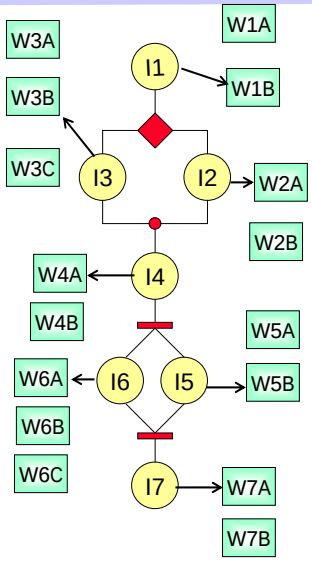
WS-Binder: a Framework to enable Dynamic Binding of Composite Web Services

Authors: M. Di Penta, R. Esposito, M.L. Villani,
RCOST – University of Sannio

R. Codato, M. Colombo and E. Di Nitto
CEFRIEL

Context

- Allow for dynamic binding in web service compositions
  - *Services can be discovered*
    - before execution and stored in lists
    - at runtime
  - *Bindings may vary*
    - for different users
      - selection based on user preferences
      - customized global QoS garantees (SLA)
    - over time
      - service availability may change
      - new services published



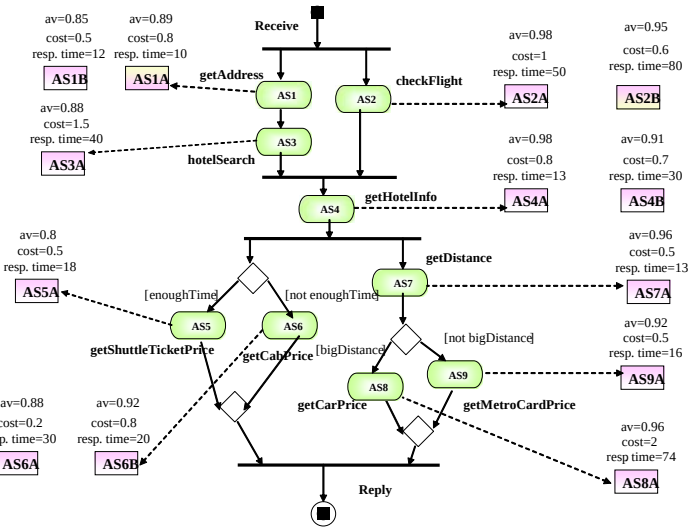
Features: Binding Types

- Two types of dynamic binding provided
  - Run-time local binding
    - What: determine each binding when needed
    - Why: to satisfy pre-defined local preferences
    - When: performed during process execution
    - How: by using context information
  - Pre-execution time global binding & re-binding
    - What: determine all bindings at once
    - Why: to satisfy pre-defined global QoS preferences
    - When: performed right before process execution (i.e., as latest as possible)
    - How: by using statistics
    - Bindings may change during execution if needed

Example

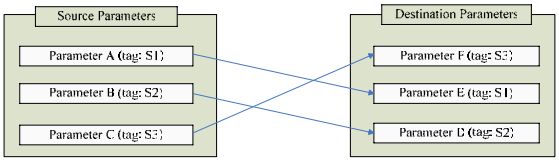
- Cost < 5$
  AND
- Resp Time < 2 min

- Maximize availability



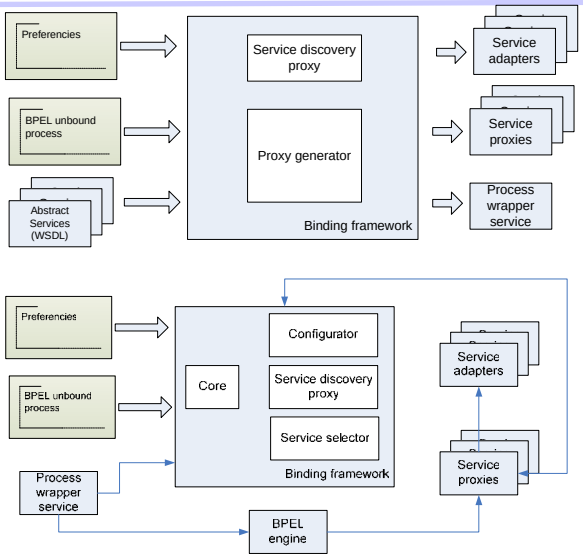
Service Interface Mapping

- Different operation signatures

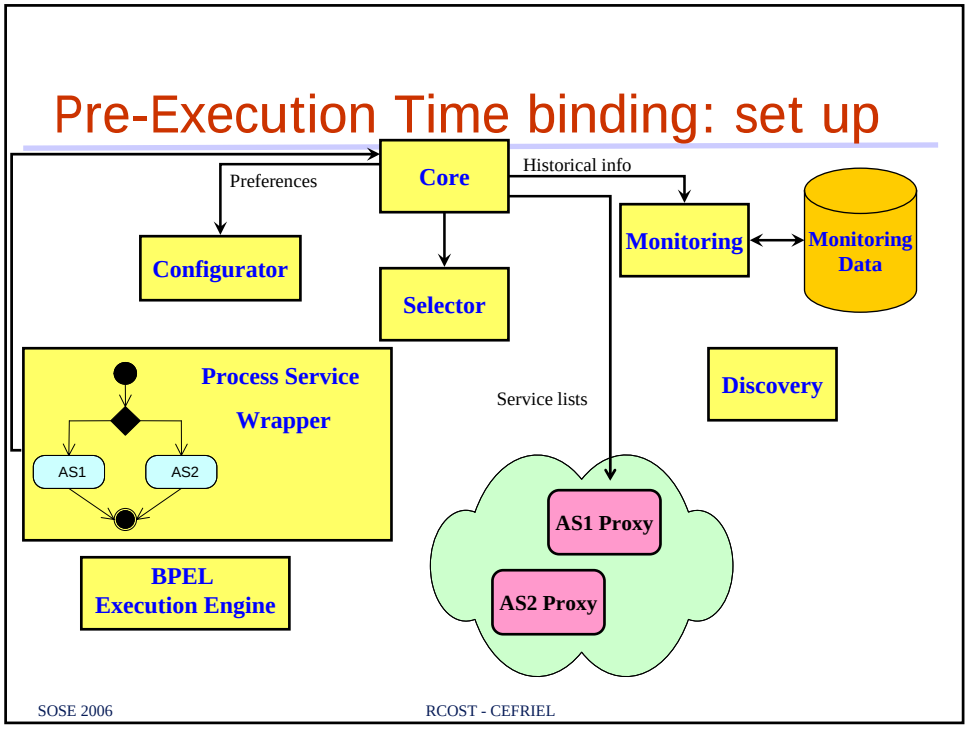


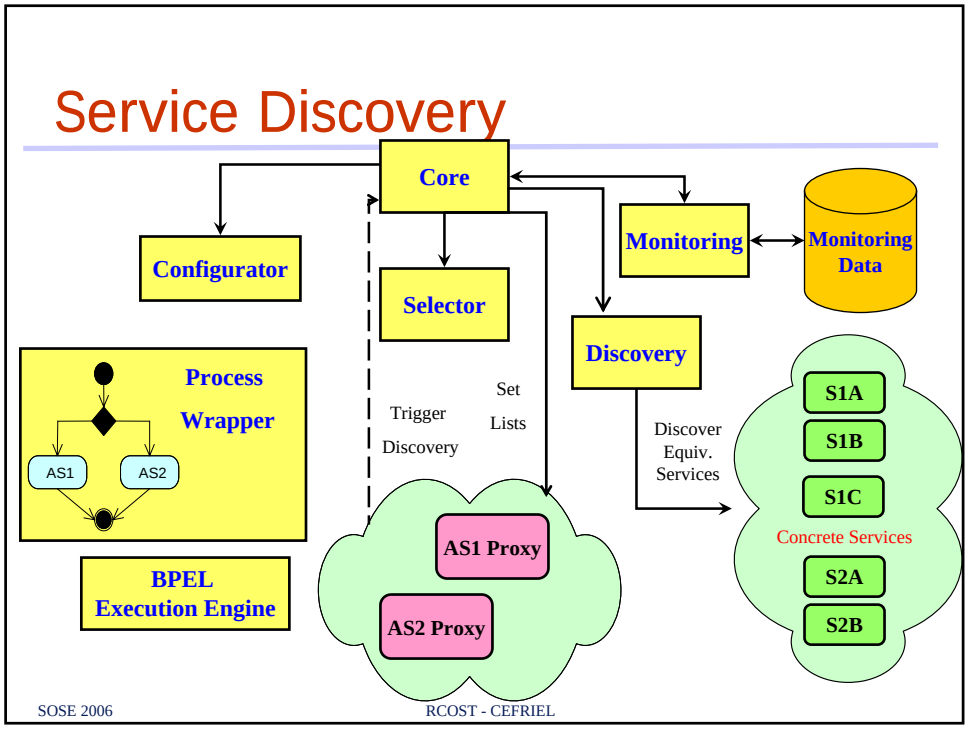
- Use of semantic facets
  - Input/Output parameters refer to ontological concepts
  - Mapping performed by adapters
    - A basic adapter looks for concepts match
    - Ad-hoc adapters may be introduced to enhance mapping by using concept relationships of the underlying ontology

Architecture

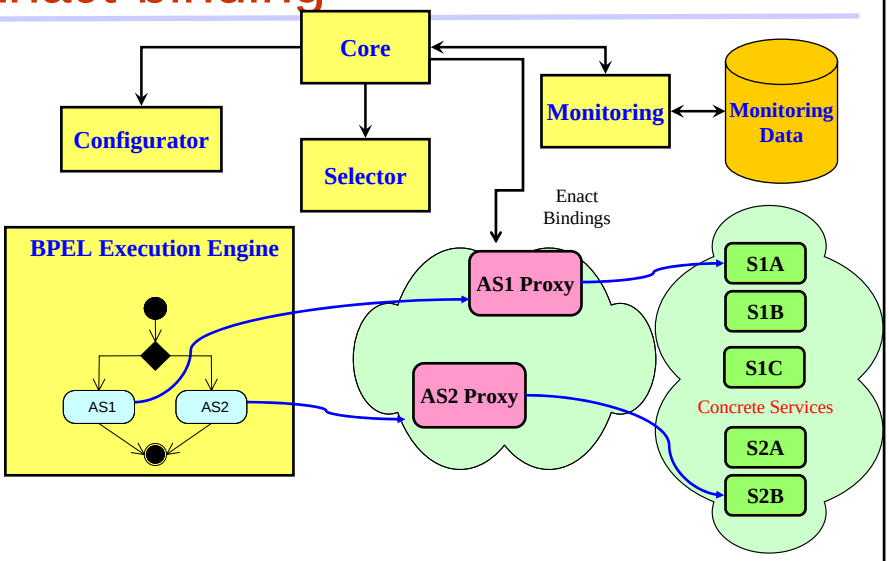


- Design time
  - Abstract services (WSDL)
  - BPEL process design
  - Preferences
- Deployment time
  - Process wrapper
  - Proxy services
  - Service adapters
- Execution time
  - Wrapper
    - Calls Binder
    - Calls engine

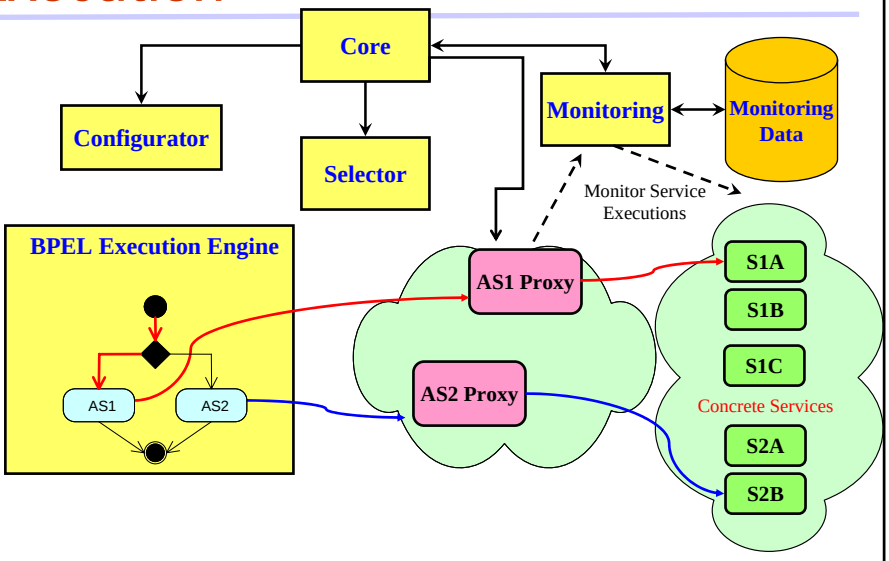

Pre-Execution Time binding: set up
Core
Historical info
Preferences
Configurator
Selector
Monitoring
Monitoring Data
Process Service
Wrapper
AS1
AS2
Service lists
Discovery
AS1 Proxy
AS2 Proxy
BPEL
Execution Engine
SOSE 2006
RCOST - CEFRIEL


Service Discovery
Core
Configurator
Selector
Monitoring
Monitoring Data
Discovery
Process
Wrapper
AS1
AS2
Trigger
Discovery
Set
Lists
Discover
Equiv.
Services
S1A
S1B
S1C
Concrete Services
S2A
S2B
AS1 Proxy
AS2 Proxy
BPEL
Execution Engine
SOSE 2006
RCOST - CEFRIEL

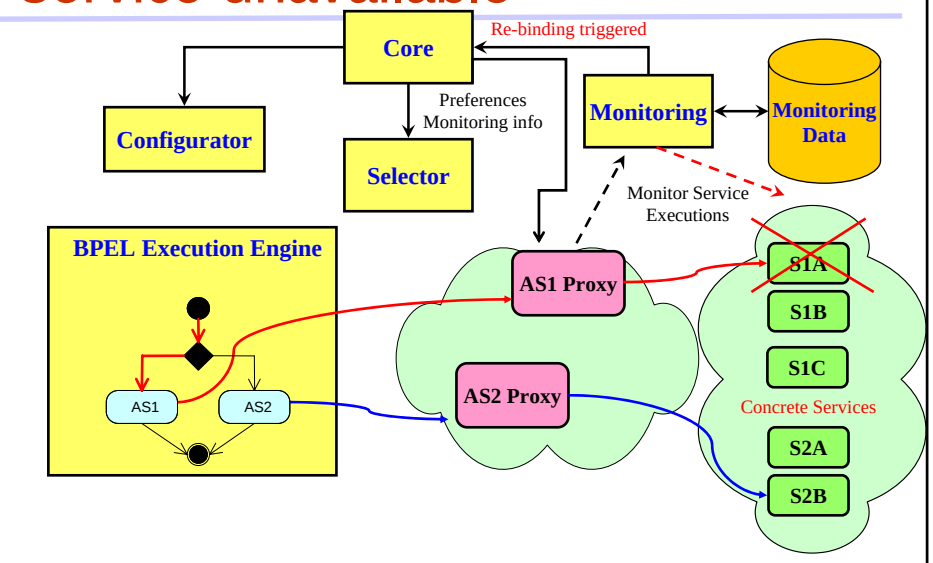
Enact binding



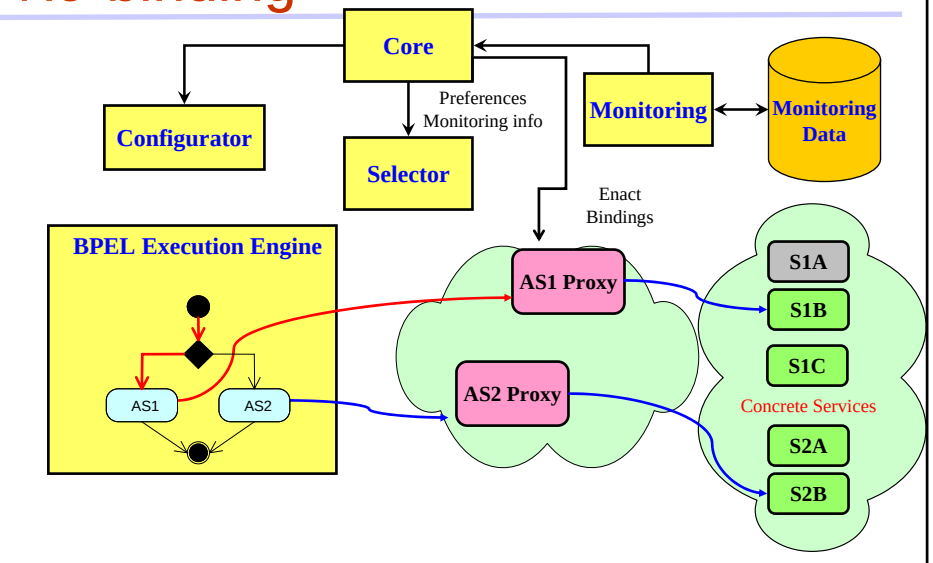
Execution



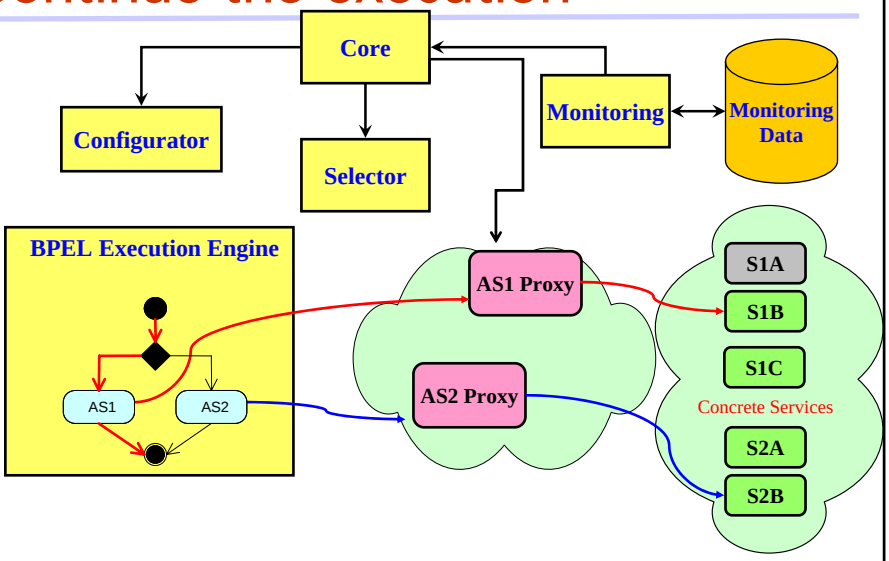
Service unavailable



Re-binding



Continue the execution



Selection algorithms

- Local dynamic binding approach
  - Selection according to algorithms referred to in the preferences

```
<selectionPreferences  name="" description="">
    <preference>
        <key>implementation</key>
        <value>LocalSelection</value>
    </preference>
    <preference>
        <key>alghoritm</key>
        <value>singleQualityRanking</value>
    </preference>
    <preference>
        <key>quality</key>
        <value>reliability</value>
    </preference>
</selectionPreferences>
```

- Pre-execution global binding
  - Estimate the QoS of concretizations
  - Use genetic algorithms (GA) to find the optimal solution
- Re-binding
  - Determine the re-binding slice
  - Apply the global binding approach to the slice

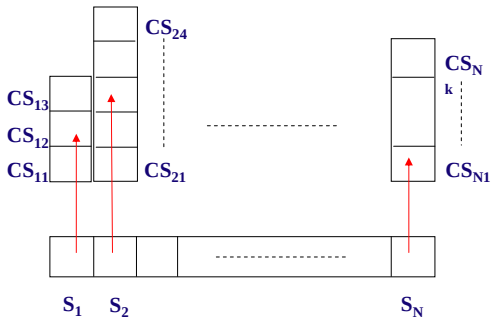
Workflow QoS estimation

- QoS Aggregation Functions
  - QoS attributes currently considered:
    - Cost
    - Response time
    - Availability
  - Formulae

QoS Attr.	Sequence	Switch	Fork	Loop
Time (T)	$\sum_{i=1}^{m} T(t_i)$	$\sum_{i=1}^{n} p_{ai} * T(t_i)$	$Max\{T(t_i)_{i\in\{1...p\}}\}$	$k * T(t)$
Cost (C)	$\sum_{i=1}^{m} C(t_i)$	$\sum_{i=1}^{n} p_{ai} * C(t_i)$	$\sum_{i=1}^{p} C(t_i)$	$k * C(t)$
Availability (A)	$\prod_{i=1}^{m} A(t_i)$	$\sum_{i=1}^{n} p_{ai} * A(t_i)$	$\prod_{i=1}^{p} A(t_i)$	$A(t)^k$
Reliability (R)	$\prod_{i=1}^{m} R(t_i)$	$\sum_{i=1}^{n} p_{ai} * R(t_i)$	$\prod_{i=1}^{p} R(t_i)$	$R(t)^k$
Custom Attr. (F)	$f_S(F(t_i)_{i\in\{1...m\}})$	$f_B((p_{ai}, F(t_i))_{i\in\{1...n\}})$	$f_F(F(t_i)_{i\in\{1...p\}})$	$f_L(k, F(t))$

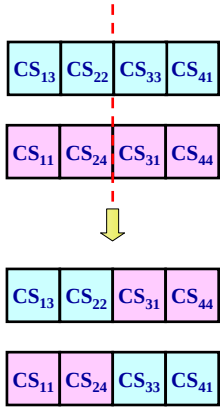
Finding a solution using Genetic Algorithm

- Representation:
  - Array chromosome
  - A slot for each abstract service invoked in the process
  - Value indicates the binding
  - S_1, ..., S_N abstract services
  - $CS_{i1, ..., } CS_{ik}$ concrete services for S_i

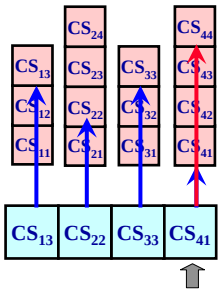


GA Operators

Crossover:



Mutation:



Configuration example

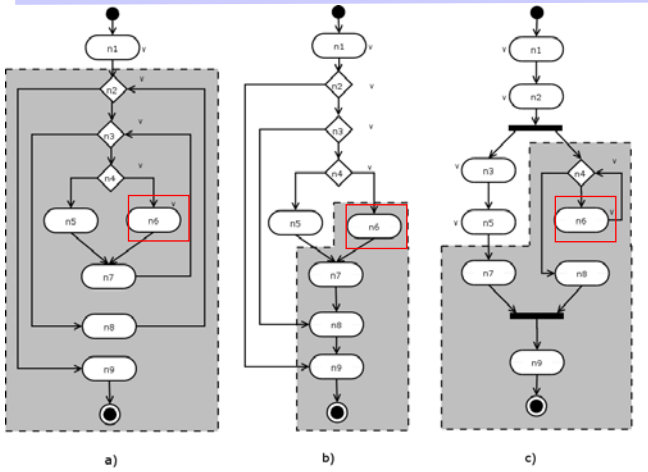
Selection: Roulette wheel

- p_{cross}=0.7
- p_{mut}=0.1
- GA type: Simple with elitism of 2 (best) individuals
- Population size: 100

Re-binding

- When:
  - $QoS_{actual-}$ $QoS_{estimated}$ >threshold
- Due to one of the following events
  1. Difference between estimated and actual number of loop iterations
  2. Choice of the path to be followed in a switch
  3. Difference between the estimated and actual QoS of some executed services
  4. A service is not available
- Triggered at specified breakpoints, e.g., while, switch and invoke nodes

Determining the re-binding slice



Loop Node	Estimated Iterations	Actual Iterations	Current Iterations
n3	k3	k'3	j3
n2	k2	k'2	j2

d)

Remaining n3 iterations: k'3-j3

Remaining n2 iterations: k'2-j2

Evaluation and future work

- Being evaluated from the industrial partners of the SeCSE project
  - Telecommunications and Automotive domains
  - We are collecting feedbacks
- Framework being enhanced with:
  - User-defined QoS attributes
    - language to specify aggregation formulae
- Focus on
  - Extension of BPEL to support the definition of rules constraining autonomic behavior of the composition
  - Negotiation of SLAs
  - Support for transactions

Related Work

- Various languages and environments for dynamic composition
  - Examples: SELFSERV, E-FLOW, MAIS, METEOR-S...
- Advantages of our approach
  - Uses/Extends the BPEL de-facto standard (as MAIS and METEOR-S)
  - Supports various kinds of bindings (pre-execution and runtime (re)binding) ... and binding policies (QoS, context-dependent, ...)