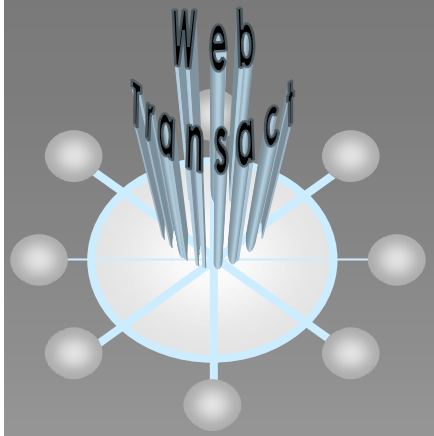
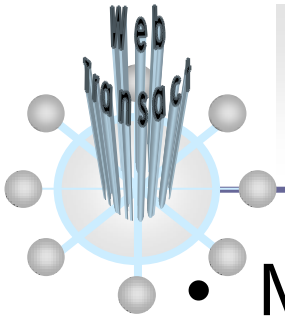


Building Reliable Web Services Compositions

Authors:
Paulo F. Pires
Mario R.F. Benevides
Marta Mattoso
[pires, mario, marta] @cos.ufrj.br

Federal University of Rio de Janeiro
COPPE/UFRJ
DCC-IM



Organization

- Motivation
- Goal and scope of the work
- WebTransact Overview
- WebTransact Layers
  - Expressing the Transaction Support of Web Services
  - Integrating Web Services in WebTransact
  - Aggregating Web Services
  - Specifying Web Services Compositions
  - Execution Model for Transaction Mediation
- Related Work
- Contributions and Future Work

Motivation

- The Current Web services Tecnology:
  - Communication Interoperability in the Web
  - underpining to build Web Services Compositions
- However, building reliable Web services compositions needs more..

- Autonomy

Provided by different organizations

Designed not to be dependent of any collective computing entity.

Transaction coordination

- Scale

Motivation

- The Current Web services Tecnology:
  - Communication Interoperability in the Web
  - underpining to build Web Services Compositions
- However, building reliable Web services compositions needs more..

- Autonomy
- Scale

Web services providing the same semantic functionality.

Existence of dissimilarities between these services

- Message formats
- Data semantics
- Service content
- Transaction support

Motivation

- The Current Web services Tecnology:
  - Communication Interoperability in the Web
  - underpining to build Web Services Compositions
- However, building reliable Web services compositions needs more..

- Autonomy
- Scale

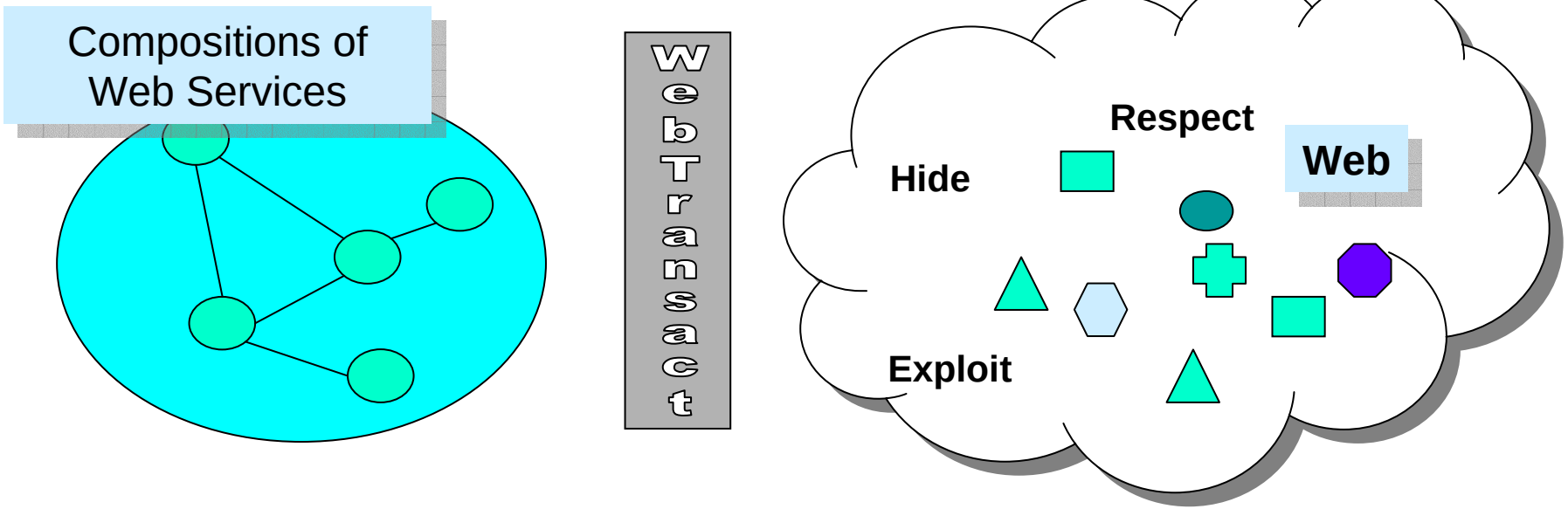
The task of building compositions has to somehow deal with these problems.

The current Web services technology does not address these problems.

Goal of the Work

- Development and specification of a framework to build reliable Web services compositions.
  - Respect the autonomy of WS
  - Exploit the existence of different WS providing semantic equivalent services
  - Hide the heterogeneity

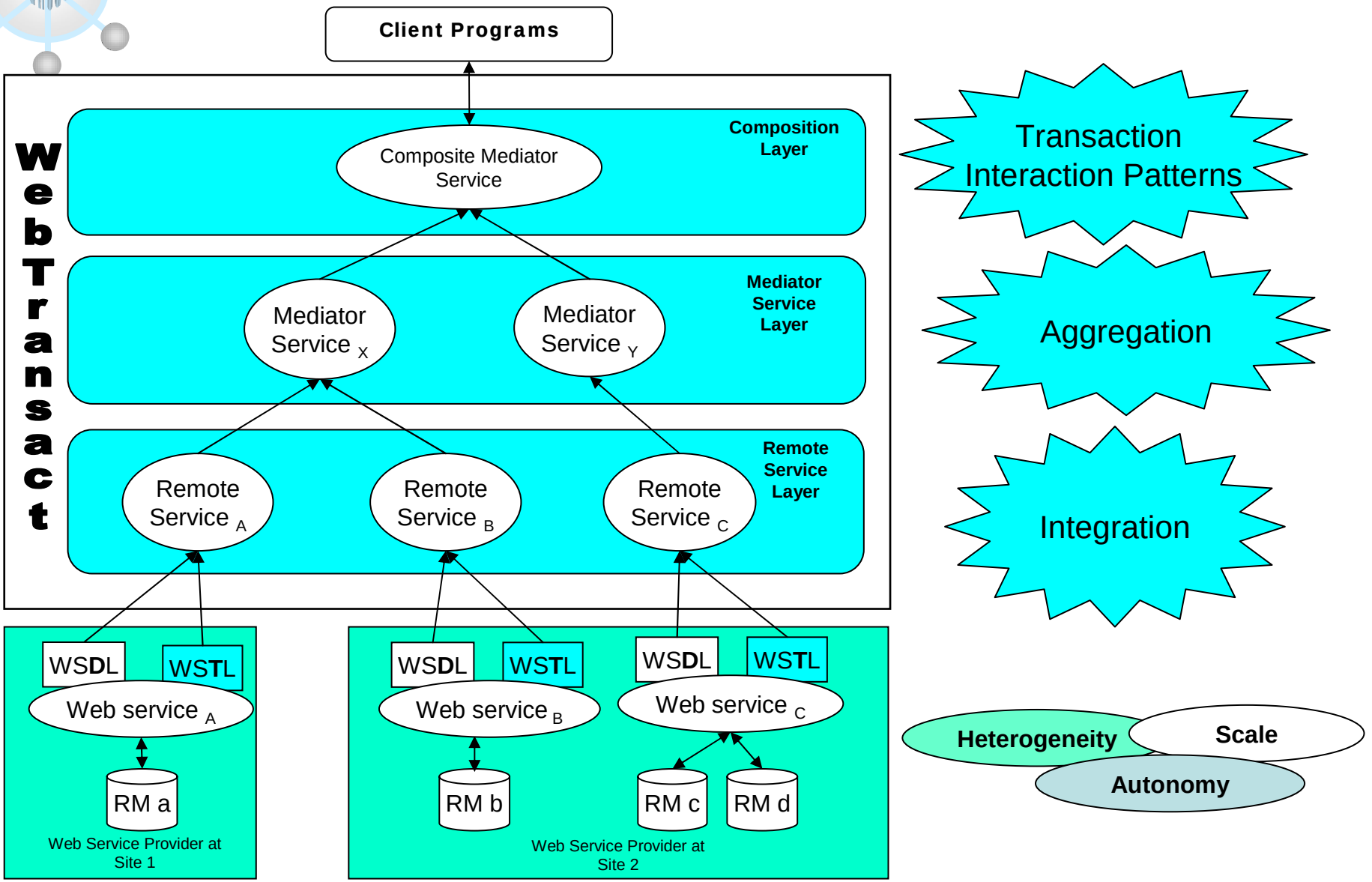
WebTransact



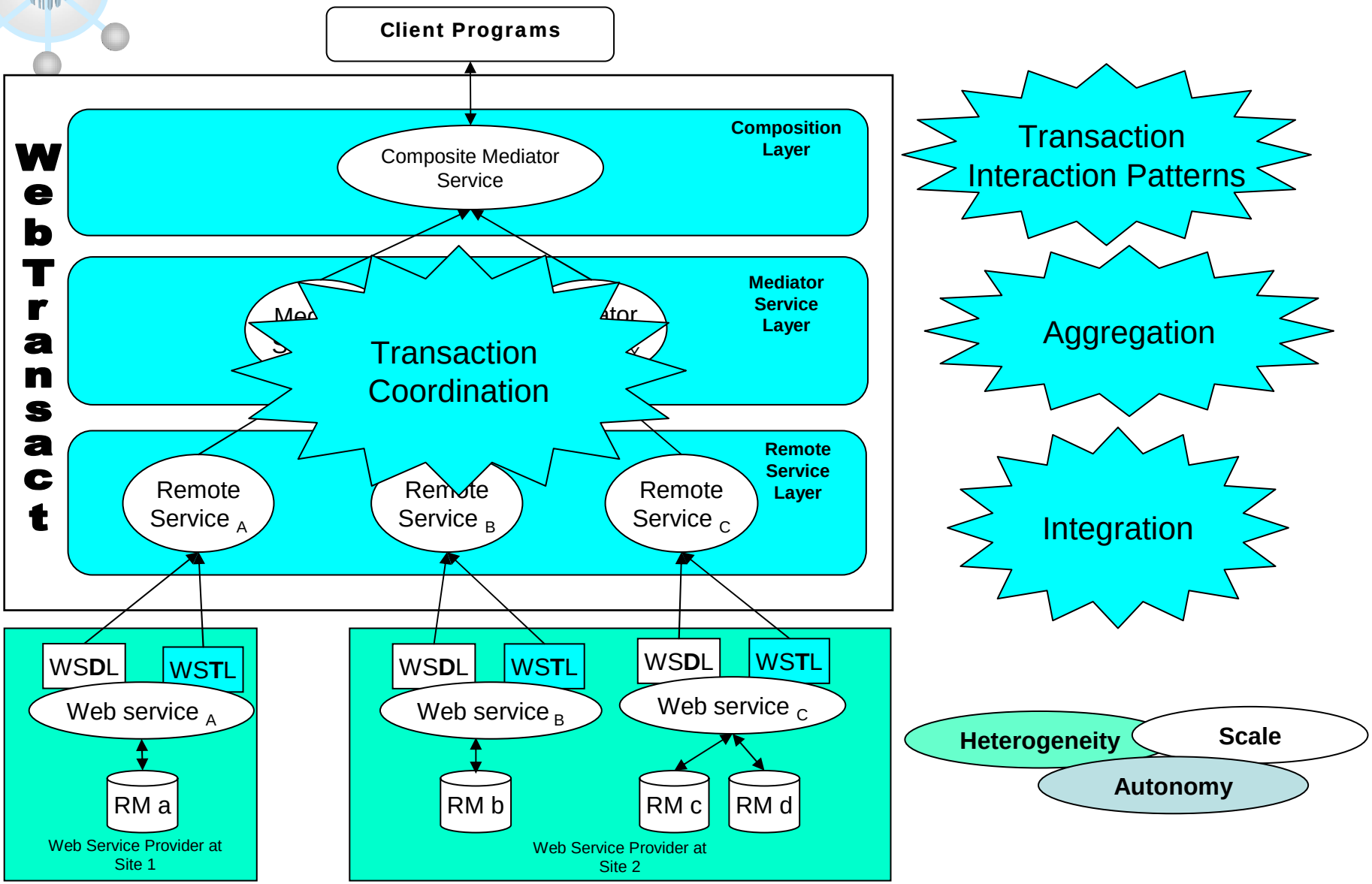
WebTransact Components

- ***The multilayered architecture:***
  - **Based on mediator wrapper technology ([Disco], [Garlic], [TSIMMIS], [Himpar]):**
    - **Mediate the *query capability* of remote (database or non-database) sources; and**
    - **Provide information integration at the mediator level.**
  - **WebTransact → adapted to resolve *syntactic, structural and content heterogeneity* of Web services.**
- ***The XML-based* language:**
  - ***Web Service Transaction language* (WSTL):**
    - **Describe the transaction support and content of Web services;**
    - **Define aggregation rules of Web services; and**
    - **Specify the transaction interaction patterns of compositions.**
  - **WSTL is an extension of WSDL**
- **The *transaction model:***
  - **Provide the adequate correctness guarantees when executing Web services compositions built with WSTL.**

Overview of WebTransact

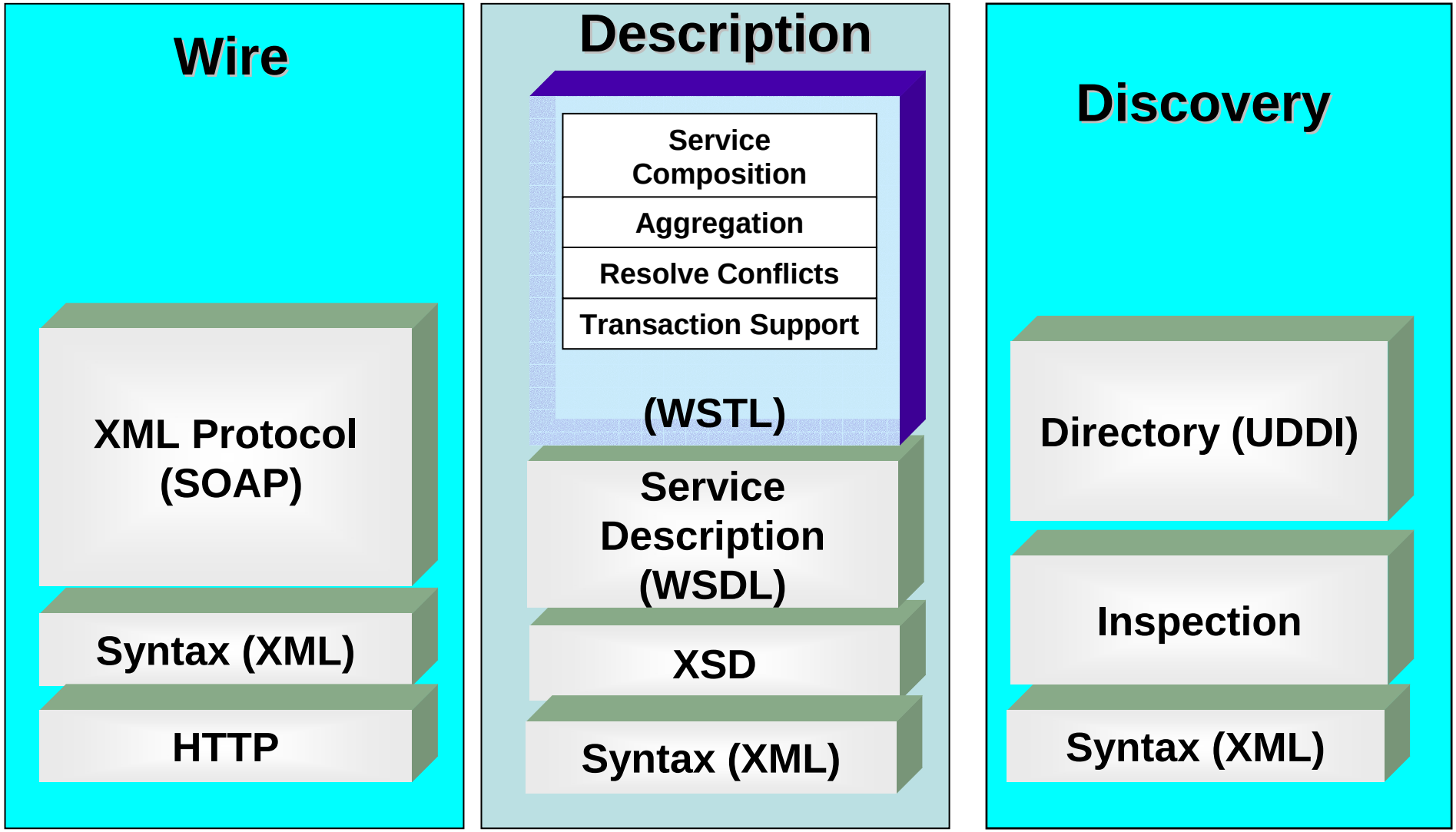


Overview of WebTransact

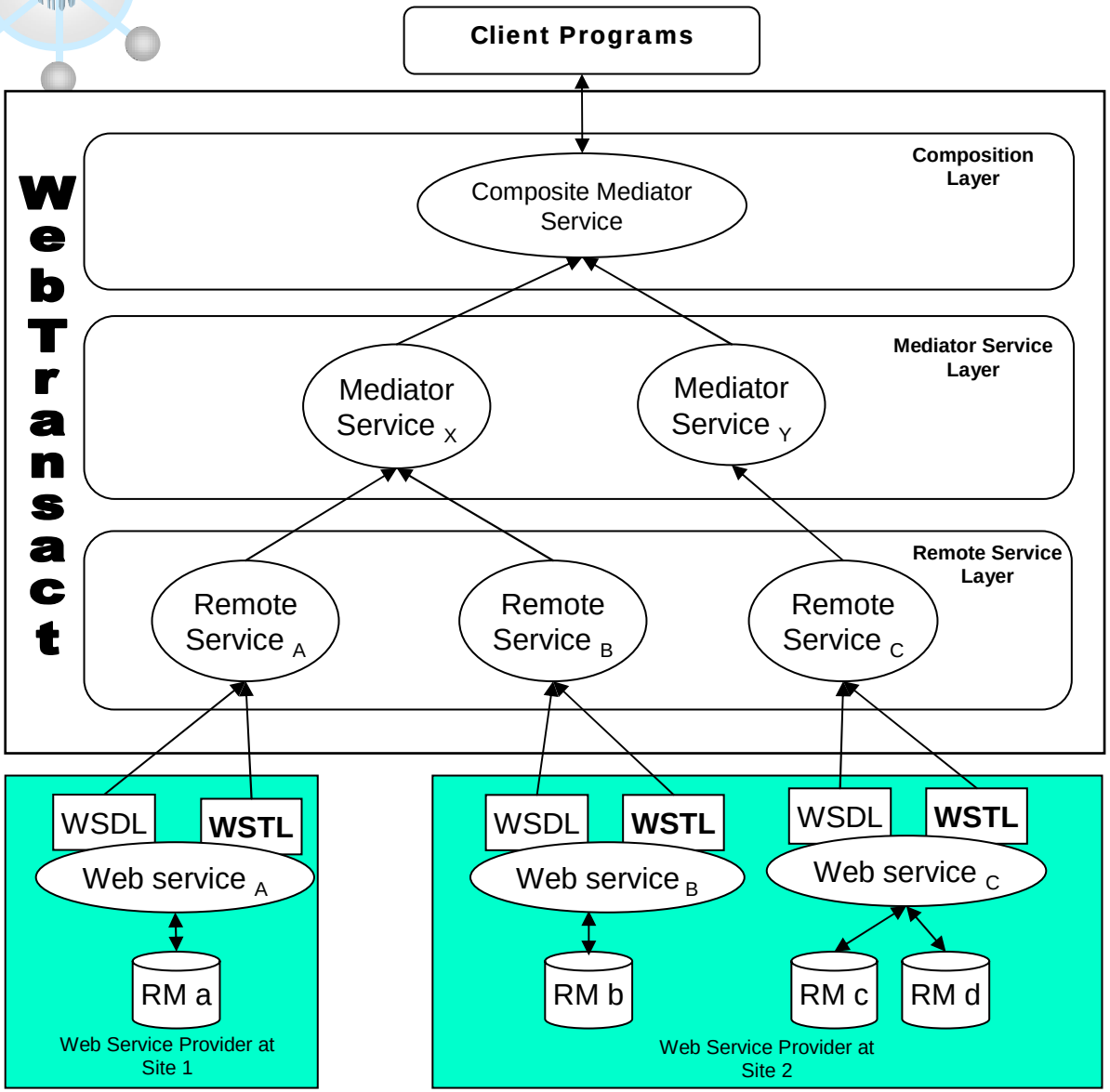


Overview of WebTransact

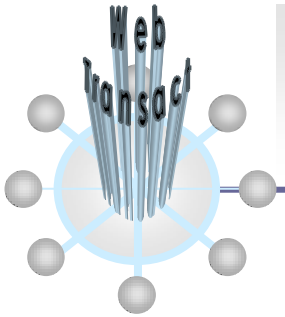
Extensions to the Web services Technology Stack



WebTransact Architecture



Transaction Support of Web services



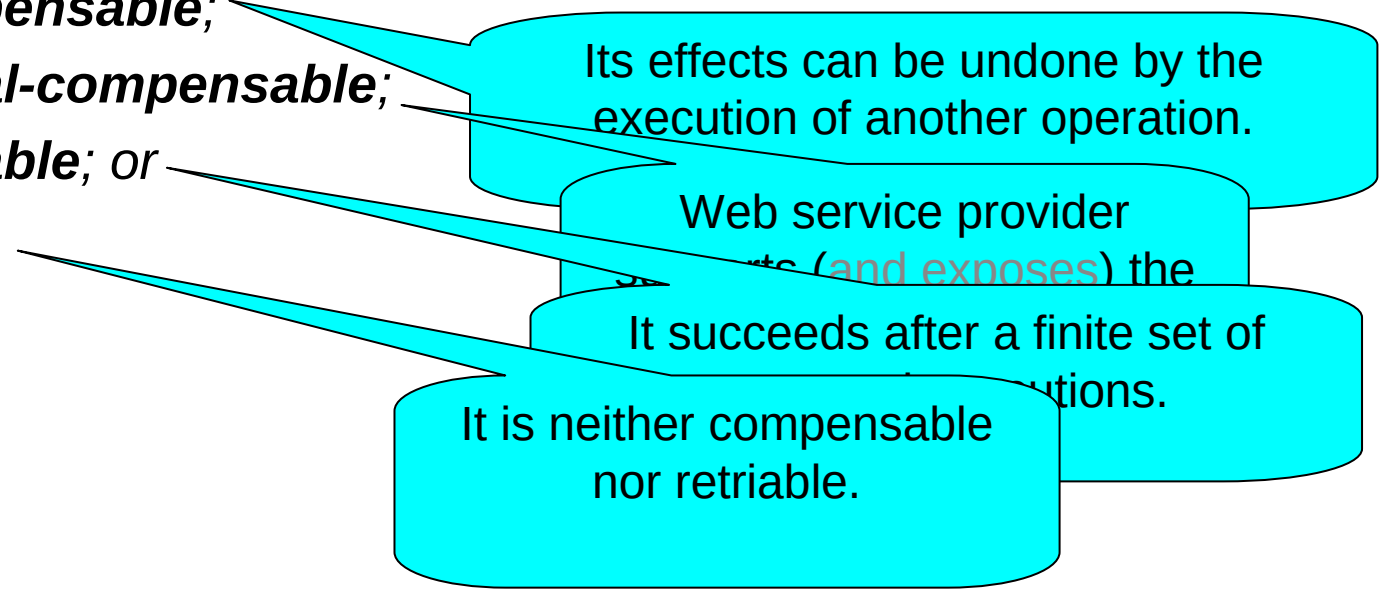
Expressing the Transaction Behavior of Web Services

Web services may support dissimilar transaction capabilities

WSTL provides mechanisms to explicitly define different levels of Transaction support of Web services.

Expressing the Transaction Behavior of Web Services

- The transaction support of Web services is defined through a concept named ***transaction behavior***.
  - The transaction behavior indicates the transaction semantics supported by a WSDL operation:
    - ***Compensable***;
    - ***Virtual-compensable***;
    - ***Retriable***; or
    - ***Pivot***.



Expressing the Transaction Behavior of Web Services - Example

```
<definitions
  xmlns:absd="http://example.com/carReservation/abstractDef/"">
   …
  <wstl:transactionDefinitions>
     <wstl:transactionBehavior operationNa     absd:reservation"
                  type="compensable">
        <wstl:activeAction portTypeName="absd:reservationSoap"
                        compensatoryOper="cancelReservation">
            <wstl:paramLink>
              <wstl:sourceParamLink
                    msgName="absd:reservationSoapOut"
                    param="reservationResponse/reservationResult"/>
          wstl:targetParamLink
                    msgName="cancelReservationSoapIn"
                    param="cancelReservation/reservationCode"/>
            </wstl:paramLink>
        </wstl:activeAction>
      </wstl:transactionBehavior>

     <wstl:transactionBehavior operationName="  absd:cancelReservation"
                               type="retriable"/>
  …
```

Transaction behavior

WSDL Port type of the compensatory operation

WSDL operation

Compensatory operation

Mapping information used to construct the input message of the compensatory operation

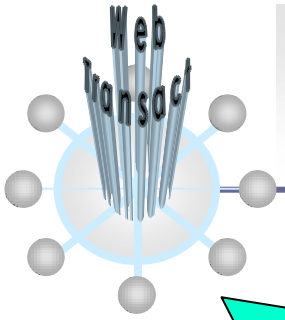
Does not require futher information

Expressing the Transaction Behavior of Web Services

- Integrating Virtual-compensable operations
  - WebTransact supports the integration of Web services that support the standard two-phase commit protocol (2PC)
  - **Concepts**:
    - 2PC communication interface of transaction managers is exposed through WSDL definitions.
    - Transaction Internet Protocol (TIP):
      - Define the standard 2PC interface; and
      - The semantics of transaction messages
    - Do not enforce any rigid format for the 2PC interface of remote TMs
      - Integration through mappings

IETF standard Compliant with DTP model

Respects the autonomy of Web services



Expressing the Transaction Behavior of Web Services

The effective use of Web services is directly related to the quality of their description

Transaction support key requirement

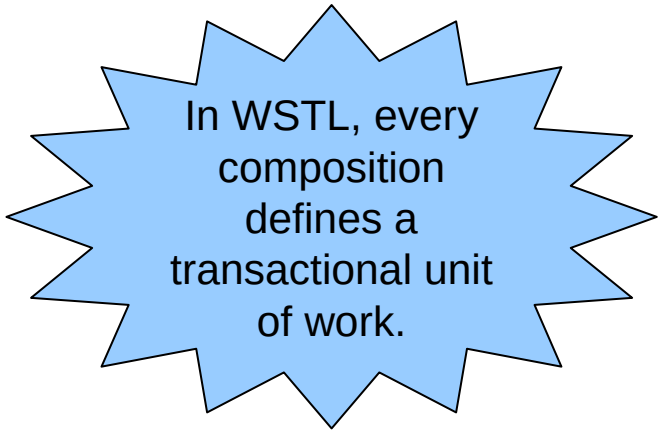
WSTL improves the Web services description through the explict definition of the transaction Support of Web Services

Mediator and Remote Services in WebTransact

- Remote Services
  - They are wrapping services responsible for integrating a Web service within the WebTransact framework.
    - link a mediator service to a WSDL port type;
    - provide mapping information between mediator service operations and WSDL port type operations; and
    - specify the content description of the remote service.
- Mediator Services
  - They are *virtual* services that aggregate semantic equivalent remote services
    - delegate its operations execution to one or more remote services.
    - Provide a homogenized interface of heterogeneous remote services

Mediator Service Composition

- WSTL provides elements for describing compositions of mediator service operations.
  - These elements specify a ***transaction interaction pattern*** of a collection of mediator service operations.



Mediator Service Composition

- WSTL models compositions as *composite tasks*.
  - Labeled directed graph:
    - nodes represent steps of execution; and
    - edges represent the flow of control and data among different steps.
  - Each step is either an atomic *task* or another composite task.
    - An atomic task is a unit of work that is executed by a mediator service operation.
- Structure:
  - Tasks are identified by a name and have:
    - a signature;
    - a set of execution dependencies;
    - a set of data links; and
    - a set of rules (optionally).
    - Mandatory tasks (composite tasks)

Transaction Model of WebTransact

- The transaction model coordinates the execution of a given mediator service composition:
  - The composite task coordination level
    - Specify the rules for interpreting a composite task specification.
  - The atomic task coordination level:
    - Specify the rules for coordinating the execution of the mediator service operations that implement atomic tasks.
      - Provide a transactional interface for all atomic tasks.

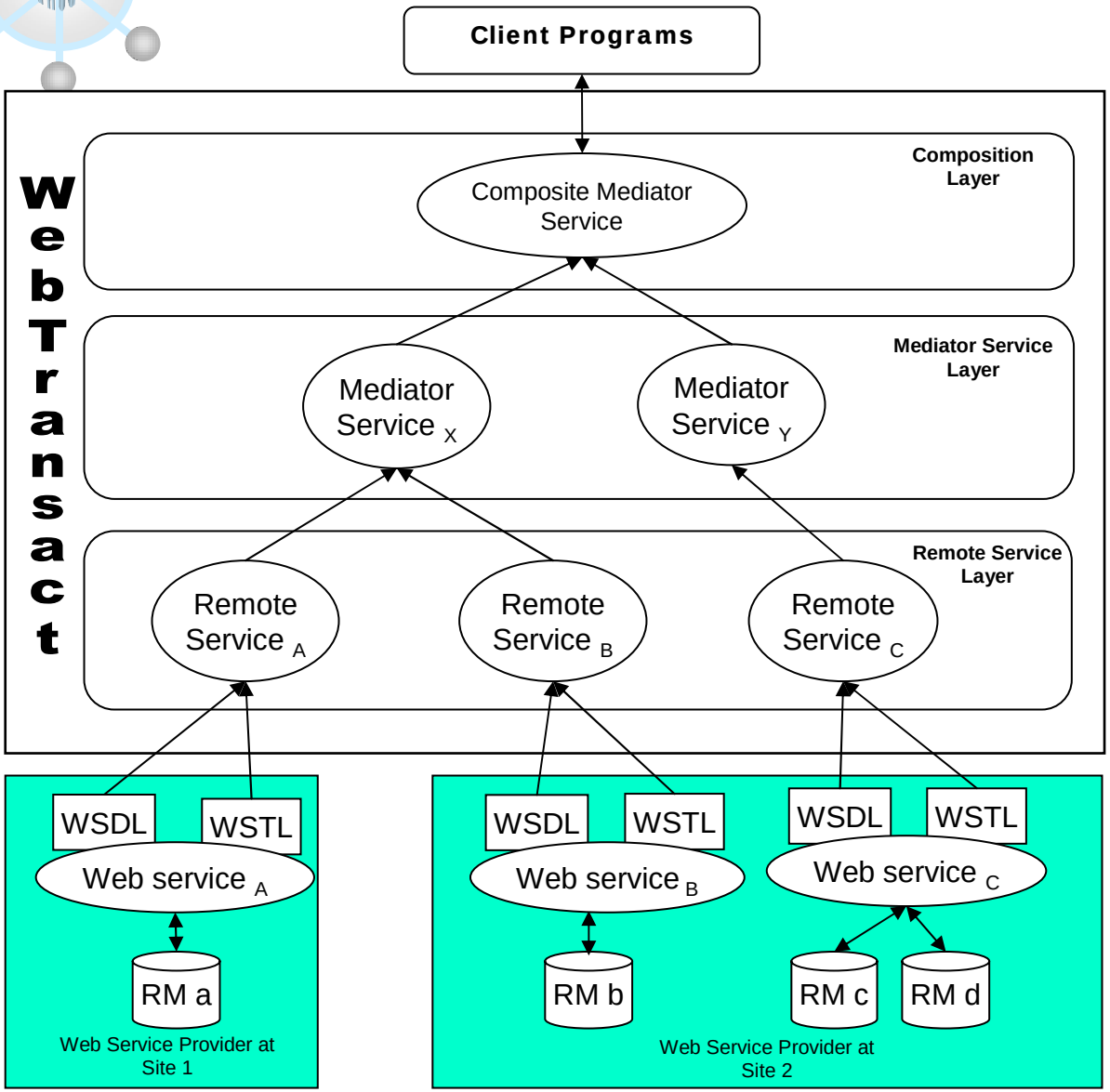
Transaction Model of WebTransact

- The transaction model coordinates the execution of a given mediator service composition:
  - The com
    - Sp
      sp
  - The atomic task coordination level:
    - Specify th xecution of the
      n mic tasks

Respect the composition specification
and
Guarentee the transaction properties.

Coordinate the one-to-many relationship that exists between mediator service operations and remote service operations
and
exploits the dissimilar transaction behavior of remote service operations

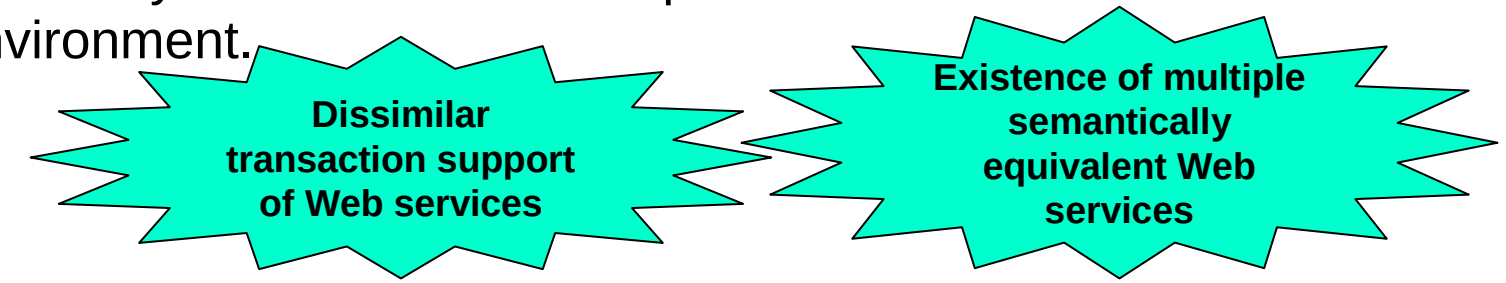
WebTransact Architecture



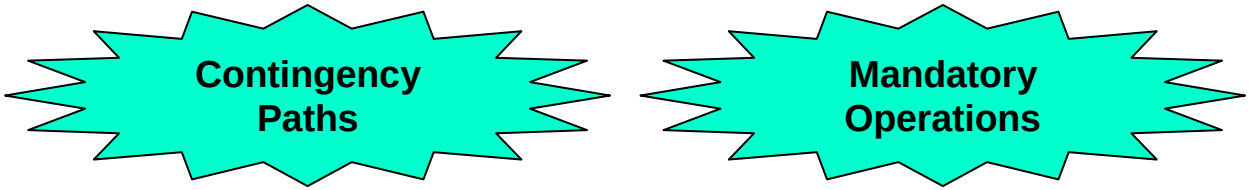
Transaction Model

Transaction Model of WebTransact

- *Discussion:*
  - The development of the transaction model of WebTransact was guided by the transactional requirements of the Web services environment.



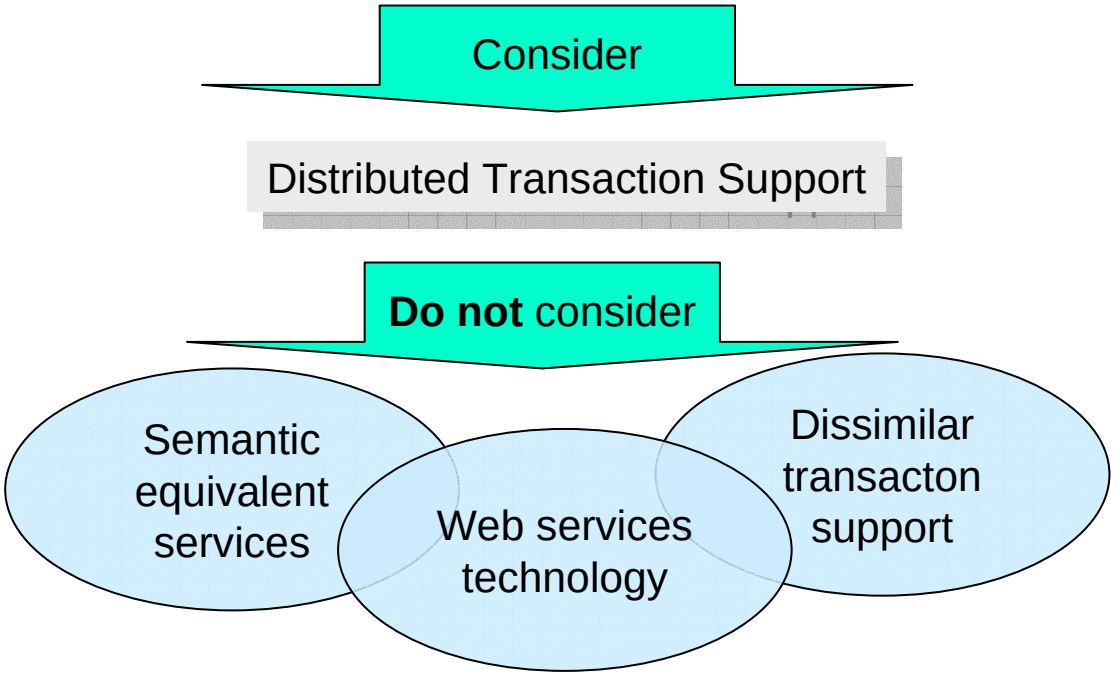
  - Support a user defined specification of reliability and atomicity



  - Guarantee the safe and correct execution of compositions based on both the user needs and dissimilar transaction behavior of Web services
    - Layered notion of the guaranteed-termination property
      - Two levels: Composite task level and the atomic task level (mediator service execution)

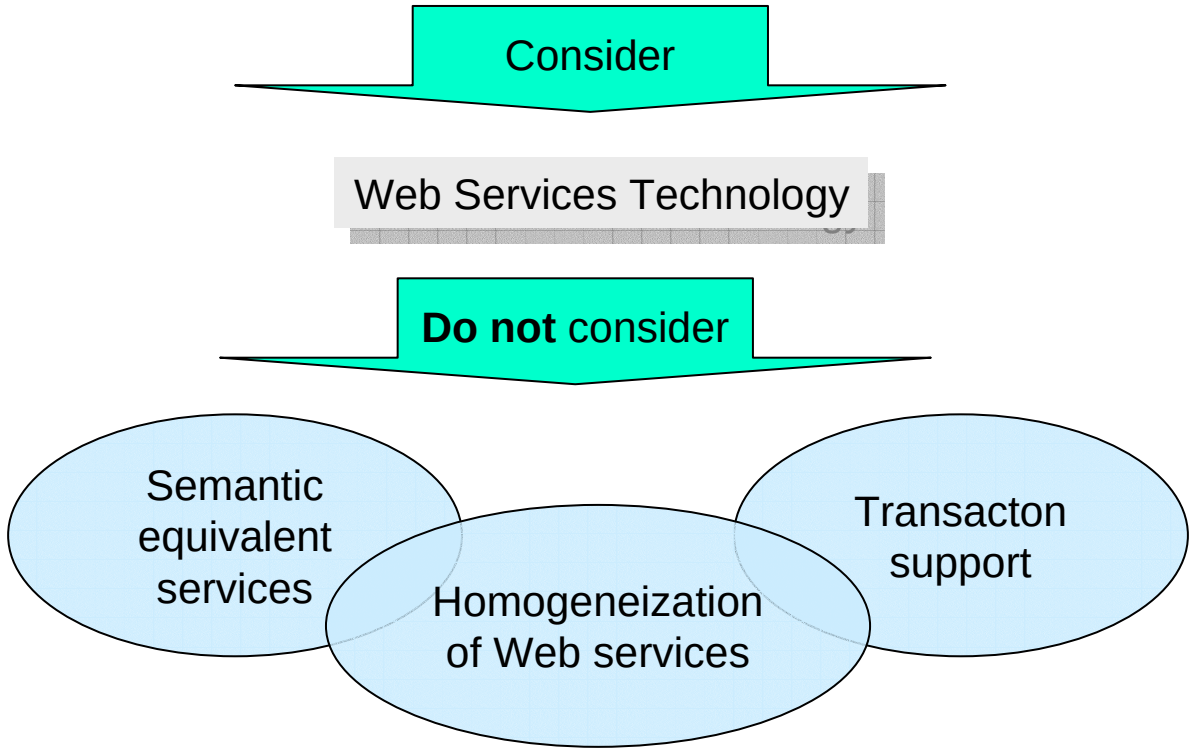
Related Work

- There has been extensive research in transaction support for:
  - Distributed computing systems:
    - [Interbase], [Garlic], [IPL], [Dayal], [Garcia-Molina], [Georgakopoulos], [TSME];
  - Transactional process coordination:
    - [Alonso], [Schuldt]); and
  - Workflow management systems:
    - [InConcert], [ConTract], [METEOR], [Mentor], [METUflow], [CoAct]).



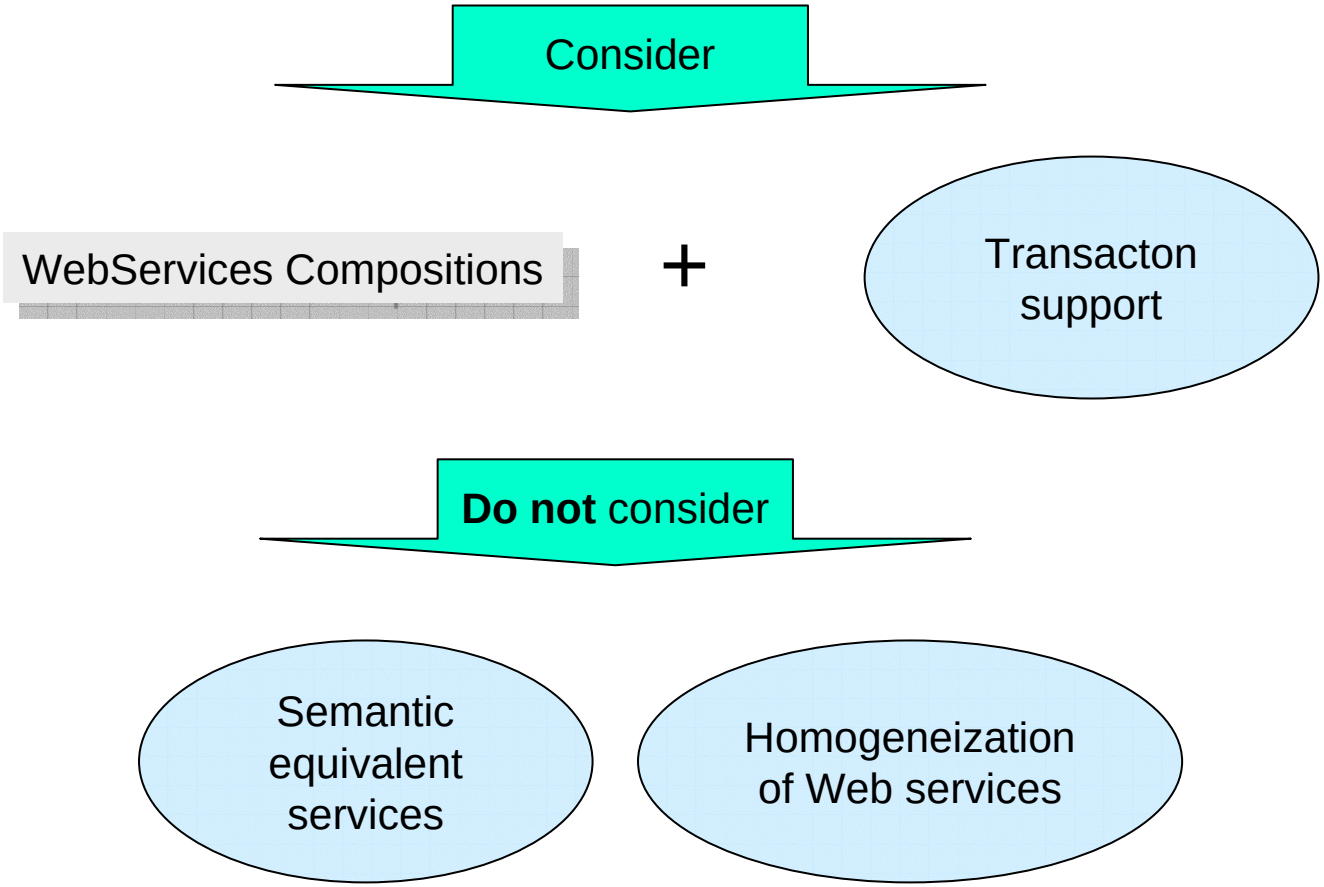
Related Work

- More recently, the area of e-service composition is attracting the interest of both academia and industry:
  - The existent work in this area (WSFL, XLANG) is concentrated in defining primitives for composing services and automating service coordination.



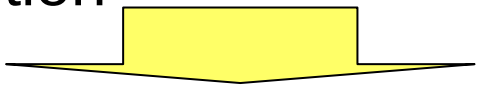
Related Work

- BPEL4WS + WS-Transaction + WS-Coordination



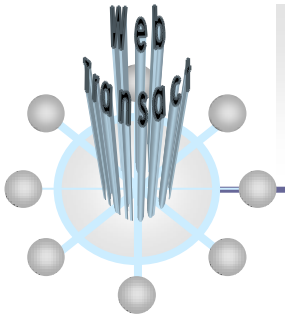
Related Work

- Web Transaction Protocols
  - W3C tentative hold protocol (THP)
  - OASIS Business Transaction Protocol (BTP)
  - WS-Transaction



- Each participant must understand the protocol
  - Protocol ubiquity

- WebTransact
  - Participant does not need to know the transaction model of WebTransaction
    - provides description of its transaction support



Contributions

- Definition of a multilayered architecture:
  - Address the problems related to the inherent autonomy,heterogeneity, scale and dynamism of the Web service environment.

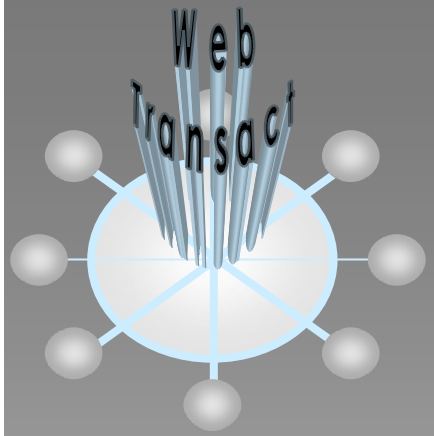
- Definition of a flexible framework for integrating Web services that have (and expose) their own local transaction support based on the two-phase commit protocol (2PC).
  - Use of a two-pipe model for communicating transaction synchronization messages; and
  - Use of the WSDL framework for exposing the 2PC communication interface of TMs.

Contributions

- Definition of the Web Service Transaction Language (WSTL).
  - WSTL provides elements for:
    - defining the supported transaction behavior of Web services;
    - resolving semantic and content dissimilarities,
    - aggregating semantically equivalent Web services,
    - defining mediator service interfaces, and
    - defining reliable interaction patterns of Web services composition.
- Definition of a transaction model and its related protocol for coordinating the execution of Web services compositions specified through WSTL.

Current and Future Works

- **Execution Model (Current)**
  - **Selection criteria for service execution**
    - **Cost model (transaction, security, QoS, proce, etc.)**
- **Security:**
  - The requirements regarding security are likely to be dissimilar between Web services.
- **Application Domains:**
  - Scientific applications (Bioinformatics)
- **Dynamic Discovering of Services:**
  - The Web services are statically integrated in WebTransact by a developer who plays the role of Web services integrator.
- **Methodologies:**
  - There is a need to investigate the impact that Web services compositions will have on the existent component-based approaches for developing internet applications.
- **Distributed Transaction Coordination:**
  - Building of TIP state machines for the most common distributed transaction processing systems such as OMG's Object Transaction Service (OTS) [95], Sun's Java Transaction Service (JTS) [126], and Microsoft's Distributed Transaction Coordinator (DTC) [85].



End of the presentation.

Thanks!