user driven (activities), recommendations (mining/models), policies
- **Short vs. Long-term Impact**
  - short-term: Context for autonomic service adaptation
  - long-term: Mining results and patterns
Autonomic Service Adaptation

Services react to changes or anticipate changes

- Based on context-information (e.g., degradation of QoS)


- Based on activity patterns (mining of activities)


- Based on service mining (e.g., mining of service dependencies)

Outline

Understanding today’s complex information systems
- Assumptions
- Complexity, Interaction, Autonomy
- Software- and Teamwork Evolution
- Emerging Team Forms

Key concepts
- Collaborative Activities & Web services
- Autonomic Services
- **Mining**
  - How can ad-hoc processes be recorded?
- Context Tunneling

- Conclusion
Different scopes/levels for mining

1. Individual Services
2. Service Selection (A or B)
3. Service Dependencies (a set of services is always used in combination with each other)
4. Workflow Mining (activities in a process)

Finding Patterns in Ad-hoc Team Interactions

- **Proxy**
  - 1:1 relation to original
  - e.g., secretary, assistant

- **Broker**
  - e.g., person who is responsible to answer all client requests

- “Master/Slave”
  - sending identical requests to multiple recipients
  - e.g., multiple participants are requested to state their cost estimates

- More patterns...

Outline

Understanding today´s complex information systems
- Assumptions
- Complexity, Interaction, Autonomy
- Software- and Teamwork Evolution
- Emerging Team Forms

Key concepts
- Collaborative Activities & Web services
- Autonomic Services
- Mining
- **Context Tunneling**
  - How can context be reused across activities/services?

- Conclusion
Context Tunneling – Context Scopes

- Tunneling is based on three notions of context scopes:
  - Individual Context
  - (Complex) Activity Context
  - Team Context

- We need “integration” of context (e.g., individuals collaborate on shared activities)
Context Model (still evolving)

- Central to the model: Activities
- Packages for domain independent and domain dependent models -> used for abstraction and extensibility
- Packages include:
  - Location
  - Device
  - Activity
  - User
Conclusion

- Activity-oriented Services (no more rigid “applications“)
- Collaborative Activities & Web services
- Autonomic Services
- Interaction Mining and Pattern detection of activities and services
- Context Tunneling (reuse across services/applications)
Thanks for your attention

Prof. Dr. Schahram Dustdar
Distributed Systems Group
Institute of Information Systems
Vienna University of Technology

http://www.infosys.tuwien.ac.at/Staff/sd/
http://www.VitaLab.tuwien.ac.at