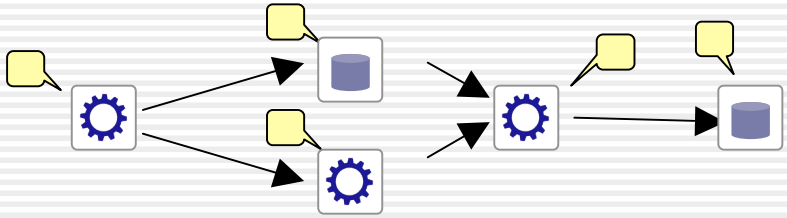


A Platform to support Decentralized and Dynamically Distributed P2P Composite OWL-S Service Execution*

Thorsten Möller, Heiko Schuldt

University of Basel, Computer Science Department, DBIS
Bernoullistrasse 16, CH-4056, Basel, Switzerland

Workshop *Middleware for Service Oriented Computing*, November 26th 2007
Middleware 2007, Newport Beach, CA, USA

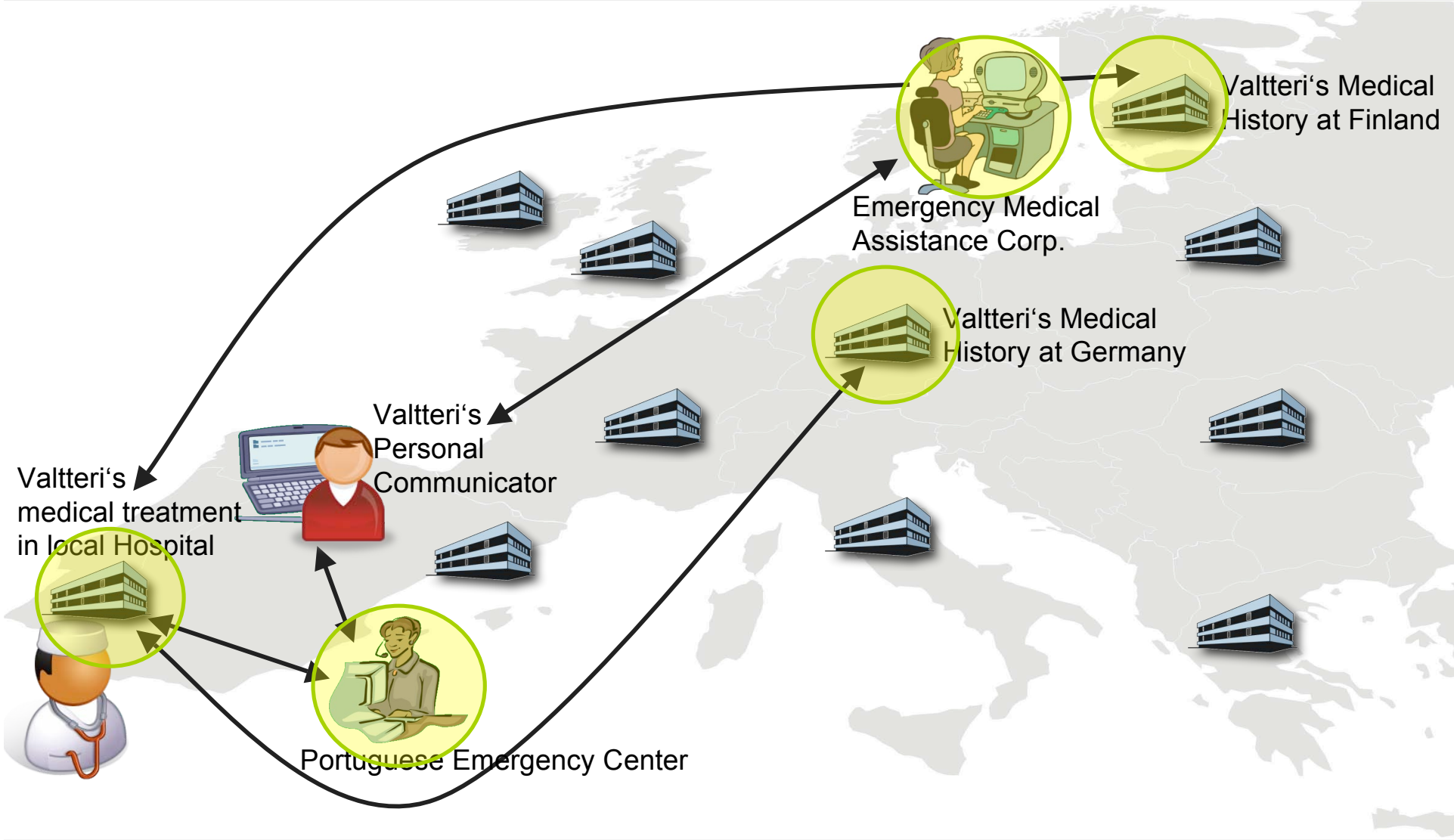
* Supported by the EU in the 6th Framework Program within the STREP CASCOM

Why do we want to have a system that is able to ...

... execute semantic composite services, and

... scales up to the Web size?

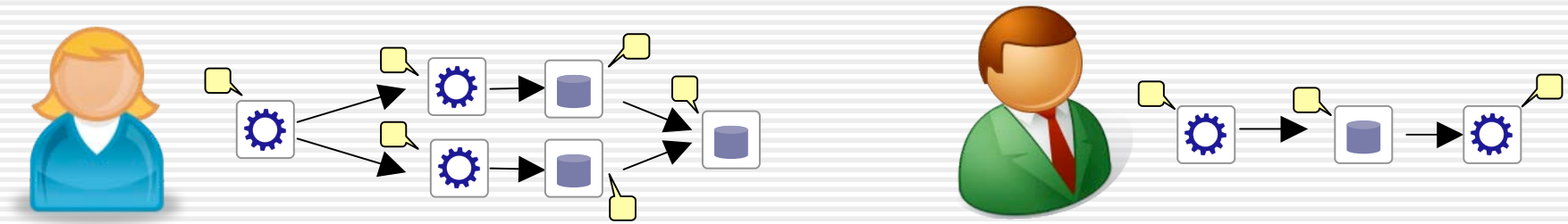
A practical Healthcare Scenario



What changes?

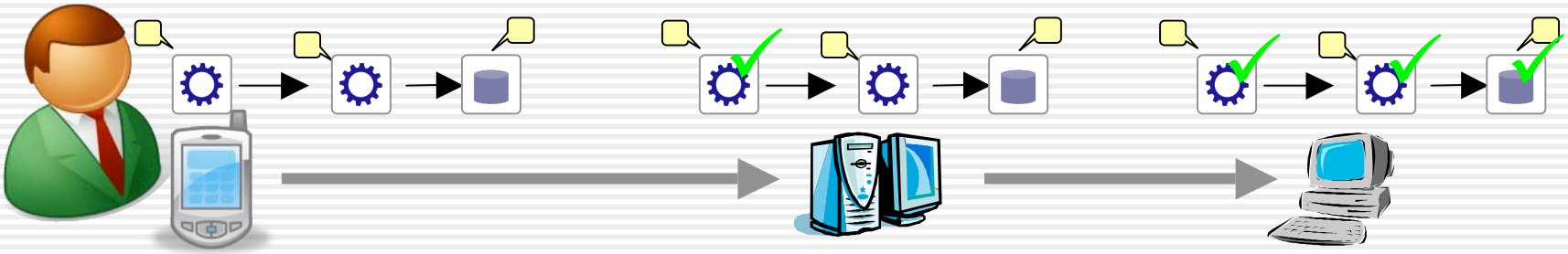
- Ad hoc created applications out of available services
  - Processes to represent workflows of applications
  - Created individually for each user (the context of a user)
  - Instantiated very view times

➔ Dynamic Service Composition (planning)

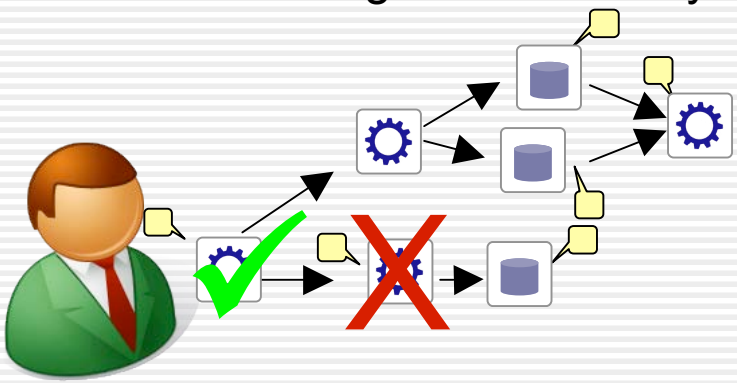


What is special?

- Vast number of users and services - requires scalable approach
  - Centralized system not appropriate
  - Migration of process instances during execution



- Users (and services) come and go
  - Sophisticated failure handling – semantically equivalent executions



Some CASCOM facts

- EU funded research project
- 8 partner (3 industry, 5 research)
  - Adetti, Lisbon, PT
  - URJC, Madrid, ES
  - EPFL, Lausanne, CH
  - UniBas, Basel, CH
  - FRAMeTech, Parma, IT
  - DFKI, Saarbrücken, D
  - TeliaSonera, Helsinki, FI
- In cooperation with
  - TILAK, Innsbruck, AT
  - Health@Net, AT
- September 2004 - December 2007

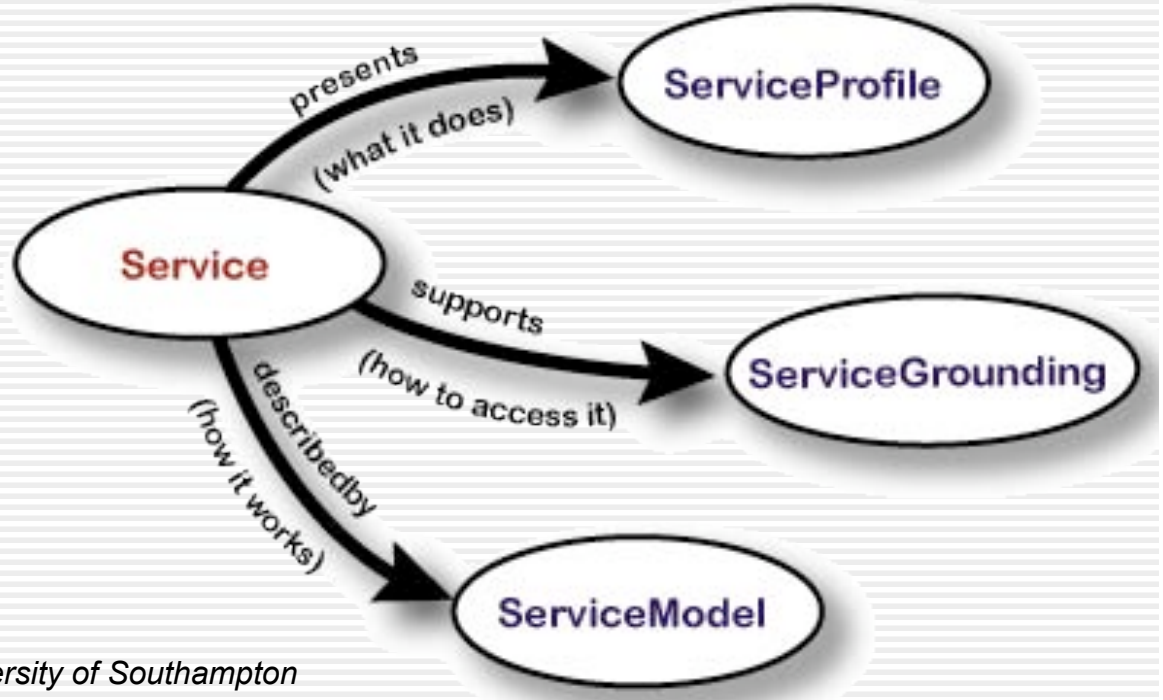
Focus on Healthcare application domain.

Overview ...

1. Background
2. Semantic Services, Processes, and OWL-S
3. Our Approach to Semantic Process Execution
    1. Process Management Systems
    2. Distribution Strategy
    3. Forward Failure Handling
4. Results
5. Conclusion & Outlook

OWL-S

- General purpose framework to:
  - Describe Web Services
  - Support automation of service management
  - Build on existing Web Service standards
  - Support entire lifecycle of service tasks
- Submission to the W3C in November 2004



Ontology image compliments of Terry Payne, University of Southampton

Semantic Service Profile

SemanticServiceProfile := $(I, O, P, E, name)$

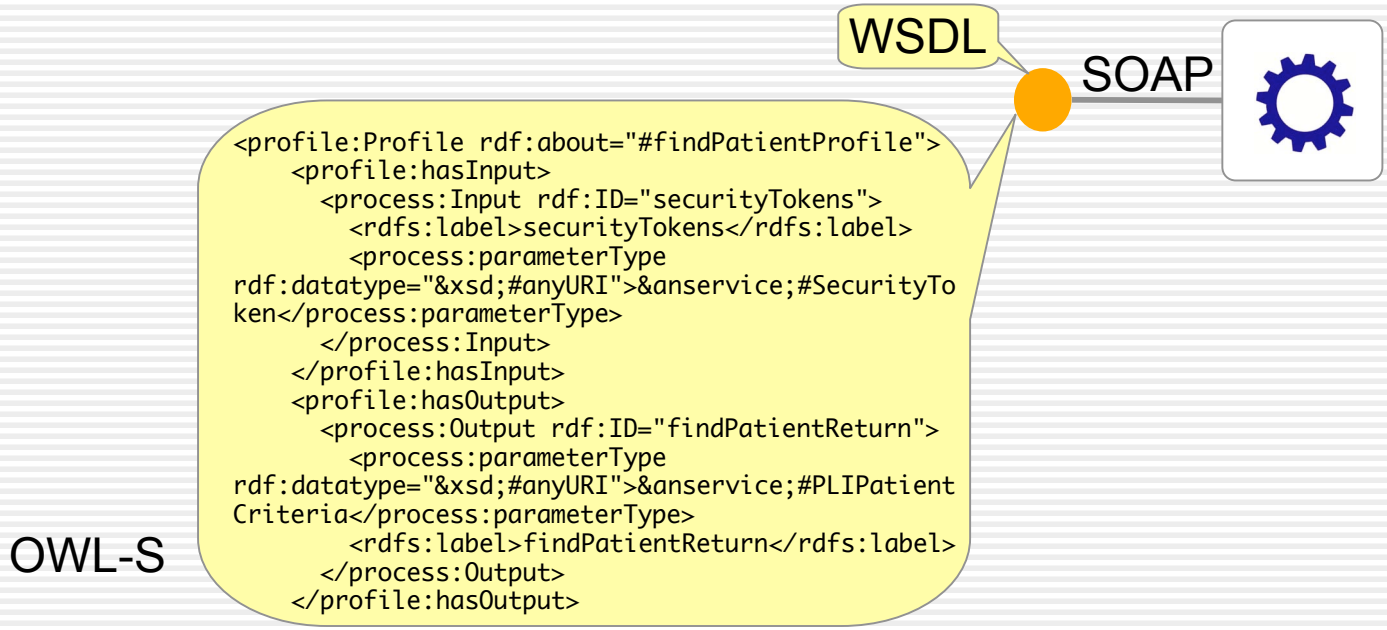
I	-	Set of inputs i, $\|I\| \geq 0$
O	-	Set of outputs o, $\|O\| \geq 0$
i, o	-	data message [of type t_i, $t_o \in T$]
		use semantic-aware Type system, Ontology (e.g. OWL)
P	-	Set of preconditions p, $\|P\| \geq 0$
E	-	Set of effects e, $\|E\| \geq 0$
p, e	-	logical formula using a logic language (e.g. SWRL)

A service consumes inputs, produces outputs, will not execute unless its preconditions are true, and when it does execute, it has various effects.

Example: "book flight", I = {$airport_{start}$, $airport_{target}$, person, creditCard}, O = {receipt}, P = {isValid(creditCard)}, E = {isCharged(creditCard, rate)}

Services & Semantic Services

Profile is made available, e.g. WSDL, OWL-S.
Access is done based on protocols, e.g. SOAP.

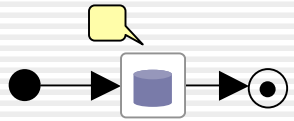


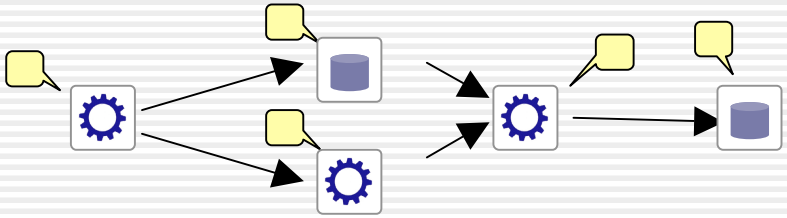
Composite Services / Processes

Primitive / Atomic Service versus Composite Service / Process

Primitive Service:

- Realizes **indivisible operation**: Web Service invocation





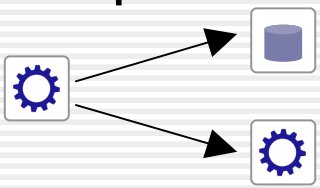
Composite Service / Process:

- Realizes well defined composition of multiple sub services.
- Defines order in which invocations are to be done: **Control Flow**
- Processing of inputs/outputs among the services defined by: **Data Flow**

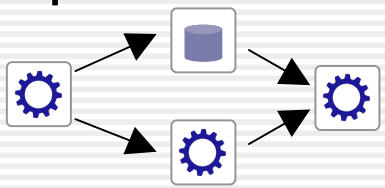
OWL-S Control Constructs

Sequence

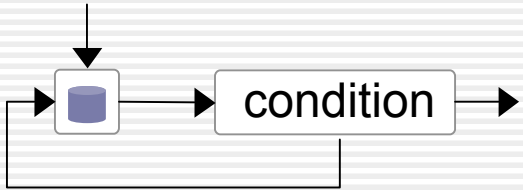
Split



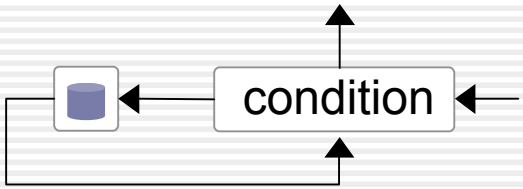
Split+Join



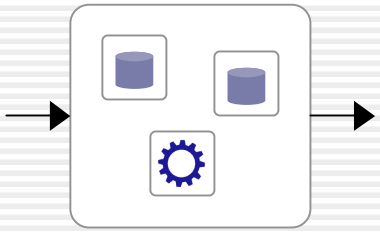
Repeat-While



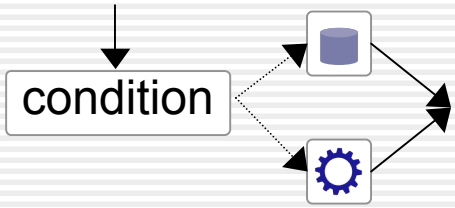
Repeat-Until



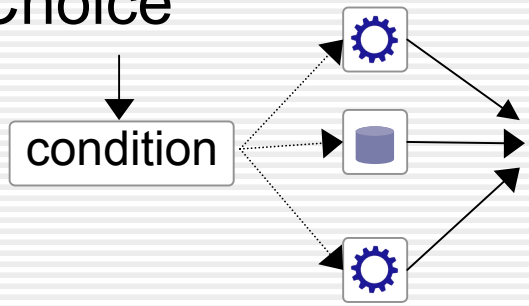
Any-Order



If-Then-Else



Choice



1. Background
2. Semantic Services, Processes, and OWL-S
3. Our Approach to Semantic Process Execution
    1. Process Management Systems
    2. Distribution Strategy
    3. Forward Failure Handling
4. Results
5. Conclusion & Outlook

Process Management Systems

Its duty is to:

- Work off the control flow while respecting its operational semantics
- Adhere to the data flow defined

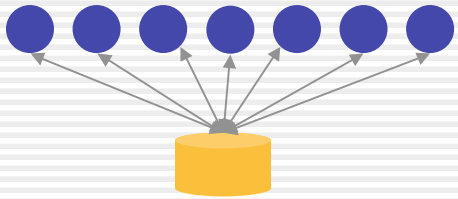
We also would like to:

- Provide guarantees for this procedure
  - Guaranteed termination
  - Correctness
  - Failure handling
  - Concurrency control

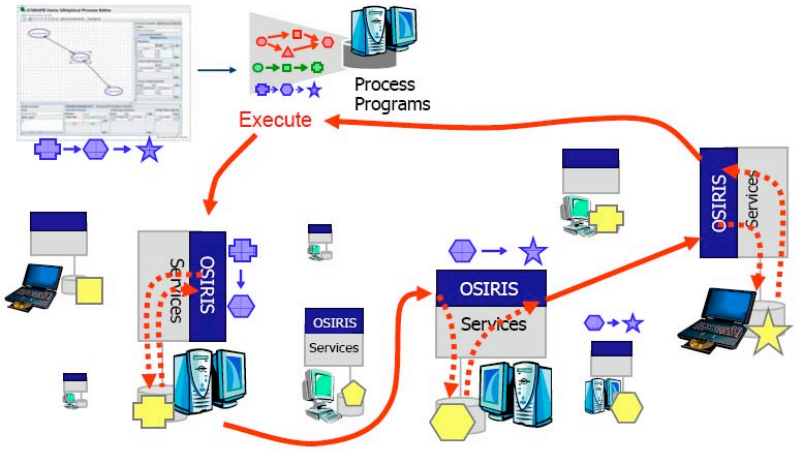
Distributed PMS

Different ways to organize distribution:

1. Execution nodes on top of centralized process instance store.
   - IBM WebShere



2. Execution nodes share metadata on processes and forward process instance among them according to control flow.
   - OSIRIS, Schuler et al, 2004 [1]
   - Nanda et al, 2004 [2]
   - Ye, 2006 [3]



3. One Execution node manages process instance at a time and may migrate it if beneficial to other node that continues.

Contribution

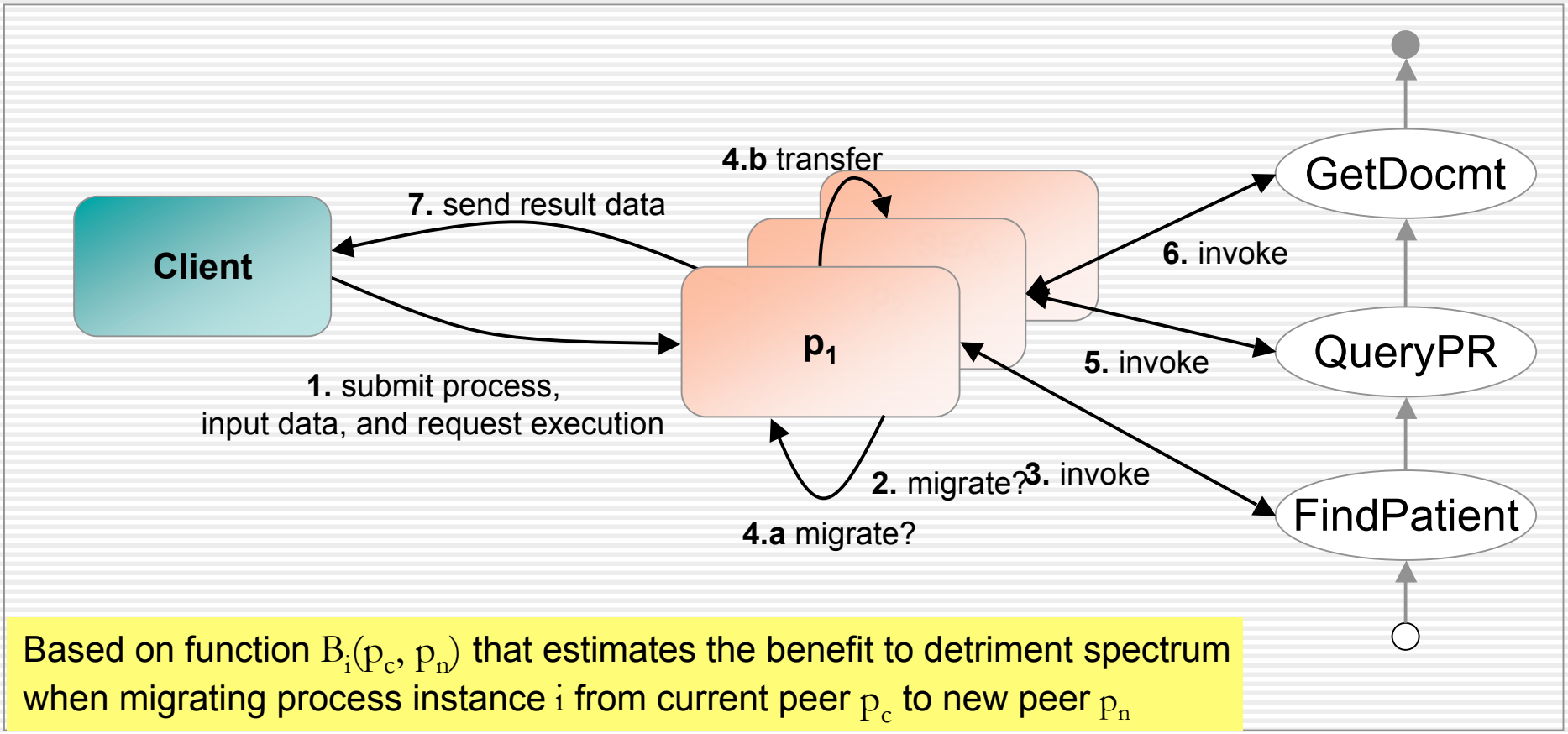
1. Method for execution of semantic processes that ...
   - Scales:
     - Dynamically distributed and decentralized ➔ macro level.
     - Wide range of computing devices: mobile phones up to high end computing devices ➔ micro level.
   - Taking into account:
     - Processes that are instantiated only once/very few times.
     - No need for additional software layers at service provider side.

2. Support for forward failure handling
   - Interfacing with planner component
   - Interaction Protocol

Developed as a continuation of the OSIRIS system.

1. Background
2. Semantic Services, Processes, and OWL-S
3. Our Approach to Semantic Process Execution
    1. Process Management Systems
    2. Distribution Strategy
    3. Forward Failure Handling
4. Results
5. Conclusion & Outlook

Dynamic Distribution



Based on function $B_i(p_c, p_n)$ that estimates the benefit to detriment spectrum when migrating process instance i from current peer p_c to new peer p_n

Benefit Function

General observation:

- Execution of the remaining part of a process instance i causes different costs at each peer; in terms of time and resource utilization
- Migration causes communication overhead / costs

$$cost_x \qquad cost_m \qquad (\text{co-domain} > 0)$$

$$B_i(p^c, p^n) = cost_x(c_{pc}) - (cost_x(c_{pn}) + cost_m(p^c, p^n))$$

$B_i : P \times P \rightarrow \mathbf{R}$

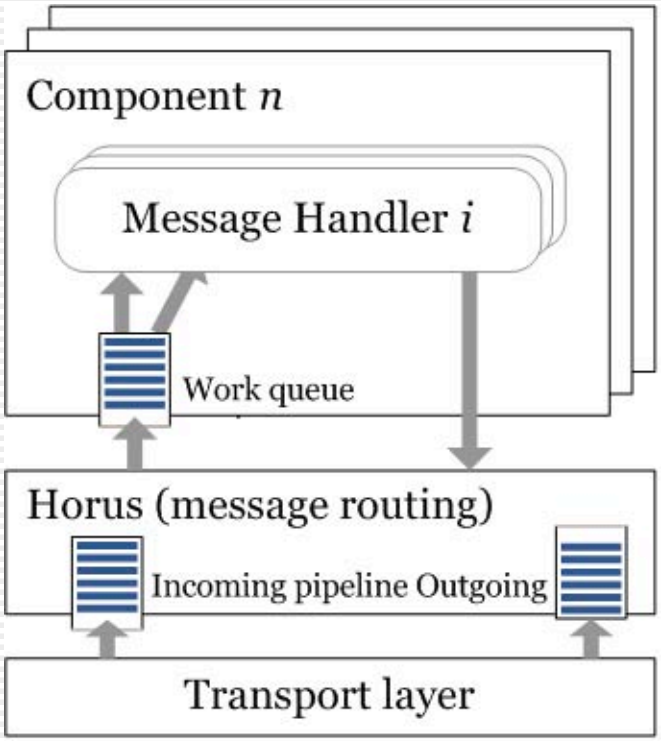
$p^c, p^n \in P$ (set of peers)

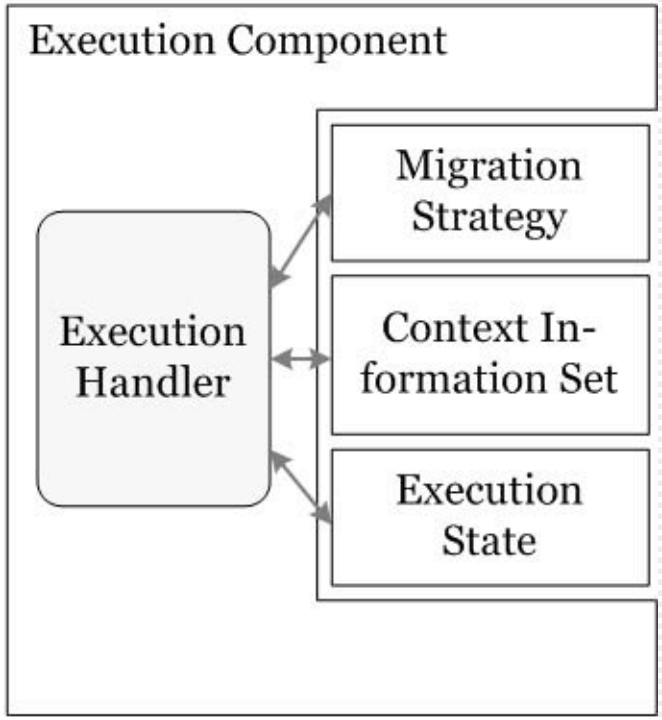
c_{pc}, c_{pn} context information set of current/new peer

$B_i<0$: adverse; $B_i=0$: can continue at either place; $B_i>0$: beneficial

Scaling at Micro Level

- Peer: internal multithreaded design, related to the SEDA approach*
- Allows for concurrent execution of several processes by each peer
  ➔ efficient use of available (computing) resources



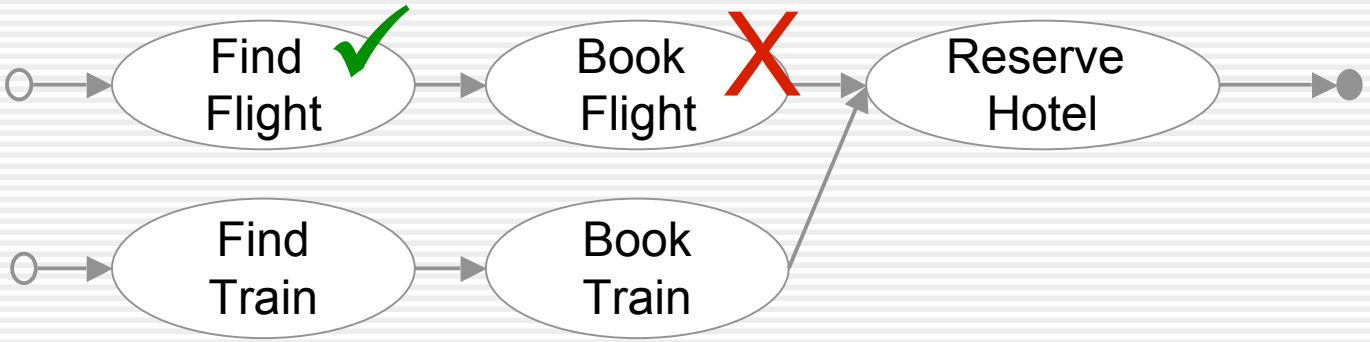


* M. Welsh. The Staged Event-Driven Architecture for highly concurrent server applications. Ph.D. Thesis, UC Berkeley, 2000.

Forward Failure Handling

Basic idea:

- Handling of service failures delegated to the planner (instead of classical rollback strategies)



- Approach:
  - Roll-back to a consistent state (if necessary)
  - Inform planner about current state (effects) and try to do a contingency re-planning ➔ planner might come up with semantically similar services
  - If successful, re-start/continue execution; otherwise, stop

The Rest of the Talk ...

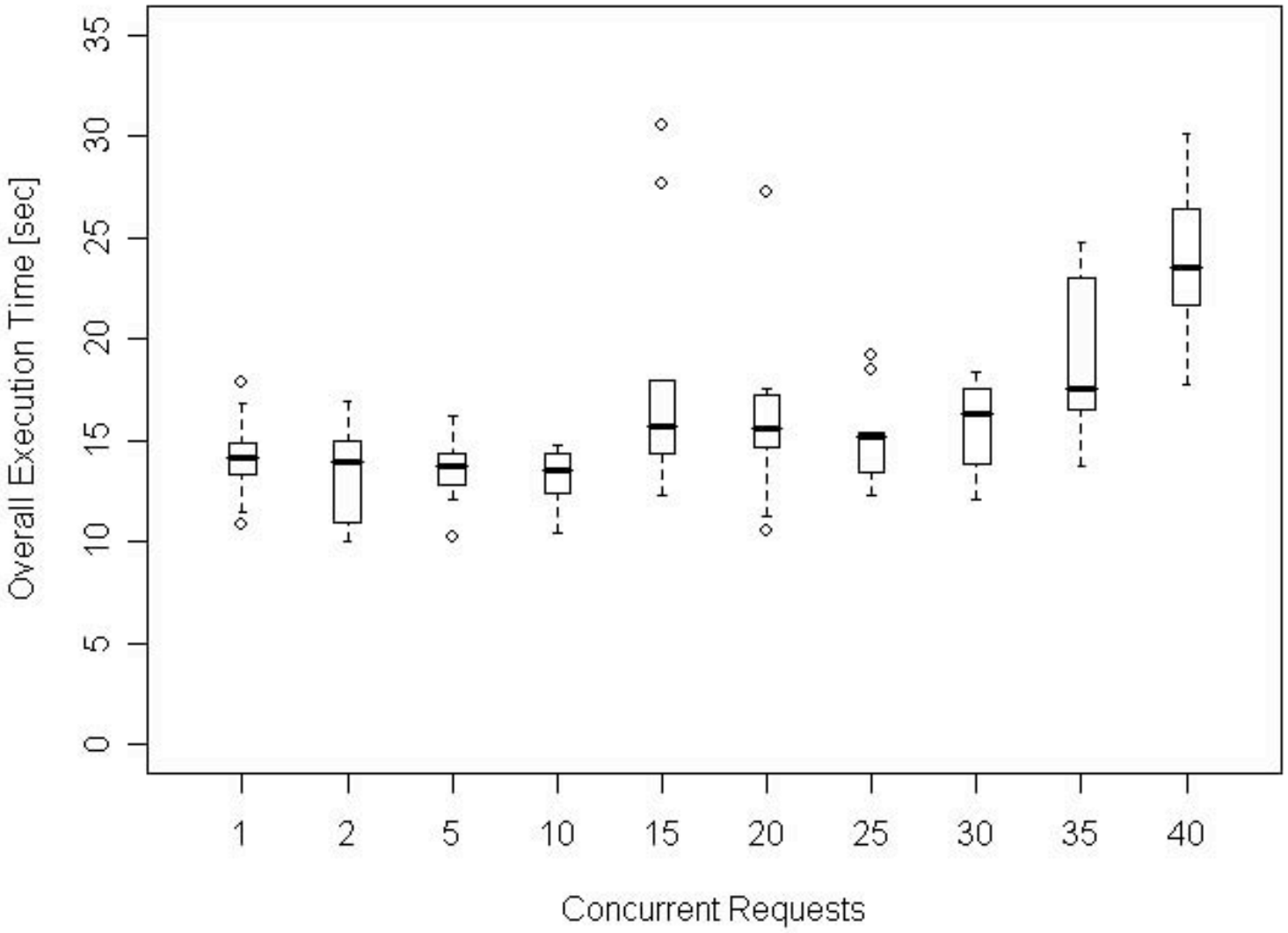
1. Background
2. Semantic Services, Processes, and OWL-S
3. Our Approach to Semantic Process Execution
    1. Process Management Systems
    2. Distribution Strategy
    3. Forward Failure Handling
4. **Results**
5. **Conclusion & Outlook**

Results

First quantitative evaluation:

- Analyze scaling characteristics at micro level (internal thread model).
- Measure overall execution time as a function of increasing load (concurrent execution requests)
  - Identical process, varying input data.
  - Standard Intel Pentium based hardware
  - Test rounds with 1, 5, 10, ... Concurrent requests
  - Each test round repeated 10 times.
- Uncontrolled test environment using real existing Web services.

Quantitative Evaluation



Qualitative Evaluation

- Qualitative Evaluation as part of the CASCOM system:
  - PMS used during several trial rounds in real healthcare environments
    - University Hospital Innsbruck TILAK (AT)
    - Christophorus Air Rescue Association Innsbruck (AT)
    - health@net project (AT): provide data from clinical information systems via Web Services
  - Using, for instance, processes to collect medical data from the health record of a person

The Rest of the Talk ...

1. Background
2. Semantic Services, Processes, and OWL-S
3. Our Approach to Semantic Process Execution
    1. Process Management Systems
    2. Distribution
    3. Forward Failure Handling
4. Results
5. **Conclusion & Outlook**

Conclusion & Outlook

- Prototype System implemented (in Java)
  - First distributed PMS for semantic processes.
  - Available open source soon (end of 2007).

- Model for dynamic distribution requires practical examples and further evaluation.
- Concept of forward failure handling by re-planning still in its infancy.
  - Potential for further optimization: “some” service failure types can be handled directly by the PMS without re-planning.

Thank you …

… Questions ?

References

[1] C. Schuler, R. Weber, H. Schuldt, and H.-J. Schek. *Scalable Peer-to-Peer Process Management - The OSIRIS Approach*. Proc. of the 2nd International Conference on Web Services, 2004, San Diego, CA, USA.

[2] M. G. Nanda, S. Chandra and V. Sarkar. *Decentralizing Execution of Composite Web Services*. Proceedings of the 19th annual ACM SIGPLAN Conf. on OO programming, systems, languages, and applications, 2004, Vancouver, BC, Canada.

[3] X. Ye. *Towards a Reliable Distributed Web Service Execution Engine*. Proc. of the IEEE International Conference on Web Services, 2006, Washington, DC, USA.