QoS-driven Selection of Web Services for Transactional Composition

Joyce El Haddad, Maude Manouvrier, Guillermo Ramirez, Marta Rukoz
Université Paris Dauphine, Universidad Central de Venezuela, Université Paris X-Nanterre

IEEE International Conference on Web Services

September 23-26, 2008
Outline

1. Introduction

2. Our approach
   - Context
   - TQoS-driven Web services selection

3. Related work

4. Conclusion and Perspectives
Goals

1. To support **user-tailored composition** of web services

2. To **exploit the transactional properties** of the component Web services to derive the transactional properties for the composite Web service

3. To **select the best QoS** component Web services

Our contribution

A selecting algorithm for composing Web services not only according to their functional requirements but also to their transactional properties and QoS characteristics
Outline

1. Introduction

2. Our approach
   - Context
   - TQoS-driven Web services selection

3. Related work

4. Conclusion and Perspectives
System Architecture

End-User → Planner Engine → Plan → Execution Engine → Composition Manager

Preferences → Planner Engine

Workflow

Web Services Registry

Transaction QoS composite service
Web Service Description

1. Web service **behavioral** properties

---

Behavioral properties

The set of all possible combinations for the behavioral property of a Web service is \{p; c; pr; cr\}

(Dauphine, Central de Venezuela, Paris X)
Web Service Description

1. Web service **behavioral** properties
   - transactional behavior
Web Service Description

1. Web service **behavioral** properties

   ▶ transactional behavior

     - pivot (p)

   ![Diagram of transactional behavior]

   The set of all possible combinations for the behavioral property of a Web service is \{p, c, pr, cr\} (Dauphine, Central de Venezuela, Paris X).
Web Service Description

1. Web service **behavioral** properties
   - transactional behavior
     - pivot (p)
     - compensatable (c)

   ![Diagram of transactional behavior]

   ![Diagram of compensatable behavior]
Web Service Description

1. Web service *behavioral* properties

   - transactional behavior
     - pivot (p)
     - compensatable (c)

   - non-transactional behavior
     - retrievable (r)
Web Service Description

Web service **behavioral** properties

- **transactional behavior**
  - pivot (p)
  - compensatable (c)

- **non-transactional behavior**
  - retrievable (r)

**Pivot and retrievable service**
Web Service Description

1 Web service behavioral properties

- transactional behavior
  - pivot (p)

- compensatable (c)

- non-transactional behavior
  - retrievable (r)

Compensatable and retrievable service
Web Service Description

1. Web service **behavioral** properties

   ▶ transactional behavior
   - pivot (p)

   ![Diagram of transactional behavior](chart)

   - compensatable (c)

   ![Diagram of compensatable behavior](chart)

   ▶ non-transactional behavior
   - retrievable (r)

   ![Diagram of non-transactional behavior](chart)

**Behavioral properties**

The set of all possible combinations for the behavioral property of a Web service is \{p;c;pr;cr\}
Web Service Description

1. Web service non-functional properties

- **Price** ($q_{ep}(s)$): the fee that a requester has to pay for invoking of the Web service $s$

- **Duration** ($q_{ed}(s)$): the measure of the expected delay time between the moment when a requester of Web service $s$ is sent and when the results are received

- **Reputation** ($q_{r}(s)$): the measure of trustworthiness of service $s$, generally this measure is defined as the average ranking given to the service by end users

- **Successful execution rate** ($q_{sr}(s)$): the probability that service $s$ responds correctly to the user request

- **Availability** ($q_{a}(s)$): the probability that a service $s$ is accessible
Composite transactional model

L. Li, C. Liu and J. Wang
Deriving Transactional Properties of Composite Web Services

Atomic workflow ($\bar{a}$)

- All the activities complete successfully $\implies$ their effect remain forever and cannot be semantically undone
- One activity fails $\implies$ all previous activities compensate
Atomic workflow (\(\bar{a}\))

- All the activities complete successfully \(\implies\) their effect remain forever and cannot be semantically undone
- One activity fails \(\implies\) all previous activities compensate
Composite transactional model

L. Li, C. Liu and J. Wang
Deriving Transactional Properties of Composite Web Services

Atomic workflow (\(\bar{a}\))

- All the activities complete successfully \(\implies\) their effect remain forever and cannot be semantically undone
- One activity fails \(\implies\) all previous activities compensate

\[\text{c}\]
Composite transactional model

L. Li, C. Liu and J. Wang
Deriving Transactional Properties of Composite Web Services

Atomic workflow ($\vec{a}$)

- All the activities complete successfully $\implies$ their effect remain forever and cannot be semantically undone
- One activity fails $\implies$ all previous activities compensate

![Diagram of atomic workflow]
Atomic workflow ($\vec{a}$)

- All the activities complete successfully $\implies$ their effect remain forever and cannot be semantically undone
- One activity fails $\implies$ all previous activities compensate
Composite transactional model

L. Li, C. Liu and J. Wang
Deriving Transactional Properties of Composite Web Services

Atomic workflow ($\vec{a}$)

- All the activities complete successfully $\implies$ their effect remain forever and cannot be semantically undone
- One activity fails $\implies$ all previous activities compensate
Composite transactional model

L. Li, C. Liu and J. Wang
Deriving Transactional Properties of Composite Web Services

Atomic workflow ($\vec{a}$)

- All the activities complete successfully $\Rightarrow$ their effect remain forever and cannot be semantically undone
- One activity fails $\Rightarrow$ all previous activities compensate

\[\begin{align*}
\vec{a} & \quad \text{(c)} \quad \text{(p)} \\
\vec{a} & \quad \text{(p)} \quad \text{(c)}
\end{align*}\]
Composite transactional model

L. Li, C. Liu and J. Wang
Deriving Transactional Properties of Composite Web Services

Atomic workflow ($\vec{a}$)

- All the activities complete successfully $\implies$ their effect remain forever and cannot be semantically undone
- One activity fails $\implies$ all previous activities compensate

\[\begin{align*}
\text{c} & \rightarrow \text{p} \\
\vec{a} & \rightarrow \\
\text{p} & \rightarrow \text{c} \\
\tilde{a} & \\
\text{c} & \rightarrow \text{p} \\
\text{p} & \rightarrow \text{c}
\end{align*}\]
Composite transactional model

L. Li, C. Liu and J. Wang
Deriving Transactional Properties of Composite Web Services

Atomic workflow (\(\vec{a}\))

- All the activities complete successfully \(\implies\) their effect remain forever and cannot be semantically undone
- One activity fails \(\implies\) all previous activities compensate
Composite transactional model

L. Li, C. Liu and J. Wang
Deriving Transactional Properties of Composite Web Services

Atomic workflow ($\vec{a}$)

- All the activities complete successfully $\implies$ their effect remain forever and cannot be semantically undone
- One activity fails $\implies$ all previous activities compensate

\[
\text{Atomic workflow (}\vec{a}\text{)}
\]

\[
\begin{align*}
\vec{a} & : c \rightarrow p \\
\vec{a} & : p \rightarrow c \\
\vec{a} & : \text{compensation}
\end{align*}
\]
Compensatable workflow (c)

A workflow is compensatable if all its activities can be compensated.
Composite transactional model

Compensatable workflow (c)

A workflow is compensatable if all its activities can be compensated.
Compensatable workflow (c)

A workflow is compensatable if all its activities can be compensated.
Compensatable workflow (c)

A workflow is compensatable if all its activities can be compensated.
Composite transactional model

Compensatatable workflow (c)

A workflow is compensatable if all its activities can be compensated

Transactionable Composite Web Service (TCWS)

A TCWS is a workflow that can be atomic or compensatable → The set of behavioral property of a TCWS is \(\{\ddot{a}; pr; c; cr\}\)
Composite quality model

L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam and H. Chang
QoS-Aware Middleware for Web Services Composition

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Aggregation function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Price</td>
<td>( q_{ep}(CWS) = \sum_{i=1}^{n} q_{ep}(s_i) )</td>
</tr>
<tr>
<td>Duration</td>
<td>( q_{ed}(CWS) = \sum_{i=1}^{n} q_{ed}(s_i) )</td>
</tr>
<tr>
<td>Reputation</td>
<td>( q_r(CWS) = \frac{1}{n} \sum_{i=1}^{n} q_r(s_i) )</td>
</tr>
<tr>
<td>Success rate</td>
<td>( q_{sr}(CWS) = \prod_{i=1}^{n} q_{sr}(s_i) )</td>
</tr>
<tr>
<td>Availability</td>
<td>( q_a(CWS) = \prod_{i=1}^{n} q_a(s_i) )</td>
</tr>
</tbody>
</table>

\( Score(s_i) = \sum w_j q_{ij}, \) where \( w_j \in [0, 1] \) is the weight assigned to the quality criterion, \( \sum w_j = 1 \) and \( q_{ij} \) is the value of criterion \( j \)
Outline

1 Introduction

2 Our approach
   • Context
   • TQoS-driven Web services selection

3 Related work

4 Conclusion and Perspectives
Definition of risk

The users can express their transactional criteria

- **Risk 0**: the system guarantees that if the execution is successful, the obtained results can be compensated by the user.
Definition of risk

The users can express their transactional criteria

- **Risk 0**: the system guarantees that if the execution is successful, the obtained results can be compensated by the user $\implies$ the selecting process generates a compensatable workflow.

- **Risk 1**: the system does not guarantee the successful execution but if it achieves the results cannot be compensated by the user.
Definition of risk

The users can express their transactional criteria

- **Risk 0**: the system guarantees that if the execution is successful, the obtained results can be compensated by the user $\implies$ the selecting process generates a compensatable workflow

- **Risk 1**: the system does not guarantee the successful execution but if it achieves the results cannot be compensated by the user
Definition of risk

The users can express their transactional criteria

- **Risk 0**: the system guarantees that if the execution is successful, the obtained results can be compensated by the user → the selecting process generates a compensatable workflow.

- **Risk 1**: the system does not guarantee the successful execution but if it achieves the results cannot be compensated by the user → the selecting process generates an atomic workflow.
Risk 0
\[ \forall a_i, \text{getBestQoS}(c \cup cr) \]
Service selection algorithm

∀ ai, getBestQoS(c ∪ cr)

Risk 0
Service selection algorithm

\[ \forall a_i, \text{getBestQoS}(c \cup cr) \]
Service selection algorithm

∀ \alpha_i, getBestQoS(c \cup cr)

Risk 0
Service selection algorithm

∀ $a_i$, $getBestQoS(c \cup cr)$
Service selection algorithm

**Risk 1**

- **sequential path** \( (a_{i-1}; a_i) \): previously selected service has either
  - a transactional property in \( \{p, pr\} \) or \( nbp = 1 \) ⇒ for \( a_i \) to \( a_n \), \( \text{getBestQoS}(pr \cup cr) \)
  - a transactional property in \( \{c, cr\} \) ⇒ for \( a_i \), \( \text{getBestQoS}(p \cup pr \cup c \cup cr) \)

- **concurrent path** (split-pattern \( \ldots a_i \ldots a_j \) join-pattern): one previously selected service has either
  - a transactional property in \( \{p\} \) ⇒ for \( a_i \) to \( a_j \), \( \text{getBestQoS}(cr) \)
  - a transactional property in \( \{pr\} \) ⇒ for \( a_i \) to \( a_n \), \( \text{getBestQoS}(pr \cup cr) \)
  - a transactional property in \( \{c\} \) ⇒ for \( a_i \) to \( a_j \), \( \text{getBestQoS}(c \cup cr) \)
Service selection algorithm

sequential path \((a_{i-1}; a_i)\): previously selected service has either
- a transactional property in \(\{p, pr\}\) or \(nbp = 1\) \(\Rightarrow\) for \(a_i\) to \(a_n\), \(\text{getBestQoS}(pr \cup cr)\)
- a transactional property in \(\{c, cr\}\) \(\Rightarrow\) for \(a_i\), \(\text{getBestQoS}(p \cup pr \cup c \cup cr)\)

concurrent path (split-pattern \(\ldots a_i \ldots a_j\) join-pattern): one previously selected service has either
- a transactional property in \(\{p\}\) \(\Rightarrow\) for \(a_i\) to \(a_j\), \(\text{getBestQoS}(cr)\)
- a transactional property in \(\{pr\}\) \(\Rightarrow\) for \(a_i\) to \(a_n\), \(\text{getBestQoS}(pr \cup cr)\)
- a transactional property in \(\{c\}\) \(\Rightarrow\) for \(a_i\) to \(a_j\), \(\text{getBestQoS}(c \cup cr)\)
Service selection algorithm

Risk 1

- sequential path $(a_{i-1}; a_i)$: previously selected service has either
  - a transactional property in $\{p, pr\}$ or $nbp = 1 \Rightarrow$ for $a_i$ to $a_n$, $getBestQoS(pr \cup cr)$
  - a transactional property in $\{c, cr\} \Rightarrow$ for $a_i$, $getBestQoS(p \cup pr \cup c \cup cr)$

- concurrent path (split-pattern $\ldots a_i \ldots a_j$ join-pattern): one previously selected service has either
  - a transactional property in $\{p\}$ \Rightarrow for $a_i$ to $a_j$, $getBestQoS(cr)$
  - a transactional property in $\{pr\}$ \Rightarrow for $a_i$ to $a_n$, $getBestQoS(pr \cup cr)$
  - a transactional property in $\{c\}$ \Rightarrow for $a_i$ to $a_j$, $getBestQoS(c \cup cr)$
**Risk 1**

- **sequential path** 
  \((a_{i-1};a_i)\): previously selected service has either
  - a transactional property in \(\{p, pr\}\) or \(nbp = 1\) ⇒ for \(a_i\) to \(a_n\), \(\text{getBestQoS}(pr \cup cr)\)
  - a transactional property in \(\{c, cr\}\) ⇒ for \(a_i\), \(\text{getBestQoS}(p \cup pr \cup c \cup cr)\)

- **concurrent path** (split-pattern \(\ldots a_i \ldots a_j\) join-pattern): one previously selected service has either
  - a transactional property in \(\{p\}\) ⇒ for \(a_i\) to \(a_j\), \(\text{getBestQoS}(cr)\)
  - a transactional property in \(\{pr\}\) ⇒ for \(a_i\) to \(a_n\), \(\text{getBestQoS}(pr \cup cr)\)
  - a transactional property in \(\{c\}\) ⇒ for \(a_i\) to \(a_j\), \(\text{getBestQoS}(c \cup cr)\)
Service selection algorithm

Risk 1

- sequential path \((a_{i-1}; a_i)\): previously selected service has either
  - a transactional property in \(\{p, pr\}\) or \(\text{nbp} = 1\) ⇒ for \(a_i\) to \(a_n\), \(\text{getBestQoS}(pr \cup cr)\)
  - a transactional property in \(\{c, cr\}\) ⇒ for \(a_i\), \(\text{getBestQoS}(p \cup pr \cup c \cup cr)\)

- concurrent path (split-pattern \(\ldots a_i \ldots a_j\) join-pattern): one previously selected service has either
  - a transactional property in \(\{p\}\) ⇒ for \(a_i\) to \(a_j\), \(\text{getBestQoS}(cr)\)
  - a transactional property in \(\{pr\}\) ⇒ for \(a_i\) to \(a_n\), \(\text{getBestQoS}(pr \cup cr)\)
  - a transactional property in \(\{c\}\) ⇒ for \(a_i\) to \(a_j\), \(\text{getBestQoS}(c \cup cr)\)
Service selection algorithm

Risk 1

- sequential path \((a_{i-1};a_i)\): previously selected service has either
  - a transactional property in \(\{p, pr\}\) or \(nbp = 1\) \(\Rightarrow\) for \(a_i\) to \(a_n\), \(getBestQoS(pr \cup cr)\)
  - a transactional property in \(\{c, cr\}\) \(\Rightarrow\) for \(a_i\), \(getBestQoS(p \cup pr \cup c \cup cr)\)

- concurrent path (split-pattern …\(a_i\) …\(a_j\) join-pattern): one previously selected service has either
  - a transactional property in \(\{p\}\) \(\Rightarrow\) for \(a_i\) to \(a_j\), \(getBestQoS(cr)\)
  - a transactional property in \(\{pr\}\) \(\Rightarrow\) for \(a_i\) to \(a_n\), \(getBestQoS(pr \cup cr)\)
  - a transactional property in \(\{c\}\) \(\Rightarrow\) for \(a_i\) to \(a_j\), \(getBestQoS(c \cup cr)\)
Service selection algorithm

Risk 1

- **sequential path** \((a_{i-1}; a_i)\): previously selected service has either
  - a transactional property in \(\{p, pr\}\) or \(nbp = 1\) ⇒ for \(a_i\) to \(a_n\), \(\text{getBestQoS}(pr \cup cr)\)
  - a transactional property in \(\{c, cr\}\) ⇒ for \(a_i\), \(\text{getBestQoS}(p \cup pr \cup c \cup cr)\)

- **concurrent path** (split-pattern \(\ldots a_i \ldots a_j\) join-pattern): one previously selected service has either
  - a transactional property in \(\{p\}\) ⇒ for \(a_i\) to \(a_j\), \(\text{getBestQoS}(cr)\)
  - a transactional property in \(\{pr\}\) ⇒ for \(a_i\) to \(a_n\), \(\text{getBestQoS}(pr \cup cr)\)
  - a transactional property in \(\{c\}\) ⇒ for \(a_i\) to \(a_j\), \(\text{getBestQoS}(c \cup cr)\)
Service selection algorithm

![Diagram](image.png)

**Risk 1**

- **Sequential path** \((a_{i-1}; a_i)\): previously selected service has either
  - a transactional property in \(\{p, pr\}\) or \(nbp = 1\) ⇒ for \(a_i\) to \(a_n\), \(\text{getBestQoS}(pr \cup cr)\)
  - a transactional property in \(\{c, cr\}\) ⇒ for \(a_i\), \(\text{getBestQoS}(p \cup pr \cup c \cup cr)\)

- **Concurrent path** (split-pattern \(\ldots a_i \ldots a_j\) join-pattern): one previously selected service has either
  - a transactional property in \(\{p\}\) ⇒ for \(a_i\) to \(a_j\), \(\text{getBestQoS}(cr)\)
  - a transactional property in \(\{pr\}\) ⇒ for \(a_i\) to \(a_n\), \(\text{getBestQoS}(pr \cup cr)\)
  - a transactional property in \(\{c\}\) ⇒ for \(a_i\) to \(a_j\), \(\text{getBestQoS}(c \cup cr)\)
Service selection algorithm

Risk 1

- sequential path \((a_{i-1}; a_i)\): previously selected service has either
  - a transactional property in \(\{p, pr\}\) or \(nbp = 1\) \(\Rightarrow\) for \(a_i\) to \(a_n\), \(\text{getBestQoS}(pr \cup cr)\)
  - a transactional property in \(\{c, cr\}\) \(\Rightarrow\) for \(a_i\), \(\text{getBestQoS}(p \cup pr \cup c \cup cr)\)

- concurrent path (split-pattern \(\ldots a_i \ldots a_j\) join-pattern): one previously selected service has either
  - a transactional property in \(\{p\}\) \(\Rightarrow\) for \(a_i\) to \(a_j\), \(\text{getBestQoS}(cr)\)
  - a transactional property in \(\{pr\}\) \(\Rightarrow\) for \(a_i\) to \(a_n\), \(\text{getBestQoS}(pr \cup cr)\)
  - a transactional property in \(\{c\}\) \(\Rightarrow\) for \(a_i\) to \(a_j\), \(\text{getBestQoS}(c \cup cr)\)
Service selection algorithm

Risk 1

- sequential path \((a_{i-1}; a_i)\): previously selected service has either
  - a transactional property in \(\{p, pr\}\) or \(nbp = 1\) ⇒ for \(a_i\) to \(a_n\), \(getBestQoS(pr \cup cr)\)
  - a transactional property in \(\{c, cr\}\) ⇒ for \(a_i\), \(getBestQoS(p \cup pr \cup c \cup cr)\)

- concurrent path (split-pattern \(\ldots a_i \ldots a_j\) join-pattern): one previously selected service has either
  - a transactional property in \(\{p\}\) ⇒ for \(a_i\) to \(a_j\), \(getBestQoS(cr)\)
  - a transactional property in \(\{pr\}\) ⇒ for \(a_i\) to \(a_n\), \(getBestQoS(pr \cup cr)\)
  - a transactional property in \(\{c\}\) ⇒ for \(a_i\) to \(a_j\), \(getBestQoS(c \cup cr)\)
Implementation

- A workflow
Implementation

- A workflow
- A random generation of different services that can implement the activities
  - For each activity, uniformly generate 15 Web services
  - For each service, randomly generates transactional property and a QoS vector

<table>
<thead>
<tr>
<th>Criteria</th>
<th>(1)</th>
<th>(2)</th>
<th>(3)</th>
<th>(4)</th>
<th>(5)</th>
<th>(6)</th>
<th>(7)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_{ep}(s)$</td>
<td>0.20 – 0.30</td>
<td>0.20 – 0.30</td>
<td>0.00 – 0.10</td>
<td>0.00 – 0.10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q_{ed}(s)$</td>
<td>0.20 – 0.30</td>
<td>0.01 – 0.10</td>
<td>0.20 – 0.30</td>
<td>0.01 – 0.10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q_r(s)$</td>
<td>0 – 5</td>
<td>0 – 5</td>
<td>0 – 5</td>
<td>0 – 5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q_{sr}(s)$</td>
<td>0.00 – 1.00</td>
<td>0.00 – 1.00</td>
<td>0.00 – 1.00</td>
<td>0.00 – 1.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$q_a(s)$</td>
<td>0.00 – 1.00</td>
<td>0.00 – 1.00</td>
<td>0.00 – 1.00</td>
<td>0.00 – 1.00</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Implementation

A workflow

A random generation of different services that can implement the activities

- For each activity, uniformly generate 15 Web services
- For each service, randomly generates transactional property and a QoS vector

User assigned weight with price and duration constraints have always 60% of the total weight

<table>
<thead>
<tr>
<th>Criteria</th>
<th>cr</th>
<th>c</th>
<th>pr</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td>q_{ep}(s)</td>
<td>0.20 – 0.30</td>
<td>0.20 – 0.30</td>
<td>0.00 – 0.10</td>
<td>0.00 – 0.10</td>
</tr>
<tr>
<td>q_{ed}(s)</td>
<td>0.20 – 0.30</td>
<td>0.01 – 0.10</td>
<td>0.20 – 0.30</td>
<td>0.01 – 0.10</td>
</tr>
<tr>
<td>q_r(s)</td>
<td>0 – 5</td>
<td>0 – 5</td>
<td>0 – 5</td>
<td>0 – 5</td>
</tr>
<tr>
<td>q_{sr}(s)</td>
<td>0.00 – 1.00</td>
<td>0.00 – 1.00</td>
<td>0.00 – 1.00</td>
<td>0.00 – 1.00</td>
</tr>
<tr>
<td>q_a(s)</td>
<td>0.00 – 1.00</td>
<td>0.00 – 1.00</td>
<td>0.00 – 1.00</td>
<td>0.00 – 1.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Criteria</th>
<th>(1)</th>
<th>(2)</th>
<th>(3)</th>
<th>(4)</th>
<th>(5)</th>
<th>(6)</th>
<th>(7)</th>
</tr>
</thead>
<tbody>
<tr>
<td>q_{ep}(s)</td>
<td>0</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td>60</td>
</tr>
<tr>
<td>q_{ed}(s)</td>
<td>60</td>
<td>50</td>
<td>40</td>
<td>30</td>
<td>20</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>q_r(s)</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>q_{sr}(s)</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>q_a(s)</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
</tbody>
</table>
More the price criteria is important to the user, the best is a composition with risk 1

More the duration criteria is important to the user, the best is a composition with risk 0
Related work

L. Zeng, A. N. B. Benatallah, M. Dumas, J. Kalagnanam and H. Chang
QoS-Aware Middleware for Web services Composition

M. C. Jaeger, G. Muehl and S. Golze
Qos-aware composition of web services: An evaluation of selection algorithms

W. Zhang, Y. Yang, S. Tang and L. Fang
Qos-driven service selection optimization model and algorithms for composite web services

+ QoS-aware Web services selection
− no transactional behavior for the composite Web service
Related work

S. Bhiri, O. Perrin and C. Godart
Ensuring required failure atomicity of composite web services
14th Int. Conf. on World Wide Web (WWW’2005), 2005.

F. Montagut and R. Molva
Augmenting web services composition with transactional requirements
IEEE Int. Conf. on Web Services (ICWS’2006), September 2006.

L. Li, C. Liu and J. Wang
Deriving Transactional Properties of Composite Web Services
IEEE Int. Conf. on Web Services (ICWS’2007), July 2007.

+ Transactional composition mechanism

— no QoS-aware selection
A. Liu, L. Huang and Q. Li.
QoS-Aware Web Services Composition Using Transactional Composition Operator
7th Int. Conf. Advances in Web-Age Information Management (WAIM), LNCS 4016, June 2006.

+ Composition of Web services with various transactional requirements
  - Transactional property of the composite service is determined by its component services (if all component are $p$ then the composite service is $p$)

+ Evaluation of the QoS of the composite service
  - not a QoS-aware approach to service composition
Conclusion

- **Web service selection** approach supporting transactional and quality driven Web service composition

- Transactional properties of composite Web service are established based on the transactional properties of its component Web services

- The selection is realized depending on **transactional and QoS user requirements**
  - User transactional requirements are established by means of a **risk notion** that indicates if the user has or not the obligation to take the execution results
  - User QoS requirements are expressed as **weight** over each QoS criterion
Perspectives

- **Replaning** the selection of Web services in order to take into account dynamic changes
- Web service selection at **run-time**
- Take into account other considerations about the component Web services such as **non-retriable** services with higher execution time than **retriable** ones
- Experimentations
References

F. Montagut and R. Molva
Augmenting web services composition with transactional requirements
IEEE Int. Conf. on Web Services (ICWS’2006), September 2006.

L. Li, C. Liu and J. Wang
Deriving Transactional Properties of Composite Web Services
IEEE Int. Conf. on Web Services (ICWS’2007), July 2007.

L. Zeng, A. N. B. Benatallah, M. Dumas, J. Kalagnanam and H. Chang
QoS-Aware Middleware for Web services Composition

A. Liu, L. Huang and Q. Li.
Qos-aware web services composition using transactional composition operator
7th Int. Conf. Advances in Web-Age Information Management (WAIM),
LNCS 4016, June 2006.