Basis for Automatic Service Composition

Daniela Berardi *
Giuseppe De Giacomo
Massimo Mecella

Dipartimento di Informatica e Sistemistica
"Antonio Ruberti"
Università di Roma "La Sapienza"
{berardi,degiacomo,mecella}@dis.uniroma1.it

* now at *Innovazione Italia*
Italian Ministery for Technological Innovation

Instructors

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- **Daniela Berardi**
  Research Interests:
  - AI: Knowledge Representation, Reasoning about Action, Planning
  - Software Engineering: Conceptual Modeling
  - Services: Modeling and Composition
  - Applications: e-Health Systems

- **Giuseppe De Giacomo**
  Research Interests:
  - AI: Knowledge Representation, Reasoning about Action, Planning, Cognitive Robotics
  - Databases: Databases with Incomplete Information, Data Integration, Semantic Interoperability
  - Dynamic Systems: Reasoning on Dynamic Systems, Process Verification and Synthesis
  - Services: Composition

- **Massimo Mecella**
  Research Interests:
  - Software Architectures: Cooperative Information Systems for e-Government and e-Business
  - Information Systems: Workflow Management, Data Quality
  - Distributed Systems: Middleware and Mobile Applications
  - Services: Modeling, Compatibility and Substitution, Composition and Orchestration

Outline

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Part 1 (Massimo Mecella)
  - Basic Concepts
  - Relevant Technologies and Abstractions
- Part 2 (Daniela Berardi)
  - State of the Art of the Research in Composition
  - Classification of Different Approaches
- Part 3 (Giuseppe De Giacomo)
  - Basic Research Perspective
  - The Roman Approach

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

Basic Concepts

e-Services, Web Services, Services … (1)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- An e-Service is often defined as an application accessible via the Web, that provides a set of functionalities to businesses or individuals. What makes the e-Service vision attractive is the ability to automatically discover the e-Services that fulfill the users' needs, negotiate service contracts, and have the services delivered where and when users needs them

  Guest editorial. In [VLDBJ01]

- e-Service: an application component provided by an organization in order to be assembled and reused in a distributed, Internet-based environment; an application component is considered as an e-Service if it is: (i) open, that is independent, as much as possible, of specific platforms and computing paradigms; (ii) developed mainly for inter-organizations applications, not only for intra-organization applications; (iii) easily composable; its assembling and integration in an inter-organizations application does not require the development of complex adapters. e-Application: a distributed application which integrates in a cooperative way the e-Services offered by different organizations

  M. Mecella, B. Pernici: Designing Wrapper Components for e-Services in Integrating Heterogeneous Systems. In [VLDBJ01]

e-Services, Web Services, Services … (2)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

A Web service is a software system identified by a URI, whose public interfaces and bindings are defined and described using XML. Its definition can be discovered by other software systems. These systems may then interact with the Web service in a manner prescribed by its definition, using XML based messages conveyed by Internet protocols

Web Services Architecture Requirements, W3C Working Group Note, 11 Feb. 2004, http://www.w3.org/TR/wsa-reqs/

e-Services, Web Services, Services … (3)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

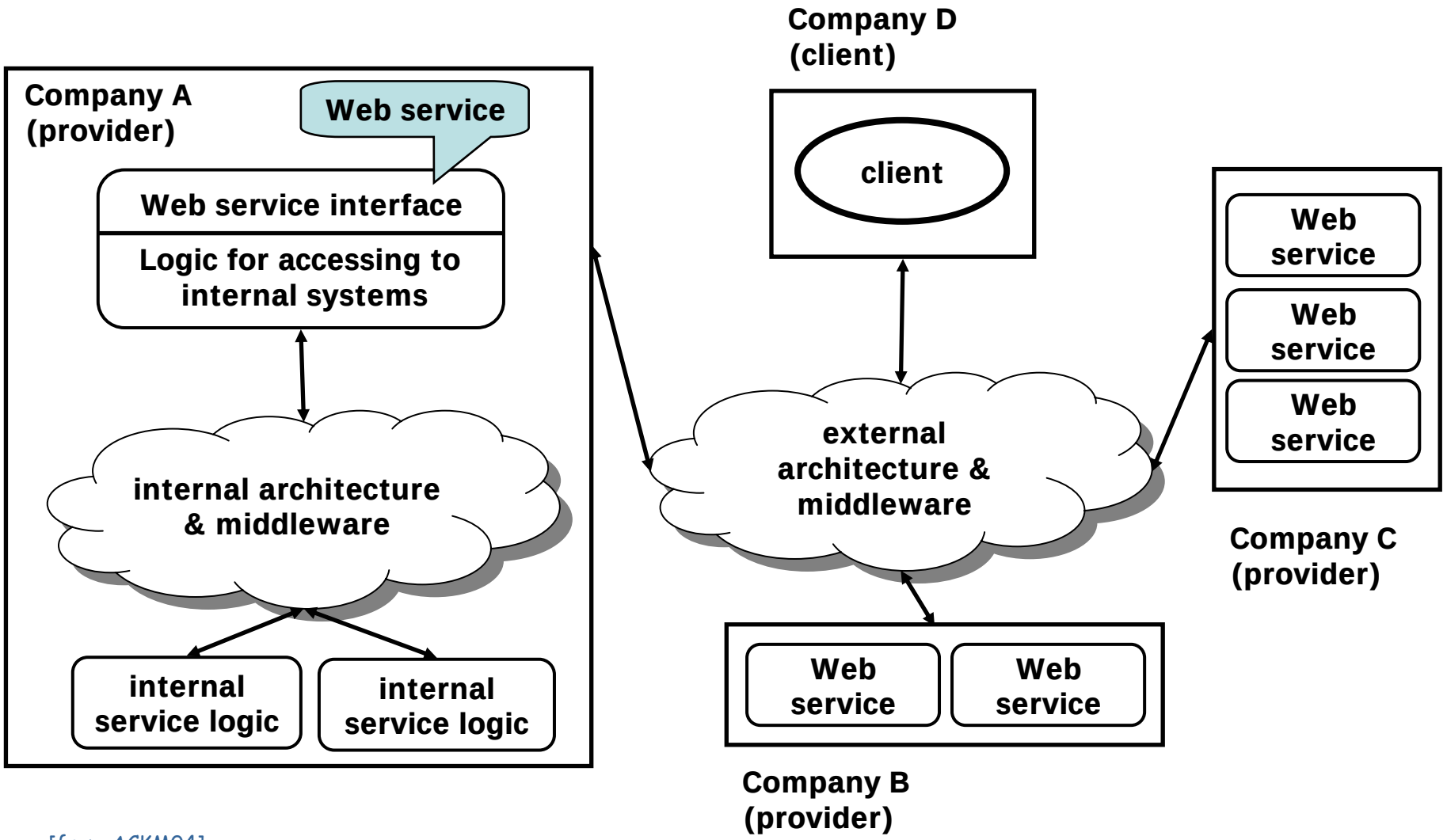
- Services are self-describing, open components that support rapid, low-cost composition of distributed applications. Services are offered by service providers — organizations that procure the service implementations, supply their service descriptions, and provide related technical and business support.

  Since services may be offered by different enterprises and communicate over the Internet, they provide a distributed computing infrastructure for both intra and cross-enterprise application integration and collaboration.

  Service descriptions are used to advertise the service capabilities, interface, behavior, and quality. Publication of such information about available services provides the necessary means for discovery, selection, binding, and composition of services. In particular, the service capability description states the conceptual purpose and expected results of the service (by using terms or concepts defined in an application-specific taxonomy). The service interface description publishes the service signature (its input/output/error parameters and message types). The (expected) behavior of a service during its execution is described by its service behavior description. Finally, the Quality of Service (QoS) description publishes important functional and nonfunctional service quality attributes […]. Service clients (end-user organizations that use some service) and service aggregators (organizations that consolidate multiple services into a new, single service offering) utilize service descriptions to achieve their objectives.
- The application on the Web (including several aspects of the SOA) is manifested by Web services

Guest editorial. In [CACM03]

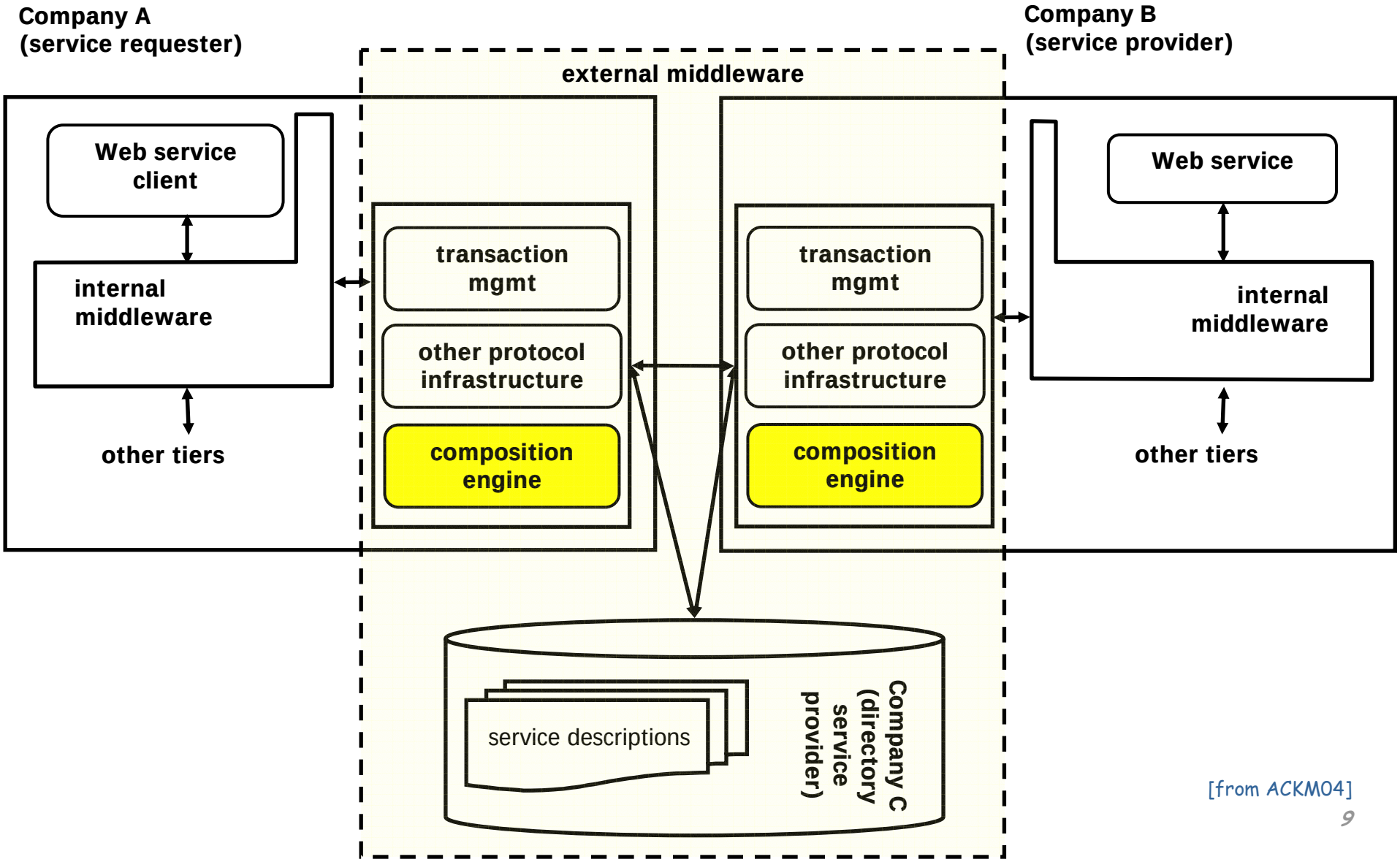
Two Architectures (and Middlewares) (1)



[from ACKM04]

Two Architectures (and Middlewares) (2)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

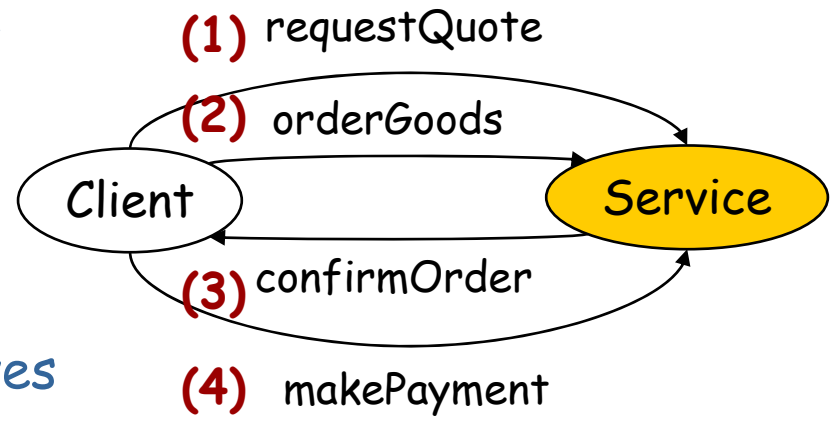


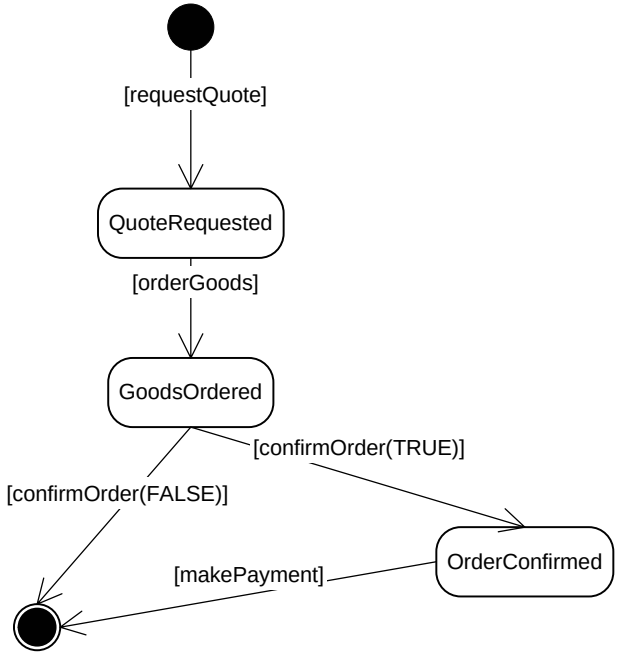
[from ACKM04]

Services

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

- A service is characterized by the set of (atomic) operations that it exports …
- … and possibly by constraints on the possible conversations
  - Using a service typically involves performing sequences of operations in a particular order (conversations)
  - During a conversation, the client typically chooses the next operation to invoke (on the basis of previous results, etc.) among the ones that the service allows at that point

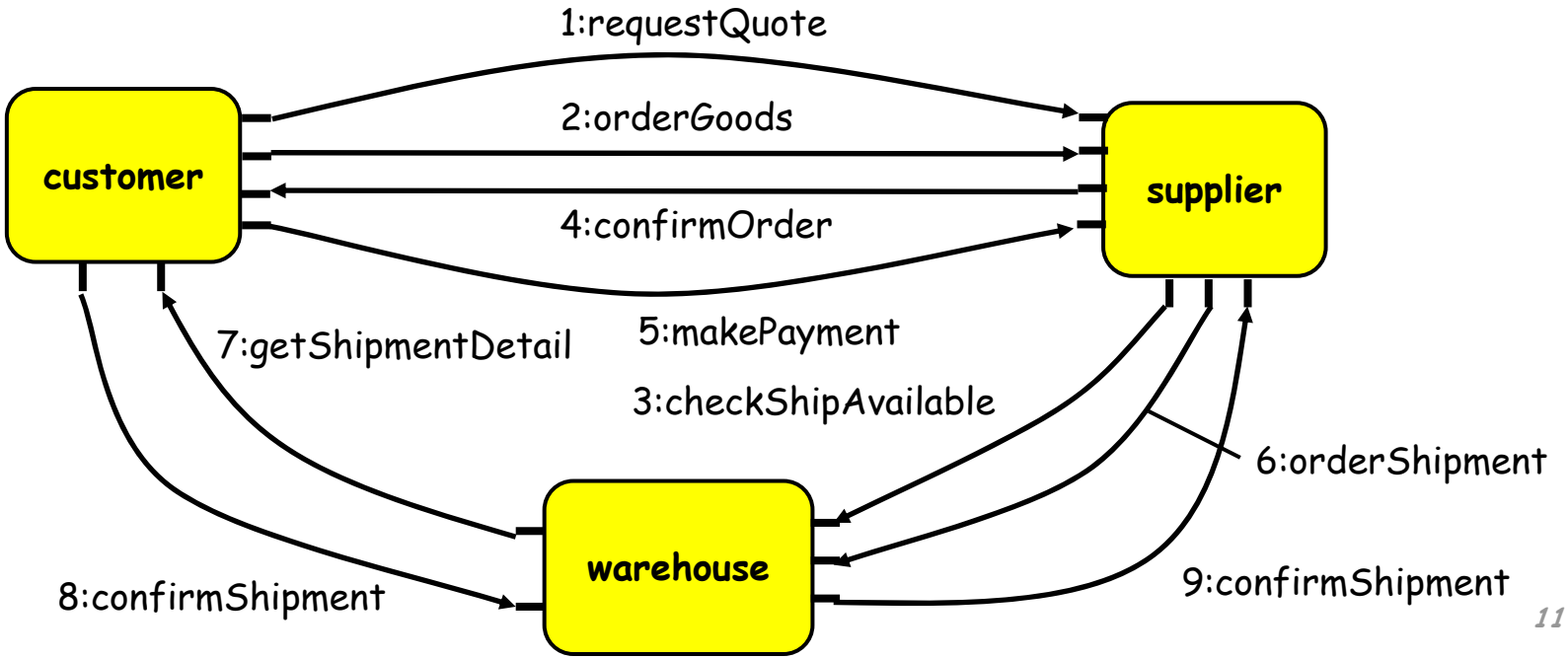




Choreography: Coordination of Conversations of N Services

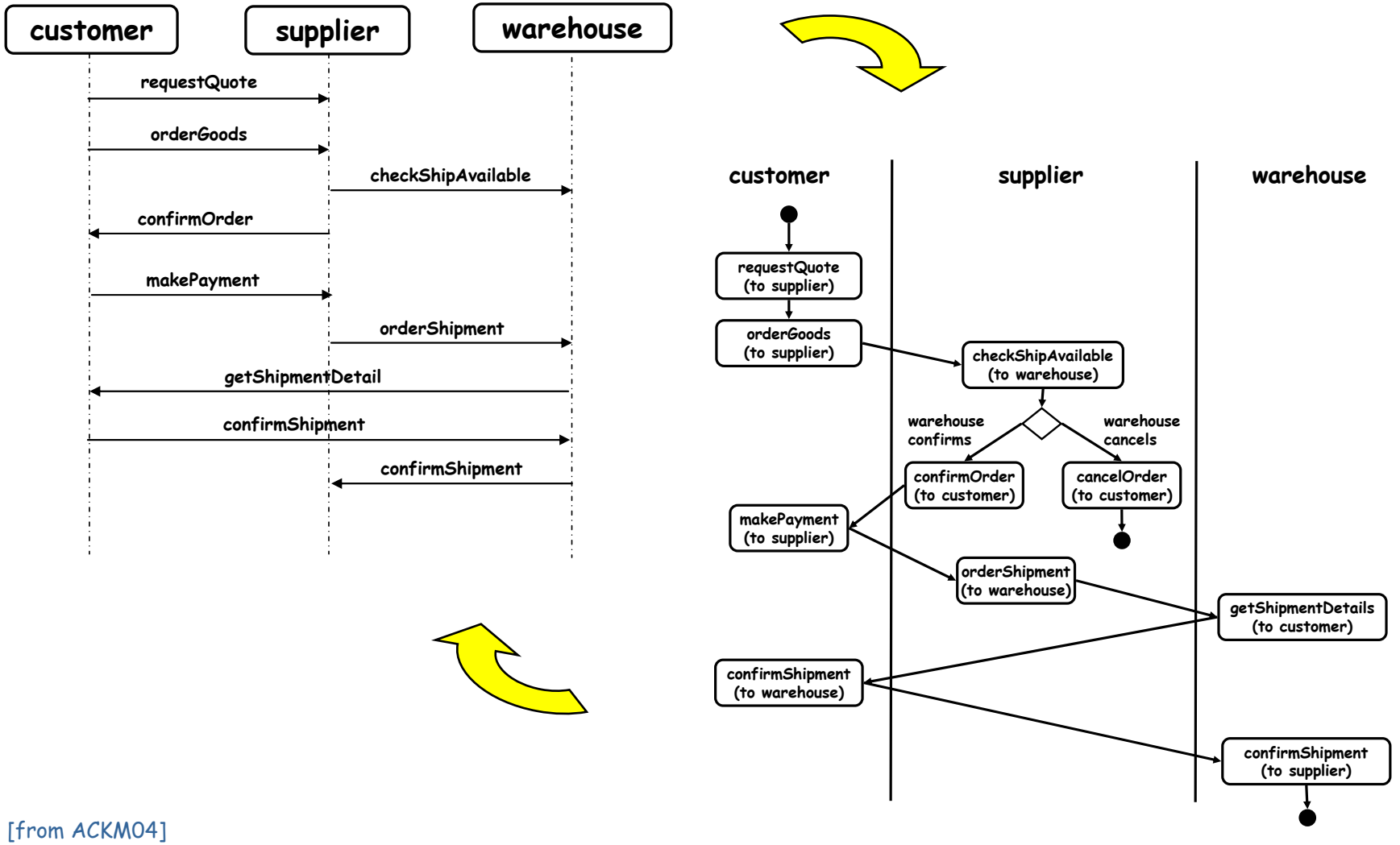
Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

- Global specification of the conversations of N peer services (i.e., multi-party conversations)
  - Roles
  - Message exchanges
  - Constraints on the order in which such exchanges should occur



Choreography: Coordination of Conversations of N Services

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"



[from ACKM04]

Composition

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Deals with the implementation of an application (in turn offered as a service) whose application logic involves the invocation of operations offered by other services
  - The new service is the *composite* service
  - The invoked services are the *component* services

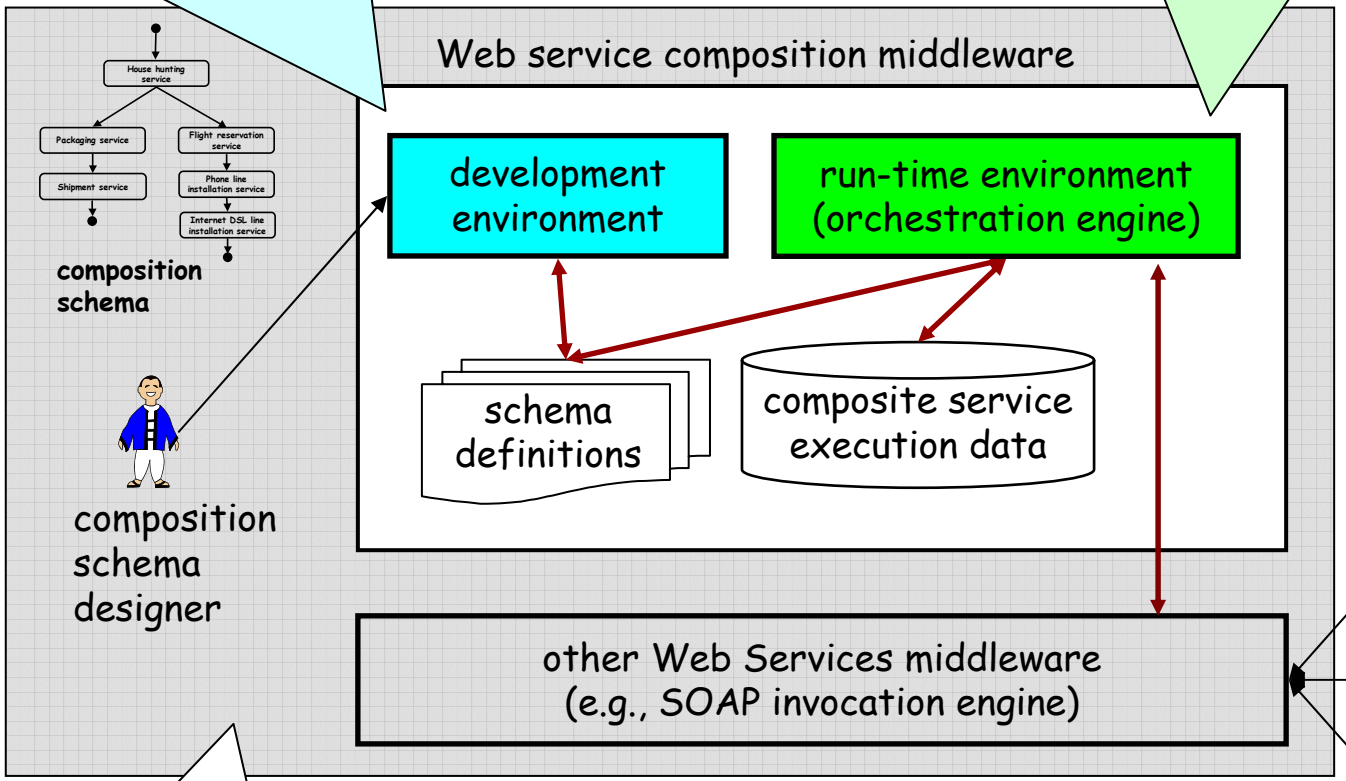
The Composition Engine/Middleware

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS

Università di Roma "La Sapienza"

Through the development environment, a **composition schema** is **synthesized**, either manually or (semi-)automatically. A service composition model and a language (maybe characterized by a graphical and a textual representation) are adopted

Orchestration: the run-time environment executes the composite service business logic by invoking other services (through appropriate protocols)



[from ACKM04]

Synthesis and Orchestration

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- (Composition) Synthesis: building the specification of the composite service (i.e., the composition schema)
  - Manual
  - Automatic
- Orchestration: the run-time management of the composite service (invoking other services, scheduling the different steps, etc.)
  - Composition schema is the "program" to be executed
  - Similarities with WfMSs (Workflow Management Systems)

Composition Schema

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- A composition schema specifies the "process" of the composite service
  - The "workflow" of the service
- Different clients, by interacting with the composite service, satisfy their specific needs (reach their goals)
  - A specific execution of the composition schema for a given client is an orchestration instance

Choreography (Coordination) vs. Composition (Orchestration)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

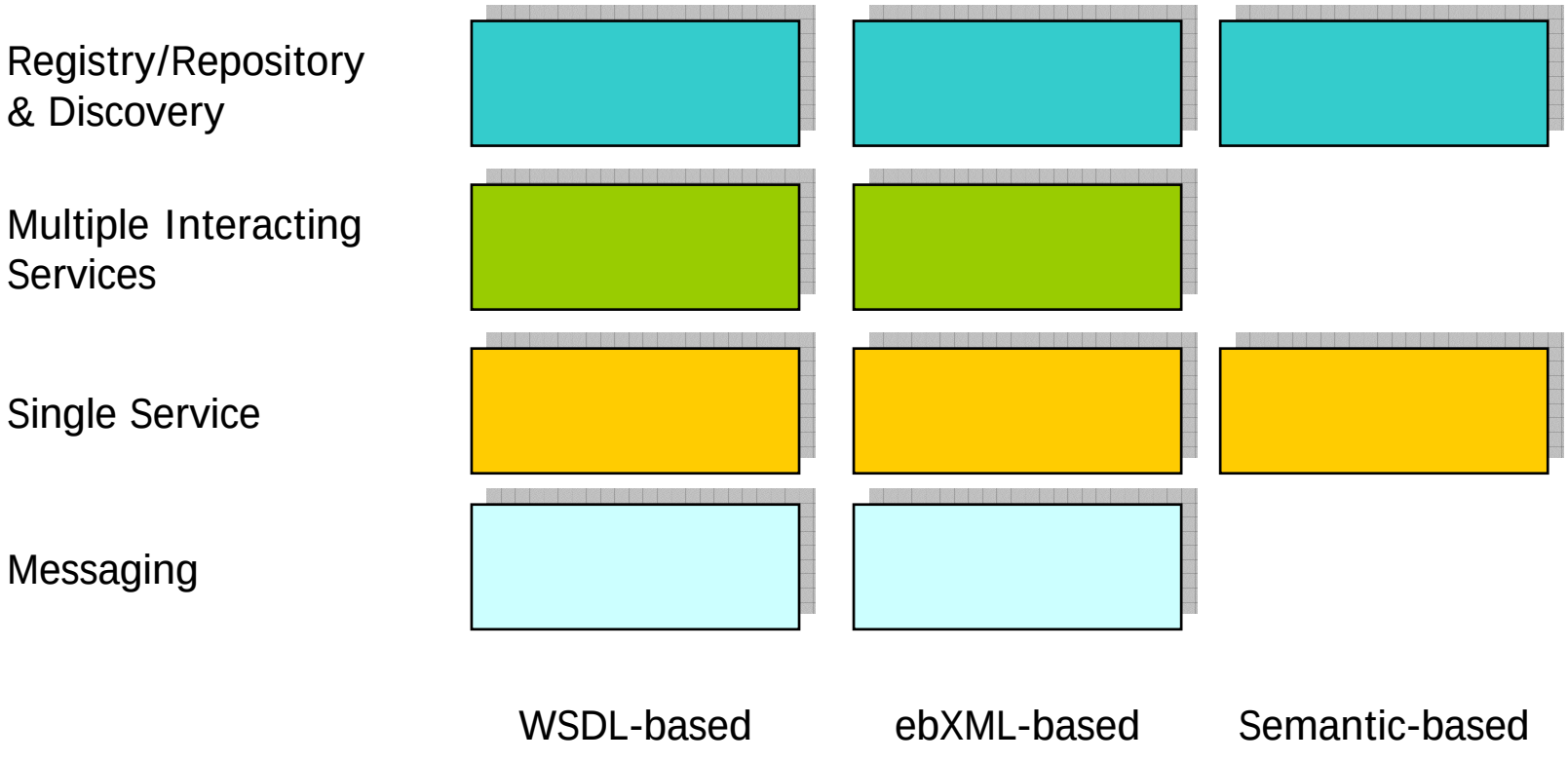
- Composition is about implementing new services
  - From the point of view of the client, a composite service and a basic (i.e., implemented in a traditional programming language) one are indistinguishable
- Choreography is about global modeling of N peers, for proving correctness, design-time discovery of possible partners and run-time bindings
- N.B.: There is a strong relationship between a service internal composition and the external choreographies it can participate in
  - if A is a composite service that invokes B, the A's composition schema must reflect the coordination protocol governing A – B interactions
  - in turn, the composition schema of A determines the coordination protocols that A is able to support (i.e., the choreographies it can participate in)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

Relevant Technologies and Abstractions

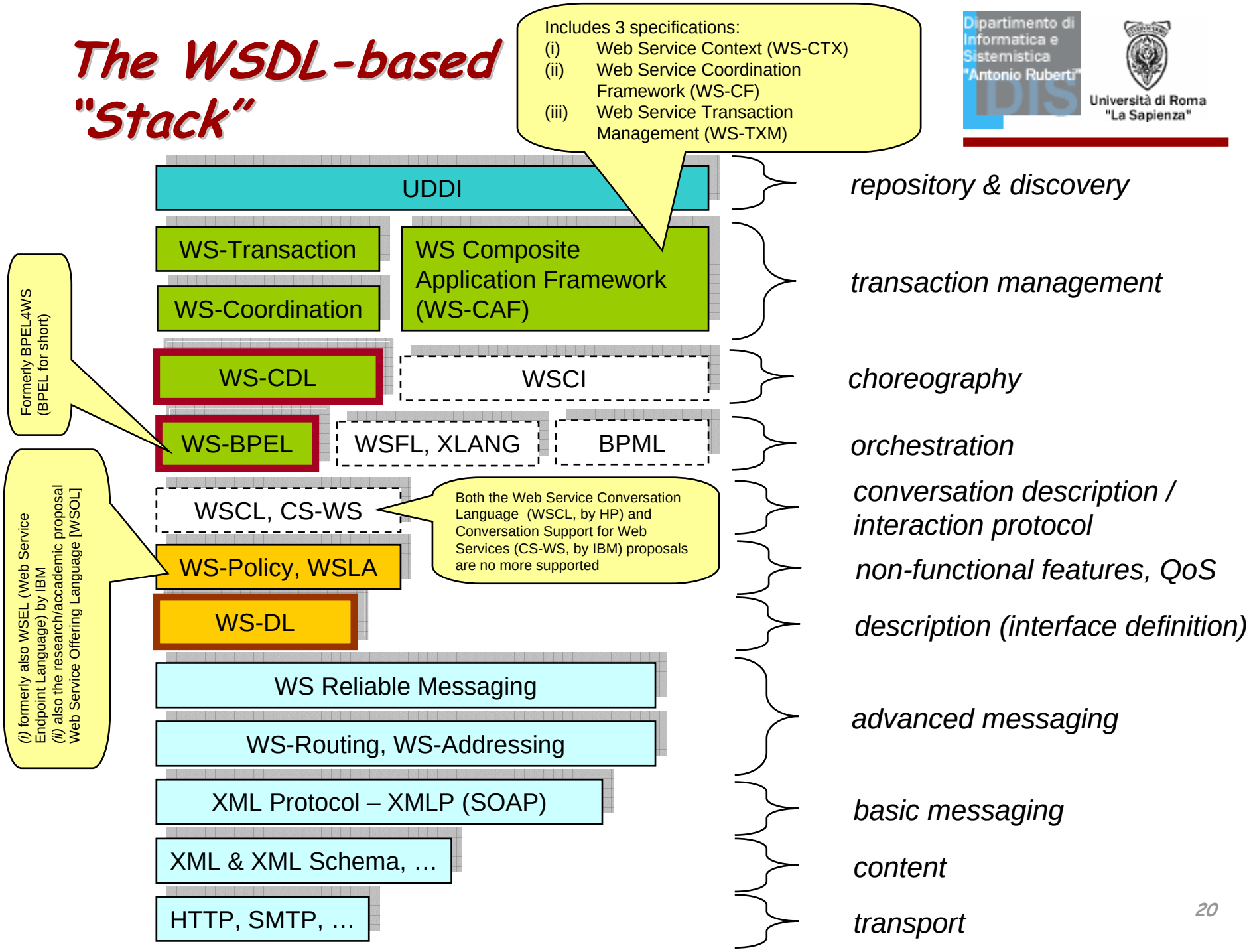
The "Stacks" of Service Technologies

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"



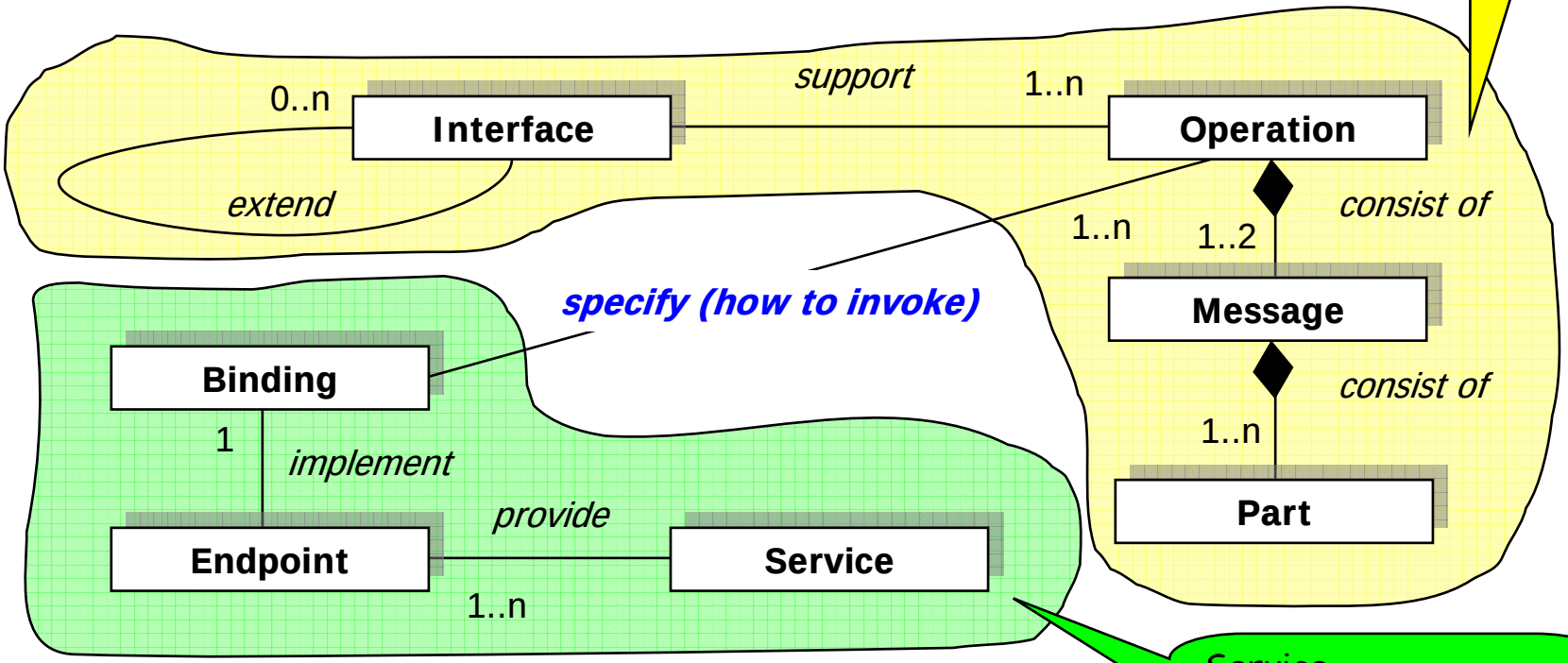
The WSDL-based "Stack"

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

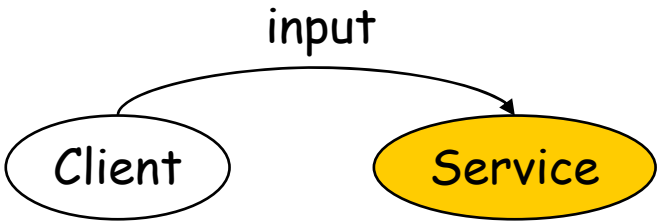


Web Service Definition Language (WS-DL)

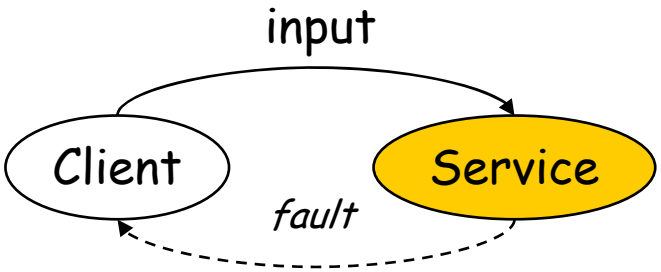
- WS-DL (v2.0) provides a framework for defining
  - Interface: operations and input/output formal parameters
  - Access specification: protocol bindings (e.g., SOAP)
  - Endpoint: the location of service



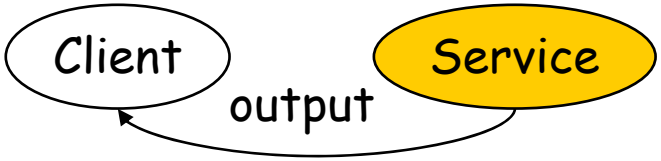
Message Exchange Patterns (1)



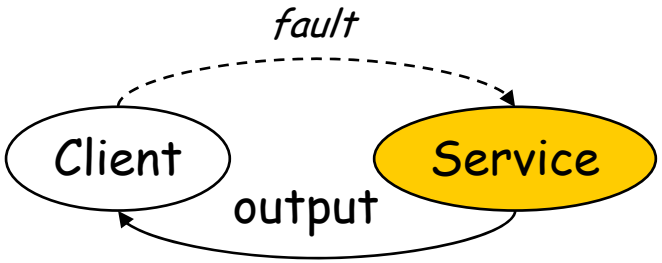
in-only (no faults)



robust in-only (message triggers fault)

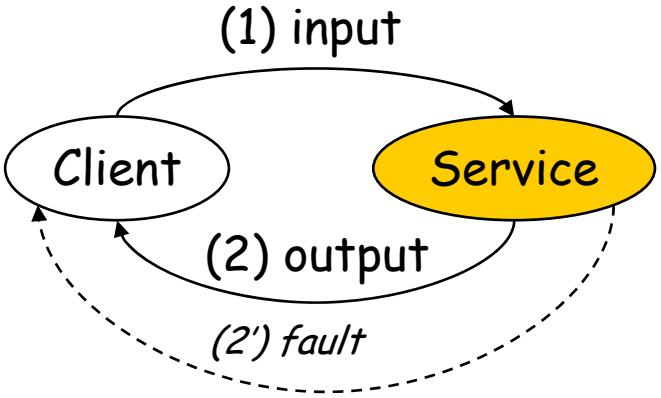


out-only (no faults)

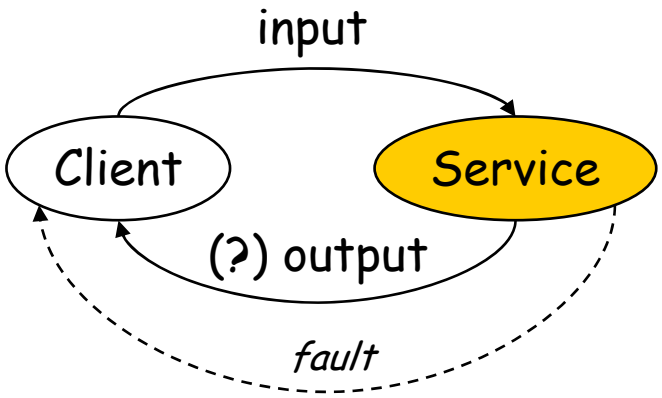


robust out-only (message triggers fault)

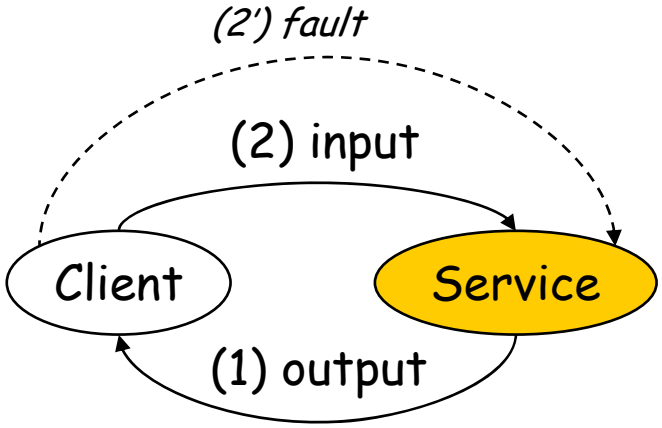
Message Exchange Patterns (2)



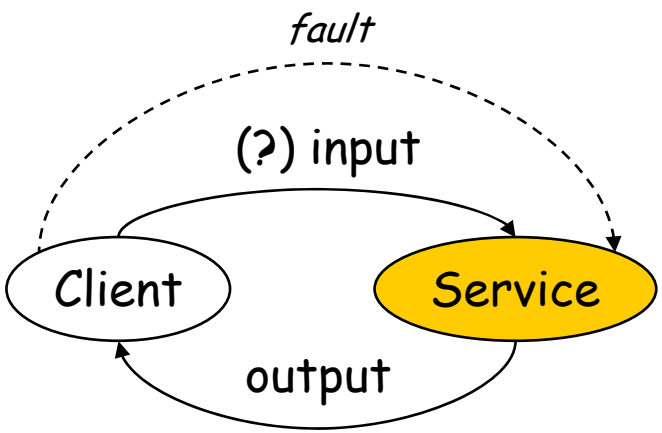
in-out (fault replaces message)



in-optional-out (message triggers fault)

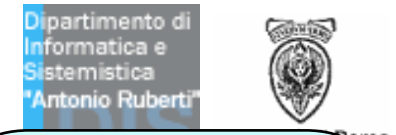


out-in (fault replaces message)



out-optional-in (message triggers fault)

An Example (1)



Definition of a message and its formal parameters

```
<definitions … >
   <types>
        <element name="ListOfSong_Type">
              <complexType><sequence>
                    <element minOccurs="0" maxOccu   "unbound“
                          name="SongTitle" type=  s:string"/>
              </sequence></complexType>
        </element>
        <element name="SearchByTitleRequest">
              <complexType><all>
                    <element name="containedInTitle“
                          type="xs:string"/>
              </all></complexType>
        </element>
        <element name="SearchByTitleResponse">
              <complexType><all>
                    <element name="matchingSongs“
                          xsi:type="ListOfSong_Type"/>
              </all></complexType>
         </element>
```

An Example (2)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

```
<element name="SearchByAuthorRequest">
        <complexType><all>
                <element name="authorName“
                        type="xs:string"/>
        </all></complexType>
</element>
<element name="SearchByAuthorResponse">
        <complexType><all>
                <element name="matchingSongs“
                        xsi:type="ListOfSong_Type"/>
        </all></complexType>
</element>
<element name="ListenRequest">
        <complexType><all>
                <element name="selectedSong“
                        type="xs:string"/>
        </all></complexType>
</element>
```

An Example (3)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

```
        <element name="ListenResponse">
                <complexType><all>
                        <element name="MP3fileURL" type="xs:string"/>
                </all></complexType>
        </element>
        <element name="ErrorMessage">
                <complexType><all>
                        <element name="cause" type="xs:string"/>
                </all></complexType>
        </element>
</types>
```

An Example (4)

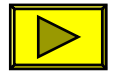
Definition of a service interface

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS

Università di Roma "La Sapienza"

```
<interface name="MP3ServiceType">
     <operation name="search_by_title" pattern="in-out">
            <input message="SearchByTitleRequest"/>
            <output message="SearchByTitleResponse"/>
            <outfault message="ErrorMessage"/>
     </operation>
     <operation name="search_by_author" pattern="in-out">
            <input message="SearchByAuthorRequest"/>
            <output message="SearchByAuthorResponse"/>
            <outfault message="ErrorMessage"/>
     </operation>
     <operation name="listen" pattern="in-out">
            <input message="ListenRequest"/>
            <output message="ListenResponse"/>
            <outfault message="ErrorMessage"/>
     </operation>
</interface>
</definitions>
```

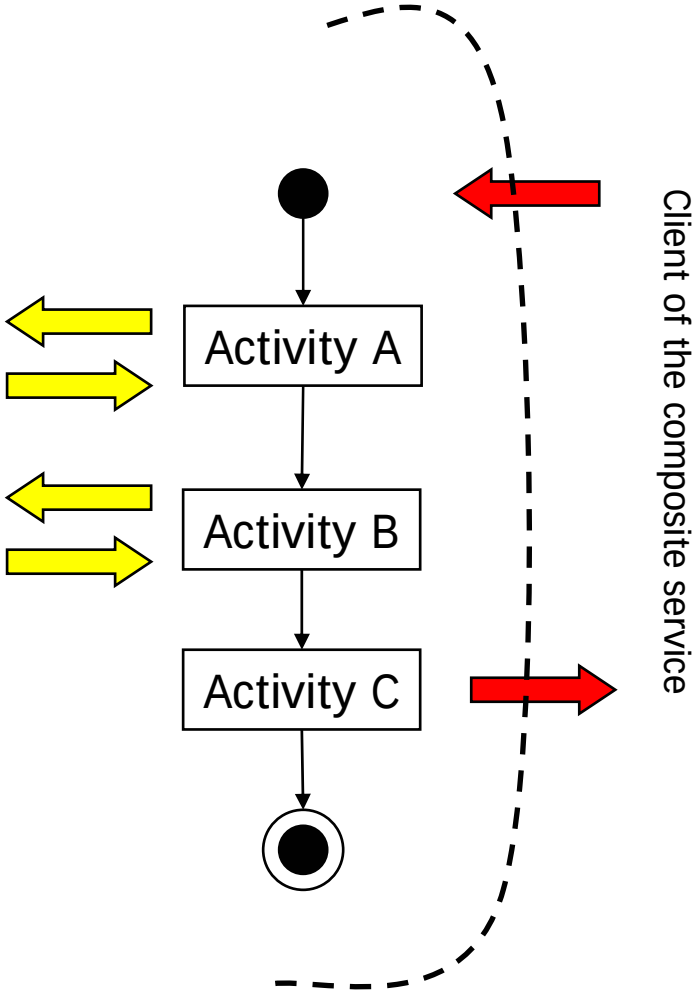
Definition of an operation and its message exchange pattern



Business Process Execution Language for Web Services (WS-BPEL)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

- Allows specification of composition schemas of Web Services
  - Business processes as coordinated interactions of Web Services
  - Business processes as Web Services
- Allows abstract and executable processes
- Influenced from
  - Traditional flow models
  - Structured programming
  - Successor of WSFL and XLANG
- Component Web Services described in WS-DL (v1.1)

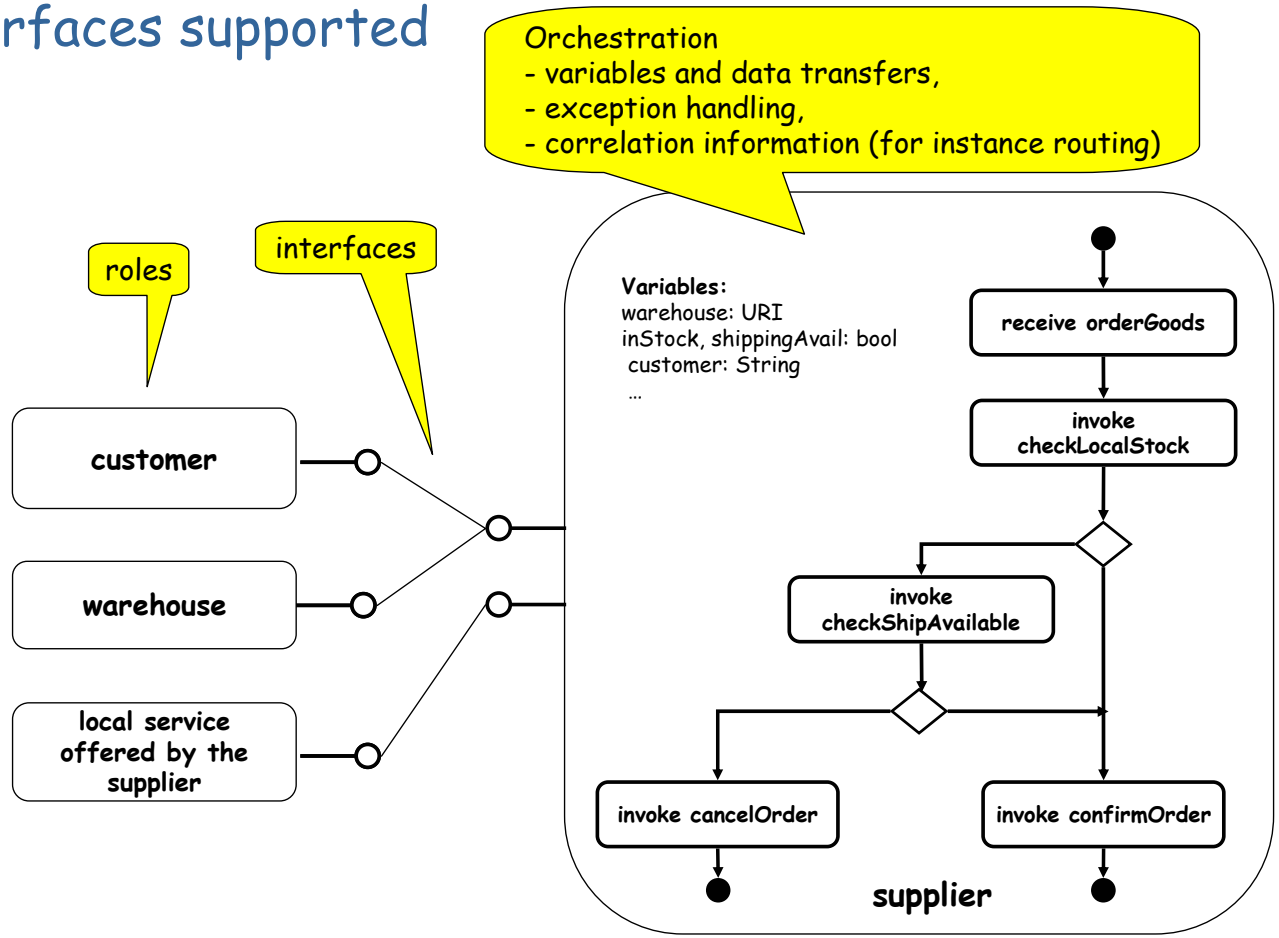


WS-BPEL Specification

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

An XML document specifying

- Roles exchanging messages with the composite service/process
- The (WSDL) interfaces supported by such roles
- The orchestration of the process
  - Variables and data transfer
  - Exception handling
  - Correlation information



Process Model (Activities)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

- Primitive
  - **invoke**: to invoke a Web Service (in-out) operation
  - **receive**: to wait for a message from an external source
  - **reply**: to reply to an external source message
  - **wait**: to remain idle for a given time period
  - **assign**: to copy data from one variable to another
  - **throw**: to raise exception errors
  - **empty**: to do nothing
- Structured
  - **sequence**: sequential order
  - **switch**: conditional routing
  - **while**: loop iteration
  - **pick**: choices based on events
  - **flow**: concurrent execution (synchronized by **links**)
  - **scope**: to group activities to be treated "transactionally" (managed by the same fault handler, within the same transactional context)

A link connects exactly one source activity S to exactly one target activity T; T starts only after S ends. An activity can have multiple incoming (possibly with join conditions) and outgoing links. Links can be guarded

Process Model
(Data Manipulation and Exception Handling)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Blackboard approach
  - a blackboard of variables is associated to each orchestration instance (i.e., a shared memory within an orchestration instance)
  - variables are not initialized at the beginning; they are modified (read/write) by assignments and messages
  - manipulation through XPath
- Try-catch-throw approach
  - definition of fault handlers
  - … but also event handlers and compensation handlers (for managing transactionality as in the SAGA model)

Choreography
(As Reported in Literature: Classical Ballet Style)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Consider a dance with more than one dancer
  - Each dancer has a set of steps that they will perform. They orchestrate their own steps because they are in complete control of their domain (their body)
  - A choreographer ensures that the steps all of the dancers make is according to some overall, pre-defined scheme. This is a choreography
  - The dancers have no control over the steps they make: their steps must conform to the choreography
  - The dancers have a single view-point of the dance
  - The choreographer has a multi-party or global view-point of the dance

Choreography
(A Possible Evolution: Jam Session Style)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Consider a jazz band with many players
  - There is a rhythm and a main theme. This is the choreography
  - Each player executes his piece by improvising variations over the main theme and following the given rhythm
  - The players still have a single view-point of the music; in addition they have full control over the music they play
  - There is a multi-party or global view-point of the music, but this is only a set of "sketchy" guidelines

WS-BPEL vs. WS-CDL

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Orchestration/WS-BPEL is about describing and executing a single peer
- Choreography/WS-CDL is about describing and guiding a global model (N peers)
- You should derive the single peer from the global model by projecting based on participant

WS-CDL Basics (1)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Participants & Roles
  - Role type
    - Enumerate the observable behavior that a collaborating participant exhibits
    - Behavior type specifies the operations supported
      - Optional WSDL interface type
  - Relationship type
    - Specify the mutual commitments, in terms of the Roles/Behavior types, **two** collaborating participants are required to provide
    - Note: all multi-party relationships are transformed into binary ones
  - Participant type
    - Enumerate a set of one or more Roles that a collaborating participant plays

WS-CDL Basics (2)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

- Channels
  - A channel realizes a *dynamic* point of collaboration, through which collaborating participants interact
    - Where & how to communicate a message
      - Specify the *Role/Behavior* and the *Reference* of a collaborating participant
      - Identify an *Instance* of a Role
    - Identify an instance of a conversation between two or more collaborating participants
      - A conversation groups a set of related message exchanges
- One or more channel(s) MAY be passed around from a Role to one or more other Role(s), possibly in a daisy fashion through one or more intermediate Role(s), creating new points of collaboration dynamically
  - A Channel type MAY restrict the types of Channel(s) allowed to be exchanged between the Web Services participants, through this Channel
  - A Channel type MAY restrict its usage, by specifying the number of times a Channel can be used

WS-CDL Basics (3)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Activities are the building blocks of a choreography
  - Basic Activity
    - **Interaction**: message exchange between participants
      - Only in-out and in-only
    - **Assign**: within one role, assign the value of a variable to another one
      - Variables can be about information (exchanged documents), states and channels
    - **No action**: do null
  - Ordering structure
    - **Sequence** (P.Q)
    - **Parallel** (P | Q)
    - **Choice** (P + Q)
  - **Perform**: a complete, separately defined choreography is performed
    - Basis for scalable modeling

Attention: *a choreography performing another one is referred to as "choreography composition" in the standard*

WS-CDL Basics (4)

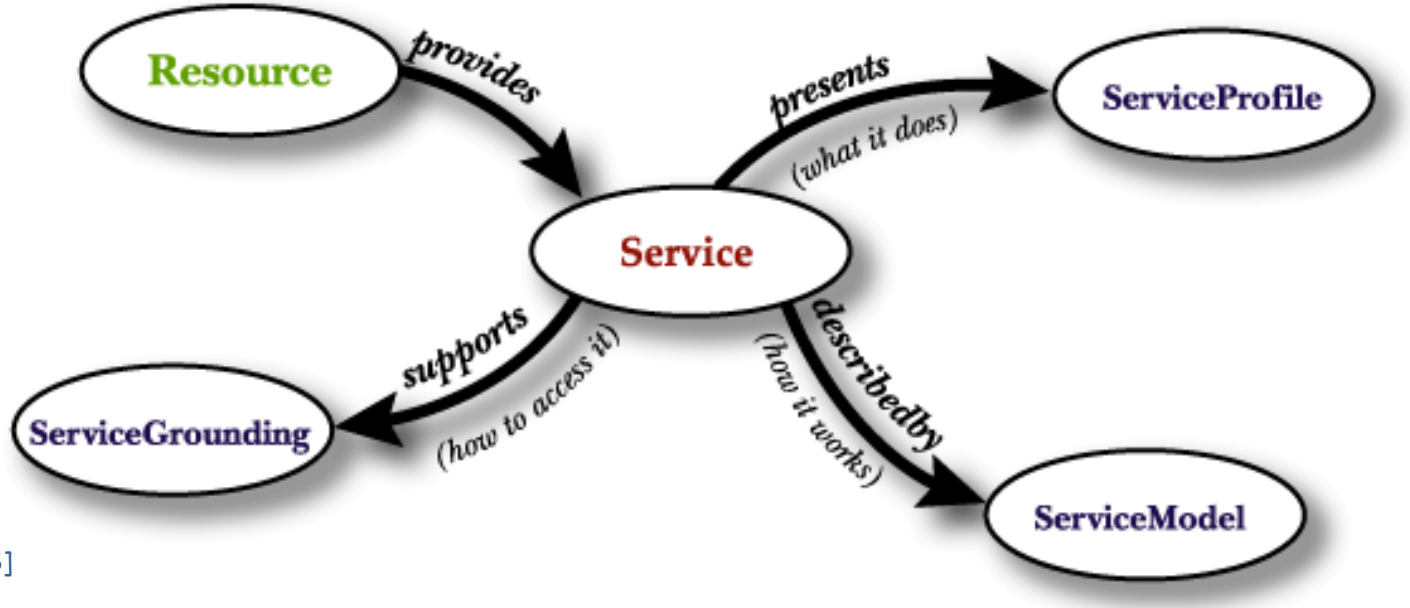
Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- A Choreography combines all previous elements, forming a collaboration unit of work
  - Enumerate all the binary relationships interactions act in
  - Localize the visibility of variables
    - Using variable definitions
  - Prescribe alternative patterns of behavior
    - Using work/units and reactions
  - Enable Recovery
    - Using work/units and reactions
    - Backward: handle exceptional conditions
    - Forward: finalize already completed activities

OWL-S (formely DAML-S)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- An emerging standard to add semantics
  - An upper ontology for describing properties & capabilities of Web Services using OWL
- Enable automation of various activities (e.g., service discovery & selection)



[from DAML-S]

OWL-S Service Profile
(What it does)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

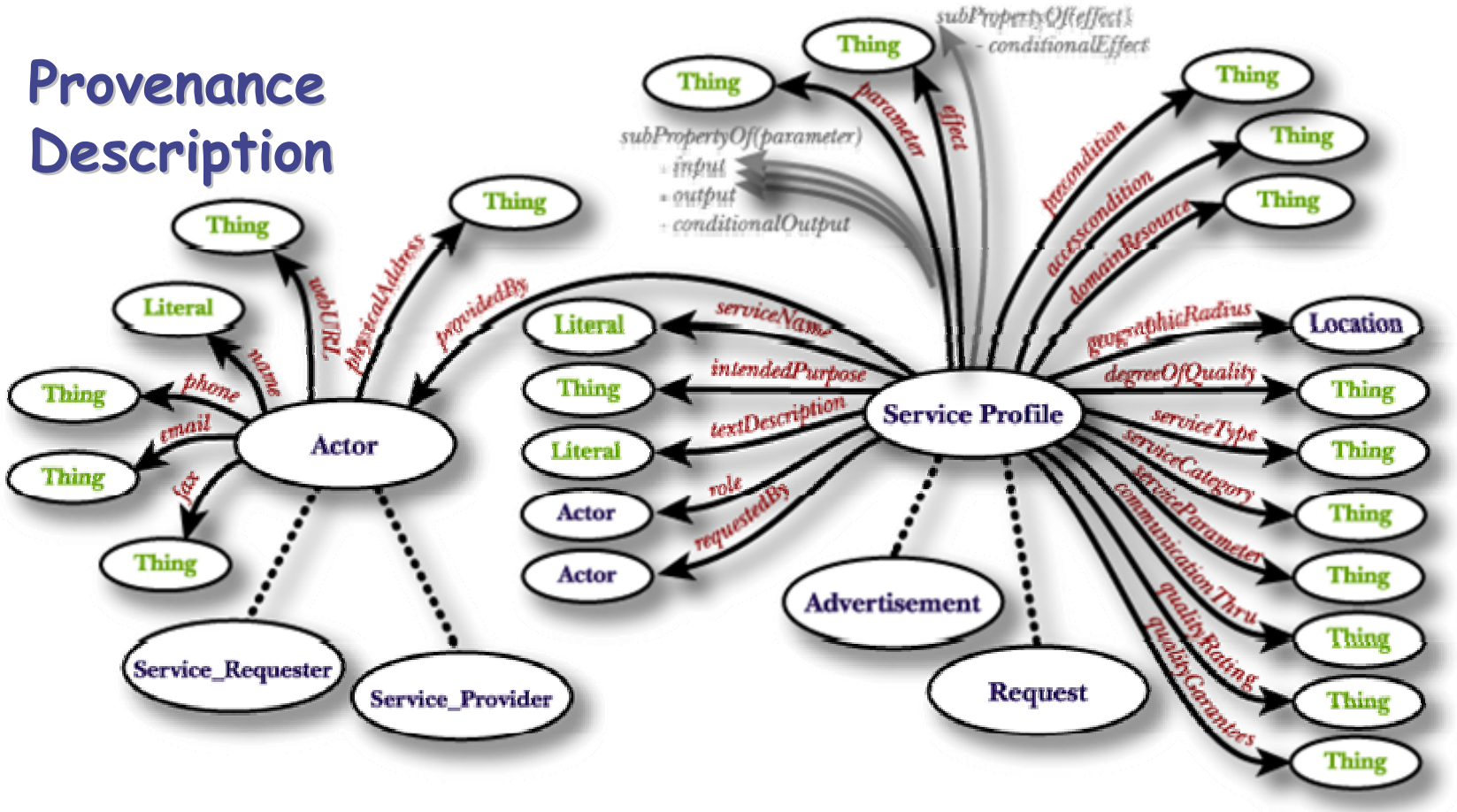
- High-level characterization/summary of a service
  - Provider & participants
  - Capabilities
  - Functional attributes (e.g., QoS, region served)
- Used for
  - Populating service registries
    - A service can have many profiles
  - Automated service discovery
  - Service selection (matchmaking)
- One can derive:
  - Service advertisements
  - Service requests

OWL-S Service Profile

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" — Università di Roma "La Sapienza"

Capability Description

Provenance Description

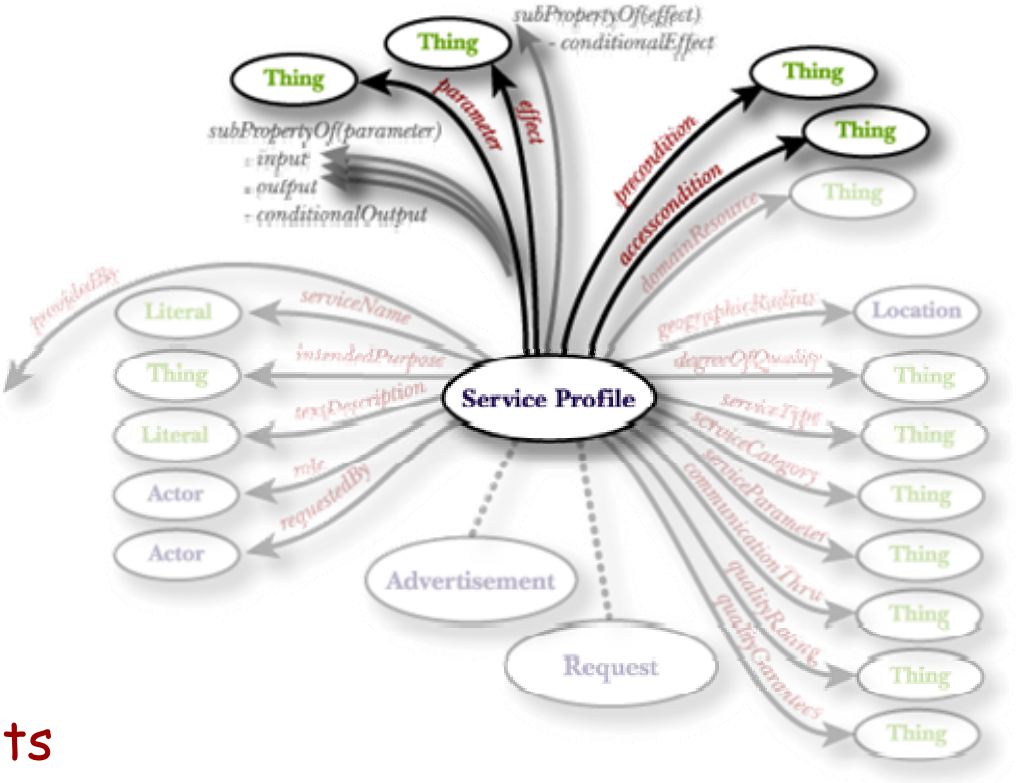


Functional Attributes

[from DAML-S]

Capability Description

- Specification of what the service provides
  - High-level functional representation in terms of:
    - preconditions
    - inputs
    - (conditional) outputs
    - (conditional) effects



IOPE

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

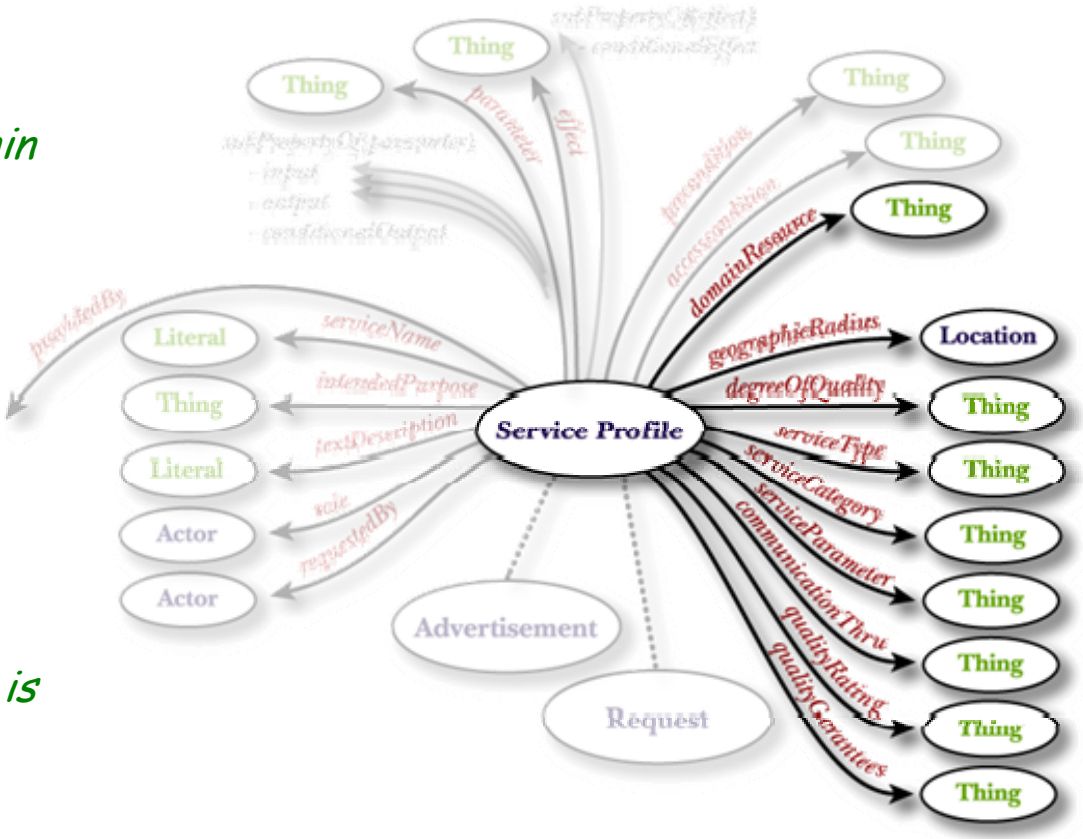
- Inputs
  - Set of necessary inputs that the requester should provide to invoke the service
- (Conditional) Outputs
  - Results that the requester should expect after interaction with the service provider is completed
- Preconditions
  - Set of conditions that should hold prior to service invocation
- (Conditional) Effects
  - Set of statements that should hold true if the service is invoked successfully
  - Often refer to real-world effects, e.g., a package being delivered, or a credit card being debited

Functional Attributes

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" — Università di Roma "La Sapienza"

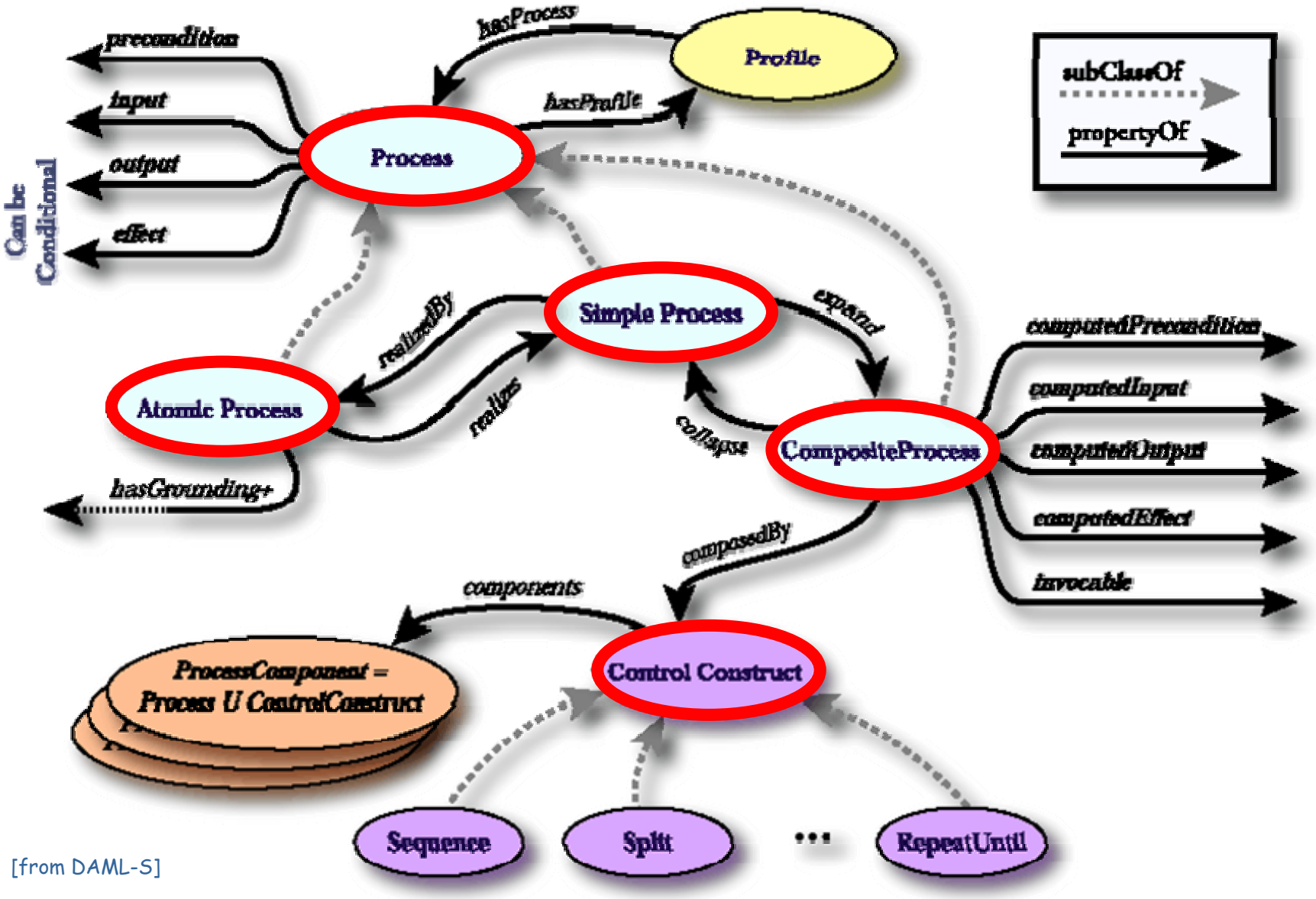
Provide supporting information about the service, including:

- geographical scope
  - *Pizza Delivery only within the Pittsburgh area*
- quality descriptions and guarantees
  - *Stock quotes delivered within 10 secs*
- service types, service categories
  - *Commercial / Problem Solving, etc.*
- service parameters
  - *Average Response time is currently ...*



OWL-S Service Model (How it works)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"



[from DAML-S]

OWL-S Process Ontology

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Atomic processes: directly invokable, no subprocesses, executed in a single step
- Composite processes: consist of other (non-composite or composite) processes
- Simple processes: a virtual view of atomic process or composite process (as a "black box")

Process Model

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Constructs for complex processes
  - Sequence
  - Concurrency: Split; Split+Join; Unordered
  - Choice
  - If-Then-Else
  - Looping: Repeat-Until; Iterate (non-deterministic)
- Data Flow
  - No explicit variables, no internal data store
  - Predicate "sameValues" to match input of composite service and input of subordinate service
- Less refined than, e.g., WS-BPEL

Enhancements

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Recent proposals aim at improving and detailing process modeling and dynamic semantics …
  - WSMO (Web Service Modeling Ontology)
  - SWSL (Semantic Web Service Language)
- … work in progress !!

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

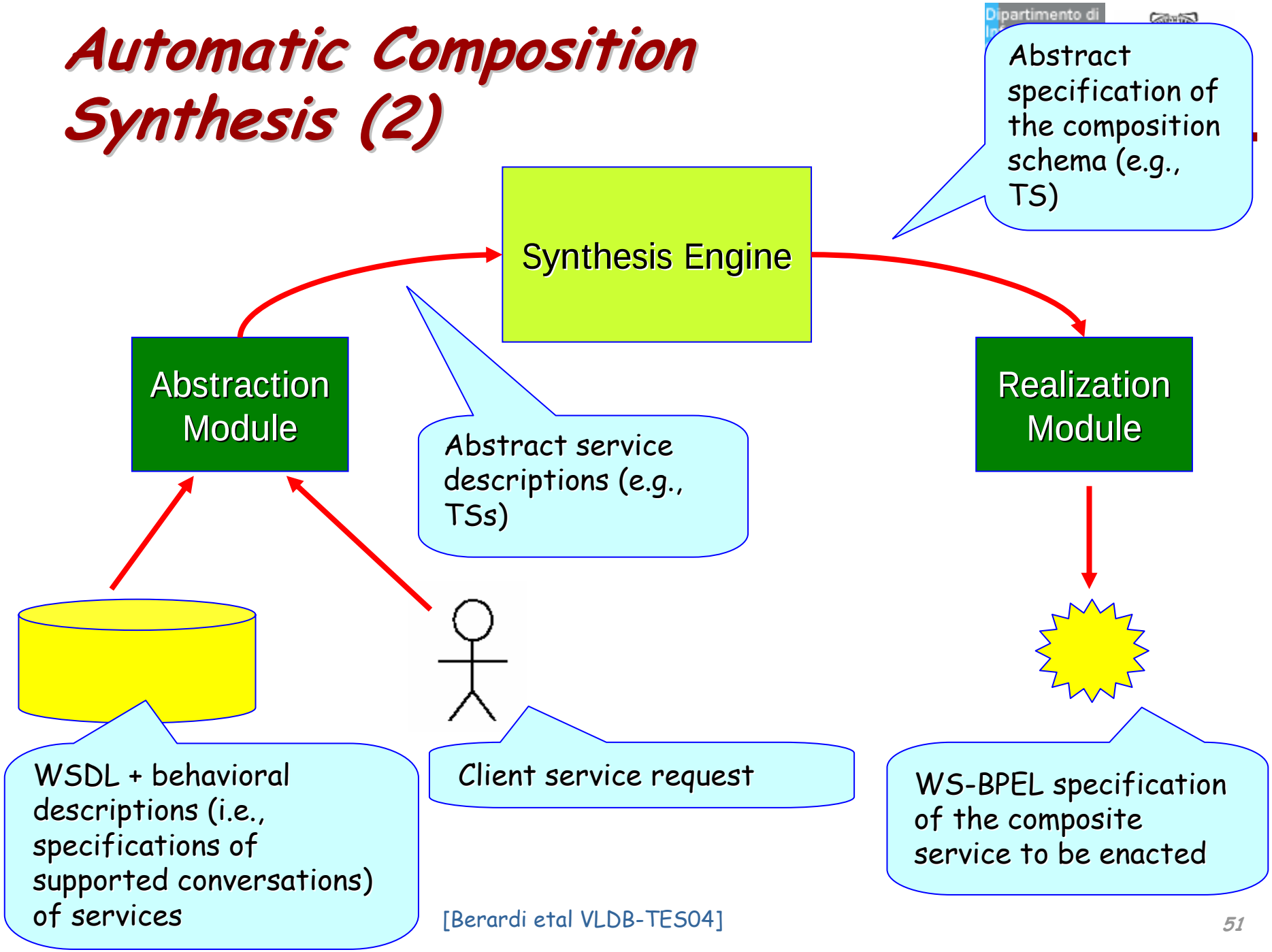
Introducing Automatic Composition

Automatic Composition Synthesis (1)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Given:
  - a set (S_1, …, S_n) of component services
  - a client service request T
- Automatically build:
  - a composition schema CS that fulfills T by suitably orchestrating (S_1, …, S_n)

Automatic Composition Synthesis (2)



[Berardi etal VLDB-TES04]

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

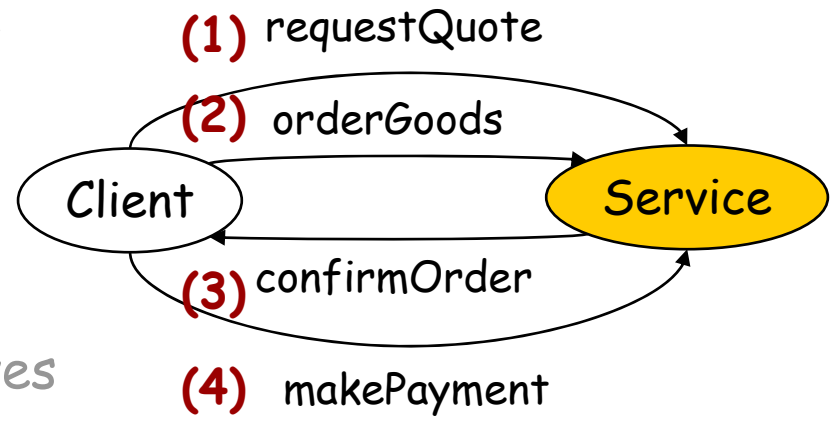
Abstracting over Technologies

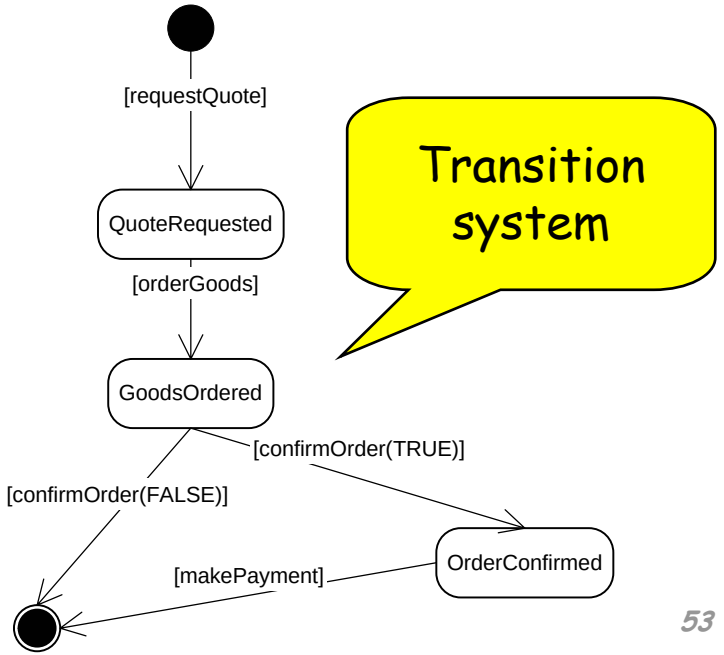
Modeling Services as Transition Systems

Services

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- A service is characterized by the set of (atomic) operations that it exports …
- … and possibly by constraints on the possible conversations
  - Using a service typically involves performing sequences of operations in a particular order (conversations)
  - During a conversation, the client typically chooses the next operation to invoke (on the basis of previous results, etc.) among the ones that the service allows at that point

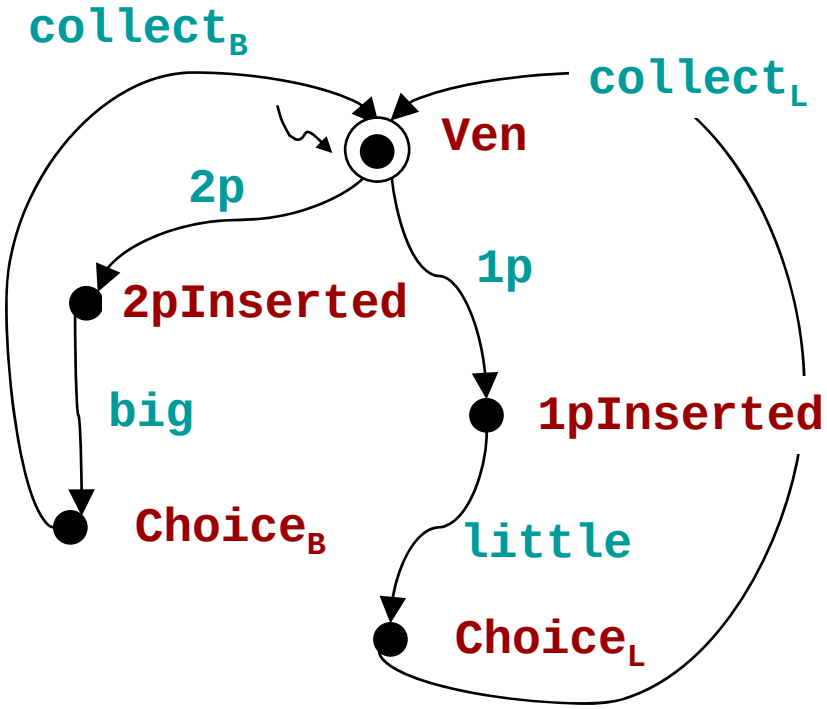




Transition Systems

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- A transition system (TS) is a tuple $T = \langle A, S, S^0, \delta, F \rangle$ where:
  - A is the set of actions
  - S is the set of states
  - $S^0 \subseteq S$ is the set of initial states
  - $\delta \subseteq S \times A \times S$ is the transition relation
  - $F \subseteq S$ is the set of final states

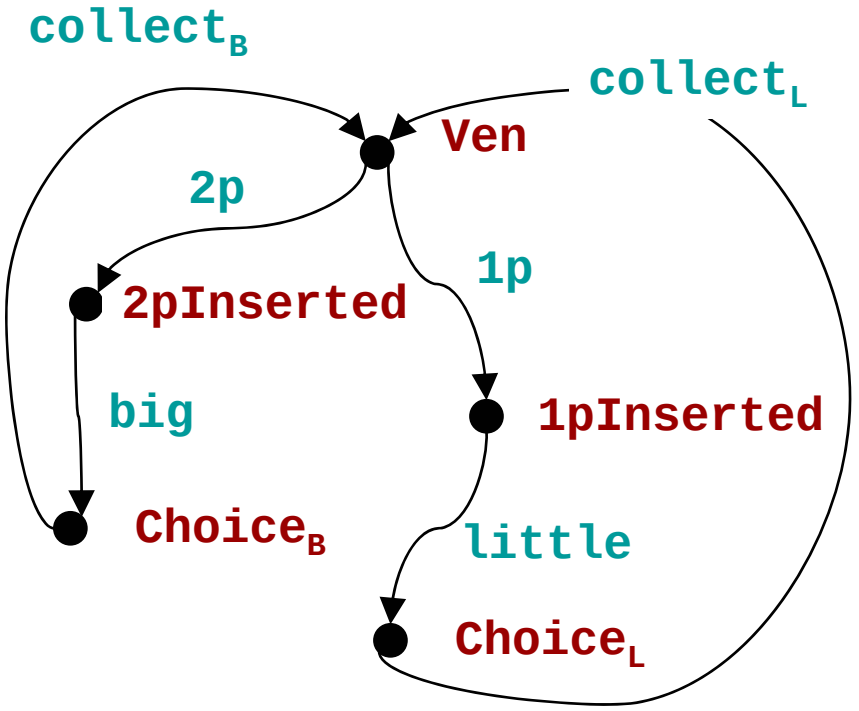


Process Algebras and TSs

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

- Process theory:
  - a process is a term of an algebraic language
  - a transition $E \rightarrow_a F$ means that process E may become F by performing (participating in, or accepting) action a
  - structured rules guide the derivation

- A graph:
  - nodes are process terms
  - labelled directed arcs between nodes



$Ven = 2p.2pInserted + 1p.1pInserted$

$2pInserted = big.Choice_B$

$1pInserted = little.Choice_L$

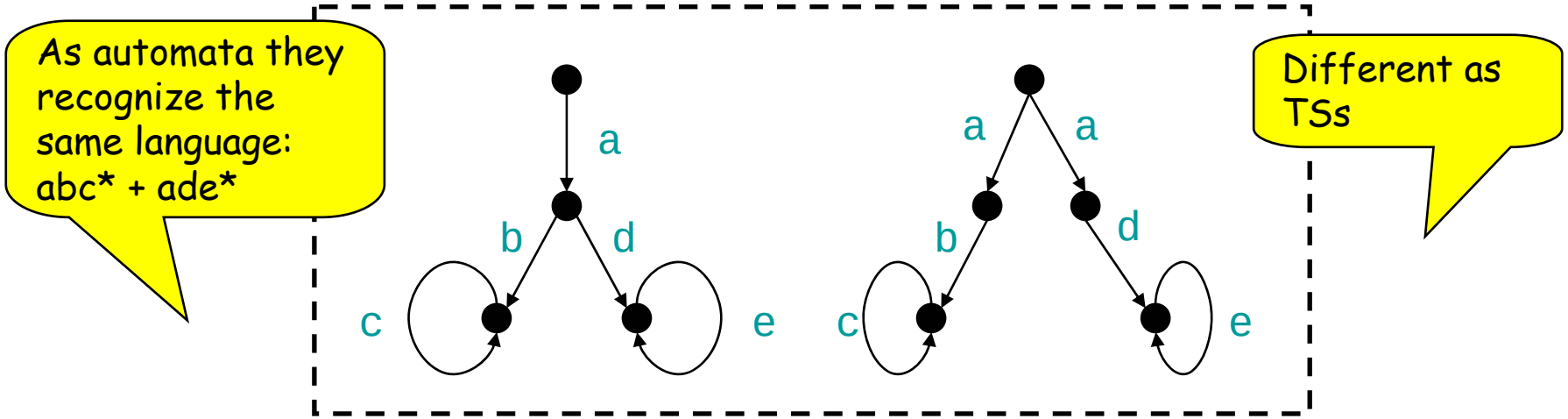
$Choice_B = collect_B.Ven$

$Choice_L = collect_L.Ven$

Automata vs. Transition Systems

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

- Automata
  - define sets of runs (or traces or strings): (finite) length sequences of actions
- TSs
  - … but I can be interested also in the alternatives "encountered" during runs, as they represent client's "choice points"



WS-DL is the Set of Actions

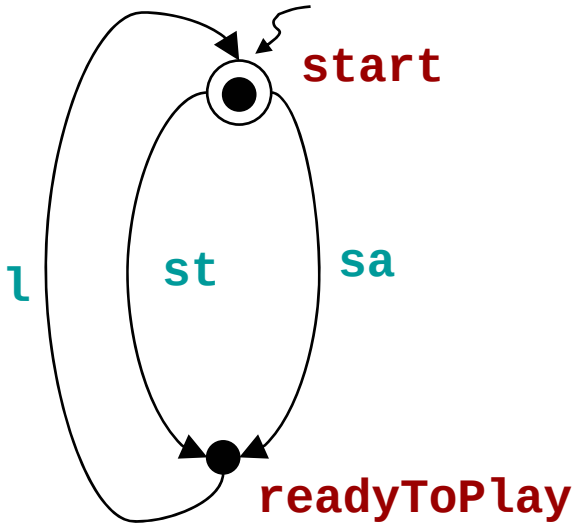
Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- A message exchange pattern (and the related operation) represents an interaction with the service client
  - an action that the service can perform by interacting with its client
- Abstracting from formal parameters, we can associate a different symbol to each operation …
- … thus obtaining the alphabet of actions

An Example

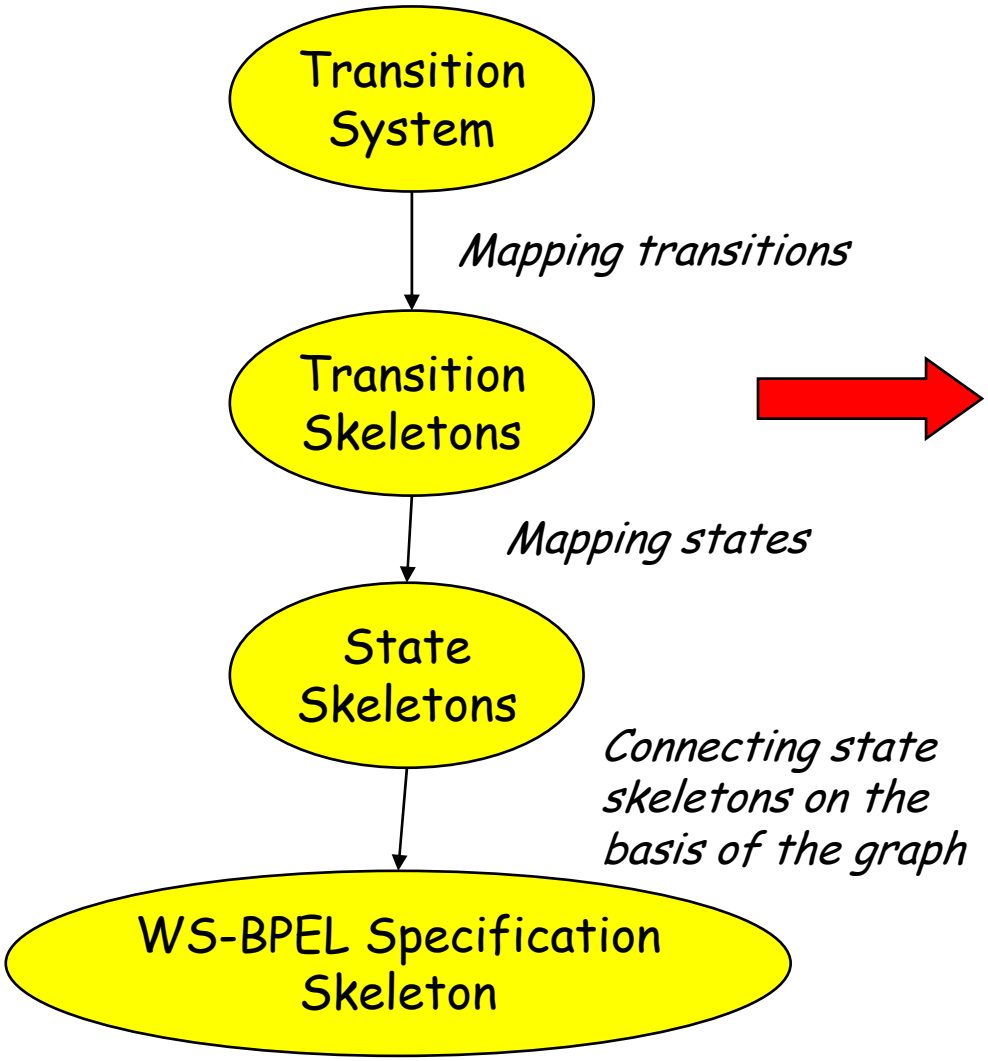
Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- The `MP3ServiceInterface` defines 3 actions:
  - search_by_title / st
  - search_by_author / sa
  - listen / l



- Formally $A = \{st, sa, l\}$

From a TS to WS-BPEL (1)




```
<process name = "…">
    <partnerLinks>
     …
    </partnerLinks>

    <variables>
     …
    </variables>

    <flow>
       <links>
         …
       </links>
       <!-- state skel. -->
       …
       <!-- state skel. -->
    </flow>

</process>
```

From a TS to WS-BPEL (2)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS

Università di Roma "La Sapienza"

***Intuition* [Baina etal CAISE04, Berardi etal VLDB-TES04]**

1. Each transition corresponds to a WS-BPEL pattern consisting of *(i)* an **<onMessage>** operation (in order to wait for the input from the client of the composite service), *(ii)* followed by the effective logic of the transition, and then *(iii)* a final operation for returning the result to the client. Of course both before the effective logic and before returning the result, messages should be copied forth and back in appropriate variables
2. All the transitions originating from the same state are collected in a **<pick>** operation, having as many **<onMessage>** clauses as transitions originating from the state
3. The WS-BPEL file is built visiting the graph in depth, starting from the initial state and applying the previous rules.

N.B.: (1) and (2) works for in-out interactions (the ones shown in the following). Simple modifications are needed for in-only, robust-in-only and in-optional-out. The other kinds of interactions implies a proactive behaviour of the composite service, possibly guarded by **<onAlarm>** blocks.
(3) works for acyclic TS. See later for cycle management.

Transition Skeletons

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

```
<onMessage … >
      <sequence>
            <assign>
               <copy>
                  <from variable="input" … />
                  <to variable="transitionData" … />
               </copy>
            </assign>
            < !-- logic of the transition -->
            <assign>
               <copy>
                  <from variable="transitionData" … />
                  <to variable="output" … />
               </copy>
            </assign>
            <reply … />
      </sequence>
</onMessage>
```

State Skeletons

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- N transitions from state S_i are mapped onto:

```
<pick name = "Si">
      <!-- transition #1 -->
      <onMessage ... >
           <!-- transition skeleton -->
      </onMessage>
      ... ... ...
      <!-- transition #N -->
      <onMessage ... >
           <!-- transition skeleton -->
      </onMessage>
</pick>
```

Mapping the TS

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

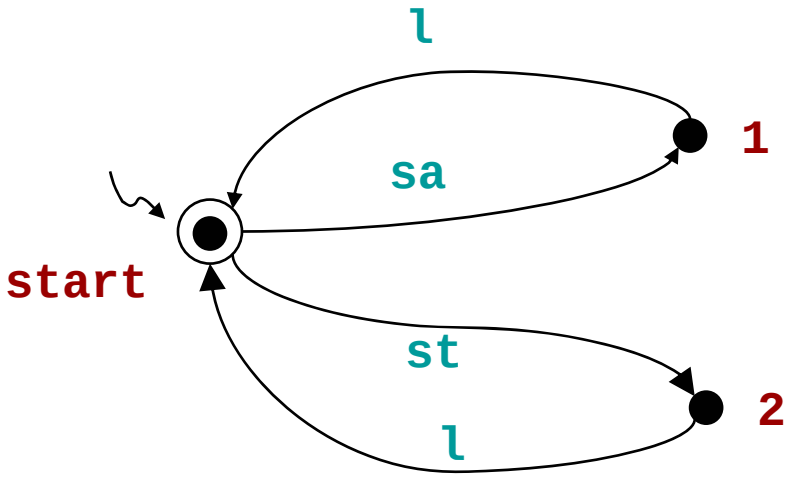
- All the **<pick>** blocks are enclosed in a surrounding **<flow>**; the dependencies are modeled as **<link>**s
  - **<link>**s are controlled by specific variables $\mathbf{S_i}$**-to-**$\mathbf{S_j}$ that are set to TRUE iff the transition $S_i \rightarrow S_j$ is executed
  - Each state skeleton has many outgoing **<link>**s as states connected in output, each going to the appropriate **<pick>** block

Mapping Cyclic TSs
(Intuition)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Identify all the cycles
- Enclose the involved state skeletons inside a **<while>** block controlled by a condition **(!exit)** (**exit** is a variable defined ad hoc)
  - **exit** is set to TRUE by any transition that "goes out" of the cycle
  - The overall **<while>** block is connected to other state skeletons by appropriate **<link>**s
- Special cases:
  - A state S with self-transitions can be represented as a **<pick>** block enclosed in a **<while>** block controlled by a condition **(V_s)** (the variable V_s is set to FALSE by other non self-transitions)
  - Cycles starting from the initial state should not be considered, as they can be represented as the start of a new instance

An Example (1)




```
<partnerLinks>
  <!-- The "client" role represents the requester of this composite service -->
      <partnerLink name="client"
                   partnerLinkType="tns:Transition"
                   myRole="MP3ServiceTypeProvider"
                   partnerRole="MP3ServiceTypeRequester"/>
      <partnerLink name="service"
                   partnerLinkType="nws:MP3CompositeService"
                   myRole="MP3ServiceTypeRequester"
                   partnerRole="MP3ServiceTypeProvider"/>
</partnerLinks>
```

An Example (2)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

```
<variables>
      <variable name="input" messageType="tns:listen_request"/>
      <variable name="output" messageType="tns:listen_response"/>
      <variable name="dataIn" messageType="nws:listen_request"/>
      <variable name="dataOut" messageType="nws:listen_response"/>
</variables>

       <pick>
          <onMessage partnerLink="client"
                     portType="tns:MP3ServiceType"
                     operation="listen"
                     variable="input">
                     <sequence>
                          <assign>
                              <copy>
                                  <from variable="input" part="selectedSong"/>
                                  <to variable="dataIn" part="selectedSong"/>
                              </copy>
                          </assign>
                          … …
                          <assign>
                              <copy>
                                  <from variable="dataOut" part="MP3FileURL"/>
                                  <to variable="output" part="MP3FileURL"/>
                              </copy>
                          </assign>
                          <reply name="replyOutput"
                                 partnerLink="client"
                                 portType="tns:MP3ServiceType"
                                 operation="listen"
                                 variable="output"/>
                     </sequence>
          </onMessage>
          … …
       </pick>
```

An Example (3)

Università di Roma "La Sapienza"

A new instance is created in the initial state. This resolve also the presence of the cycles without the need of enclosing <while>

The <sa> transition skeleton should set variables:
start-to-1 = TRUE
start-to-2 = FALSE

The <st> transition skeleton should set variables:
start-to-1 = FALSE
start-to-2 = TRUE

```
<process suppressJoinFailure = "no">
   <flow>
   <links>
      <link name="start-to-1"/>
      <link name="start-to-2"/>
    </links>

    <pick createInstance = "yes">
       <onMessage="sa">
          <sequence>
             <copy>...</copy>
             … …
             <copy>...</copy>
             <reply ... />

          </sequence>
       </onMessage>
       <onMessage="st">
          <sequence>
             <copy>...</copy>
             … …
             <copy>...</copy>
             <reply ... />
          </sequence>
       </onMessage>
       <source linkName="start-to-1" transitionCondition = "bpws:getVariableData('start-to-1') = 'TRUE' " />
       <source linkName="start-to-2" transitionCondition = "bpws:getVariableData('start-to-2') = 'TRUE' " />
   </pick>
```

An Example (4)

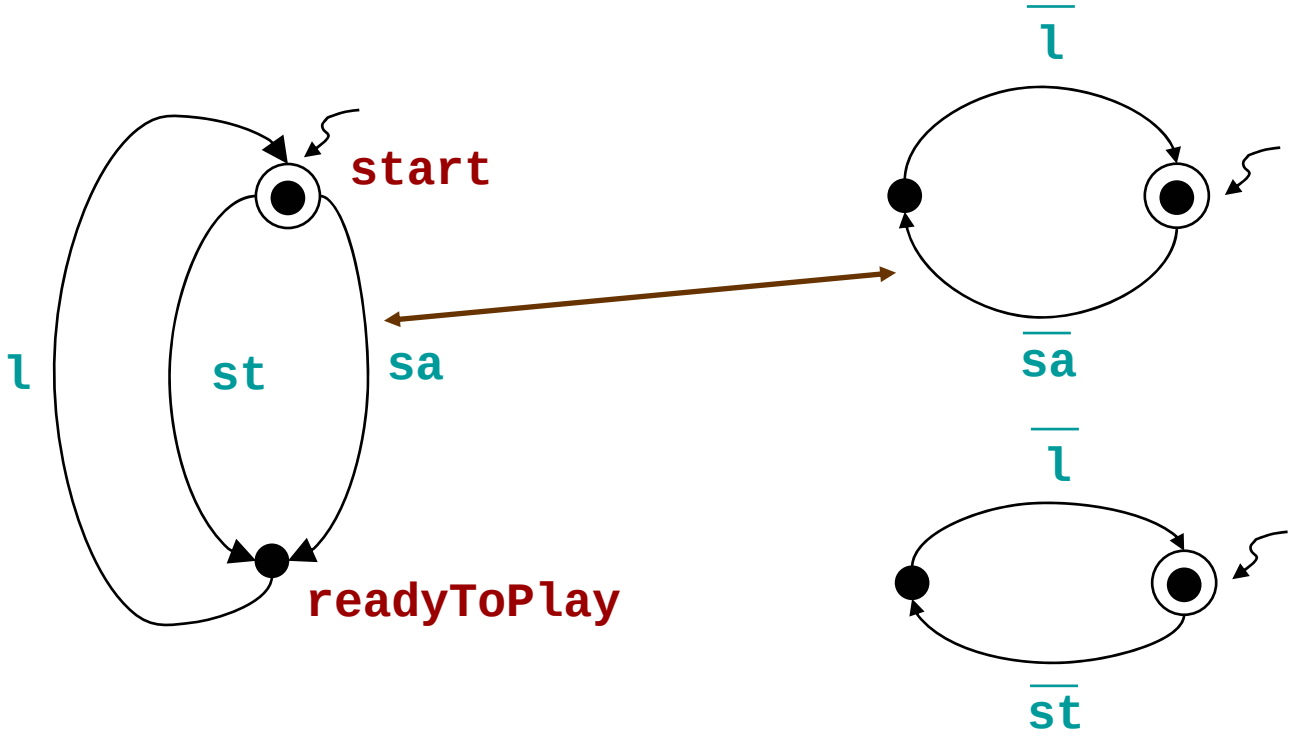
Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

```
<pick>
   <onMessage="I">
         <sequence>
            <copy>...</copy>
            … …
           <copy>...</copy>
           <reply ... />
         </sequence>
   </onMessage>
   <target linkName="start-to-1" />
</pick>
<pick>
   <onMessage="I">
         <sequence>
            <copy>...</copy>
            … …
           <copy>...</copy>
           <reply ... />
         </sequence>
   </onMessage>
   <target linkName="start-to-2" />
</pick>
</process>
```

TSs and Choreography
(only an intuition :-))

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" — Università di Roma "La Sapienza"

- A Choreography can be seen as the specification of a set of concurrent peers, each one exposing a TS, that fulfills the global model



References

[ACKM04] - G. Alonso, F. Casati, H. Kuno, V. Machiraju: Web Services. Concepts, Architectures and Applications. Springer-Verlag 2004

[VLDBJ01] - F. Casati, M.C. Shan, D. Georgakopoulos (eds.): Special Issue on e-Services. VLDB Journal, 10(1), 2001
Based on the 1st International Workshop on Technologies for e-Services (VLDB-TES 2001)

[CACM03] – M.P. Papazoglou, D. Georgakopoulos (eds.): Special Issue on Service Oriented Computing. Communications of the ACM 46(10), 2003

[WSOL] - V.Tosic, B. Pagurek, K. Patel, B. Esfandiari, W. Ma: Management Applications of the Web Service Offerings Language (WSOL). To be published in Information Systems, Elsevier, 2004.
An early version of this paper was published in Proc. of CAiSE'03, LNCS 2681, pp. 468-484, 2003

[Berardi etal WSCC04] - D. Berardi, R. Hull, M. Gruninger, S. McIlraith: Towards a First-Order Ontology for Semantic Web Services. Proc. W3C International Workshop on Constraints and Capabilities for Web Services (WS-CC), 2004, http://www.w3.org/2004/06/ws-cc-cfp.html

[Benatallah etal IJCIS04] - B. Benatallah, F. Casati, H. Skogsrud, F. Toumani: Abstracting and Enforcing Web Service Protocols, International Journal of Cooperative Information Systems (IJCIS), 13(4), 2004

References

[Baina etal CAISE04] K. Baina, B. Benatallah, F. Casati, F. Toumani: Model-driven Web Service Development, Proc. of CAiSE'04, LNCS 3084, 2004

[Berardi etal ICSOC03] - D. Berardi, D. Calvanese, G. De Giacomo, M. Lenzerini, M. Mecella: Proc. of ICSOC'03, LNCS 2910, 2004

[Berardi etal VLDB-TES04] - D. Berardi, D. Calvanese, G. De Giacomo, M. Lenzerini, M. Mecella: Post-proc. of VLDB-TES'04, to appear

[Stirling Banff '96] - C. Stirling: Modal and Temporal Logics for Processes. Banff Higher Order Workshop, LNCS 1043, 1996. Available at: http://homepages.inf.ed.ac.uk/cps/banff.ps

[ebpml] - Jean-Jacques Dubray: the ebPML.org Web Site, http://www.ebpml.org/

[DAML-S] – DAML Semantic Web Services, http://www.daml.org/services

References

[WS-Policy] - Web Services Policy Framework (WS-Policy), September 2004, http://www-106.ibm.com/developerworks/library/specification/ws-polfram/

[WSCL] - Web Services Conversation Language (WSCL) 1.0. W3C Note, 14 March 2002, http://www.w3.org/TR/wscl10/

[WSLA] - A. Dan, D. Davis et al: Web Services On Demand: WSLA-driven Automated Management. IBM Systems Journal, 43(1), 2004

[ebXML] - Electronic Business using eXtensible Markup Language, http://www.ebxml.org/

[OASIS] - Organization for the Advancement of Structured Information Standards, http://www.oasis-open.org/home/index.php

[WSDL] - R. Chinnici, M. Gudgin, J.J. Moreau, J. Schlimmer, and S. Weerawarana, Web Services Description Language (WSDL) 2.0, Available on line: http://www.w3.org/TR/wsdl20, 2003, W3C Working Draft.

[BPEL4WS] - T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic, and S. Weerawarana, Business Process Execution Language for Web Services (BPEL4WS) -Version 1.1, http://www-106.ibm.com/developerworks/library/ws-bpel/, 2004

References

[WS-CDL] - N. Kavantzas, D. Burdett, G. Ritzinger, Y. Lafon: Web Services Choreography Description Language (WS-CDL) Version 1.0, Available on line at: http://www.w3.org/TR/ws-cdl-10/, W3C Working Draft.

[UDDI] – Universal Discovery, Description and Integration, http://www.uddi.org/

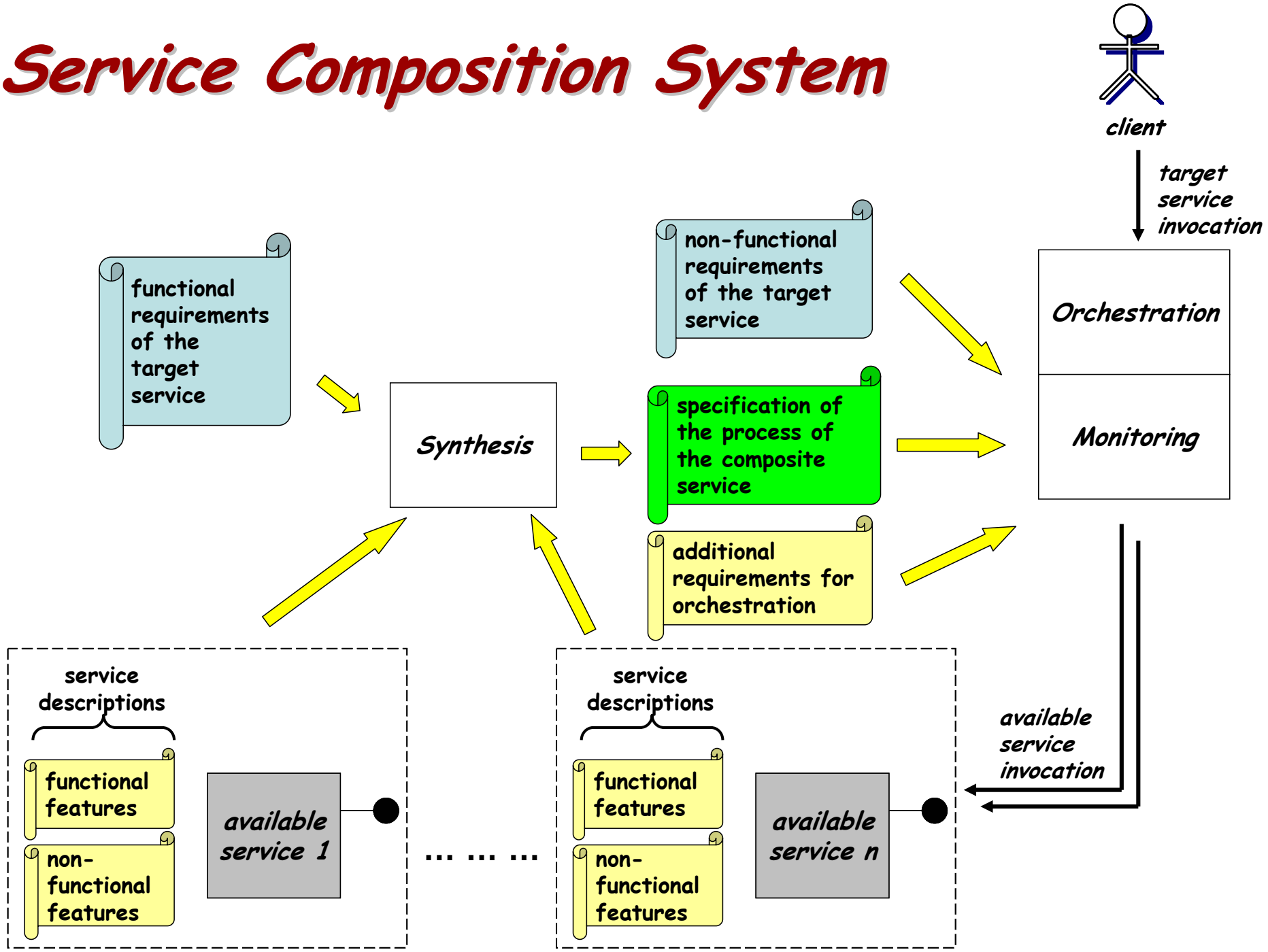
[WS-C] – Web Services Coordination (WS-C), http://www-106.ibm.com/developerworks/library/ws-coor/

[WS-T] - Web Services Transaction (WS-Transaction), http://www-106.ibm.com/developerworks/webservices/library/ws-transpec/

[WS-CAF] – Web Services Composite Application Framework, http://developers.sun.com/techtopics/webservices/wscaf/

State of the Art on Service Composition

Service Composition System



Service composition

1. Composition Synthesis:
   Input:
   - client request
   - set of available services

   Output:
   - specification of composite service
2. Orchestration:
   Input:
   - specification of composite service

   Output:
   - coordination of available services according to the composition schema
   - data flow and control flow monitoring

How to model client request ?

How to model available services ?

How to model the composite service ?

How to orchestrate the composite service ?

Service description

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Services export a view of their behavior
  - I/O interface
    - Data Access
      - focus on data
      - for information gathering
    - Atomic Actions
      - focus on actions
      - world altering services
  - Complex Behavioral Description
    (typically represented using finite states, e.g., TSs)

information oriented services

services as atomic actions

services as processes

The whole picture

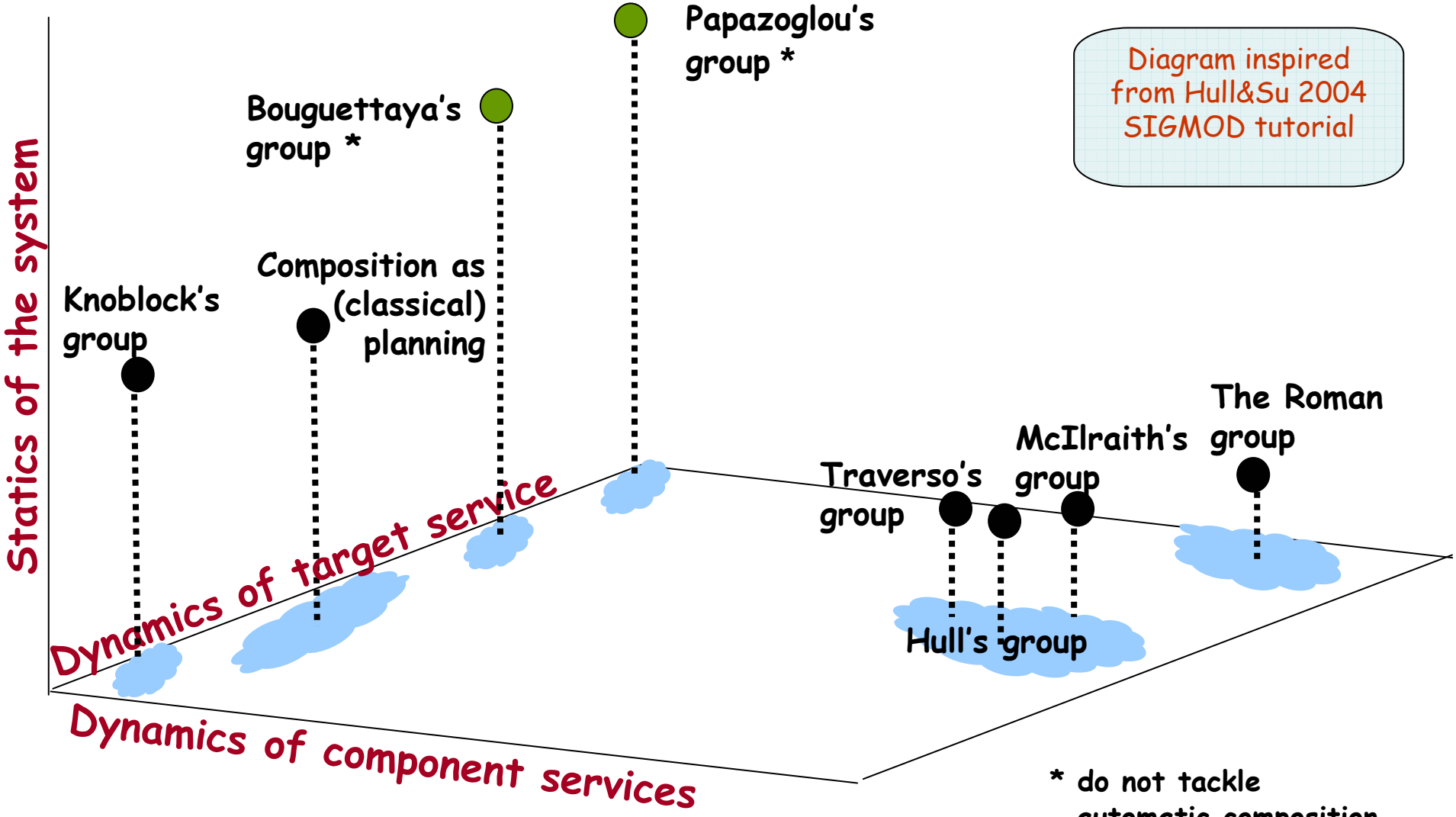


Diagram inspired from Hull&Su 2004 SIGMOD tutorial

* do not tackle automatic composition

Key dimensions in service composition (1)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

1. Statics of the composition system (i.e., static semantics):
   - e.g, ontologies of services (for sharing semantics of data/information), inputs and outputs, etc.

2. Dynamics of component services (i.e., dynamic semantics, process):
   - e.g., behavioral features, complex forms of dataflow, transactional attitudes, adaptability to varying circumstances

Key dimensions in service composition (2)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

3. Dynamics of the target service (i.e., dynamic semantics, process)

The target service exposed as:

atomic action

process

- single step
- (set of) sequencial steps
- (set of) conditional steps
- while/loops, running batch
- while/loops, running under an external control

Key dimensions in service composition: the 4th dimension

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS

Università di Roma "La Sapienza"

For simplicity not shown in the following slides

4. Degree of (in)completeness in the specification of:
   - Static Aspects (of the composition system)
   - Dynamic Aspects (of component services)
   - Target service specification

- Note: Orthogonal to previous dimensions

What is addressed from the technical point of view?

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Automatic composition techniques?
  - Which formal tools?
  - Sound and complete techniques?
  - Techniques/Problem investigated from computational point of view?

Analyzed works

- Papazoglou's group
- Bouguettaya's group

(not automatic composition)

- Knoblock's group (information oriented services)
- Composition as Planning (services as atomic actions)
- Traverso's group
- McIlraith's group
- Hull's group
- The Roman group

(services as processes)

as called by Rick Hull in his SIGMOD 2004 tutorial

Papazoglou's group

J. Yang and M.P. Papazoglou: Service Components for Managing the Life-cycle of Service Compositions, Information Systems 29 (2004), no. 2, 97 – 125

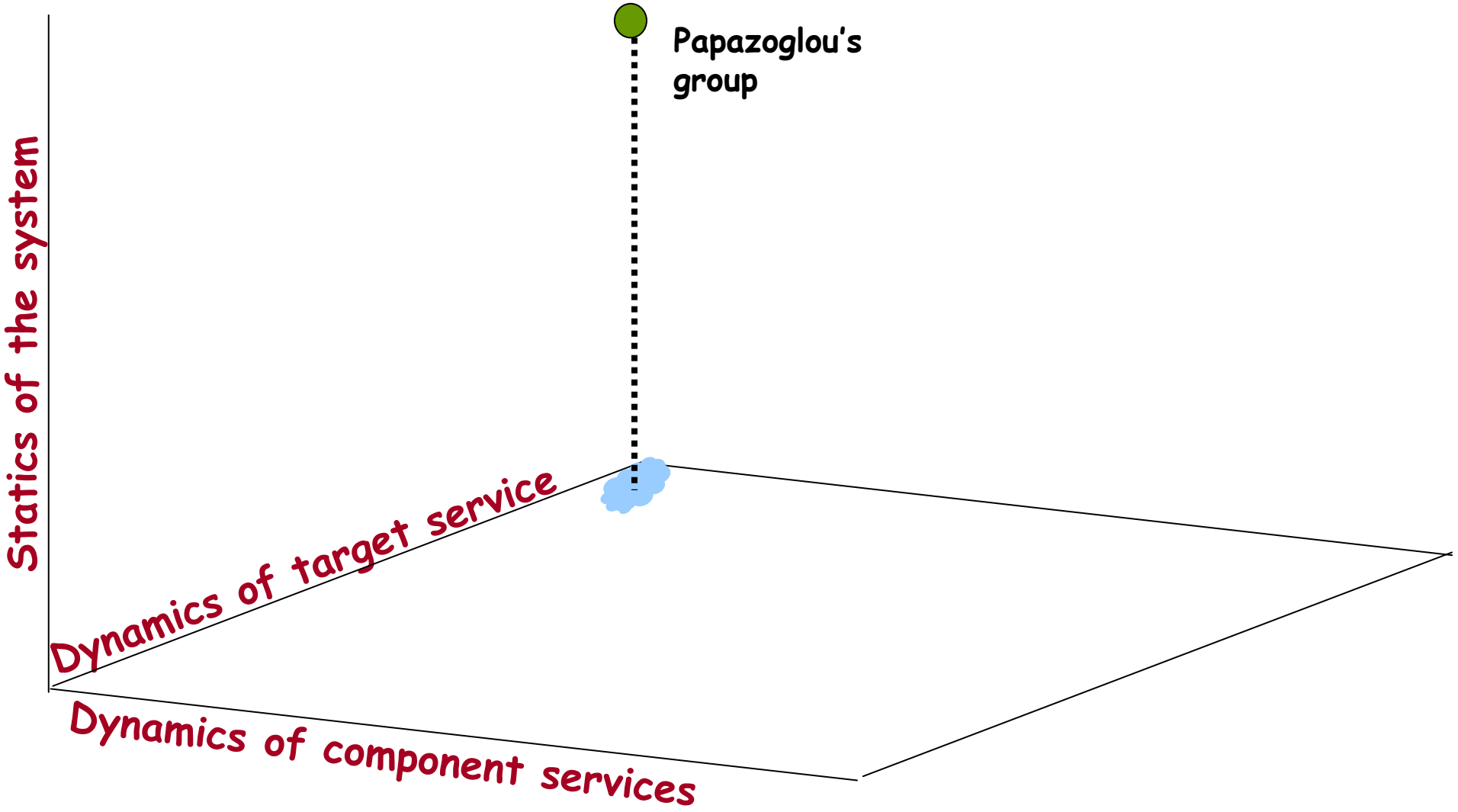
Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS

Università di Roma "La Sapienza"

- available services: I/O interfaces
  - service component: simple or complex pre-existing service wrapped into a web component
  - they are stored in a service component class library
  - operations offered through a uniform interface
- composite service: complex behavioral description
  - set of service components (from service component class library) "glued" together by composition logics
    - composition logics defines execution order (either sequential or concurrent) of service components within composition, dependencies among input and output parameters, etc.
  - support for manual composition: designer specifies composite service using the Service Scheduling Language and the Service Composition Execution Language

Papazoglou's group

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"



Bouguettaya's group

B. Medjahed, A. Bouguettaya, and A. K. Elmagarmid: Composing Web services on the Semantic Web, Very Large Data Base Journal 12 (2003), no. 4, 333-351

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"

Università di Roma "La Sapienza"

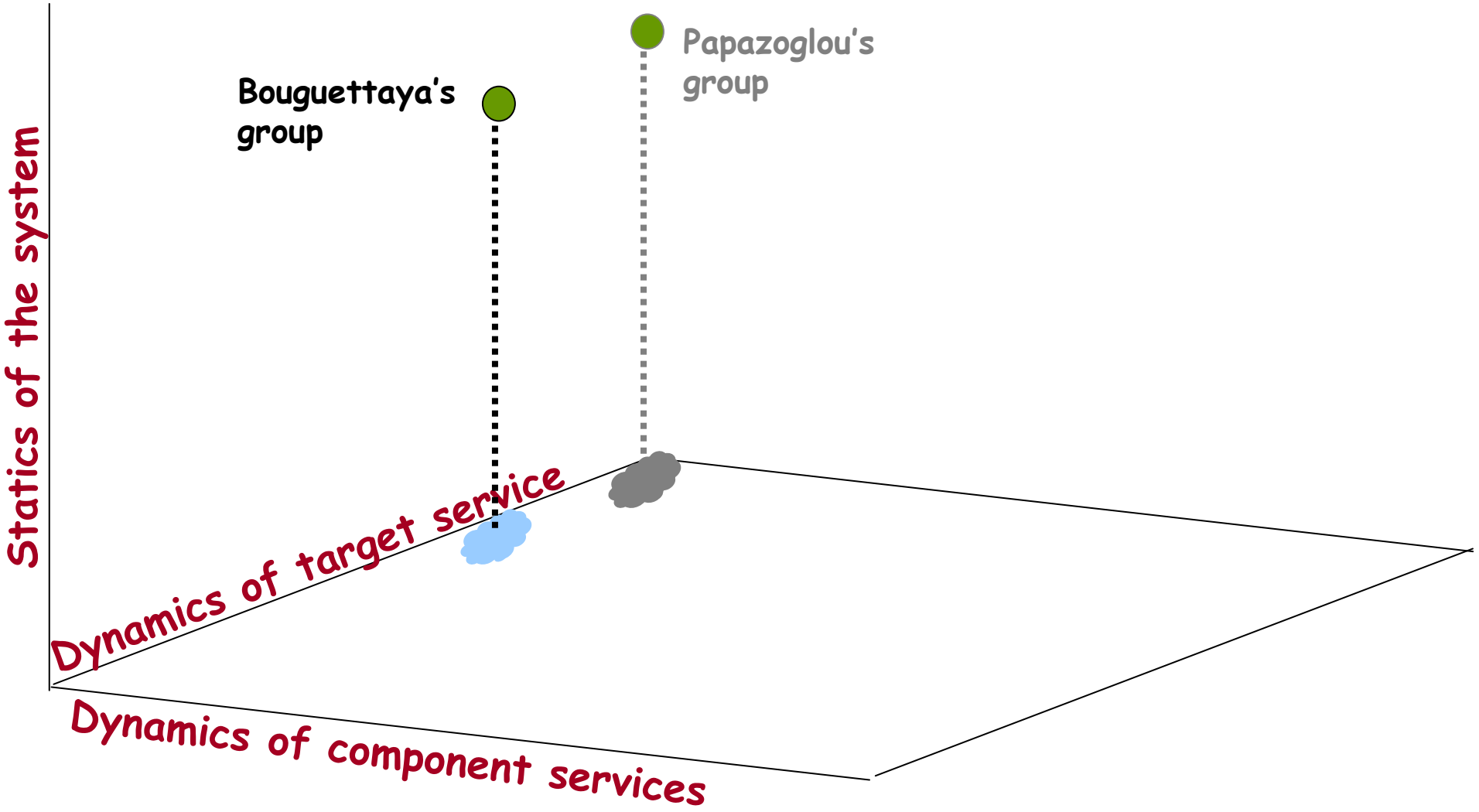
- available services: atomic actions
  - semantically described in terms of their I/O interfaces and non-functional properties such as their purpose, their category and their quality
  - Available services stored into an ontology on the basis of their non-functional properties

Bouguettaya's group

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- client request:
  - expressed in the Composite Service Specification Language (CSSL): it specifies the sequence of desired operations that the composite service should perform and control flow between operations
- service composition problem:
  - **Input**: *(i)* I/O descr. of available services
    (ii) client request expr. in CSSL
  - **Output**: composite service as sequence of operations (semi-automatically) obtained from the client specification by identifying, for each operation, the operation(s) of available services that matches it, on the basis of their I/O interface and non-functional features

Bouguettaya's group



Knoblock's group

M. Michalowski, J.L. Ambite, S. Thakkar, R. Tuchinda, C.A. Knoblock, and S. Minton: Retrieving and semantically integrating heterogeneous data from the web. IEEE Intelligent Systems, 19 (2004), no. 3, pp.72 – 79

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS

Università di Roma "La Sapienza"

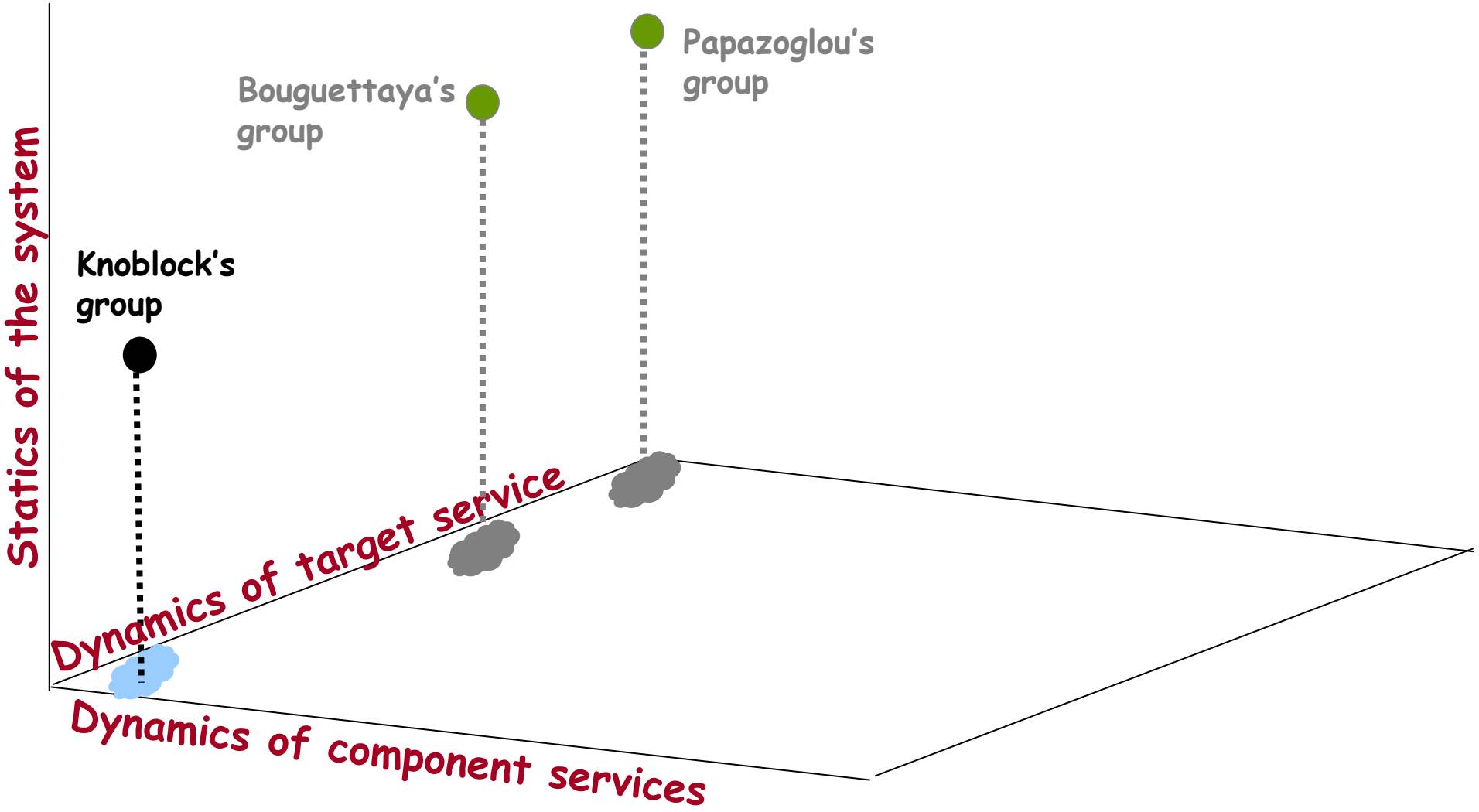
- available service: data query
  - basic idea: informative services as views over data sources
  - each service described in terms of I/O parameters (of course, the latter being provided by the data source), binding patterns and additional constraints on the source
- client request:
  - data query, expressed in terms of inputs provided by the client and requested outputs

Knoblock's group

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- service composition problem:
  - **Input**: *(i)* available services modeled as data-sources, and *(ii)* client request as user query
  - **Output**: (automatically obtained) composite service as integration plan for a *generalized* user query, so that all the user queries that differ only for intensional input values can be answered by the same (composite) service. Integration plan as a sequence of source queries, taking binding pattern into account

Knoblock's group



Composition as Planning

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- available services: atomic actions
- client request: client (propositional) goal
- service composition problem: planning problem
  - **Input**: *(i)* client goal (also encodes initial condition)
    (ii) available services as atomic actions
  - **Output**: composite service as a (possibly conditional) plan, i.e., sequence of actions that transform the initial state into a state satisfying the goal.
    - *Sirin, Parsia, Wu, Hendler & Nau [Sirin etal ICWS03]*
    - ICAPS 2003 Planning for Web Services workshop *[P4WS03]*
    - ICAPS 2004 Planning for Web and Grid Services workshop *[P4WGS04]*
- NOTE: the client has not influence over the control flow of the composite service

Example (1)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Component Services
  - S_1: True $\rightarrow$ {S_1:bookFlight} FlightBooked $\wedge$ MayBookLimo
    MayBookLimo $\rightarrow$ {S_1:bookLimo} LimoBooked
  - S_2: True $\rightarrow$ {S_2:bookHotel} HotelBooked
    HotelBooked $\rightarrow$ {S_2:bookShuttle} ShuttleBooked
  - S_3: True $\rightarrow$ {S_3:bookEvent} EventBooked
- Ontology:
  - TravelSettledUp $\equiv$ FlightBooked $\wedge$ HotelBooked $\wedge$ EventBooked
  - CommutingSettled $\equiv$ ShuttleBooked $\vee$ LimoBooked $\vee$ TaxiAvailablilityChecked
  - ...
- Client Service Request:
  - Find a composition of the actions (i.e., a sequence, a program using such actions as basic instructions) such that a given property is fulfilled

Example (2)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Component Services
  - S_1: True $\rightarrow$ {S_1:bookFlight} FlightBooked $\wedge$ MayBookLimo
    MayBookLimo $\rightarrow$ {S_1:bookLimo} LimoBooked
  - S_2: True $\rightarrow$ {S_2:bookHotel} HotelBooked
    HotelBooked $\rightarrow$ {S_2:bookShuttle} ShuttleBooked
  - S_3: True $\rightarrow$ {S_3:bookEvent} EventBooked
- Ontology:
  - TravelSettledUp $\equiv$ FlightBooked $\wedge$ HotelBooked $\wedge$ EventBooked
  - CommutingSettled $\equiv$ ShuttleBooked $\vee$ LimoBooked $\vee$ TaxiAvailablilityChecked
  - ...
- Client Service Request:
  - Starting from: $\neg$FlightBooked $\wedge$ $\neg$ HotelBooked $\wedge$ $\neg$EventBooked $\wedge$ $\neg$CommutingSettled
  - Achieve: TravelSettledUp $\wedge$ CommutingSettled

Example (3)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" — Università di Roma "La Sapienza"

- Component Services
  - S_1: True $\rightarrow$ {S_1:bookFlight} FlightBooked $\wedge$ MayBookLimo
    MayBookLimo $\rightarrow$ {S_1:bookLimo} LimoBooked
  - S_2: True $\rightarrow$ {S_2:bookHotel} HotelBooked
    HotelBooked $\rightarrow$ {S_2:bookShuttle} ShuttleBooked
  - S_3: True $\rightarrow$ {S_3:bookEvent} EventBooked
- Ontology:
  - TravelSettledUp $\equiv$ FlightBooked $\wedge$ HotelBooked $\wedge$ EventBooked
  - CommutingSettled $\equiv$ ShuttleBooked $\vee$ LimoBooked $\vee$ TaxiAvailablilityChecked
  - ...
- Client Service Request:
  Starting from:
  $\neg$FlightBooked $\wedge$ $\neg$ HotelBooked $\wedge$ $\neg$EventBooked $\wedge$ $\neg$CommutingSettled
  achieve:
  TravelSettedUp $\wedge$ CommutingSettled
- Compositions:
  - S_1:bookFlight; S_1:bookLimo; S_2:bookHotel; S_3:bookEvent
  - S_3:bookEvent; S_2:bookHotel; S_1:bookFlight; S_2:bookShuttle

Another Example (1)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Component Services:
  - S_1: Registered $\rightarrow$ {S_1:bookFlight} FlightBooked
    ¬Registered $\rightarrow$ {S_1:register} Registered
  - S_2: True $\rightarrow$ {S_2:bookHotel} HotelBooked
    HotelBooked $\rightarrow$ {S_2:bookShuttle} ShuttleBooked
  - S_3: True $\rightarrow$ {S_3:bookEvent} EventBooked
- Ontology:
  - TravelSettedUp $\equiv$ FlightBooked $\wedge$ HotelBooked $\wedge$ EventBooked
- Client Service Request:
  Starting from:
  ¬FlightBooked $\wedge$ ¬ HotelBooked $\wedge$ ¬EventBooked
  Achieve:
  TravelSettedUp

Another Example (2)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

Client Service Request:

- Starting from: ¬FlightBooked ∧ ¬ HotelBooked ∧ ¬EventBooked
- Achieve: TravelSettedUp

What about Registered?

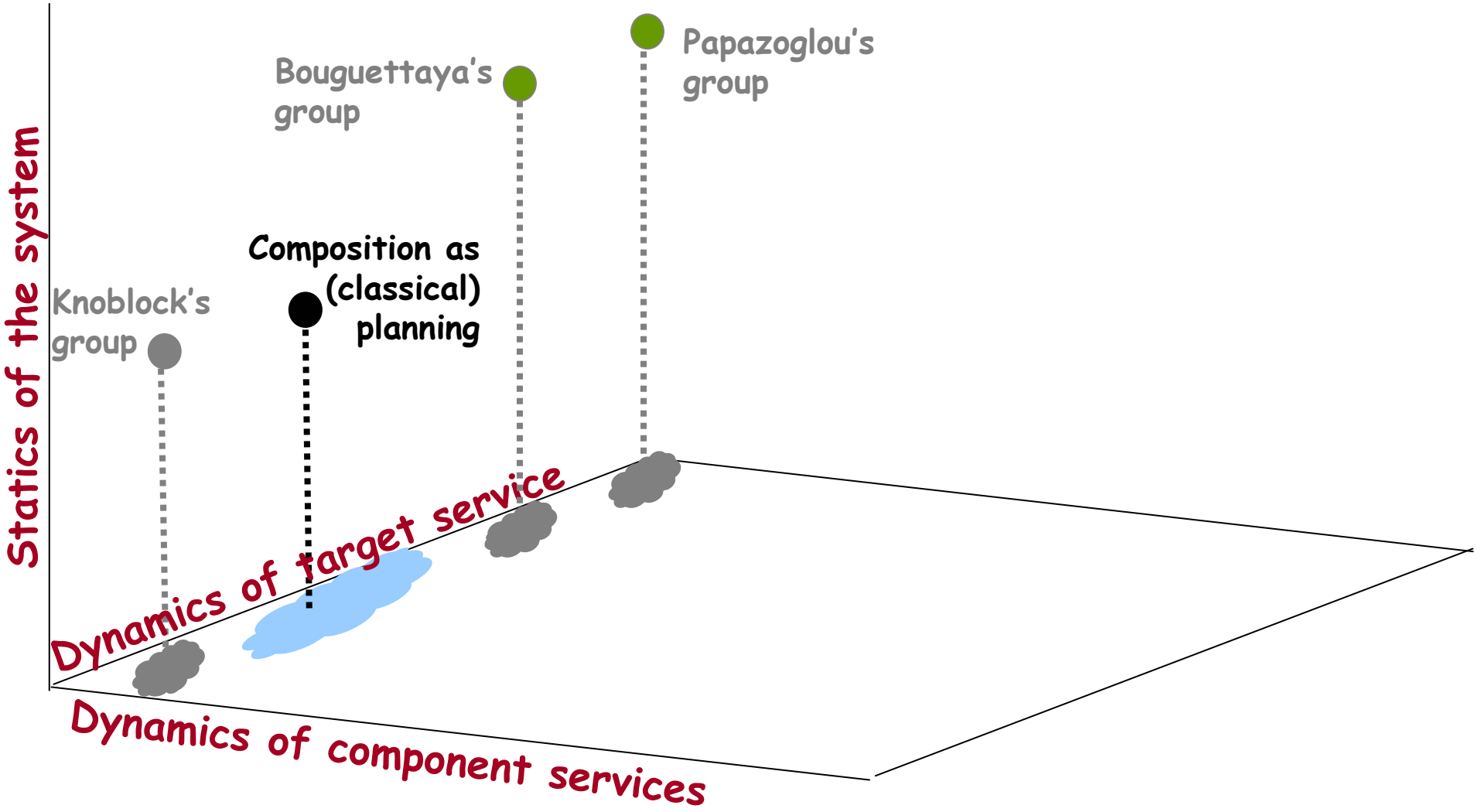
The client does not know whether he/she/it is registered or not.

The composition must resolve this at runtime:

```
if (¬Registered){
  S1:register;
}
S1:bookFlight;
S2:bookHotel;
S3:bookEvent
```

Composition as Planning

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"



Planning is a Rich Area!!!

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Sequential Planning (plans are sequences of actions)
- Conditional Planning (plans are programs with if's and while's)
- Conformant Planning (plans the work in spite of incomplete -non observable- information)
- Knowledge Producing Actions/Sensing (distinction between truth and knowledge)
- Plan Monitoring
- Interleaving Deliberation and Execution
- Form of the Goals:
  - Achieve something
  - Achieve something while keeping something else
  - Temporal goals
  - Main goal + exception handling

References on Planning

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- **Read and exploit planning and reasoning about actions literature!**

  Books

  Chapters on Planning and on Reasoning about Actions in any Artificial Intelligence textbook.

  [GNT04] M. Ghallab, D. Nau, P. Traverso. Automated Planning: Theory and Practice. Morgan Kaufmann, 2004.

  [Reiter02] R.Reiter: Knowledge in Action. MIT Press, 2002.

  Interesting papers

  [Levesque AAAI/IAAI96] H. J. Levesque: What Is Planning in the Presence of Sensing? AAAI/IAAI, Vol. 2 1996: 1139-1146

  [Bacchus&Kabanza AAAI/IAAI96] F. Bacchus, F. Kabanza: Planning for Temporally Extended Goals. AAAI/IAAI, Vol. 2 1996: 1215-1222

  [Giunchiglia&Traverso ECP99] F. Giunchiglia, P. Traverso: Planning as Model Checking. ECP 1999: 1-20

  [Calvanese etal KR02] D. Calvanese, G. De Giacomo, M. Y. Vardi: Reasoning about Actions and Planning in LTL Action Theories. KR 2002: 593-602

  [De Giacomo&Vardi ECP99] G. De Giacomo, M. Y. Vardi: Automata-Theoretic Approach to Planning for Temporally Extended Goals. ECP 1999: 226-238

  [Bylander IJCAI91] Tom Bylander: Complexity Results for Planning. IJCAI 1991: 274-279

- **See how other service-researchers have used it!**
  - Proceedings of P4WGS – ICAPS Workshop 2004
  - Proceedings of P4WS – ICAPS Workshop 2003

Traverso's group

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- available services:
  - non-deterministic transition systems characterized by a set of initial states and by a transition relation that defines how the execution of each action leads from one state to a set of states
  - among such services, one represents the client
- client request (called global goal):
  - it specifies a main execution to follow, plus some side paths that are typically used to resolve exceptional circumstances e.g., **Do Φ else Try Ψ**

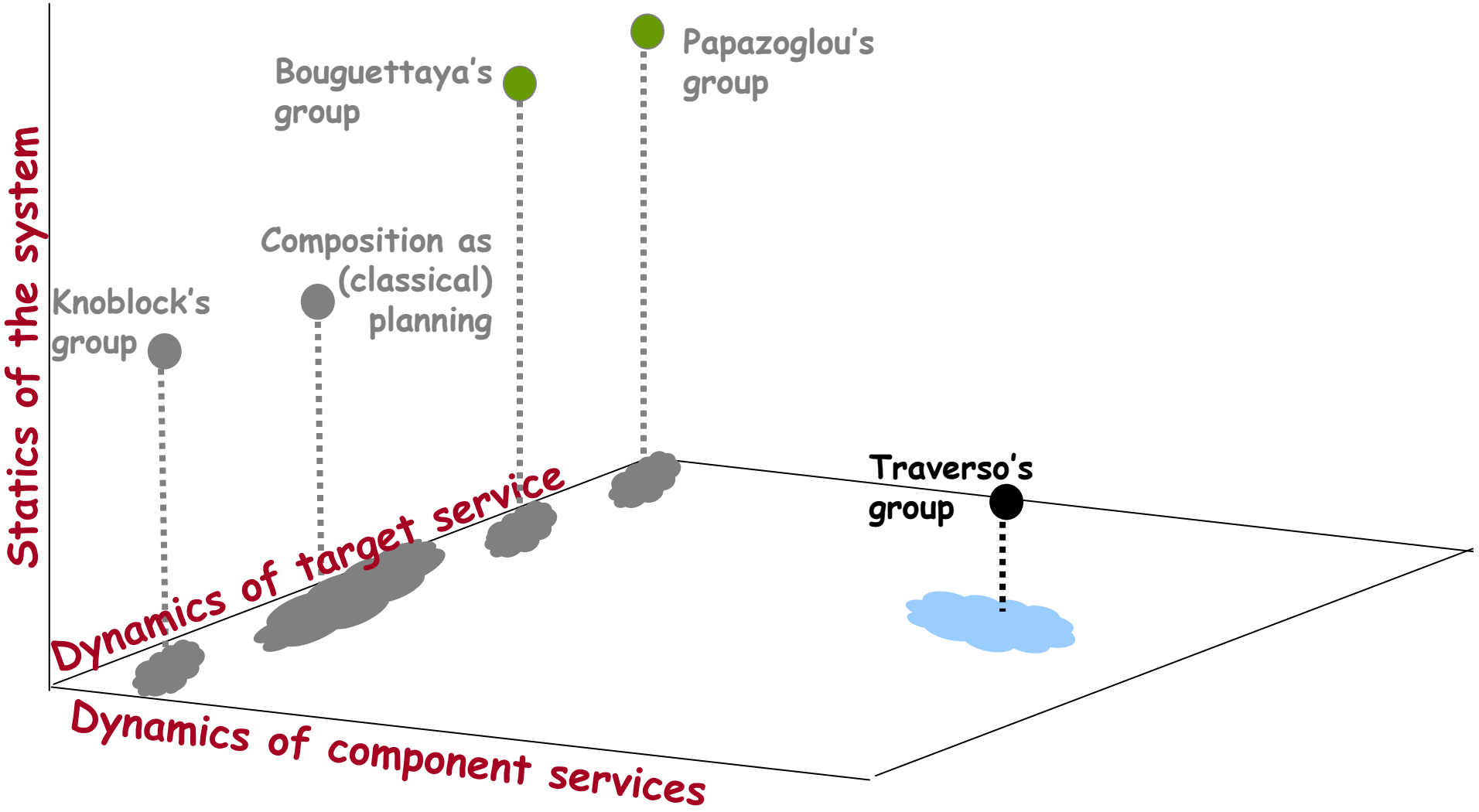
Traverso's group

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- service composition problem: (extended) planning problem
  - Input: *(i)* a set of services, including the one representing the client (behavior), and *(ii)* the global goal,
  - Output: a plan that specifies how to coordinate the execution of various services in order to realize the global goal.

- NOTE:
  - the composition is not tailored towards satisfying completely the client requested behavior, but concerns with the global behavior of the system in which some client desired executions may happen not to be fulfilled

Traverso's group



References on Traverso's group

Papers on Planning as Model Checking

[Giunchiglia&Traverso ECP99]F. Giunchiglia, P. Traverso: Planning as Model Checking. ECP 1999: 1-20

[Pistore&Traverso IJCAI01] M. Pistore, P. Traverso: Planning as Model Checking for Extended Goals in Non-deterministic Domains. IJCAI 2001: 479-486

[Bertoli etal IJCAI01] P. Bertoli, A. Cimatti, M. Roveri, P. Traverso: Planning in Nondeterministic Domains under Partial Observability via Symbolic Model Checking. IJCAI 2001: 473-478

[Dal Lago etal AAAI/IAAI02] U. Dal Lago, M. Pistore, P. Traverso: Planning with a Language for Extended Goals. AAAI/IAAI 2002: 447-454

[Cimatti etal AIJ03] A. Cimatti, M. Pistore, M. Roveri, P. Traverso: Weak, strong, and strong cyclic planning via symbolic model checking. Artif. Intell. 147(1-2): 35-84 (2003)

[Bertoli etal ICAPS03] P. Bertoli, A. Cimatti, M. Pistore, P. Traverso: A Framework for Planning with Extended Goals under Partial Observability. ICAPS 2003: 215-225

Papers on Service Composition

[Pistore&Traverso ISWC04] M. Pistore, P. Traverso: Automated Composition of Semantic Web Services into Executable Processes. ISWC2004.

[Pistore etal P4WGS04] M. Pistore, F. Barbon, P. Bertoli, D. Shaparau, P. Traverso: Planning and Monitoring Web Service Composition. P4WGS – ICAPS WS 2004

[Pistore etal AIMSA04] M. Pistore, F. Barbon, P. Bertoli, D. Shaparau, P. Traverso: Planning and Monitoring Web Service Composition. AIMSA 2004: 106-115

McIlraith's group

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- both available and composite service: behavioral description seen as procedures invokable by clients
  - Golog procedure, atomically executed, i.e., seen by its client as an atomic Situation Calculus action, presenting an I/O interface
  - each service stored in an OWL-S ontology

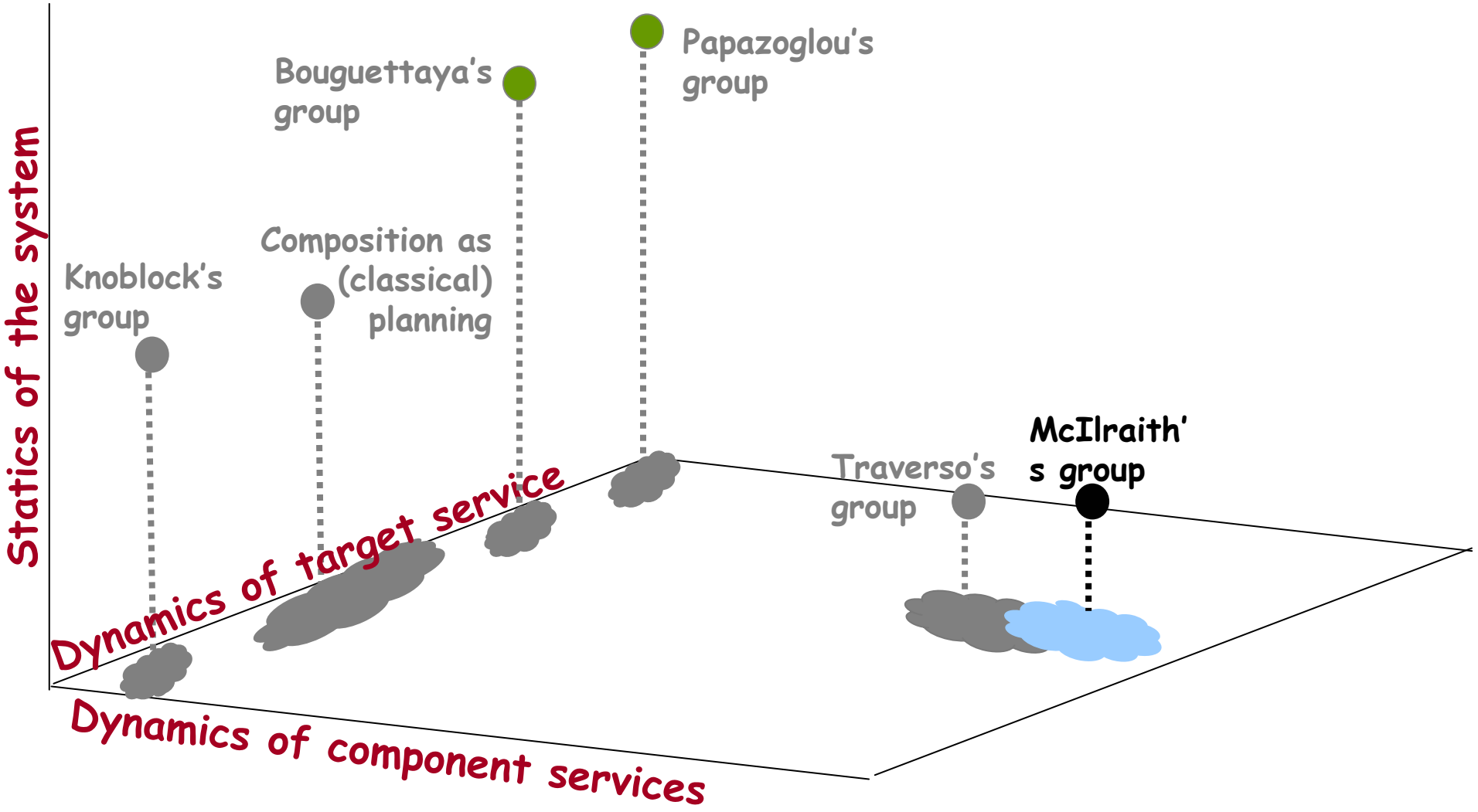
McIlraith's group

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- client request:
  - skeleton of a Golog procedure expressing also client constraints and preferences
- service composition problem:
  - **Input**: *(i)* OWL-S ontology of services as atomic actions, and *(ii)* client request
  - **Output**: Golog procedure obtained by automatically instantiating the client request with services contained in the ontology, by also taking client preferences and constraints into account
- NOTE: the client has not influence over the control flow of the composite service

McIlraith's group

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"



References on McIlraith's group

Background

[McCarthy IFIP62] J. L. McCarthy: Towards a Mathematical Science of Computation. IFIP Congress 1962: 21-28

[McCarthy&Hayes MI69] J. L. McCarthy and P. C. Hayes: Some Philosophical Problems from the Standpoint of Artificial Intelligence. Machine Intelligence 4, 1969

[Reiter 2002] R. Reiter: Knowledge in Action. MIT Press, 2002.

[Levesque etal JLP2000] H. J. Levesque, R. Reiter, Y. Lespérance, F. Lin, R. B. Scherl: GOLOG: A Logic Programming Language for Dynamic Domains. J. Log. Program. 31(1-3): 59-83 (1997)

[De Giacomo etal AIJ2000] G. De Giacomo, Y. Lespérance, H. J. Levesque: ConGolog, a concurrent programming language based on the situation calculus. Artif. Intell. 121(1-2): 109-169 (2000)

[De Giacomo etal KR02] G. De Giacomo, Y. Lespérance, H. J. Levesque, S. Sardiña: On the Semantics of Deliberation in IndiGolog: From Theory to Implementation. KR 2002: 603-614

[Scherl&Levesque AIJ03] R. B. Scherl, H. J. Levesque: Knowledge, action, and the frame problem. Artif. Intell. 144(1-2): 1-39 (2003)

Papers

[McIlraith etal IEEE01] S. A. McIlraith, T. Cao Son, H. Zeng: Semantic Web Services. IEEE Intelligent Systems 16(2): 46-53 (2001)

[Narayanan&McIlraith WWW02] S. Narayanan, S. A. McIlraith: Simulation, verification and automated composition of web services. WWW 2002:

[McIlraith&Son KR02] S. A. McIlraith, T. Cao Son: Adapting Golog for Composition of Semantic Web Services. KR 2002: 482-496

[Burstein etal ISWC02] M. H. Burstein, J. R. Hobbs, O. Lassila, D. Martin, D. V. McDermott, S. A. McIlraith, S. Narayanan, M. Paolucci, T. R. Payne, K. P. Sycara: DAML-S: Web Service Description for the Semantic Web. International Semantic Web Conference 2002: 348-363

[Narayanan&McIlraith CN03] Srini Narayanan, Sheila A. McIlraith: Analysis and simulation of Web services. Computer Networks 42(5): 675-693 (2003)

[McIlraith&Martin IEEE03] S. A. McIlraith, D. L. Martin: Bringing Semantics to Web Services. IEEE Intelligent Systems 18(1): 90-93 (2003)

Hull's group

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

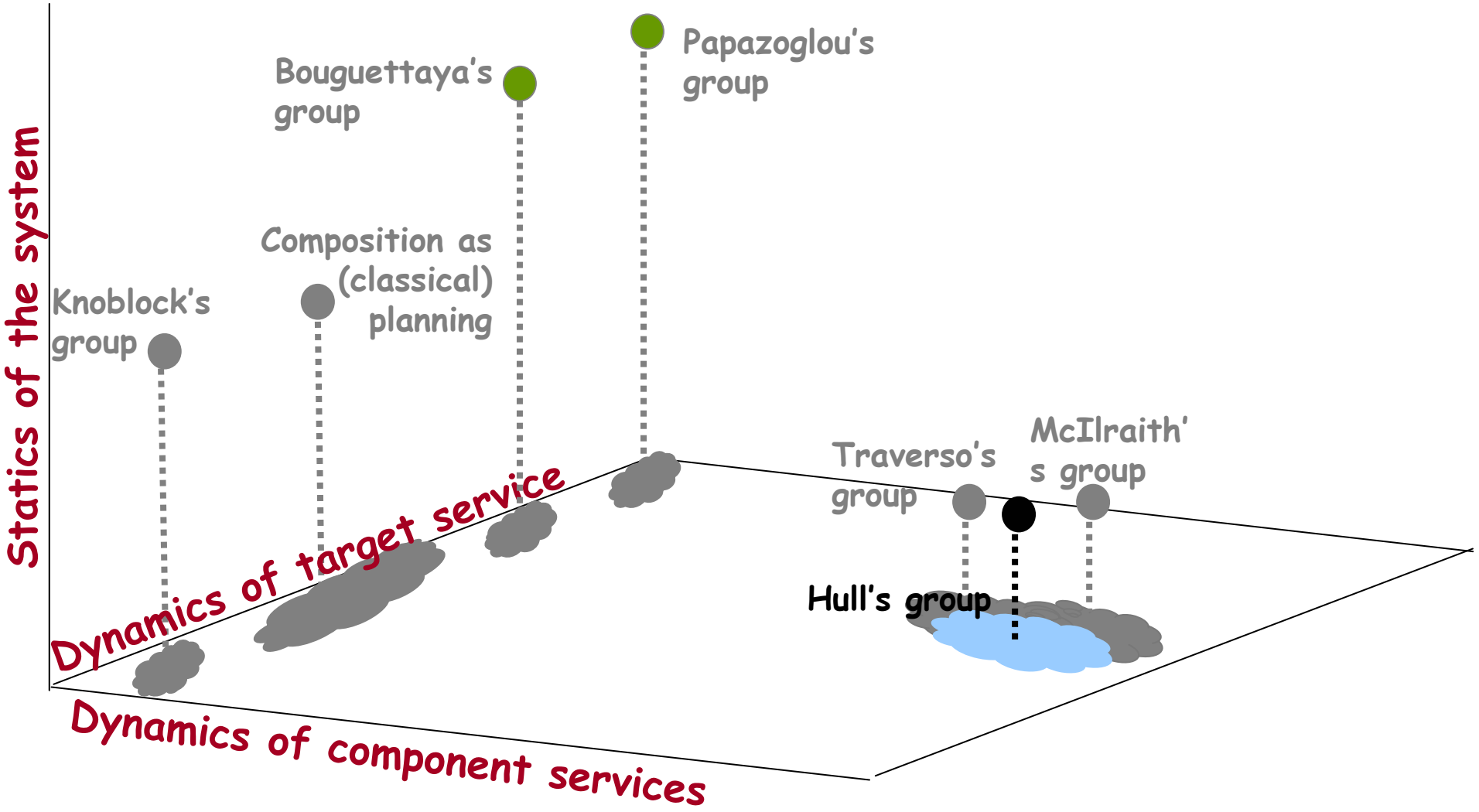
- both available and composite service (peer): behavioral description
  - Mealy machine, that exchanges messages with other peers according to a predefined communication topology (channels among peers)
  - peers equipped with (bounded) queue to store messages received but not yet processed
  - Conversation: sequence of messages exchanged by peers
  - At each step, a peer can either *(i)* send a message, or *(ii)* receive a message, or *(iii)* consume a message from the queue, or *(iv)* perform an empty move, by just changing state

Hull's group

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Choreography mapping problem:
  - **Input**: *(i)* a desired global behavior (i.e., set of desired conversations) as a Linear Temporal Logic formula, and *(ii)* an infrastructure (a set of channels, a set of peer names and a set of messages)
  - **Output**: Mealy machines (automatically obtained) for all the peers such that their conversations are compliant with the LTL specification
- NOTE: not yet a "jam session style" choreography

Hull's group


Statics of the system
Dynamics of target service
Dynamics of component services
Knoblock's group
Composition as (classical) planning
Bouguettaya's group
Papazoglou's group
Traverso's group
McIlraith's group
Hull's group

References on Hull's group

[Hull etal PODS03] R. Hull, M. Benedikt, V. Christophides, J. Su: E-services: a look behind the curtain. PODS 2003: 1-14

[Hull etal SIGMOD03] R. Hull, J. Su: Tools for Design of Composite Web Services. SIGMOD Conference 2004: 958-961

[Bultan etal WWW03] T. Bultan, X. Fu, R. Hull, J. Su: Conversation specification: a new approach to design and analysis of e-service composition. WWW 2003: 403-410

The Roman group

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

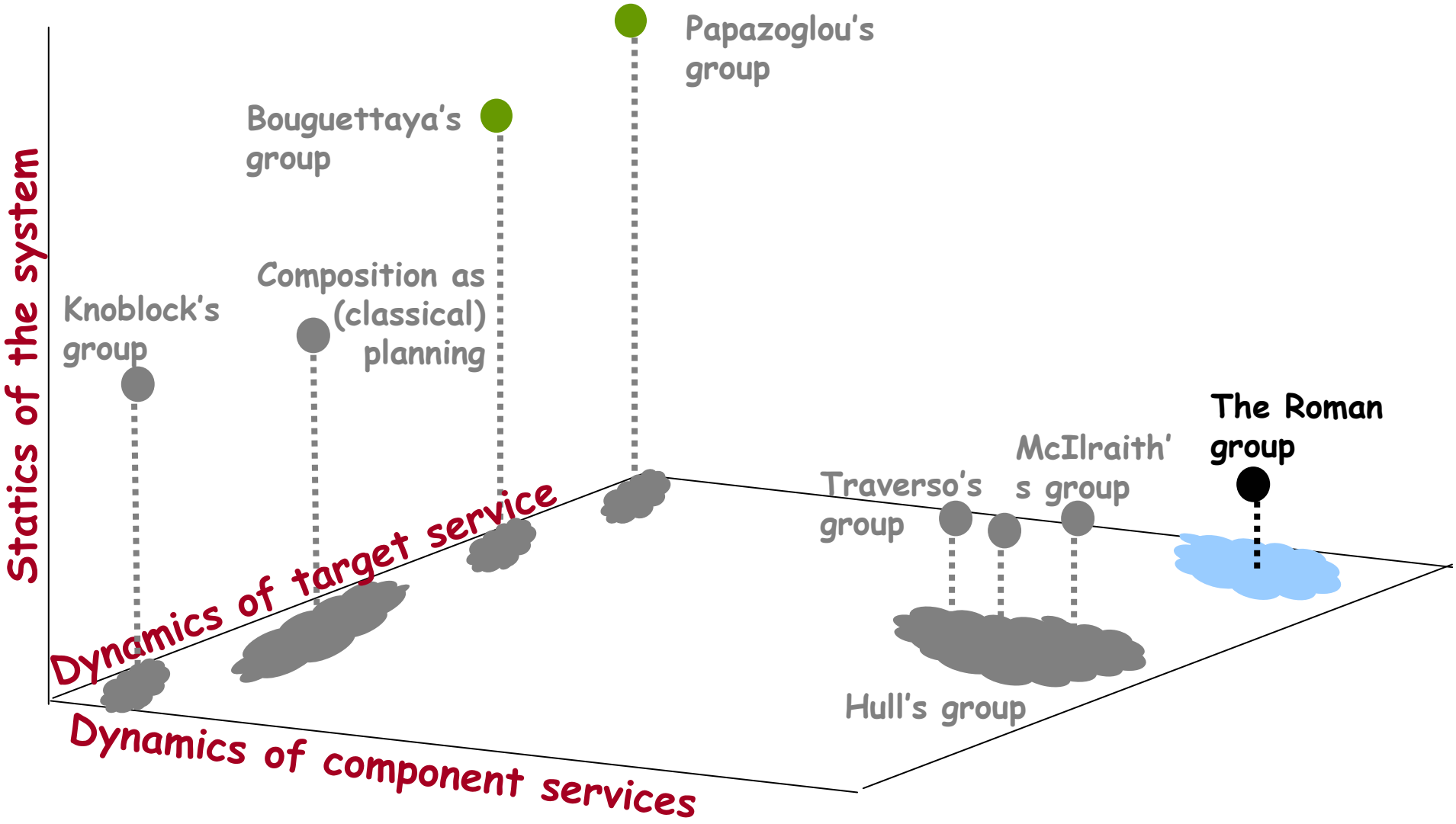
- available service: behavioral description
  - service as an interactive program: at each step it presents the client with a set of actions among which to choose the next one to be executed
  - client choice depends on outcome of previously executed actions, but the rationale behind this choice depends entirely on the client
  - behavior modeled by a finite state transition system, each transition being labeled by a deterministic (atomic) action, seen as the abstraction of the effective input/output messages and operations offered by the service

The Roman group

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- client request (target service):
  - set of executions organized in a (finite state) *transition system* of the activities he is interested in doing
- service composition problem:
  - **Input**: *(i)* finite state transition system of available services, and *(ii)* finite state transition system of target service
  - **Output**: (automatically obtained) composite service that realizes the client request, such that each action of the target service is delegated to at least one available service, in accordance with the behavior of such service.
- NOTE: the client "strongly" influence the composite service control flow

The Roman group



References on the Roman group

[Berardi etal ICSOC03] D. Berardi, D. Calvanese, G. De Giacomo, M. Lenzerini, M. Mecella: Automatic Composition of E-services That Export Their Behavior. ICSOC 2003: 43-58

[Berardi etal ICSOC04] D. Berardi, G. De Giacomo, M. Lenzerini, M. Mecella, D. Calvanese: Synthesis of Underspecified Composite e-Services based on Automated Reasoning. ICSOC 2004

[Berardi etal WES03] D. Berardi, D. Calvanese, G. De Giacomo, M. Lenzerini, M. Mecella: A Foundational Vision of e-Services. WES 2003: 28-40

[Berardi etal P4WS03] D. Berardi, D. Calvanese, G. De Giacomo, and M. Mecella: Composing e-Services by Reasoning about Actions, ICAPS 2003 Workshop on Planning for Web Services (P4WS03).

[Berardi etal DL03] D. Berardi, D. Calvanese, G. De Giacomo, M. Lenzerini, M. Mecella: e-Service Composition by Description Logics Based Reasoning. Description Logics 2003

[Berardi etal P4WGS04] D. Berardi, D. Calvanese, G. De Giacomo, M. Lenzerini, M. Mecella: Synthesis of Composite e-Services based on Automated Reasoning. ICAPS 2004 Workshop on Planning and Scheduling for Web and Grid Services (P4WGS04).

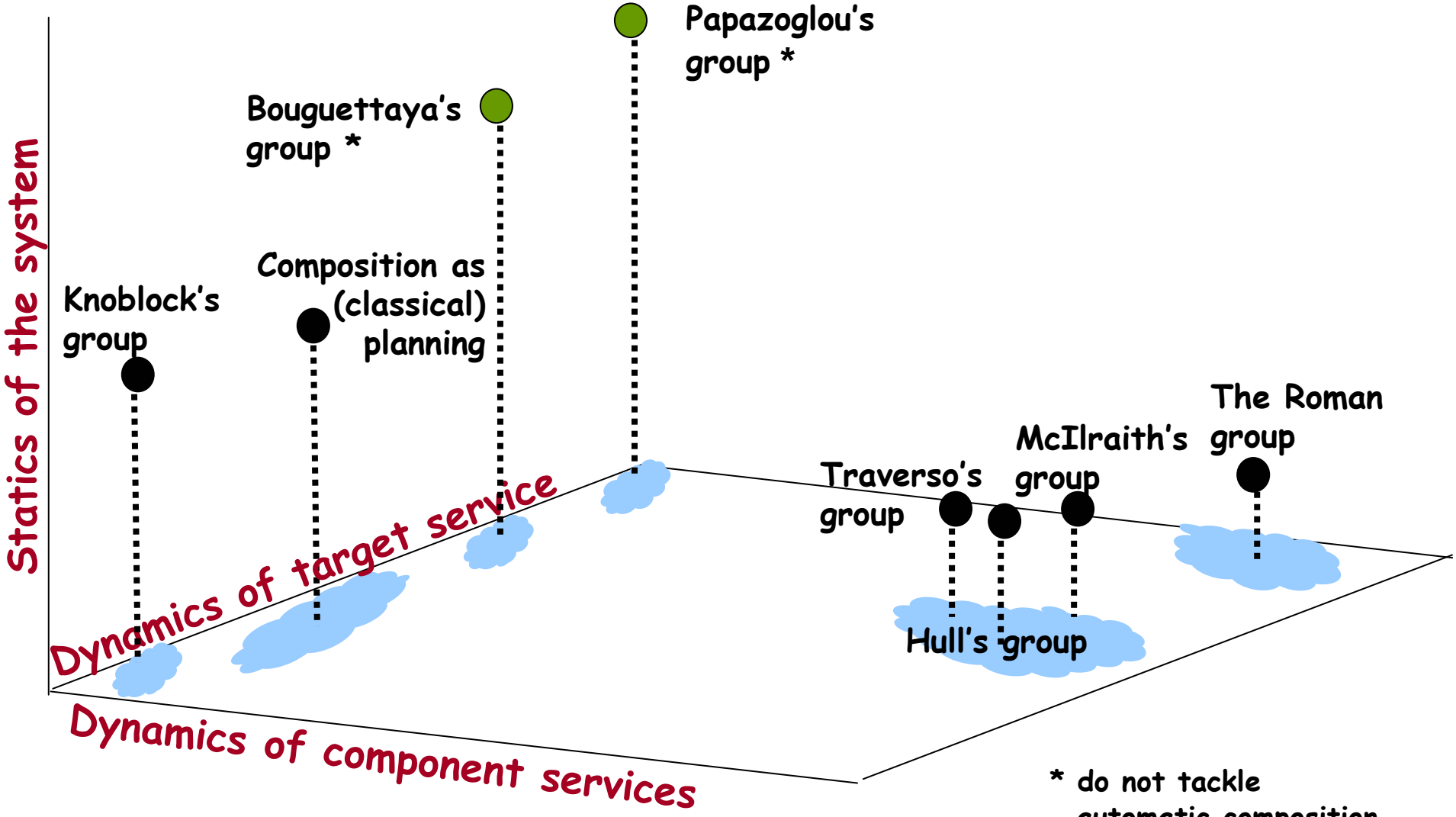
[Berardi etal TES04] D. Berardi, D. Calvanese, G. De Giacomo, M. Lenzerini, M. Mecella: ESC: A Tool for Automatic Composition of e-Services based on Logics of Programs, VLDB-TES 2004

[Berardi Ph.D] D. Berardi Automatic Service Composition.Models, Techniques and Tools. Ph.D. thesis, Dipartimento di Informatica e Sistemistica – Universita' di Roma "La Sapienza", Rome, Italy, 2005.

[IJCIS 2004] D. Berardi, G. De Giacomo, M. Lenzerini, M. Mecella, D. Calvanese: Automatic Service Composition based on Behavioral Description. To appear in IJCIS 2005

[Gerede etal ICSOC04] C. E. Gerede, R. Hull, O. H. Ibarra, J. Su: Automated Composition of E-services: Lookaheads. ICSOC 2004

The whole picture



Other Relevant Works

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Approaches proposing interesting conceptual models for services, not targeted towards composition:
  - Vianu 's group
  - Benatallah & Casati's group

Vianu's group

A. Deutsch, L. Sui, and V. Vianu: Specification and Verification of Data-driven Web Services, In Proceedings of the 23nd ACM SIGACT SIGMOD SIGART Symposium on Principles of Database Systems (PODS 2004), ACM, 2004, pp. 71-82

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS

Università di Roma "La Sapienza"

- available service: data query + behavioral descr.
  - service as a data-driven entity characterized by a database and a tree of web pages
  - At each step, set of input choices presented to client: some generated as queries over the database; specific client data treated as constants. The client chooses one of such inputs, and in response, the service produces as output updates over the service database and/or performs some actions, and makes a transition from a web page to another
- automatic verification of service properties:
  - both over runs (linear setting) and over sets of runs (branching setting)
  - they characterize the complexity of verifying such properties for various classes of services

Benatallah & Casati's group

B. Benatallah, F. Casati, and F. Toumani:
Web services conversation modeling: The Cornerstone for E-Business Automation.
IEEE Internet Computing, 8 (2004), no. 1, pp.46 – 54

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

- available service: behavioral description
  - behavior of a service as finite state transition system in terms of message exchanged with the clients (conversations)
  - transitions labeled by messages, and states labeled with the status of the conversation (e.g., effect of the message exchange leading to it, if clearly defined)
- they study how to automatically generate the skeleton of a BPEL4WS spec. starting from the transition system modeling the service behavior
- they also study properties of service behavior in order for two services to correctly interact

(Only) Orchestration

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Two main kinds of orchestration *[Hull etal PODS03]*:
  - *(i)* the mediated approach, based on a hub-and-spoke topology, in which one service is given the role of process mediator/delegator, and all the interactions pass through such a service, and
  - *(ii)* the peer-to-peer approach, in which there is no centralized control

Mediated Orchestration Engines

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- *e*-Flow *[Casati & Shan, ISO1]*:
  - Platform for specifying, enacting and monitoring composite service
  - Composite *E*-Service (CES) is a service process engine offered as (meta-) service that performs coordination of services, with some process adaption/evolution mechanisms
  - A provider can offer a value added service as coordination of different services: it registers the new service to the CES and let the CES enact its execution
- AZTEC *[Christophides etal TESO1]*:
  - Framework for orchestration of session-oriented, long running telecommunication services is studied. It is based on active flowcharts thus coping with asynchronous events that can happen during active telecom sessions

Mediated Orchestration Engines

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- WISE *[Lazcano etal CSSE2000]*:
  - Orchestration engine that coordinates the execution of distributed applications (virtual processes), and a set of brokers enables the interaction with already existing systems that are to be used as building blocks.
  - Process meta-model based on Petri Nets, with the possibility to add Event-Condition-Action (ECA) rules
- MENTOR-lite *[Shegalov etal VLDBJ01]*:
  - Workow management system based on a XML mediator for coordinating services which are distributed among different organizations and deployed on heterogeneous platforms
  - Process meta-model is based on a specific statechart dialect

Peer-to-Peer Orchestration Engines

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Self-Serv *[Benatallah etal IEEE03]*:
  - Platform for composing services and executing new composed services in a decentralized way, through peer-to-peer interactions
  - Composite service modeled as an activity diagram
  - Its enactment carried out through the coordination of different state coordinators (one for each service involved in the specification and one for the composite service itself)
- PARIDE Orchestrator *[Mecella etal VLDB-TES02]*:
  - A composition schema, modeled as a specific Coloured Petri Net, is orchestrated by a set of organizations, which moves it (as a "token") along the execution
  - Separation between the responsibility of the orchestration and the providing of services (suitable in specific scenarios)
  - Services can be substituted with other compatibles

References

[Casati & Shan, IS01] - F. Casati and M.C. Shan, Dynamic and Adaptive Composition of e-Services, Information Systems 6 (2001), no. 3, 143 – 163.

[Christophides etal TES01] - V. Christophides, R. Hull, G. Karvounarakis, A. Kumar, G. Tong, and M. Xiong. Beyond Discrete e-Services: Composing Session-oriented Services in Telecommunications. In Proc. of VLDB-TES, 2001.

[Lazcano etal CSSE2000] - A. Lazcano, G. Alonso, H. Schuldt, and C. Schuler, The WISE approach to Electronic Commerce, International Journal of Computer Systems Science & Engineering 15 (2000), no. 5

[Shegalov etal VLDBJ01] - G. Shegalov, M. Gillmann, and G. Weikum, XML-enabled Workflow Management for e- Services across Heterogeneous Platforms, Very Large Data Base Journal 10 (2001), no. 1, 91–103.

[Benatallah etal IEEE03] - B. Benatallah, Q. Z. Sheng, and M. Dumas. The Self-Serv Environment for Web Services Composition. IEEE Internet Computing, 7(1):40–48, 2003

[Mecella etal VLDB-TES02] – M. Mecella, F. Parisi Presicce, B. Pernici: Modeling e-Service Orchestration Through Petri Nets. Proc. VLDB-TES 2002, LNCS 2444. An extended version as M. Mecella, B. Pernici: Building Flexible and Cooperative Applications Based on eServices, Technical Report 21-02, DIS Univ. Roma "La Sapienza", 2002

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

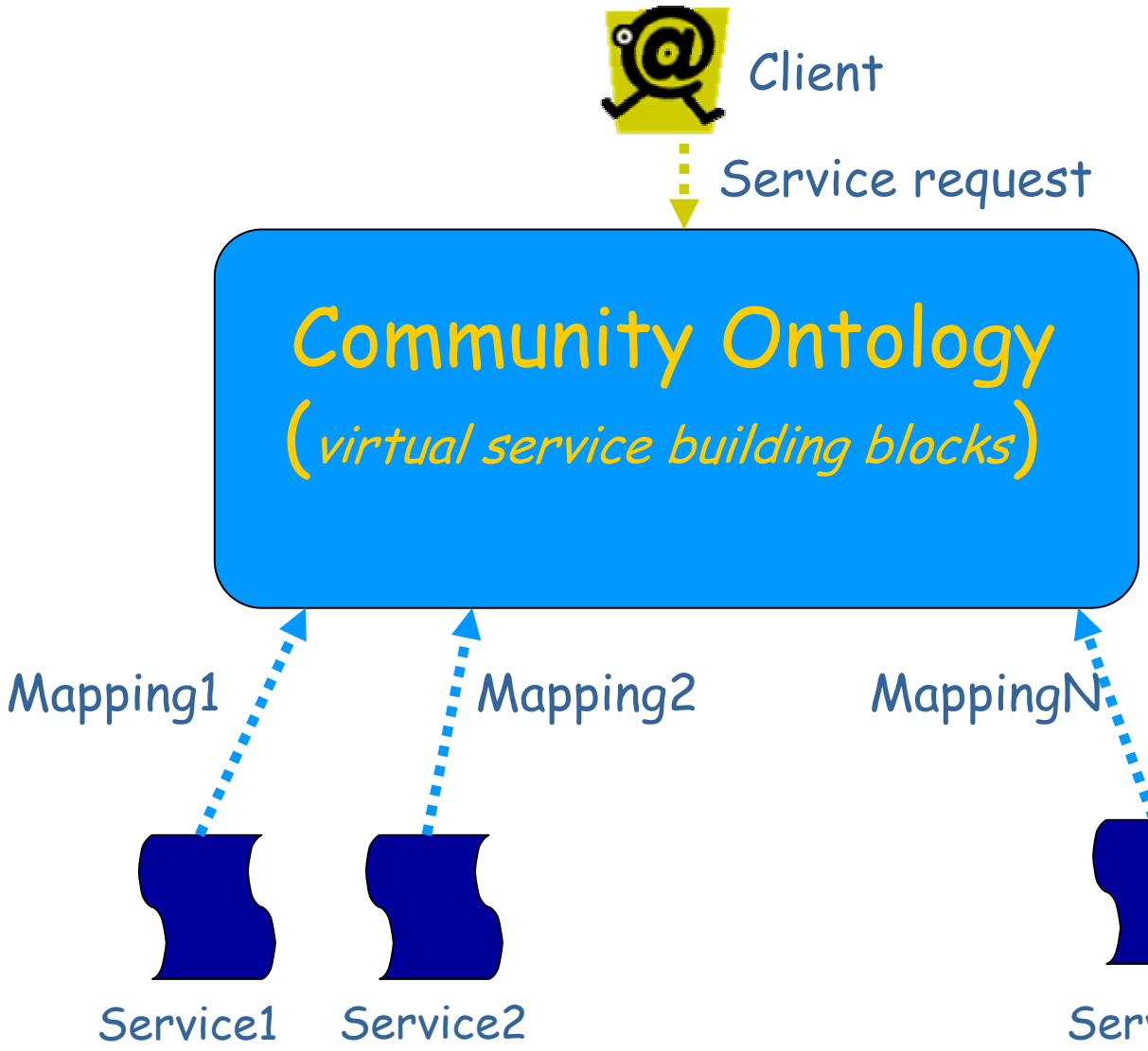
Automatic Composition: A Basic Research Perspective

Basic Research

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Envision of a sort of
  "Service Semantic Integration System"
- Semantic integration via composition synthesis
- Several directions (as we have seen):
  - Information Oriented Services
  - Services as Atomic Actions
  - Services as Processes

Semantic Service Integration



Community Basics

Client makes a **service request** in term of the **community ontology**

Available services express their behavior in terms of the **community ontology**

The **community** realizes the **client service request** making use of the **available services**

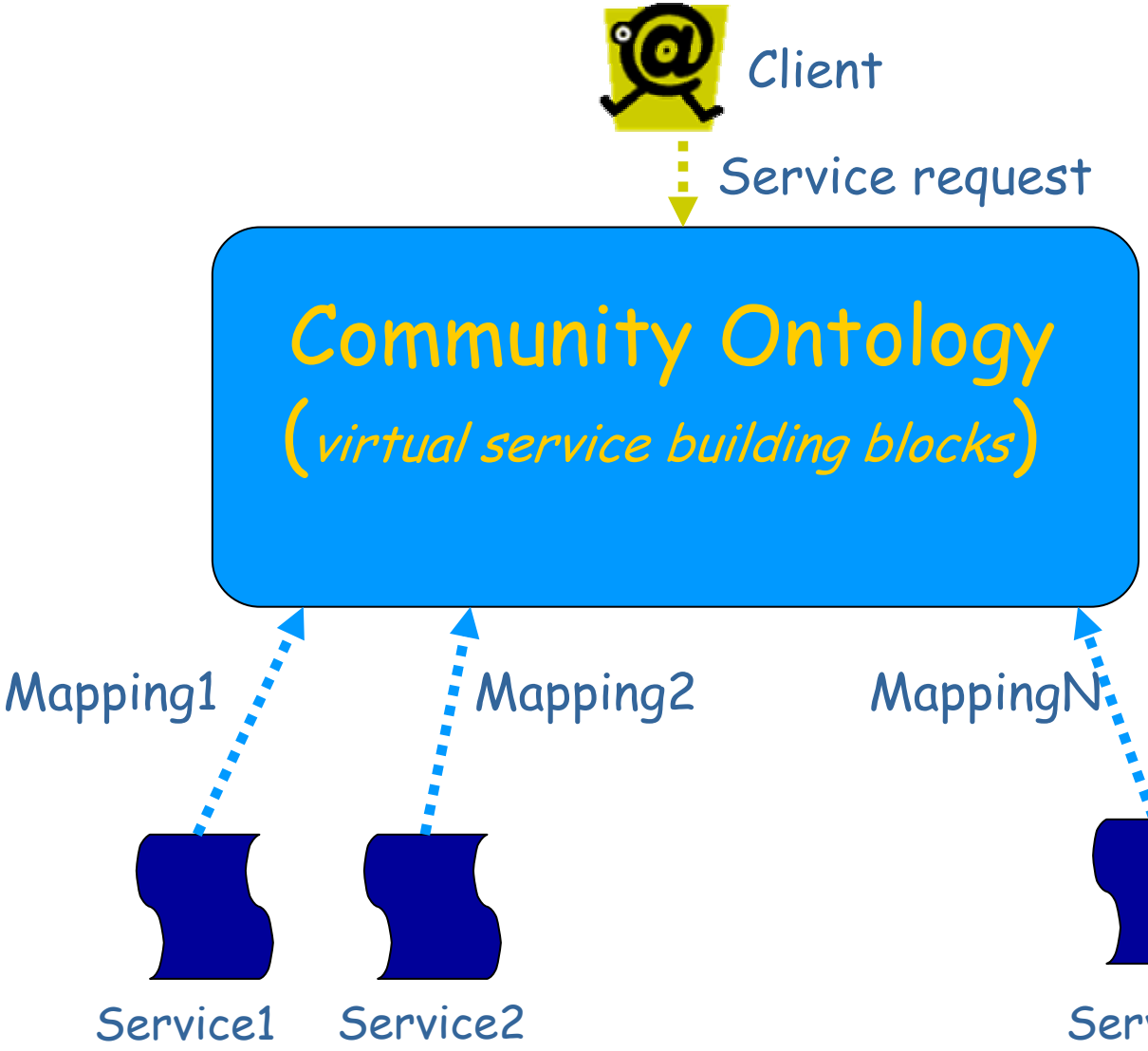
Service Integration Systems

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS

Università di Roma "La Sapienza"

- In building such system we can take two general approach:
  - Service-tailored
  - Client-tailored

Service-Tailored Approach



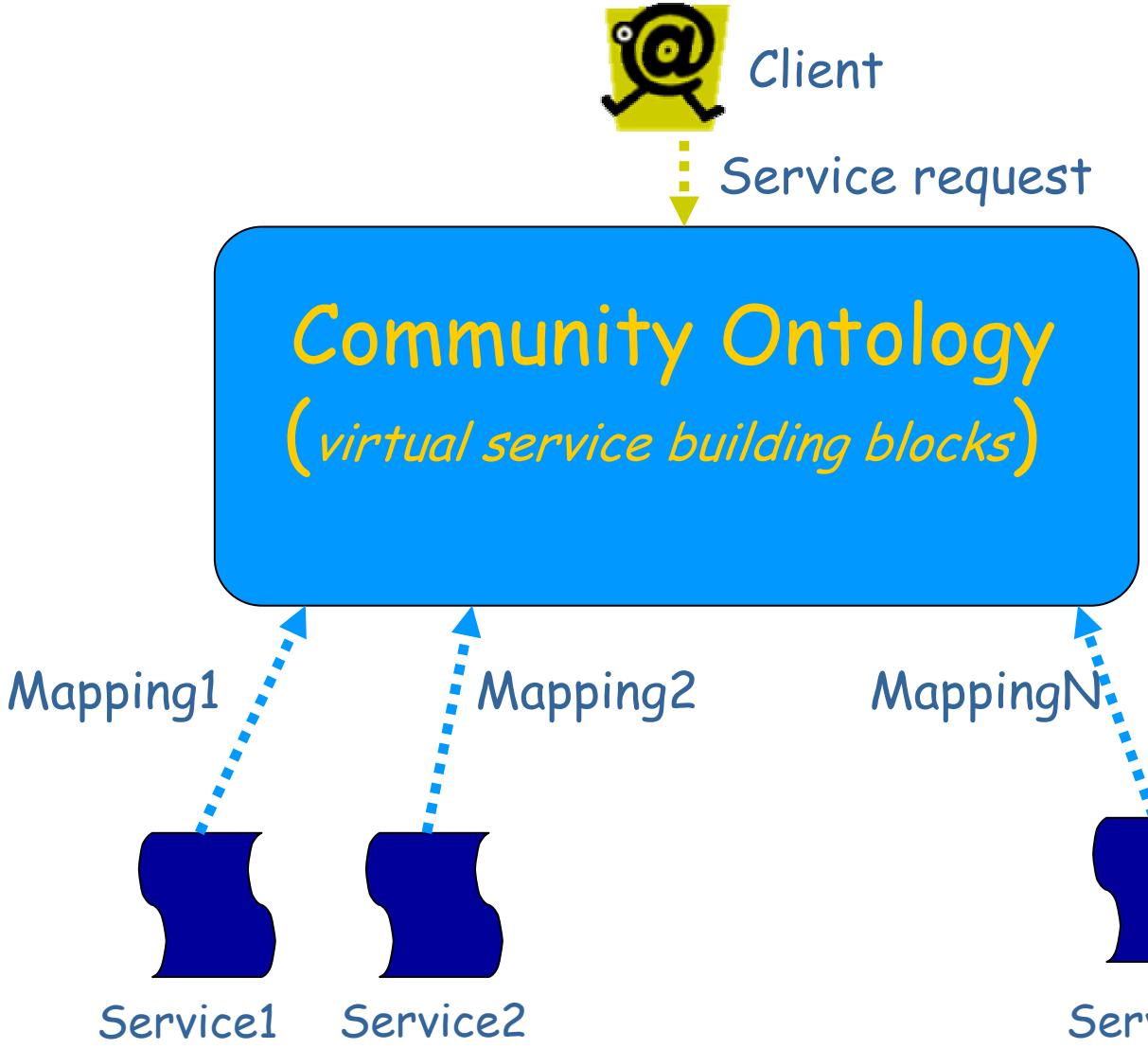
Service-tailored approach

Build the community ontology oriented by suitably reconciling the available services

Map the available services as elements of the community ontology

Compose the service request by directly applying the mappings for accessing concrete computations

Client-Tailored Approach



Client-tailored approach

Build the community ontology oriented to the client, independently from the services available

Describe (map) the available services using the community ontology

Compose the service request by reversing these mappings for accessing concrete computations

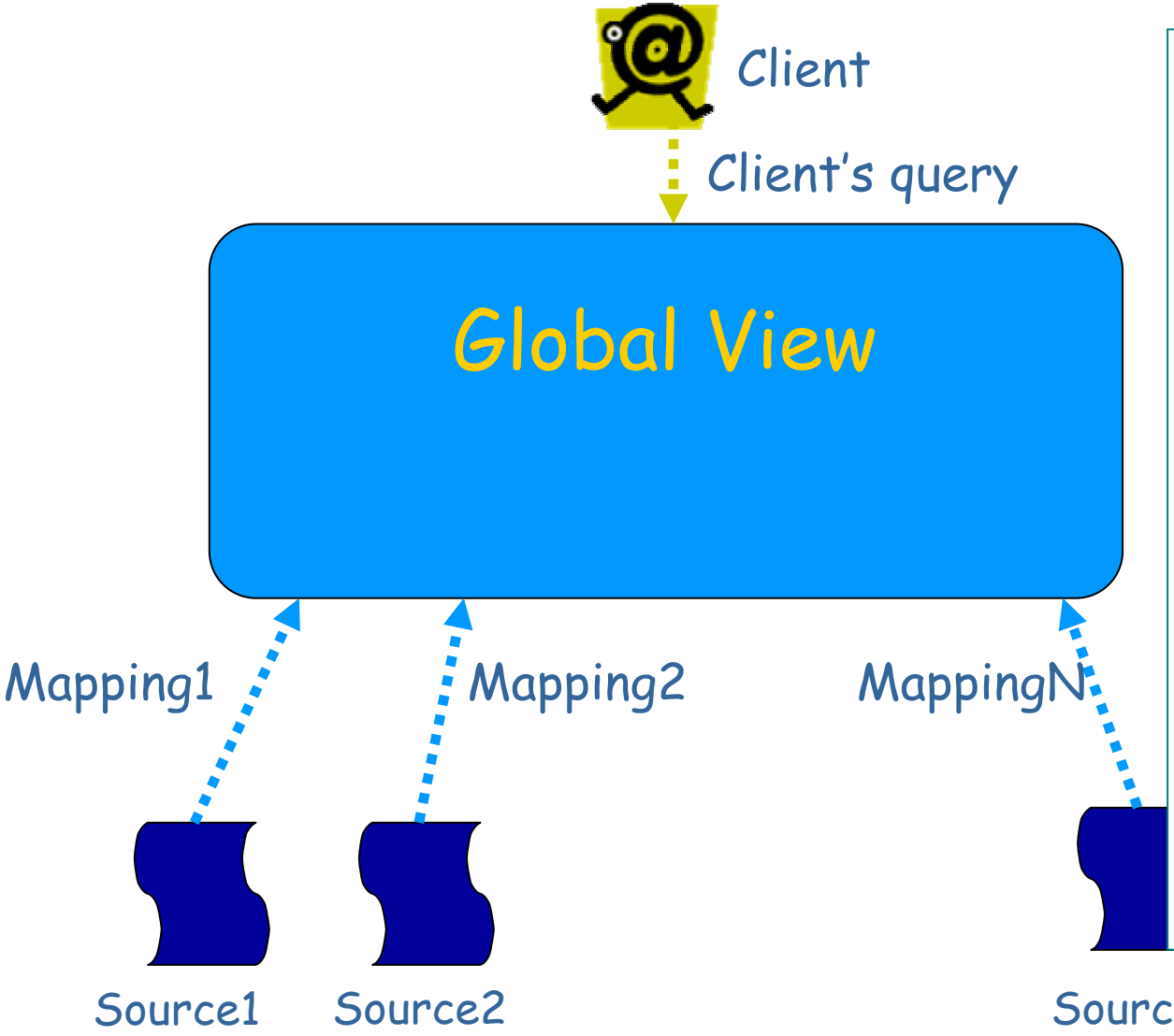
Data Integration

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS

Università di Roma "La Sapienza"

- The Service-tailored vs Client-tailored distinction mimics the GAV (Global As View) vs LAV (Local As View) approach in data integration …

Data Integration System



Integration System Basics

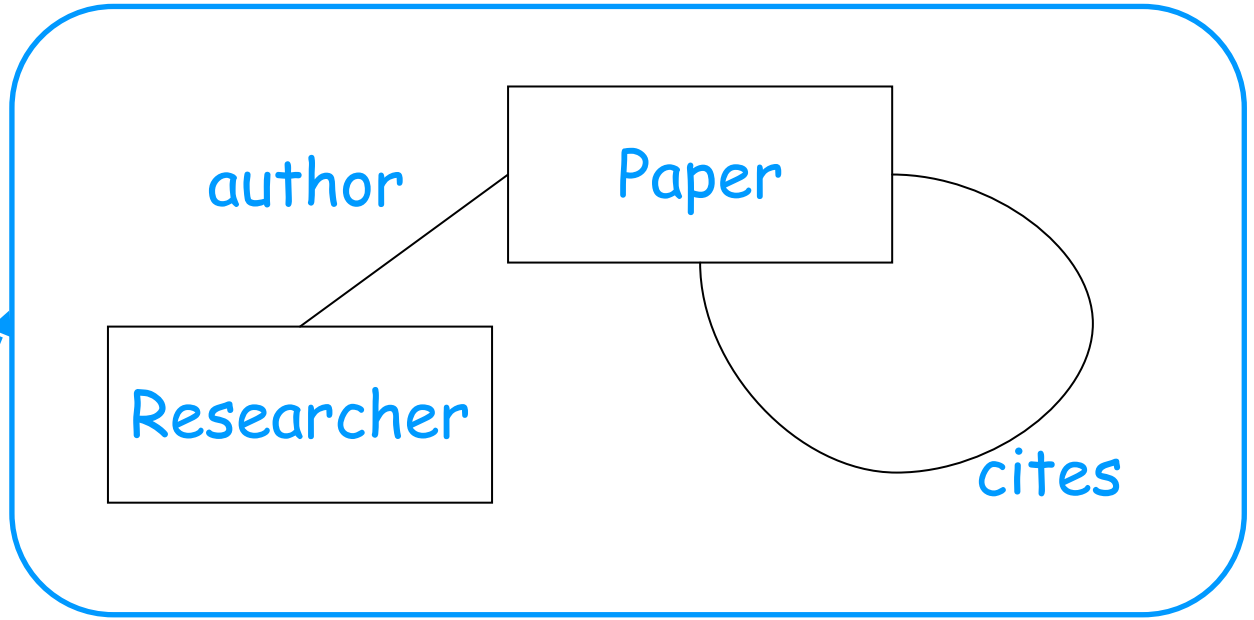
Client's request: query over the **global view**

Available sources express their information in terms of a **query** over the **global view**

The **integration system** answers the **client's query** by **reformulating/rewriting** it in terms of the information in the **available sources**

Example

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"



$\text{Selfcitation}(x) \rightsquigarrow \exists z, y.\ \text{cite}(x,y) \wedge \text{author}(z,x) \wedge \text{author}(z,y)$

Selfcitation

...

Selfcitation: contains papers that cite (other) papers by the same authors

GAV and LAV Mappings in Data Integration

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

- In data integration, we can distinguish two approaches in defining the mapping:
  - GAV (Global As View): terms of the Global View are mapped to queries over the sources
  - LAV (Local As View): sources are described by mapping them to a query over the Global View (cf. previous example)

GAV vs LAV

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- GAV:
  - Adopted in early Data Integration Systems
  - Typical setting
    - Sources are relational DBs
    - Global View is a relational schema
    - Mapping associate relations in the global view with a relational query over the sources
  - Query Answering is performed by
    - "unfolding" (substituting) each relation in the client's query with the corresponding query over the sources (the mapping),
      c.f., computing the composition
    - the evaluating the resulting query
      c.f., executing the composition
  - If constrains are present in the Global View, QA becomes more involved

GAV vs LAV

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- LAV:
  - Recent research on data integration favors the LAV approach over the GAV approach
  - Better support of dynamic changes in the system: sources (services) can be added and deleted without restructuring the global view (community ontology), and hence without impacting the clients
  - Query Answering is a challenge because it needs to deal with incomplete information
  - QA is conceptually performed by
    - first, "rewriting" the client query to an "equivalent" query over the sources,

      c.f., computing the composition
    - then, evaluating the resulting query

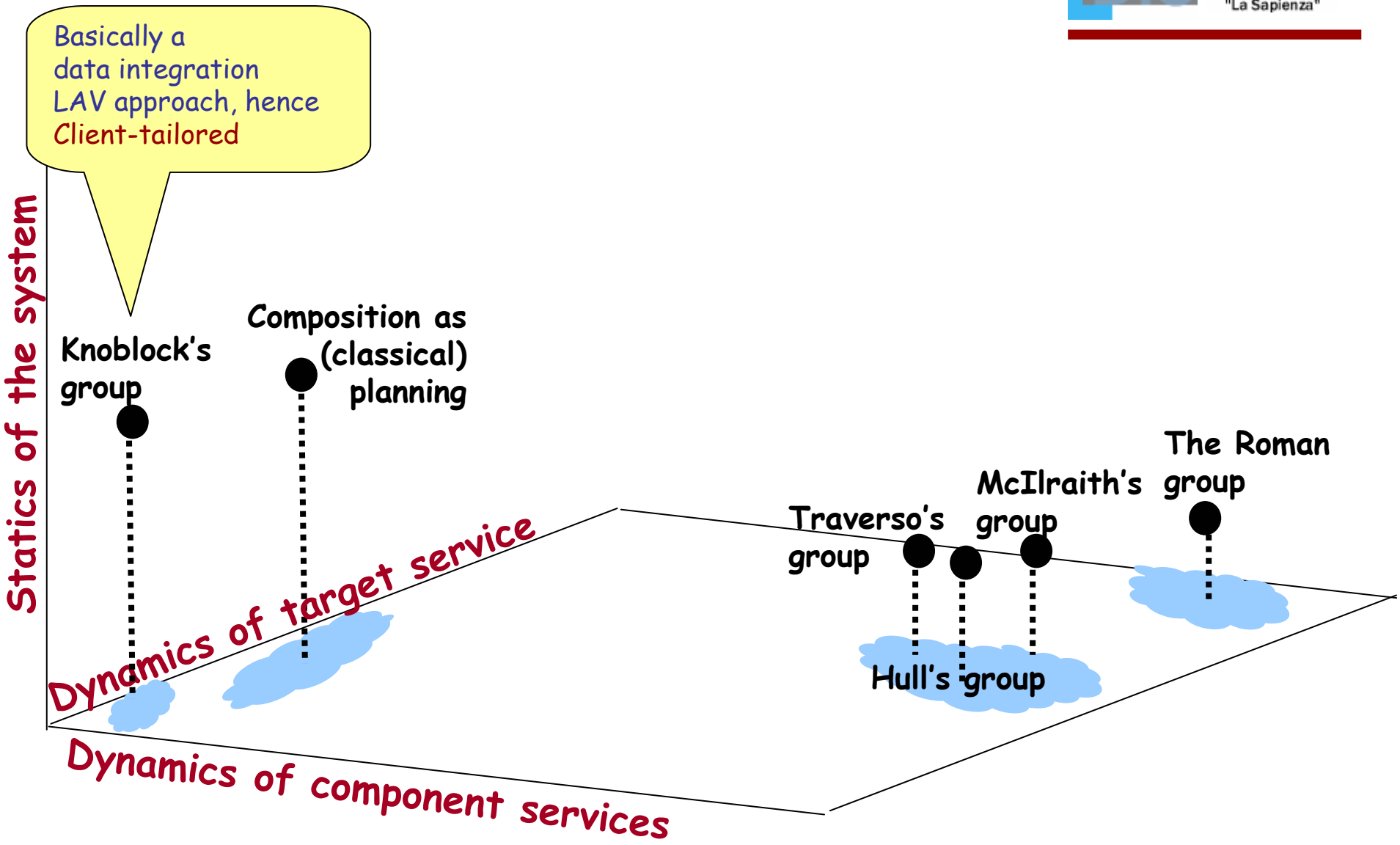
      c.f., executing the composition
  - Often rewriting is not obvious and/or may require query languages that are different from the one used by the client and the mappings

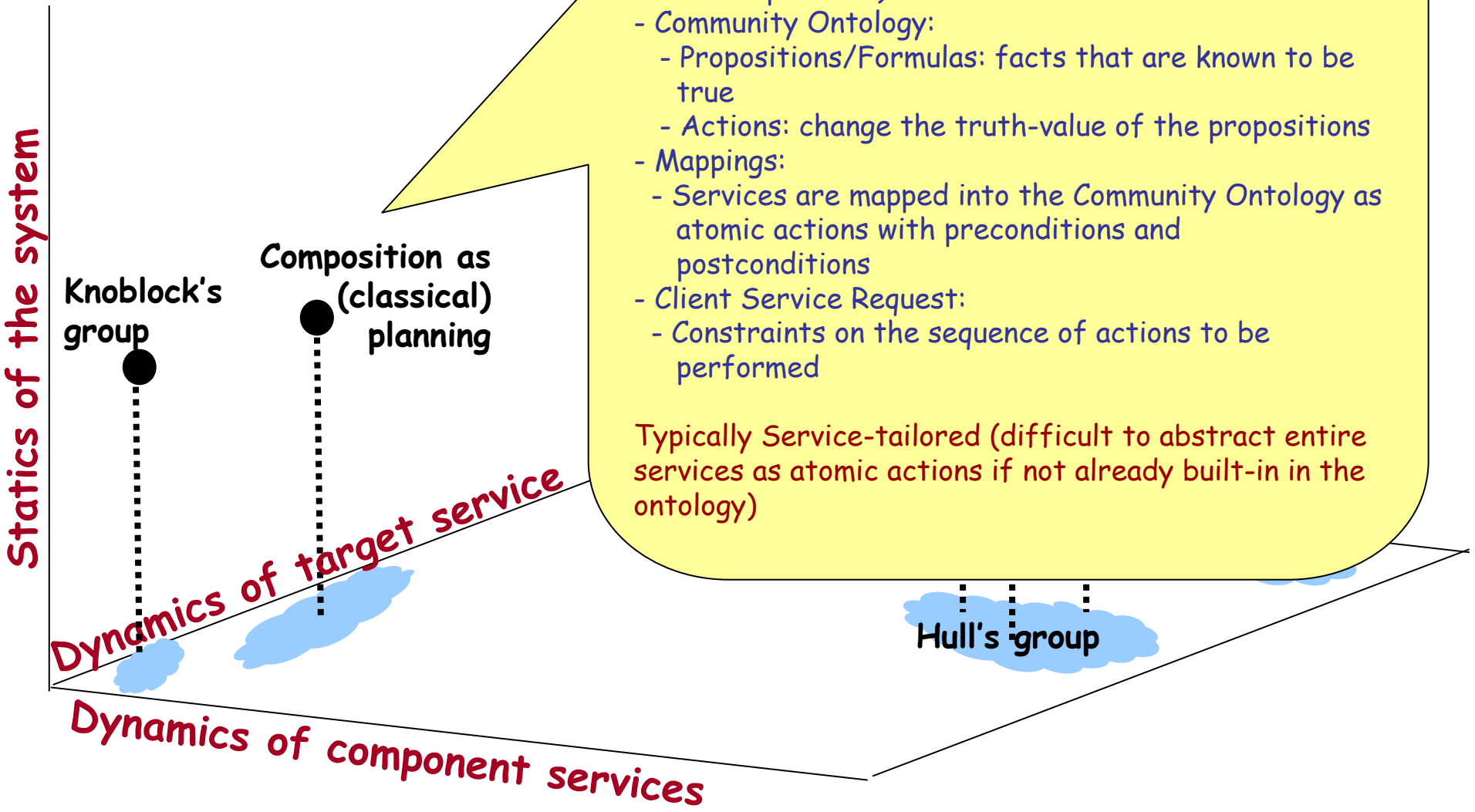
Impact on Service Composition

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Work in data integration as a direct impact on information-based service composition systems
  - Techniques developed there can be often used off-the-shelf or with minor adaptation for information-based services
- More generally data integration research has deeply looked at systems that share many conceptual notions with service composition systems:
  - Insights in data integration systems can be applied to service composition systems
  - Examples:
    - The distinction between GAV and LAV
    - The distinction between query evaluation and query rewriting (execution time vs. composition time)
- However Data Integration has not looked at the procedural aspects typical of services (except for binding patterns)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"



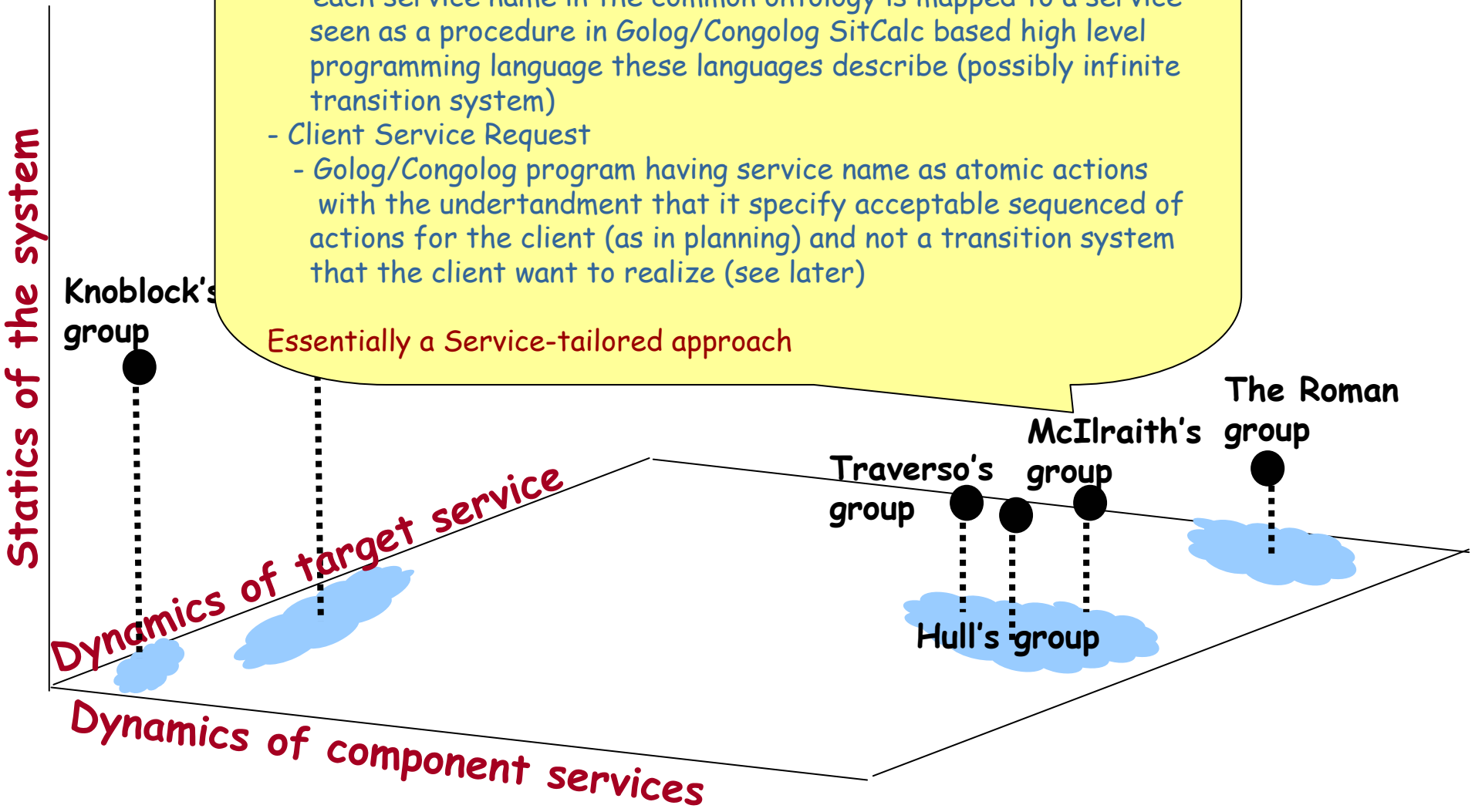


- Services seen as atomic – only I/O behavior modeled (no entry points other than the start and the end of the computation)
- Community Ontology:
  - Propositions/Formulas: facts that are known to be true
  - Actions: change the truth-value of the propositions
- Mappings:
  - Services are mapped into the Community Ontology as atomic actions with preconditions and postconditions
- Client Service Request:
  - Constraints on the sequence of actions to be performed

Typically Service-tailored (difficult to abstract entire services as atomic actions if not already built-in in the ontology)

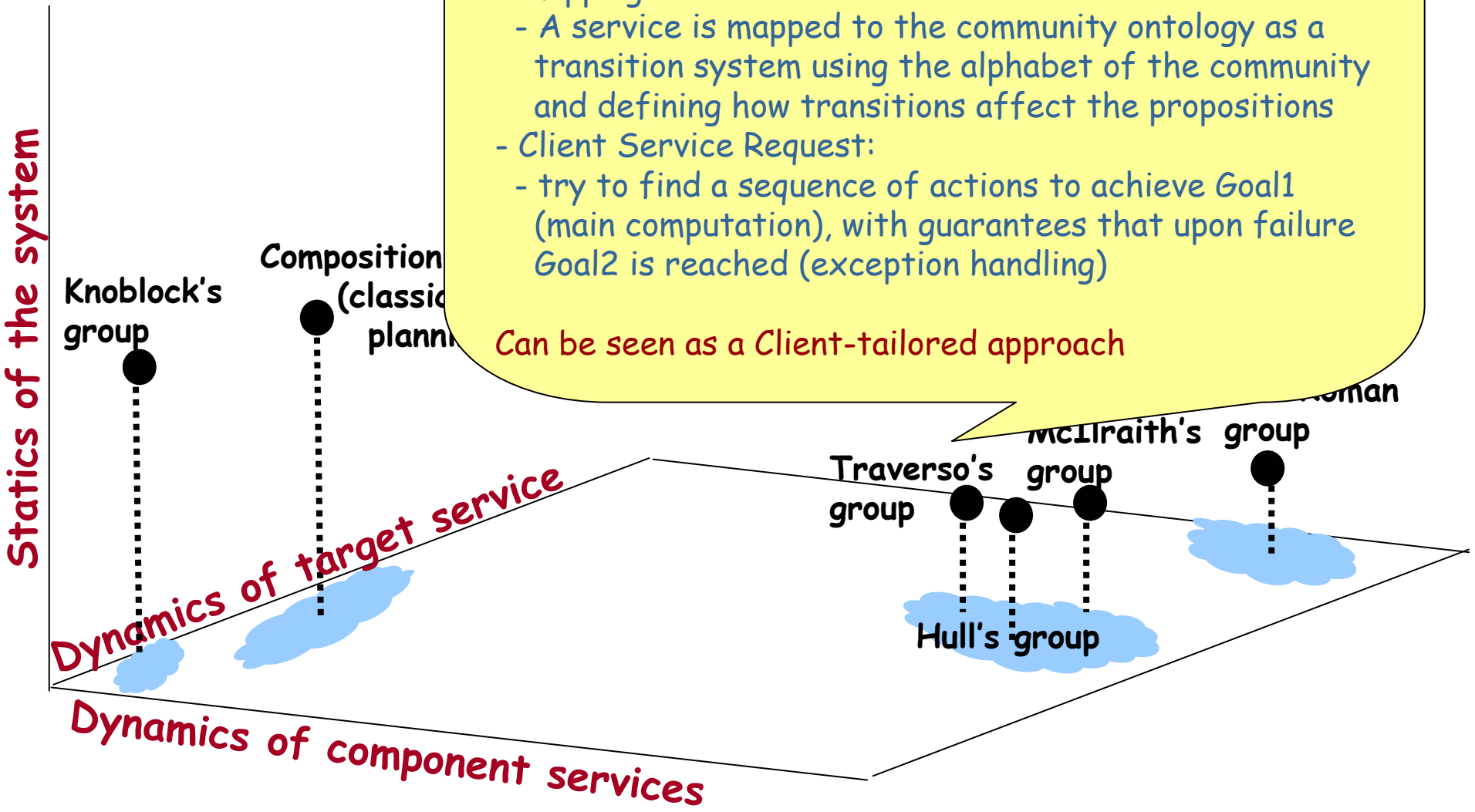
- Services seen as (possibly infinite) transition systems
- Common ontology is a Situation Calculus Theory and service names
- Mapping:
  - each service name in the common ontology is mapped to a service seen as a procedure in Golog/Congolog SitCalc based high level programming language these languages describe (possibly infinite transition system)
- Client Service Request
  - Golog/Congolog program having service name as atomic actions with the undertandment that it specify acceptable sequenced of actions for the client (as in planning) and not a transition system that the client want to realize (see later)

Essentially a Service-tailored approach

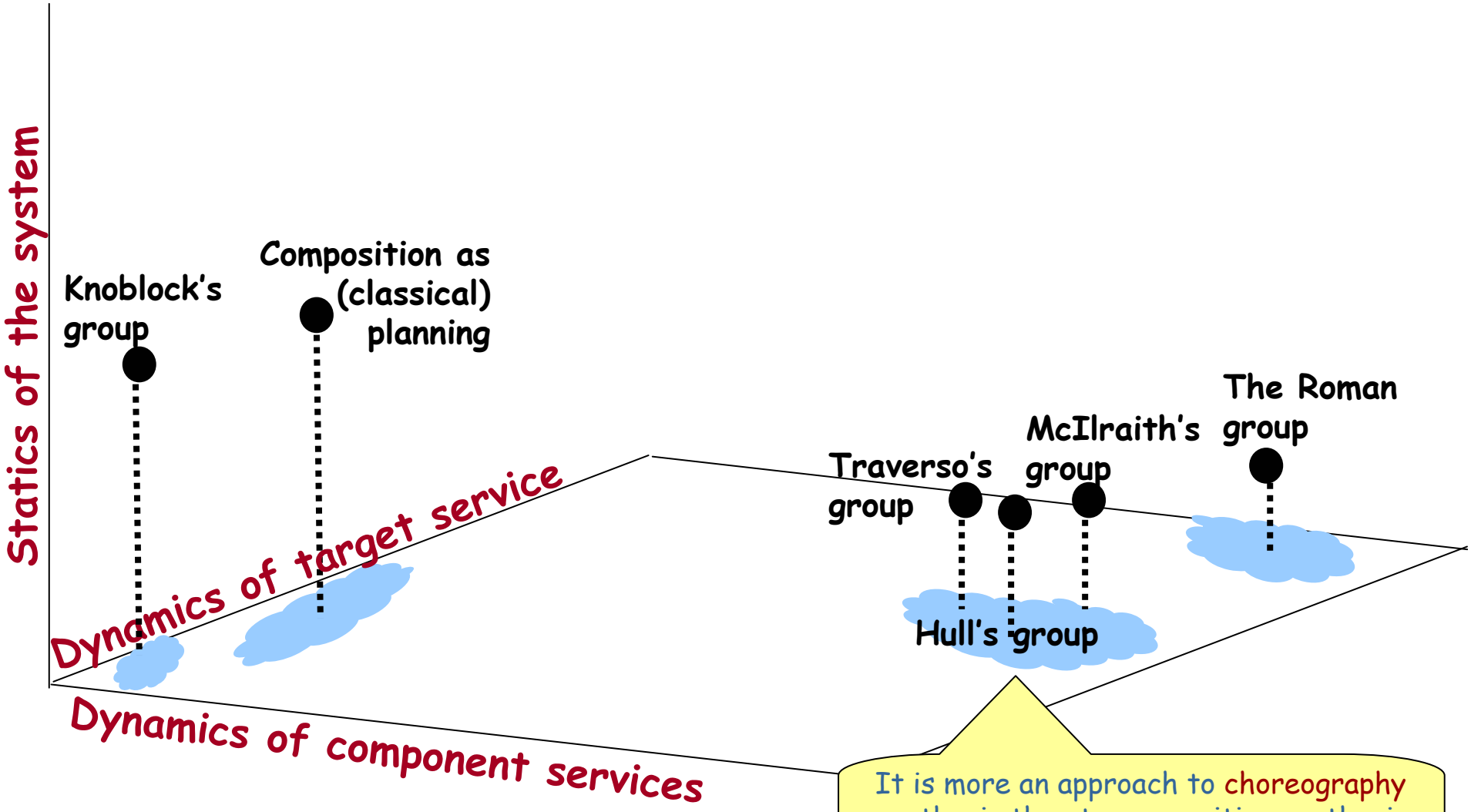


- Services seen as (finite) transition systems
- Common ontology:
  - atomic actions and propositions, as in Planning
- Mapping:
  - A service is mapped to the community ontology as a transition system using the alphabet of the community and defining how transitions affect the propositions
- Client Service Request:
  - try to find a sequence of actions to achieve Goal1 (main computation), with guarantees that upon failure Goal2 is reached (exception handling)

Can be seen as a Client-tailored approach

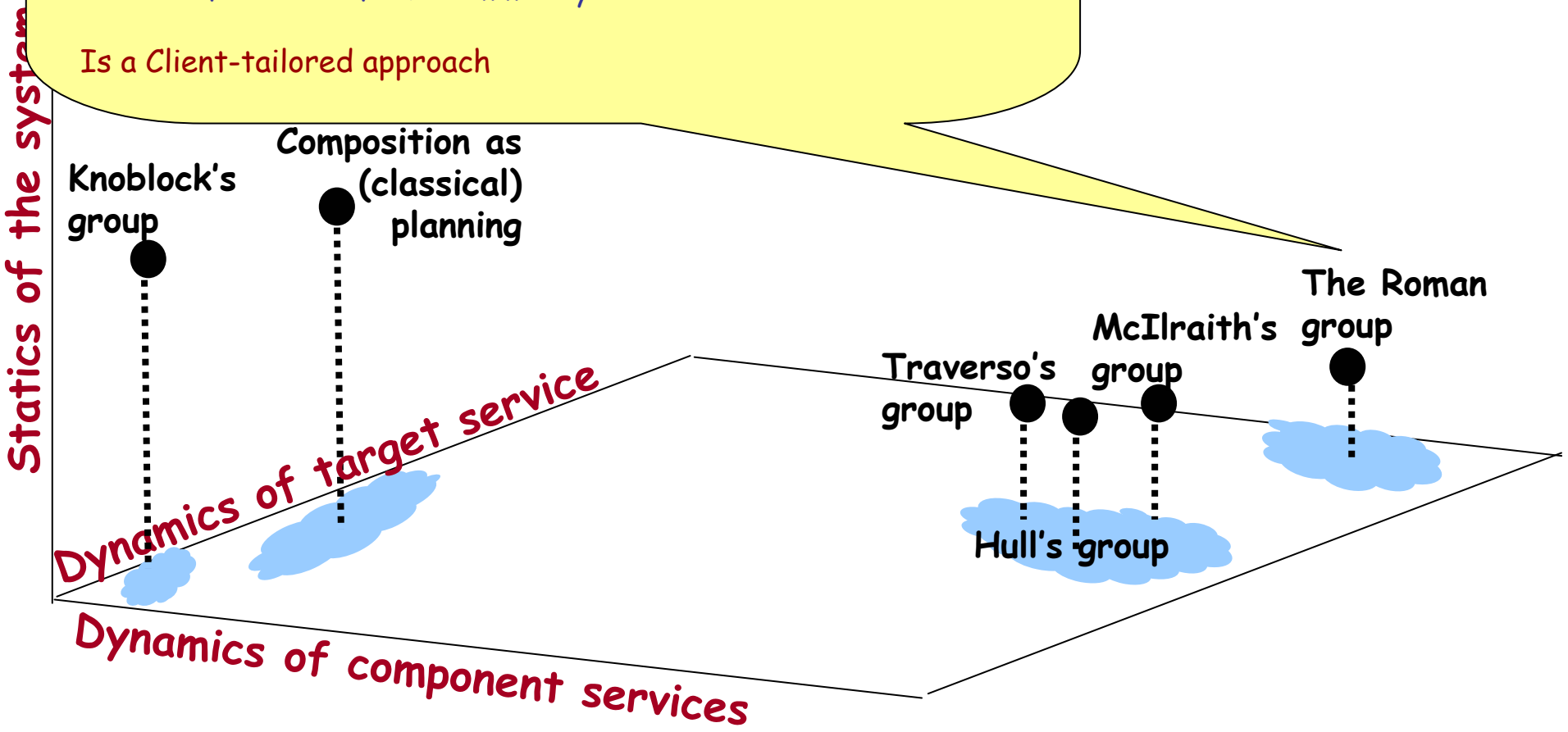


Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"



- Services seen as (finite) transition systems
- Common ontology: Alphabet of atomic actions
- Mapping:
  - A service is mapped to the community ontology as a transition system using the alphabet of the community
- Client Service Request:
  - Desired service behavior, i.e., a (finite) TS using the common set of actions of the community

Is a Client-tailored approach



References

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- **Read and exploit data integration literature!**

  Survey on data integration

  [Halevy VLDBJ01] A. Y. Halevy: Answering queries using views: A survey. VLDB J. 10(4): 270-294 (2001)
  [Lenzerini PODS02] M. Lenzerini: Data Integration: A Theoretical Perspective. PODS 2002: 233-246
  [Ullman ICDT97] J. D. Ullman: Information Integration Using Logical Views. ICDT 1997: 19-40

  Seminal papers

  [Levy etal PODS05] A. Y. Levy, A. O. Mendelzon, Y. Sagiv, D. Srivastava: Answering Queries Using Views. PODS 1995: 95-104
  [Abiteboul etal PODS98] S. Abiteboul, O. M. Duschka: Complexity of Answering Queries Using Materialized Views. PODS 1998: 254-263
  [Duschka etal PODS97] O. M. Duschka, M. R. Genesereth: Answering Recursive Queries Using Views. PODS 1997: 109-116
  [Calvanese etal JCSS02] D. Calvanese, G. De Giacomo, M. Lenzerini, M. Y. Vardi: Rewriting of Regular Expressions and Regular Path Queries. J. Comput. Syst. Sci. 64(3): 443-465 (2002)
  [Rajaraman etal PODS95] A. Rajaraman, Y. Sagiv, J. D. Ullman: Answering Queries Using Templates with Binding Patterns. PODS 1995: 105-112

- **See how other service-researchers have used it!**

  [Ghandeharizadeh etal ICWS03] S. Ghandeharizadeh, C. A. Knoblock, C. Papadopoulos, C. Shahabi, E. Alwagait, J. L. Ambite, M. Cai, C. Chen, P. Pol, R. R. Schmidt, S. Song, S. Thakkar, R. Zhou: Proteus: A System for Dynamically Composing and Intelligently Executing Web Services. ICWS 2003: 17-21
  [Thakkar etal P4WGS] S. Thakkar, J. L. Ambite, C. A. Knoblock: A Data Integration Approach to Automatically Composing and Optimizing Web Services. P4WGS –ICAPS WS 2004: 86-93

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

Transition Systems (in more details)

Transition Systems

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

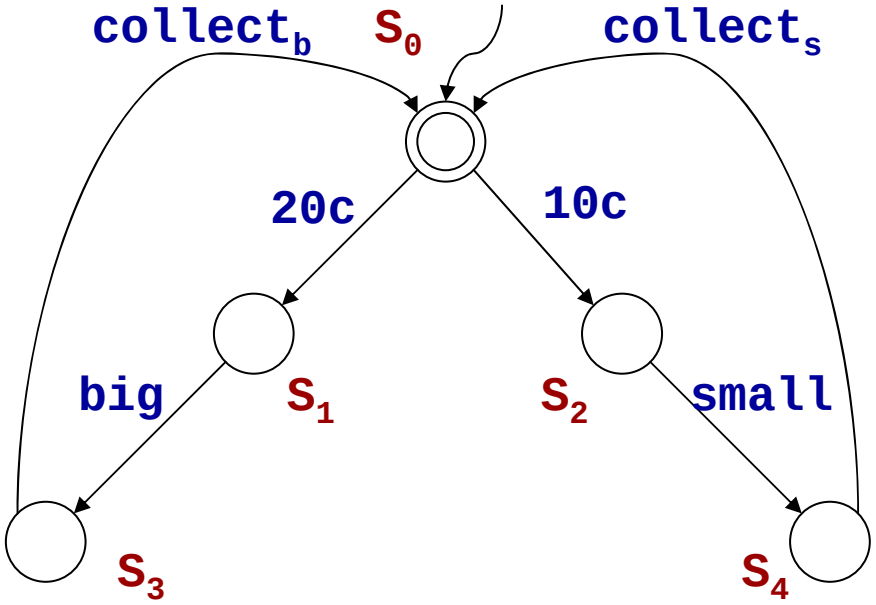
- A transition system TS is a tuple $T = \langle A, S, S^0, \delta, F \rangle$ where:
  - A is the set of actions
  - S is the set of states
  - $S^0 \subseteq S$ is the set of initial states
  - $\delta \subseteq S \times A \times S$ is the transition relation
  - $F \subseteq S$ is the set of final states

(c.f. Kripke Structure)

- Variants:
  - No initial states
  - Single initial state
  - Deterministic actions
  - States labeled by propositions other than Final/¬Final

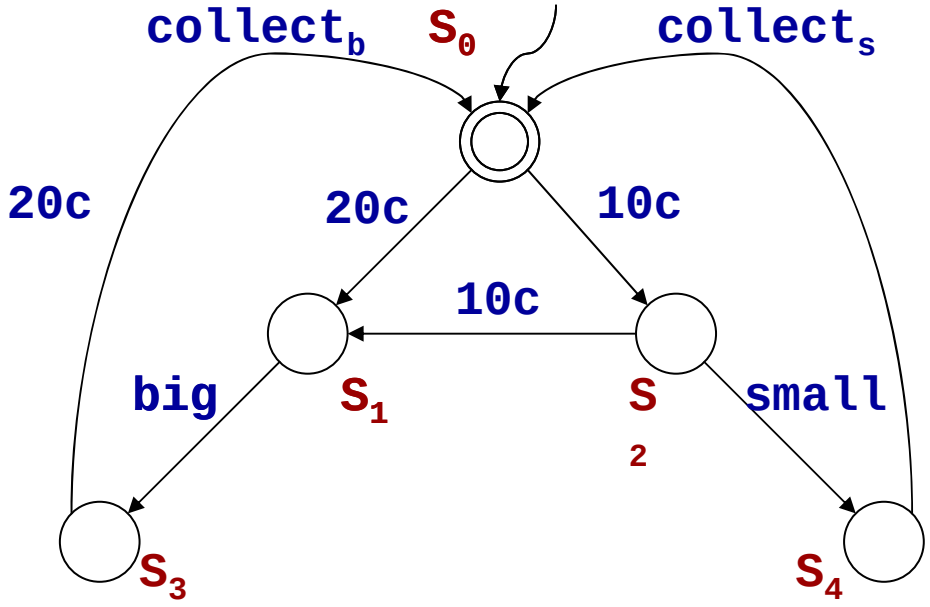
Example (Vending Machine)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"



Example (Another Vending Machine)

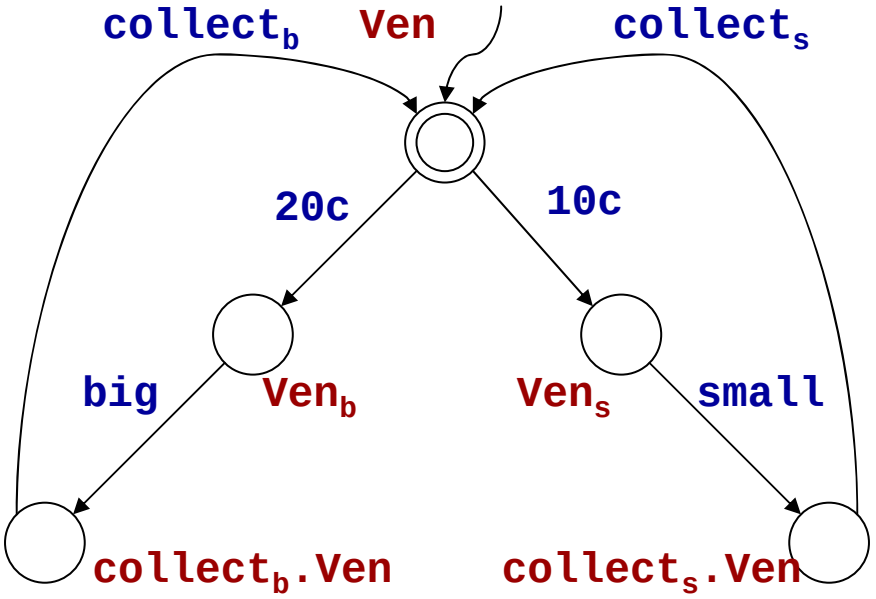
Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"



Process Algebras are Formalisms for Describing TS

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

- Trans (a la CCS)
  - $Ven = 20c.Ven_b + 10c.Ven_s$
  - $Ven_b = big.collect_b.Ven$
  - $Ven_l = small.collect_s.Ven$
- Final
  - $\surd$ Ven

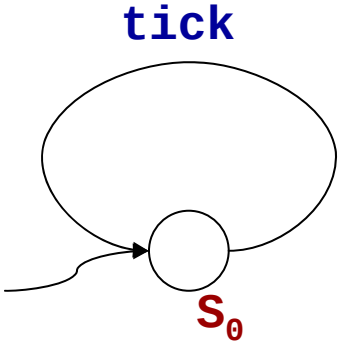


- *TS may have infinite states - e.g., this happens when generated by process algebras involving iterated concurrency*
- *However we have good formal tools to deal only with finite states TS*

Example (Clock)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

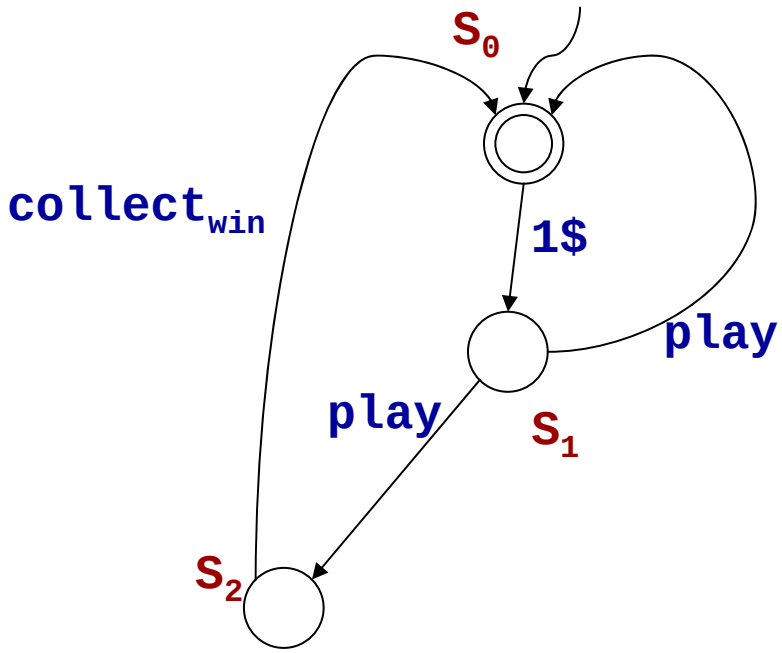
TS may describe (legal) nonterminating processes



Example (Slot Machine)

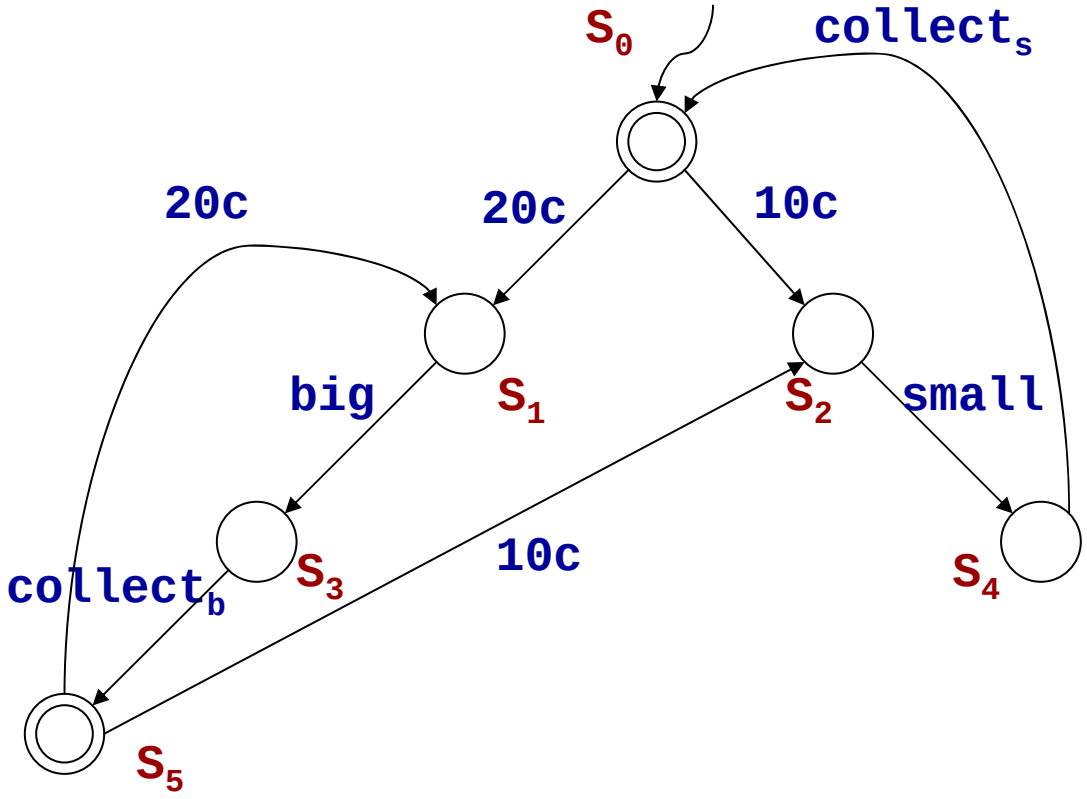
Dipartimento di Informatica e Sistemistica "Antonio Ruberti" — Università di Roma "La Sapienza"

Nondereminisic transitions express
choice that is not under the control of clients



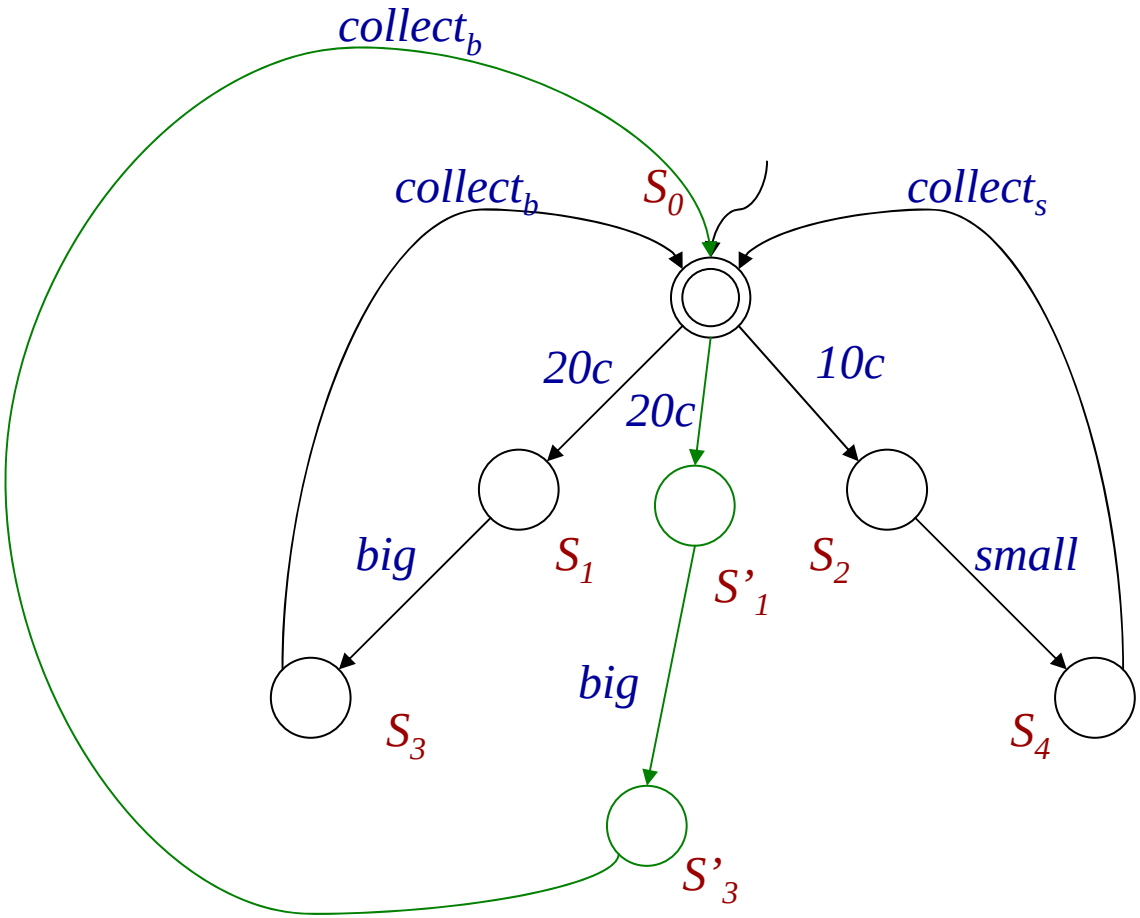
Example (Vending Machine - Variant 1)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"



Example (Vending Machine - Variant 2)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"



Bisimulation

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

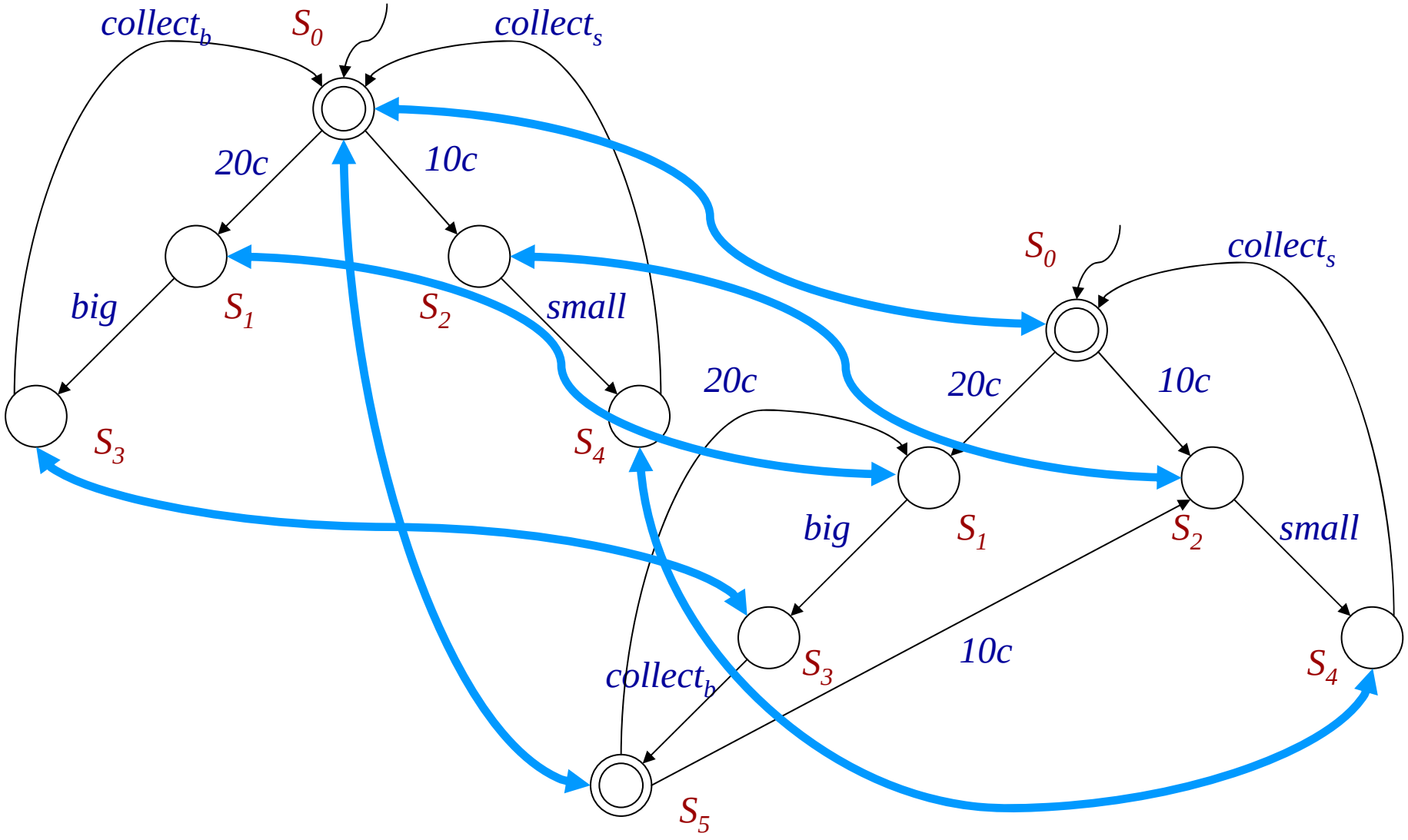
- A binary relation R is a **bisimulation** iff:

  $(s,t) \in R$ implies that
  - s is *final* iff t is *final*
  - for all actions a
    - if $s \rightarrow_a s'$ then $\exists\, t' \,.\, t \rightarrow_a t'$ and $(s',t') \in R$
    - if $t \rightarrow_a t'$ then $\exists\, s' \,.\, s \rightarrow_a s'$ and $(s',t') \in R$

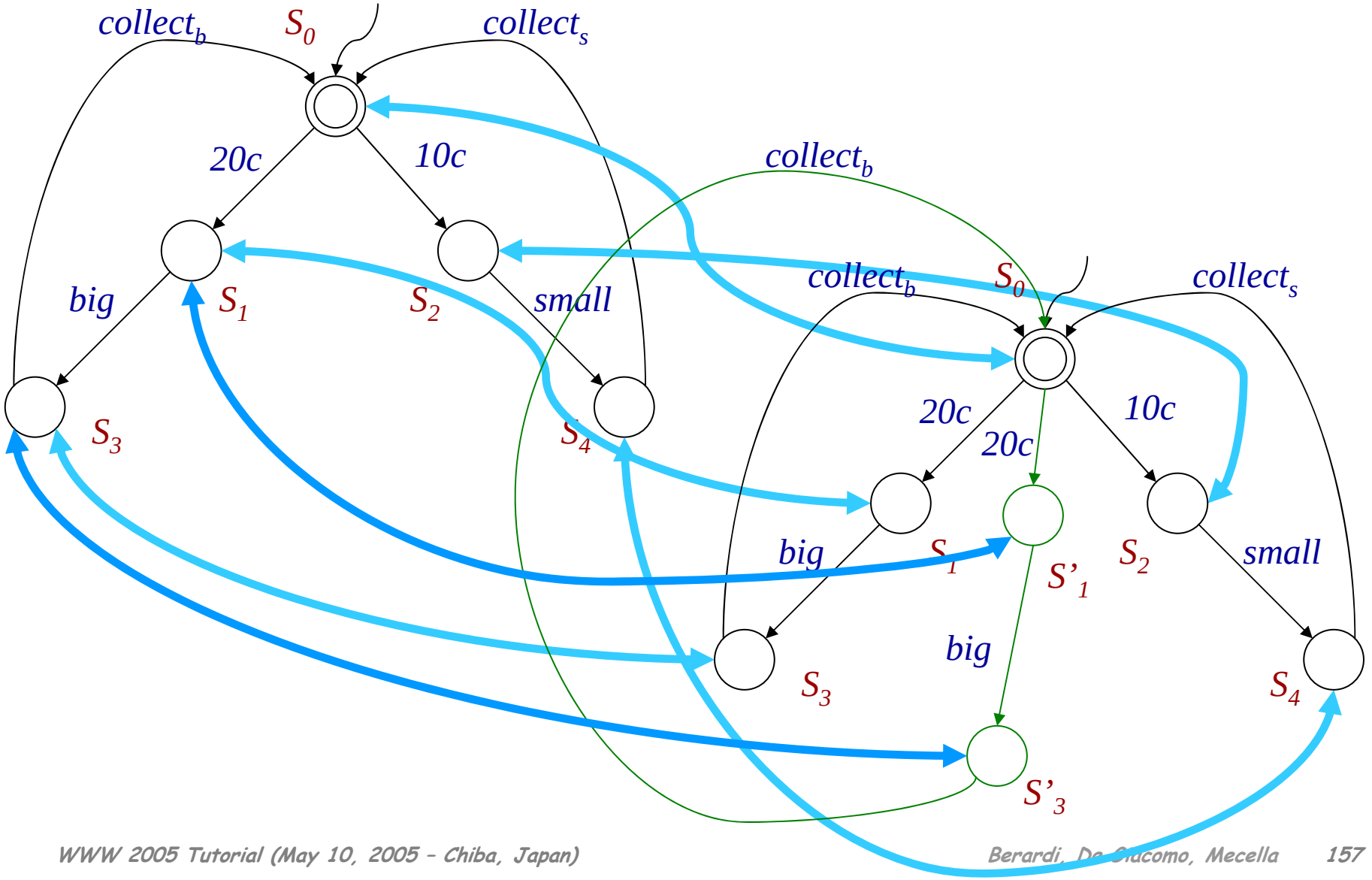
  Note it is a co-inductive definition!

- A state s_0 of transition system S is **equivalent** to a state t_0 of transition system T iff there **exists** a **bisimulation** between the initial states s_0 and t_0.

Example of Bisimulation



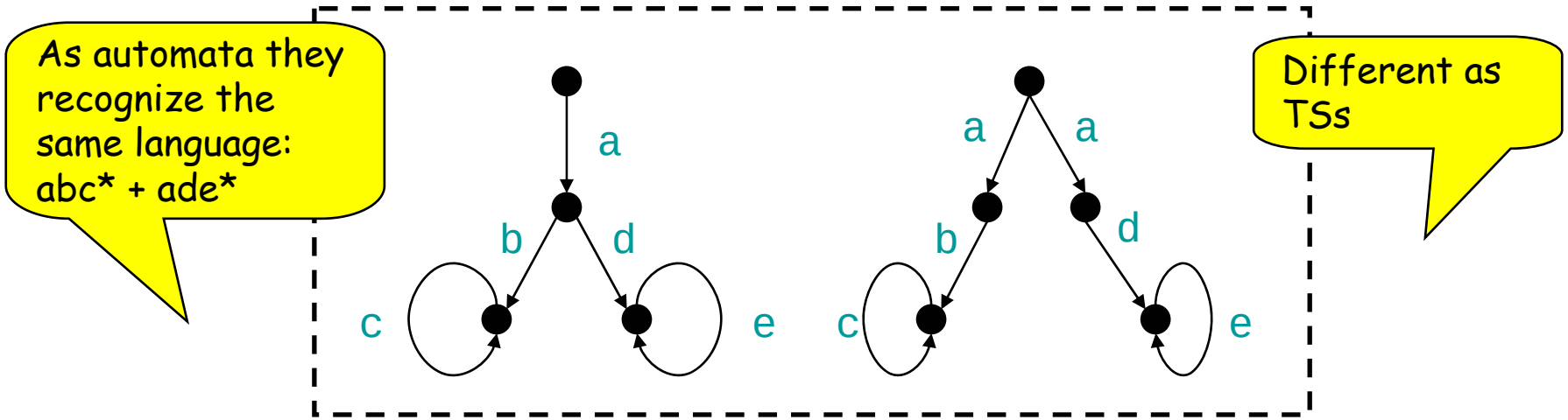
Example of Bisimulation



Automata vs. Transition Systems

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

- Automata
  - define sets of runs (or traces or strings): (finite) length sequences of actions
- TSs
  - … but I can be interested also in the alternatives "encountered" during runs, as they represent client's "choice points"



Logics of Programs

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Are modal logics that allow to describe properties of transition systems

- Examples:
  - HennesyMilner Logic
  - Propositional Dynamic Logics
  - Modal (Propositional) Mu-calculus

- Perfectly suited for describing transition systems: they can tell apart transition systems modulo bisimulation

HennessyMilner Logic

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- $\Phi := P \mid$ *(atomic propositions)*
  $\neg \Phi \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid$ *(closed under boolean operators)*
  $[a]\Phi \mid \langle a \rangle \Phi$ *(modal operators)*

- Propositions are used to denote final states

- $\langle a \rangle \Phi$ means there exists an a-transition that leads to a state where Φ holds; i.e., expresses the capability of executing action a bringing about Φ

- $[a]\Phi$ means that all a-transitions lead to states where Φ holds; i.e., express that executing action a brings about Φ

Logics of Programs: Examples

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Usefull abbreviation:
  - $\langle any \rangle \Phi$ stands for $\langle a_1 \rangle \Phi \vee \cdots \vee \langle a_n \rangle \Phi$
  - $[any] \Phi$ stands for $[a_1] \Phi \wedge \cdots \wedge [a_n] \Phi$
  - $\langle any - a_1 \rangle \Phi$ stands for $\langle a_2 \rangle \Phi \vee \cdots \vee \langle a_\nu \rangle \Phi$
  - $[any - a_1] \Phi$ stands for $[a_2] \Phi \wedge \cdots \wedge [a_\nu] \Phi$

- Examples:
  - $\langle a \rangle$true — *cabability of performing action a*
  - $[a]$false — *inability of performing action a*
  - $\neg$Final $\langle any \rangle$true $\wedge$ $[any\text{-}a]$false
    necessity/inevitability of performing action a
    i.e., action a is the only action possible
  - $\neg$Final $\wedge$ $[any]$false — *deadlock!*

Propositional Dynamic Logic

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" — Università di Roma "La Sapienza"

- $\Phi := P \mid$ *(atomic propositions)*
  $\neg \Phi \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid$ *(closed under boolean operators)*
  $[r]\Phi \mid \langle r \rangle \Phi$ *(modal operators)*

  $r := a \mid r_1 + r_2 \mid r_1;r_2 \mid r^* \mid P?$ *(complex actions as regular expressions)*

- Essentially add the capability of expressing partial correctness assertions via formulas of the form
  - $\Phi_1 \rightarrow [r]\Phi_2$ *under the conditions Φ_1 all possible executions of r that terminate reach a state of the TS where Φ holds*

- Also add the ability of asserting that a property holds in all nodes of the transition system
  - $[(a_1 + \cdots + a_\nu)^*]\Phi$ *in every reachable state of the TS Φ holds*

- Useful abbereviations:
  - any stands for $(a_1 + \cdots + a_\nu)$ *- observe that + can be expressed in HM Logic*
  - u stands for any* *- this is the so called master/universal modality*

Modal Mu-Calculus

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Φ := P | *(atomic propositions)*
  $\neg \Phi \mid \Phi_1 \wedge \Phi_2 \mid \Phi_1 \vee \Phi_2 \mid$ *(closed under boolean operators)*
  $[r]\Phi \mid \langle r \rangle \Phi$ *(modal operators)*

  $\mu X.\Phi(X) \mid \nu X.\Phi(X)$ *(fixpoint operators)*

- It is the most expressive logic of the family of logics of programs.
- It subsumes
  - PDL (modalities involving complex actions are translated into fomulas involving fixpoints)
  - LTL (linear time temporal logic),
  - CTS, CTS* (branching time temporal logics)

- Examples:
- $[any^*]\Phi$ can be expressed as $\nu X. \Phi \wedge [any]X$

- $\mu X. \Phi \vee [any]X$ *along all runs eventually Φ*
- $\mu X. \Phi \vee \langle any \rangle X$ *along some run eventually Φ*
- $\nu X. [a](\mu Y. \langle any \rangle true \wedge [any\text{-}b]Y) \wedge X$
  every run that that contains a contains later b

Model Checking

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Model checking is polynomial in the size of the TS for
  - HennessyMilner Logic
  - PDL
  - Mu-Calculus
- Also model checking is wrt the formula
  - Polynomial for HennessyMiner Logic
  - Polynomial for PDL
  - Polynomial for Mu-Calculus with bounded alternation of fixpoints and NP∩coNP in general

Model Checking

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Given a TS T, one of its states s, and a formula Φ verify whether the formula holds in s. Formally:

$$T,s \models \Phi$$

- Examples (TS is our vending machine):
  - $S_0 \models$ Final
  - $S_0 \models$ <10c>true — *capability of performing action 10c*
  - $S_2 \models$ [big]false — *inability of performing action big*
  - $S_0 \models$ [10c][big]false — *after 10c cannot execute big*
  - $S_i \models \mu X.\ \text{Final} \vee [\text{any}]\ X$ — *eventually a final state is reached*
  - $S_0 \models \nu Z.\ (\mu X.\ \text{Final} \vee [\text{any}]\ X) \wedge [\text{any}]\ Z$ — *or equivalently*
    $S_0 \models [\text{any}^*](\mu X.\ \text{Final} \vee [\text{any}]\ X)$
    from everywhere eventually final

Planning as Model Checking

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- **Build the TS of the domain:**
  - Consider the set of states formed all possible truth value of the propositions (this works only for propositional setting).
  - Use Pre's and Post of actions for determining the transitions

  Note: the TS is exponential in the size od the description.

- **Write the goal in a logic of program**
  - typically a single least fixpoint formula of Mu-Calculus

- **Planning:**
  - model check the formula on the TS starting from the given initial state.
  - use the path (paths) used in the above model checking for returning the plan.

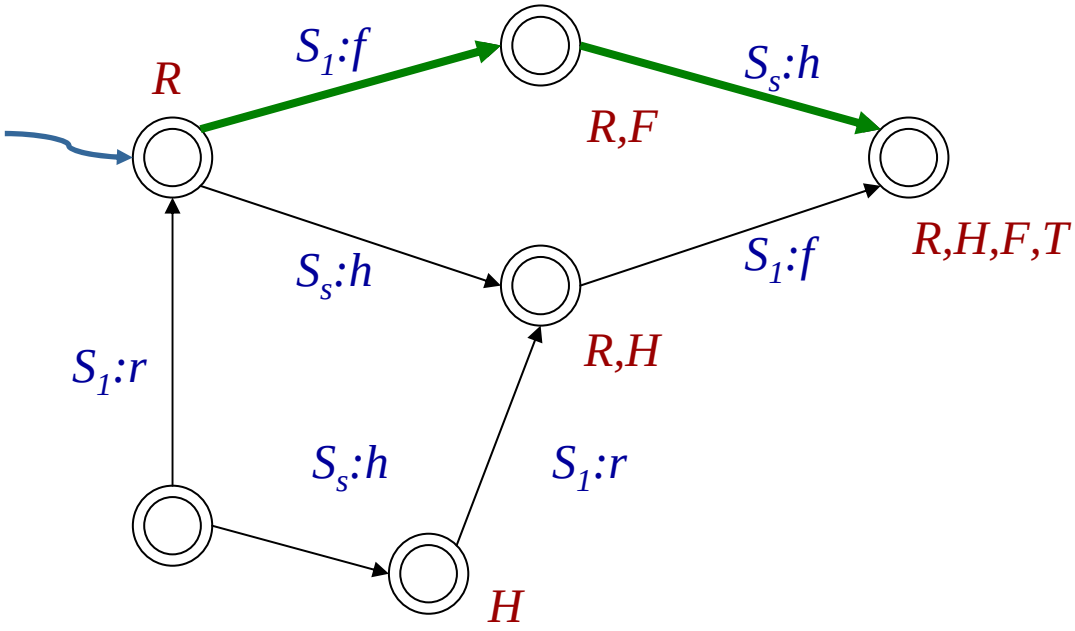
- *This basic technique works only when we have complete information (or at least total observability on state):*
  - *Sequiential plans if initial state known and actions are deterministic*
  - *Conditional plans if many possible initial states and/or actions are nondeterministic*

Example

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Operators (Services + Mappings)
  - $Registered \wedge \neg FlightBooked \rightarrow [S_1:bookFlight]\ FlightBooked$
  - $\neg Registered \rightarrow [S_1:register]\ Registered$
  - $\neg HotelBooked \rightarrow [S_2:bookHotel]\ HotelBooked$
- Additional constraints (Community Ontology):
  - $TravelSettledUp \equiv FlightBooked \wedge HotelBooked \wedge EventBooked$
- Goals (Client Service Requests):
  - Starting from state
    $Registered \wedge \neg FlightBooked \wedge \neg\ HotelBooked \wedge \neg EventBooked$
    check $\langle any^* \rangle TravelSettedUp$
  - Starting from all states such that
    $\neg FlightBooked \wedge \neg\ HotelBooked \wedge \neg EventBooked$
    check $\langle any^* \rangle TravelSettledUp$

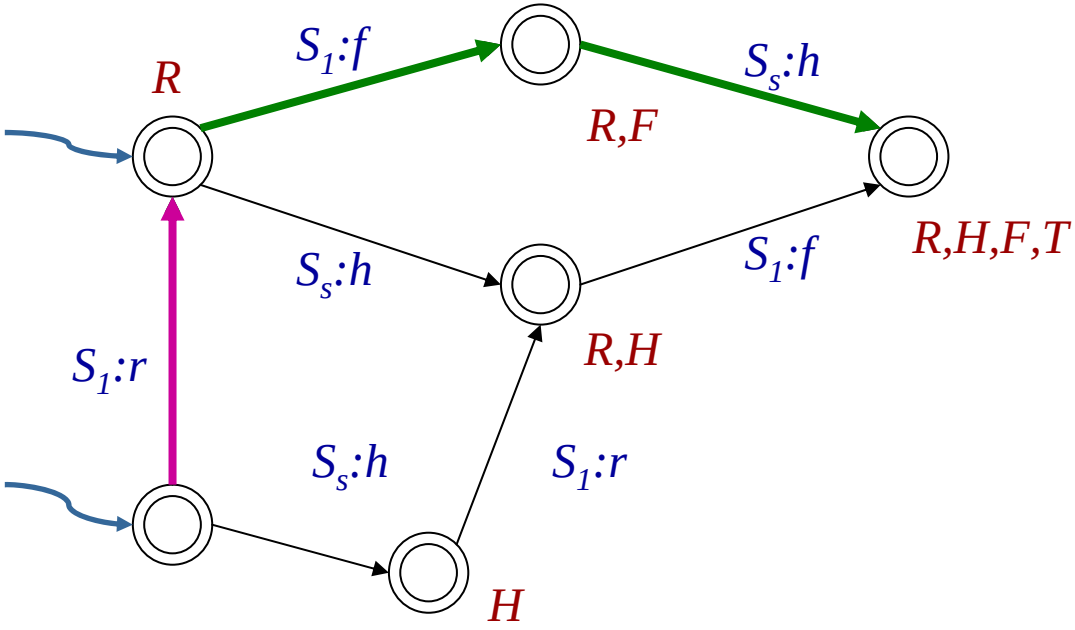
Example



Plan:
S_1:bookFlight;
S_2:bookHotel

Starting from state
Registered ∧ ¬ FlightBooked ∧ ¬ HotelBooked ∧ ¬ EventBooked
check
‹any*›TravelSettledUp

Example



Plan:

```
if(¬Registered) {
    S1:register;
}
S1:bookFlight;
S2:bookHotel
```

Starting from states where

¬ FlightBooked ∧ ¬ HotelBooked ∧ ¬ EventBooked

check

<any*>TravelSettledUp

Satisfiability

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Observe that a formula Φ may be used to select among all TS T those such that for a given state s we have that $T,s \models \Phi$
- SATISFIABILITY: Given a formula Φ verify whether there exists a TS T and a state s such that. Formally:

  check whether exists T, s such that $T,s \models \Phi$

- Satisfiability is:
  - PSPACE for HennesyMilner Logic
  - EXPTIME for PDL
  - EXPTIME for Mu-Calculus

References

[Stirling Banff96] C. Stirling: Modal and temporal logics for processes. Banff Higher Order Workshop LNCS 1043, 149-237, Springer 1996

[Bradfield&Stirling HPA01] J. Bradfield, C. Stirling: Modal logics and mu-calculi. Handbook of Process Algebra, 293-332, Elsevier, 2001.

[Stirling 2001] C. Stirling: Modal and Temporal Properties of Processes. Texts in Computer Science, Springer 2001

[Kozen&Tiuryn HTCS90] D. Kozen, J. Tiuryn: Logics of programs. Handbook of Theoretical Computer Science, Vol. B, 789–840. North Holland, 1990.

[HKT2000] D. Harel, D. Kozen, J. Tiuryn: Dynamic Logic. MIT Press, 2000.

[Clarke& Schlingloff HAR01] E. M. Clarke, B. Schlingloff: Model Checking. Handbook of Automated Reasoning 2001: 1635-1790

[CGP 2000] E.M. Clarke, O. Grumberg, D. Peled: Model Checking. MIT Press, 2000.

[Emerson HTCS90] E. A. Emerson. Temporal and Modal Logic. Handbook of Theoretical Computer Science, Vol B: 995-1072. North Holland, 1990.

[Emerson Banff96] E. A. Emerson. Automated Temporal Reasoning about Reactive Systems. Banff Higher Order Workshop, LNCS 1043, 111-120, Springer 1996

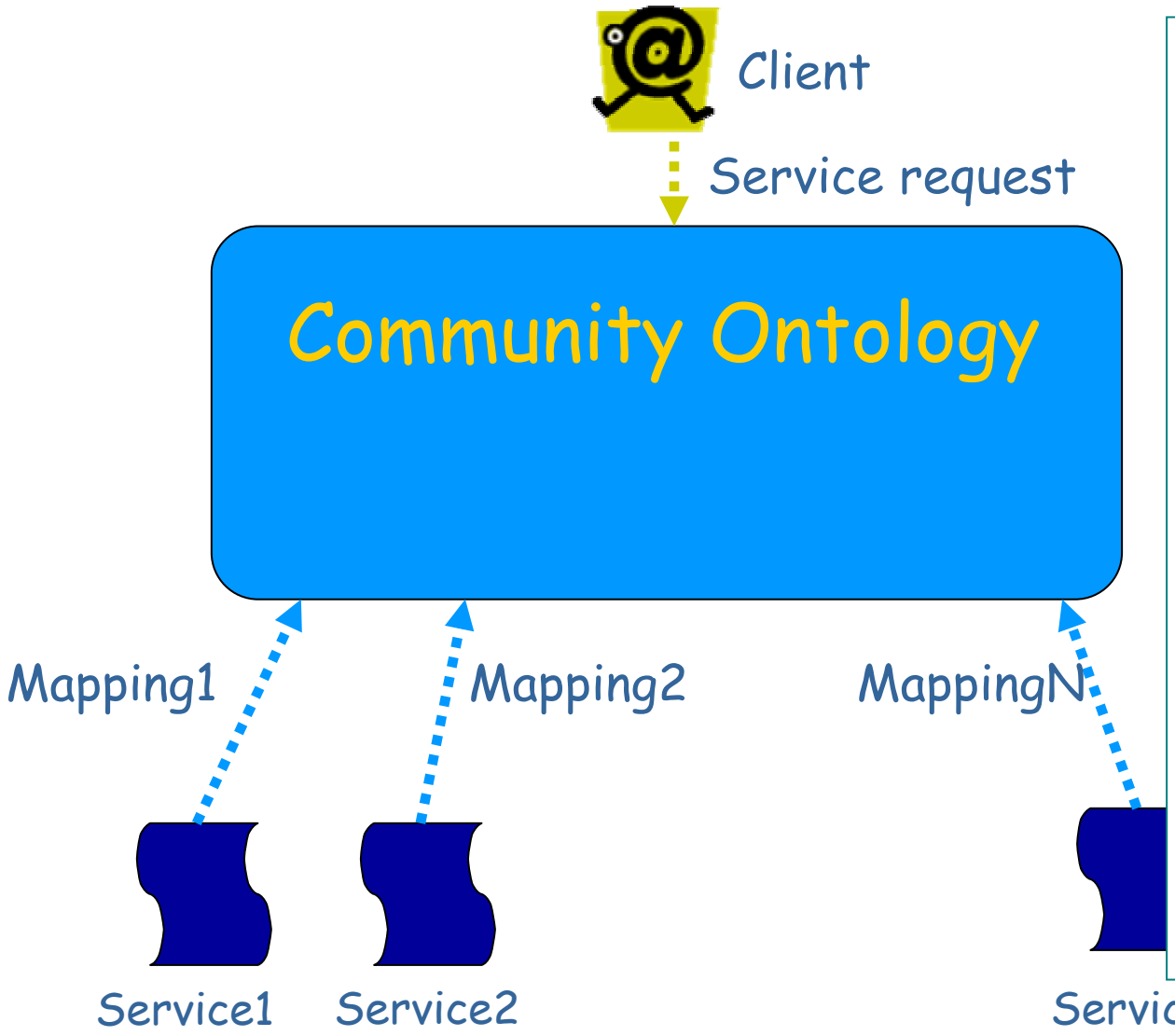
[Vardi CST] M. Vardi: Alternating automata and program verification. Computer Science Today - Recent Trends and Developments, LNCS Vol. 1000, Springer, 1995.

[Vardi etal CAV94] M. Vardi, O. Kupferman and P. Wolper: An Automata-Theoretic Approach to Branching-Time Model Checking (full version of CAV'94 paper).

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

Composition: the Roman Approach

The Roman Approach



Client-tailored!

Community ontology: just a set of **actions**

Client formulates the service it requires as a **TS** using the **actions** of the common ontology

Available services: described in terms of a **TS** using **actions** of the community ontology

The **community** realizes the **client's target service** by "reversing" the mapping and hence using **fragments** of the computation of the the **available services**

Community of Services

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- A community of Services is
  - a set of services ...
  - ... that share implicitly a *common understanding* on a common set of actions (common ontology limited to the alphabet of actions)...
  - ... and export their behavior using (finite) TS over this common set of actions
- A client specifies needs as a service behavior, i.e, a (finite) TS using the common set of actions of the community

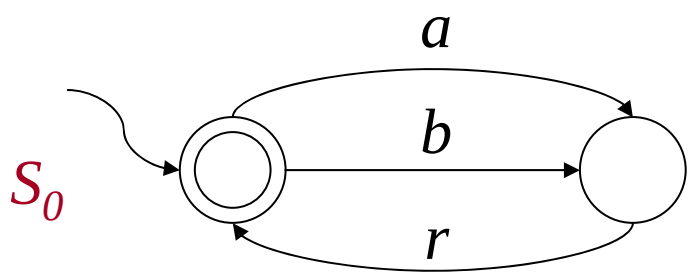
(Target & Available) Service TS

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

- We model services as finite TS $T = (\Sigma, S, s^0, \delta, F)$ with
  - single initial state *(s^0)*
  - deterministic transitions *(i.e., δ is a partial function from $S \times \Sigma$ to S)*

Note: In this way the client entirely controls/chooses the transition to execute

Example:

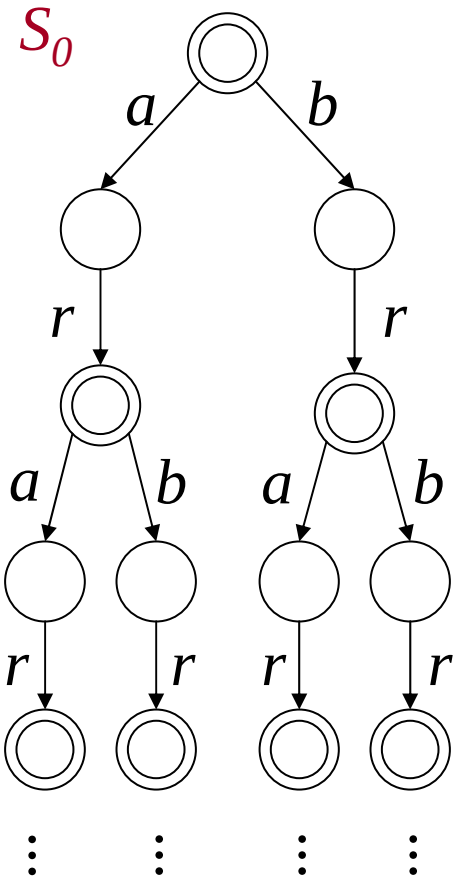


a: "search by author (and select)"
b: "search by title (and select)"
r: "listen (the selected song)"

Service Execution Tree

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

By "unfolding" a (finite) TS one gets an (infinite) execution tree
-- yet another (infinite) TS which bisimilar to the original one)



- *Nodes: history (sequence) of actions executed so far*
- *Root: no action yet performed*
- *Successor node x·a of x: action a can be executed after the sequence of action x*
- *Final nodes: the service can terminate*

Formalizing Service Composition

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS

Università di Roma "La Sapienza"

Composition:

- coordinating program …
- … that realizes the target service …
- … by suitably coordinating available services

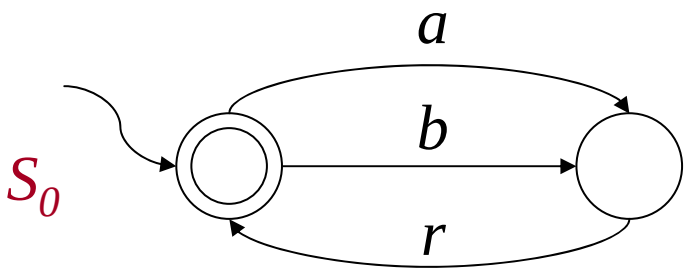
⇒ Composition can be formalized as:

- a labeling of the execution tree of the target service such that …
- … each action in the execution tree is labeled by the community service that executes it …
- … and each possible sequence of actions on the target service execution tree corresponds to possible sequences of actions on the community service execution trees, suitably interleaved

Example of Composition (1)

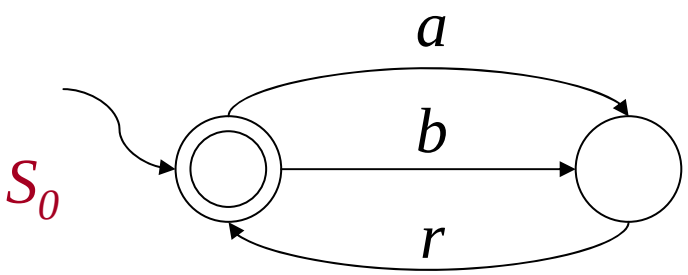
- Community services

- Target service

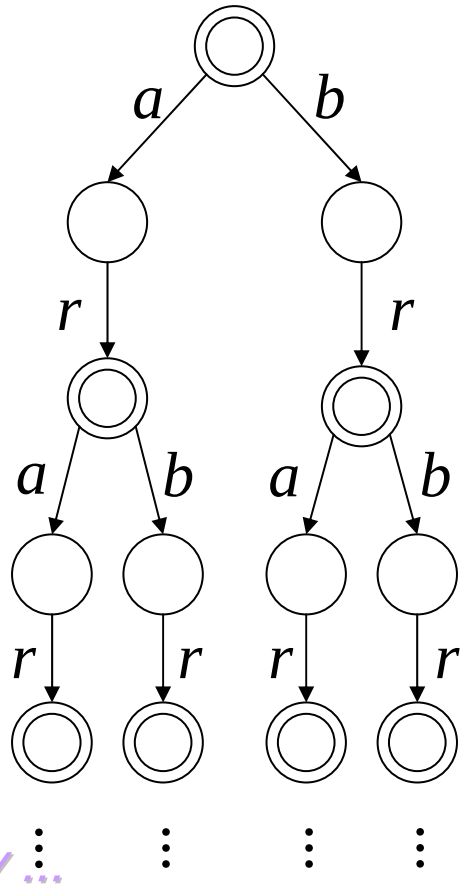


Example of Composition (2)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"



Execution tree of S_0



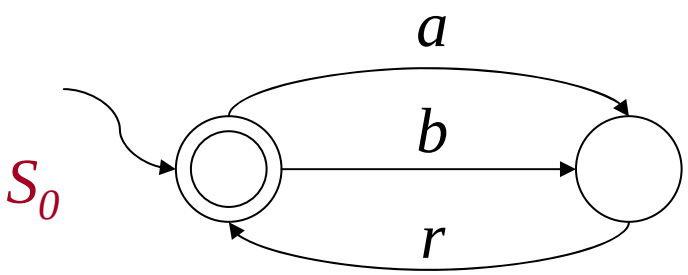
Note: we cannot label the target service TS directly ...

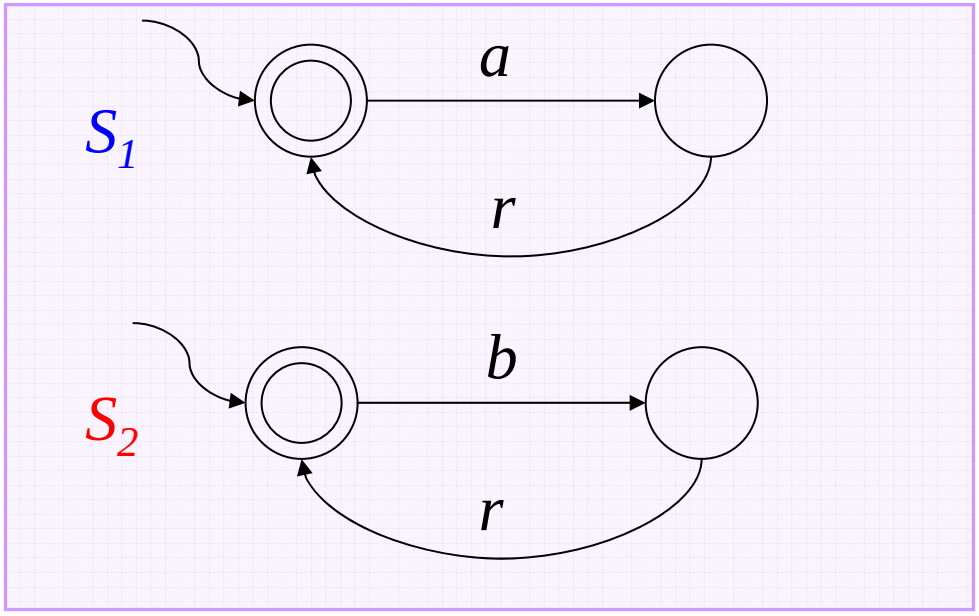
... we need to label the execution tree

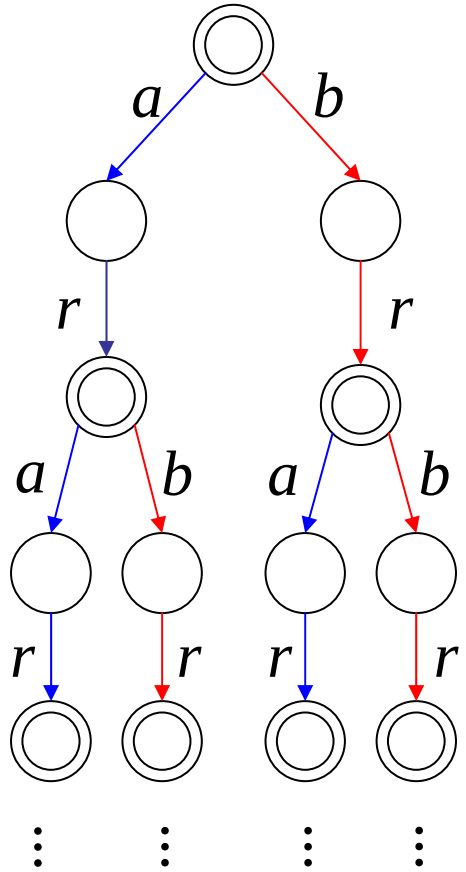
Example of Composition (3)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

$S_0 = orch(S_1 \parallel S_2)$



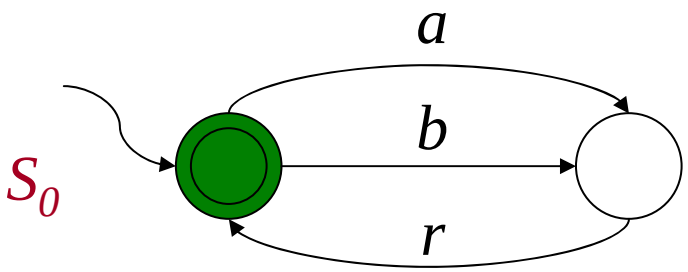


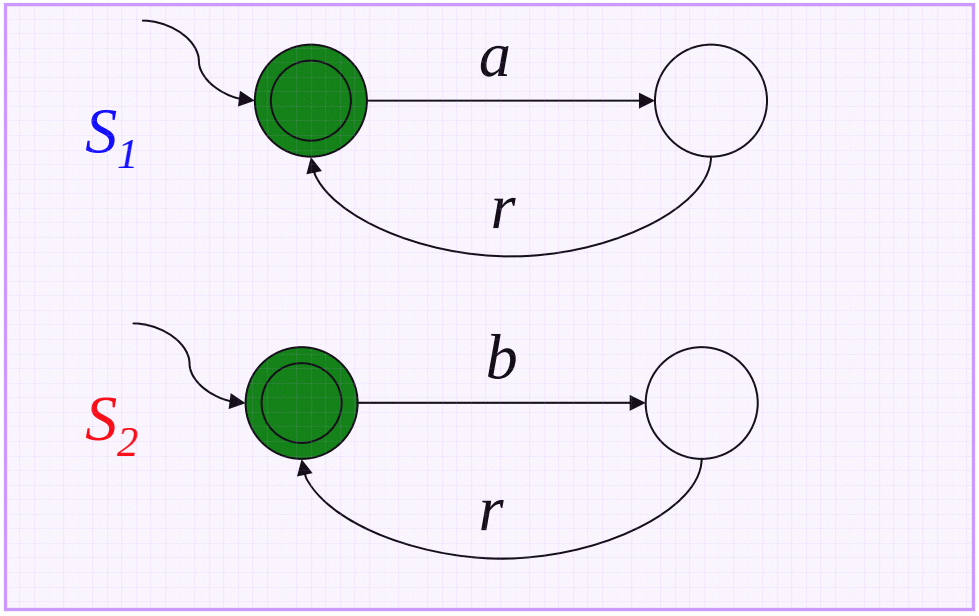


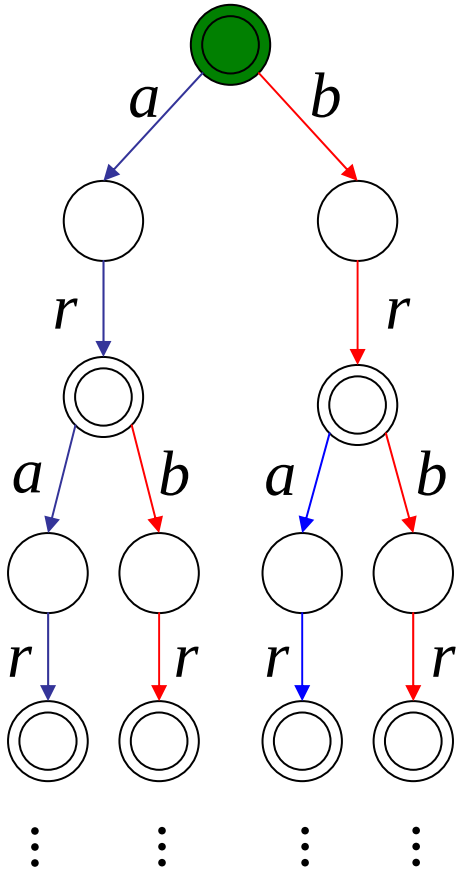
Example of Composition (4)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

$S_0 = orch(S_1 \| S_2)$





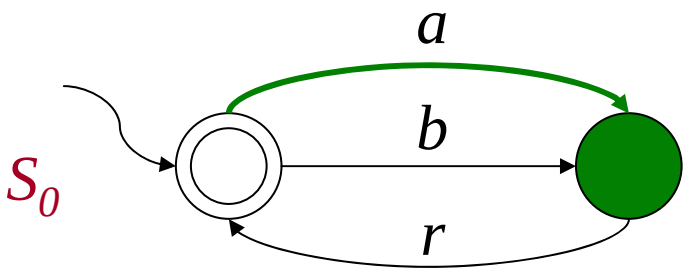


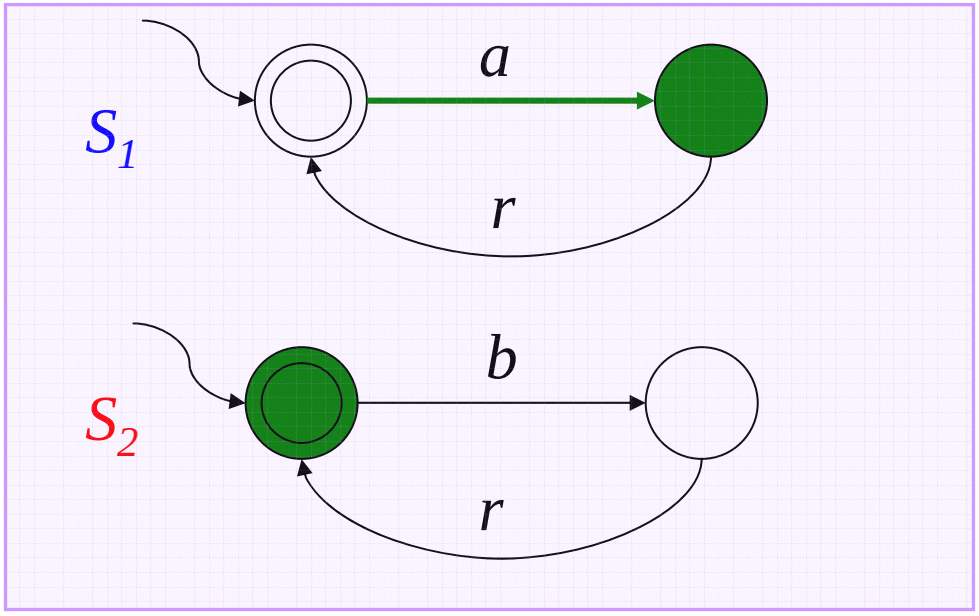
All services start from their starting state

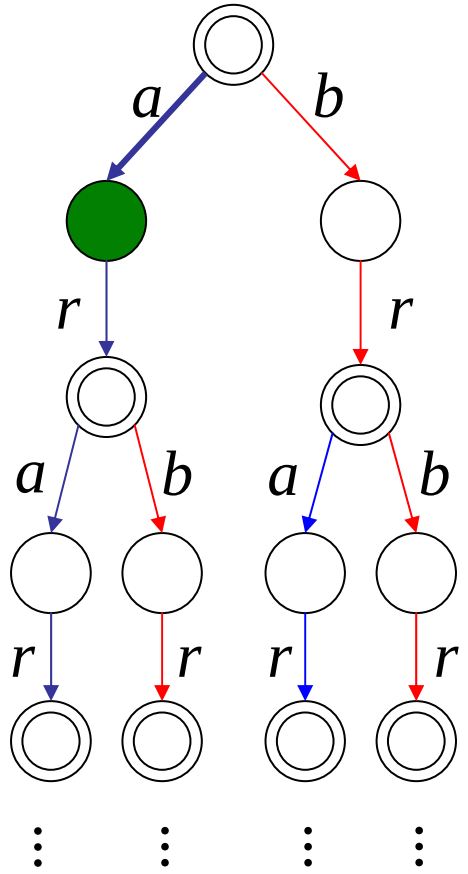
Example of Composition (5)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

$S_0 = orch(S_1 \| S_2)$





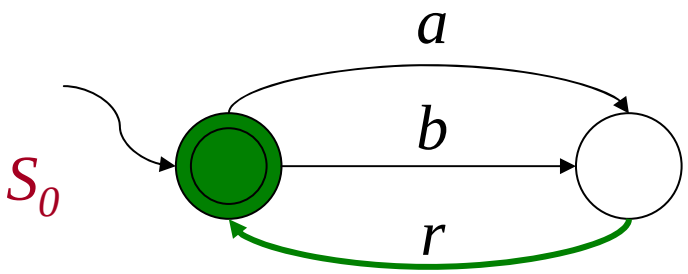


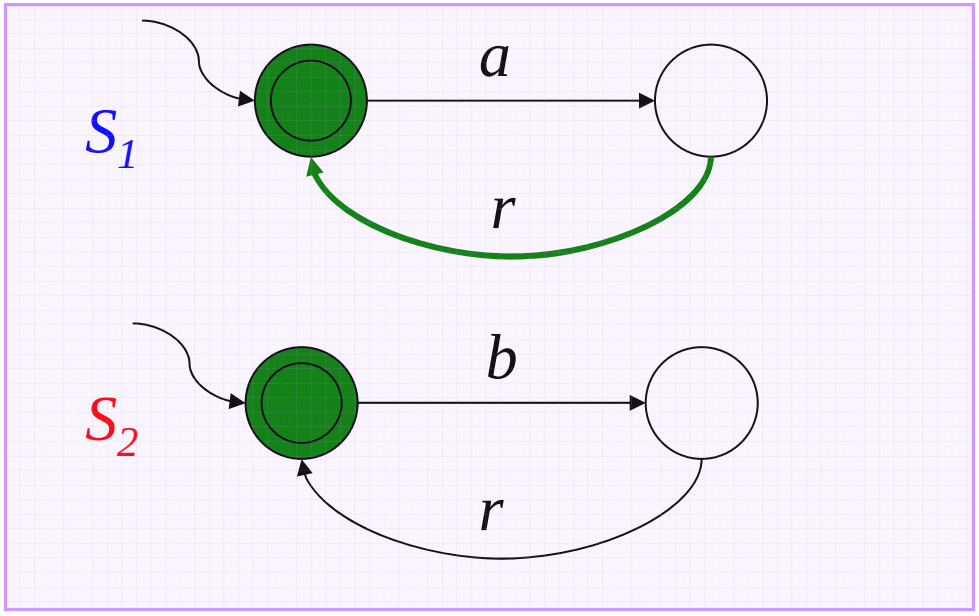
Each action of the target service is executed by at least one of the component services

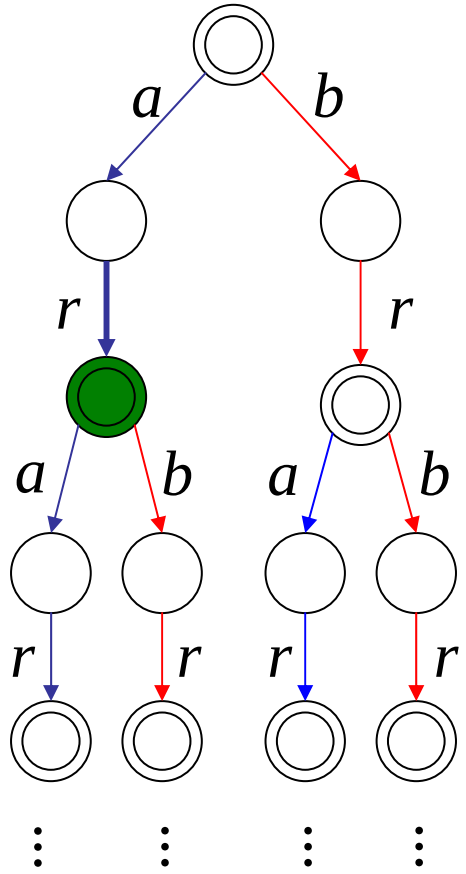
Example of composition (6)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

$S_0 = orch(S_1 \parallel S_2)$





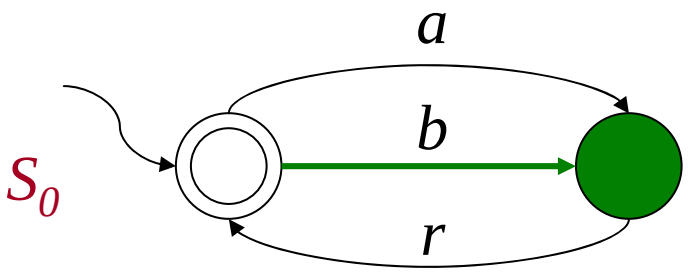


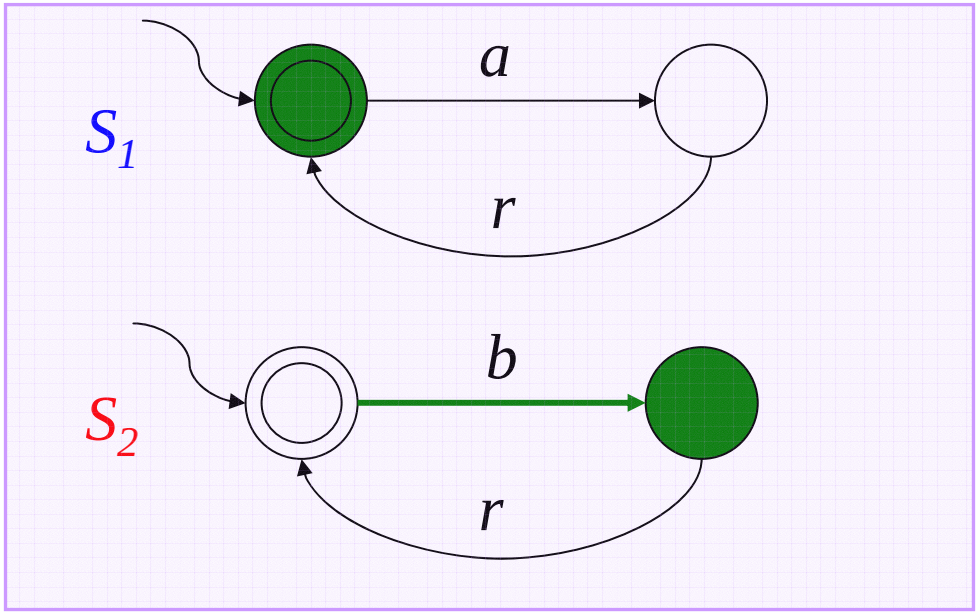
When the target service can be left, then all component services must be in a final state

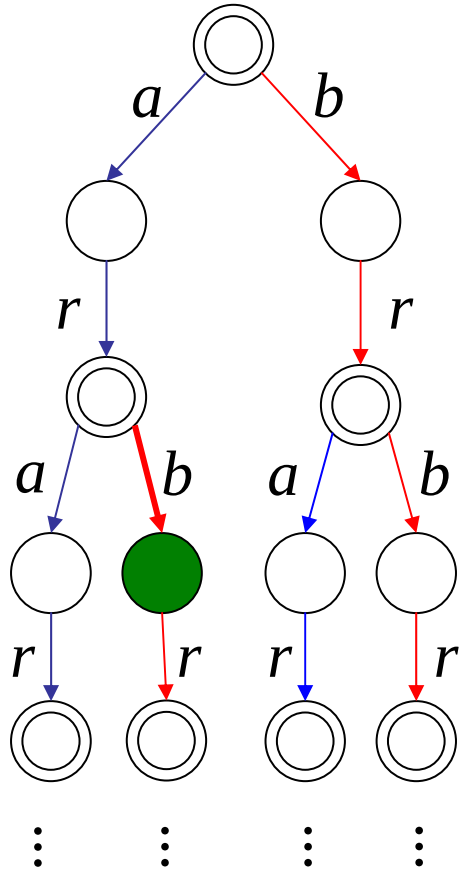
Example of composition (7)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

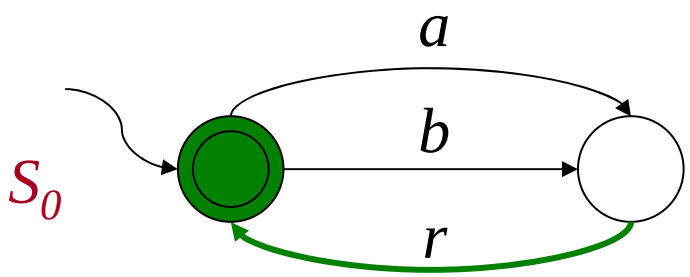
$S_0 = orch(S_1 \parallel S_2)$

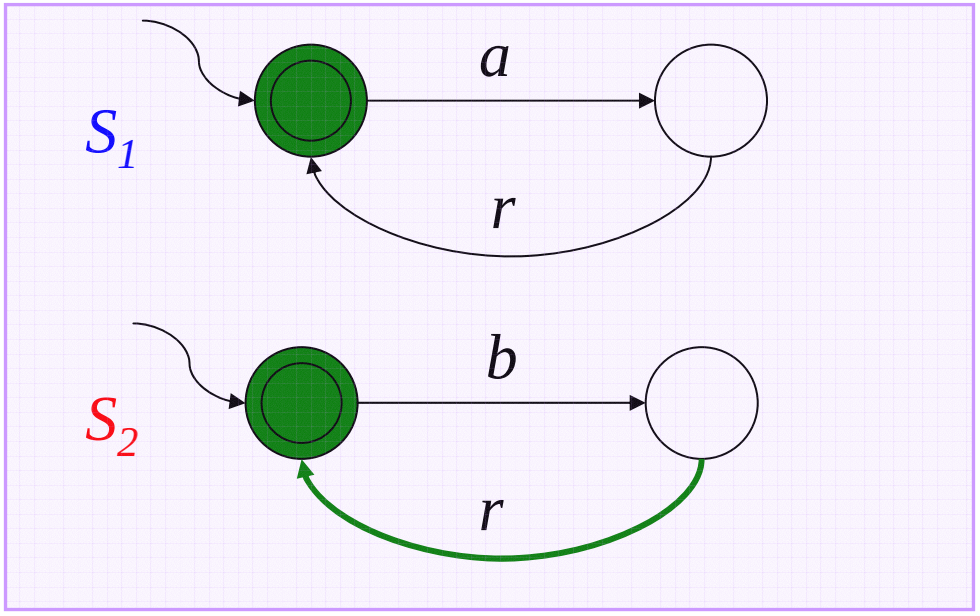




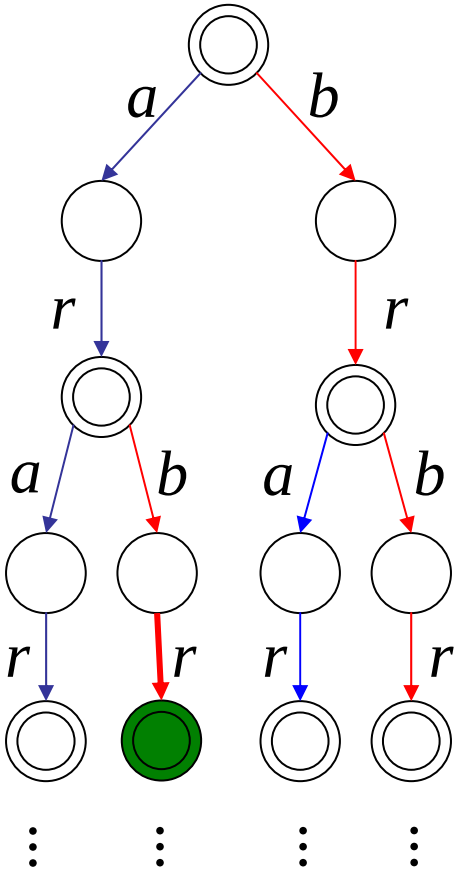


Example of composition (8)



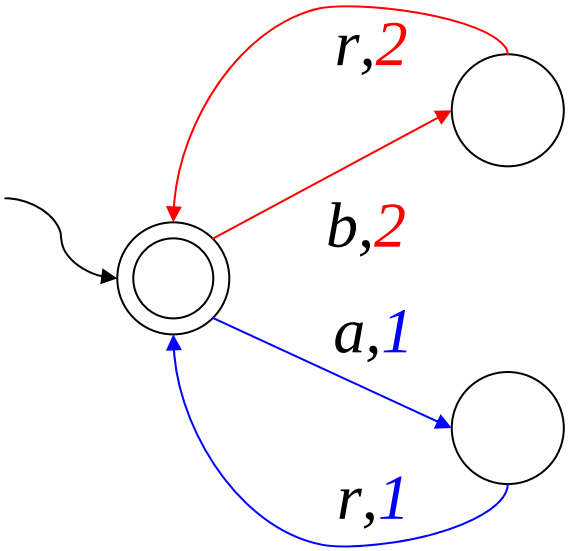


$S_0 = orch(S_1 \| S_2)$



Observation

- This labeled execution tree has a finite representation as a finite TS …
- …with transitions labeled by an action and the service performing the action

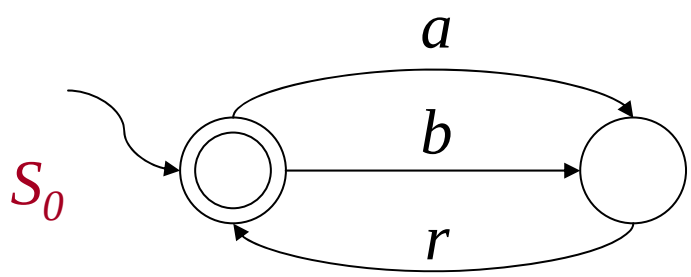


Is this always the case when we deal with services expressible as finite TS? See later…

TS for Services and TS for Composition

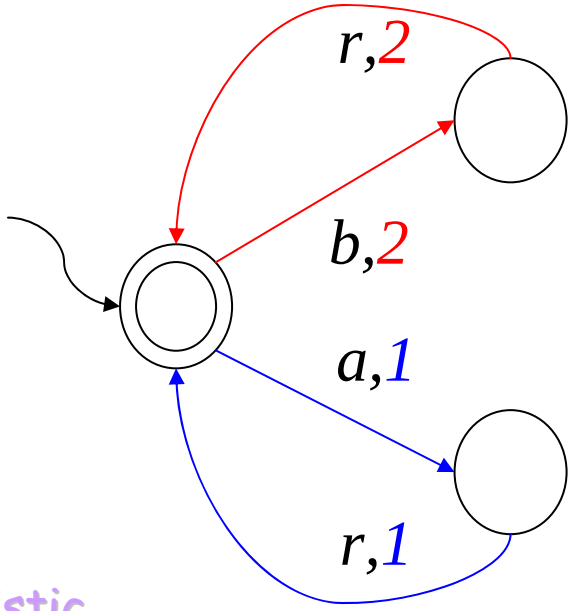
Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

Finite TS for services



- Deterministic
- Transitions labeled by actions
- Output on state to signal when final

Finite TS for composition



- Deterministic
- Transitions labeled by actions and services
- Output on transition to signal which service

Questions

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS

Università di Roma "La Sapienza"

Assume services of community and target service are finite TSs

- Can we always check composition existence?
- If a composition exists there exists one which is a finite TS?
- If yes, how can a finite TS composition by computed?

To answer we exploit PDL SAT

Answers

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

Reduce service composition synthesis to satisfability in (deterministic) PDL

- Can we always check composition existence?

  Yes, SAT in PDL is decidable in EXPTIME

- If a composition exists there exists one which is a finite TS?

  Yes, by the small model property of PDL

- How can a finite TS composition be computed?

  From a (small) model of the corresponding PDL formula

Structure of the PDL Encoding

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

$$\Phi = \mathit{Init} \wedge [u](\Phi_0 \wedge \bigwedge_{i=1,\ldots,n} \Phi_i \wedge \Phi_{aux})$$

Initial states of all services

PDL encoding of target service

PDL encoding of *i*-th component service

PDL additional domain-independent conditions

PDL encoding is polynomial in the size of the service TSs

PDL Encoding

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Target service $S_0 = (\Sigma, S_0, s^0_0, \delta_0, F_0)$ in PDL we define Φ_0 as the conjunction of:
  - $s \rightarrow \neg s'$ for all pairs of distinct states in S_0
    service states are pair-wise disjoint
  - $s \rightarrow \langle a \rangle T \wedge [a]s'$ for each $s' = \delta_0(s,a)$
    target service can do an a-transition going to state s'
  - $s \rightarrow [a] \bot$ for each $\delta_0(s,a)$ undef.
    target service cannot do an a-transition
  - $F_0 \equiv \vee_{s \in F0} s$
    denotes target service final states
- ...

PDL Encoding (cont.d)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Community services $S_i = (\Sigma, S_i, s^0_i, \delta_i, F_i)$ in PDL we define Φ_i as the conjunction of:
  - $s \rightarrow \neg s'$ for all pairs of distinct states in S_i
    Service states are pair-wise disjoint
  - $s \rightarrow [a](moved_i \wedge s' \vee \neg moved_i \wedge s)$ for each $s' = \delta_i(s,a)$
    if service moved then new state, otherwise old state
  - $s \rightarrow [a](\neg moved_i \wedge s)$ for each $\delta_i(s,a)$ undef.
    if service cannot do a, and a is performed then it did not move
  - $F_i \equiv \vee_{s \in Fi} s$
    denotes community service final states
- ...

PDL Encoding (cont.d)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Additional assertions Φ_{aux}
  - $\langle a \rangle T \rightarrow [a] \bigvee_{i=1,\ldots,n} moved_i$ for each action a

    at least one of the community services must move at each step
  - $F_0 \rightarrow \bigwedge_{i=1,\ldots,n} F_i$

    when target service is final all comm. services are final
  - $Init \equiv s^0_0 \bigwedge_{i=1\ldots.n} s^0_i$

    Initially all services are in their initial state

PDL encoding: $\Phi = Init \wedge [u](\Phi_0 \wedge_{i=1,\ldots,n} \Phi_i \wedge \Phi_{aux})$

Results

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS

Università di Roma "La Sapienza"

Thm: Composition exists iff PDL formula Φ SAT

From composition labeling of the target service one can build a tree model of the PDL formula and viceversa

Information on the labeling is encoded in predicates moved$_i$

$\Rightarrow$ Composition existence of services expressible as finite TS is decidable in EXPTIME

Results on TS Composition

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

Thm: If composition exists then finite TS composition exists.

From a small model of the PDL formula Φ, one can build a finite TS machine

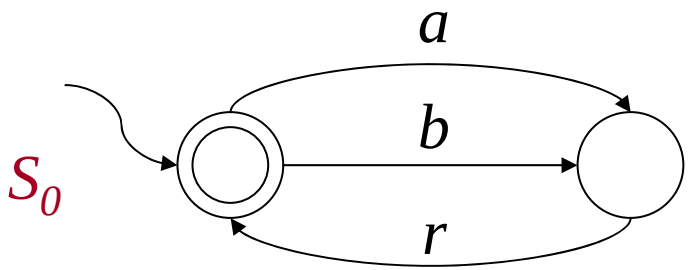
Information on the output function of the machine is encoded in predicates moved$_i$

$\Rightarrow$ finite TS composition existence of services expressible as finite TS is decidable in EXPTIME

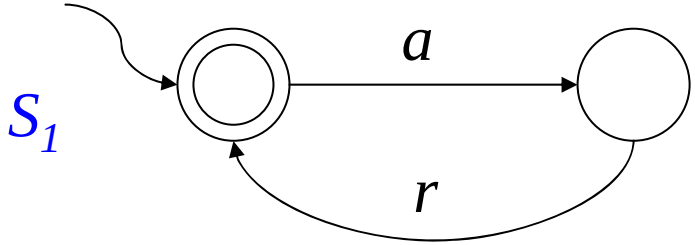
Example (1)

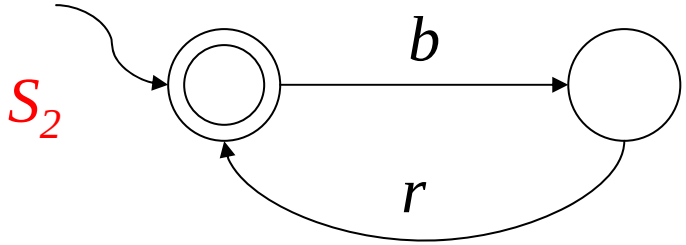
Dipartimento di Informatica e Sistemistica "Antonio Ruberti" — Università di Roma "La Sapienza"

Target service



Community services





DPDL

...

...

...

$s_0^0 \wedge s_1^0 \wedge s_2^0$

$\langle a \rangle T \rightarrow [a] (moved_1 \vee moved_2)$

$\langle b \rangle T \rightarrow [b] (moved_1 \vee moved_2)$

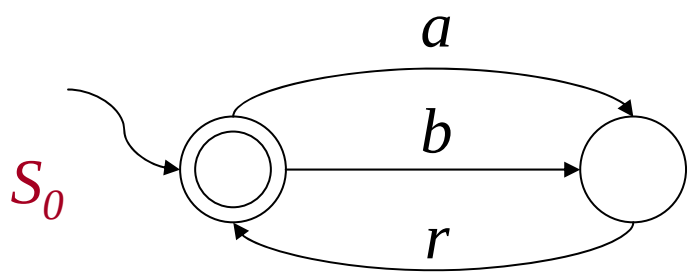
$\langle r \rangle T \rightarrow [r] (moved_1 \vee moved_2)$

$F_0 \rightarrow F_1 \wedge F_2$

Example (2)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

Target service



$s_0^0 \rightarrow \neg\, s_0^1$

$s_0^0 \rightarrow \langle a \rangle\, T \wedge [a]\, s_0^1$

$s_0^0 \rightarrow \langle b \rangle\, T \wedge [b]\, s_0^1$

$s_0^1 \rightarrow \langle r \rangle\, T \wedge [r]\, s_0^0$

$s_0^0 \rightarrow [r] \bot \wedge [r]\, s_0^0$

$s_0^1 \rightarrow [a] \bot$

$s_0^1 \rightarrow [b] \bot$

$F_0 \equiv s_0^0$

...

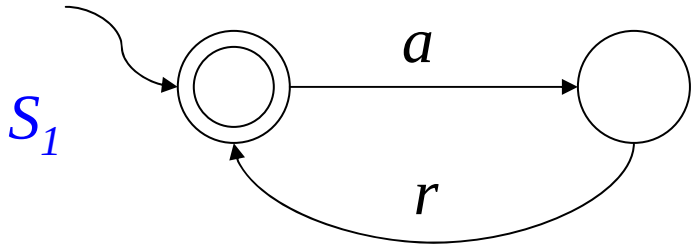
...

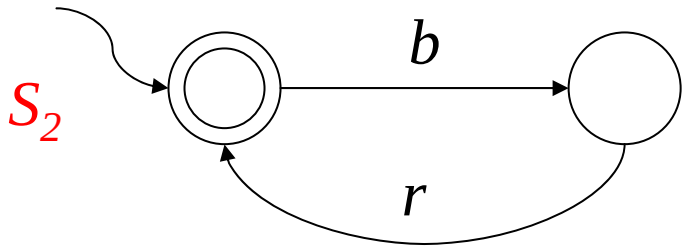
...

Example (3)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" — Università di Roma "La Sapienza"

Community services





...

$s_1^0 \rightarrow \neg\, s_1^1$

$s_1^0 \rightarrow [a]\,(moved_1 \wedge s_1^1 \vee \neg\, moved_1 \wedge s_1^0)$

$s_1^0 \rightarrow [r]\neg\, moved_1 \wedge s_1^0$

$s_1^0 \rightarrow [b]\neg\, moved_1 \wedge s_1^0$

$s_1^1 \rightarrow [a]\neg\, moved_1 \wedge s_1^1$

$s_1^1 \rightarrow [b]\neg\, moved_1 \wedge s_1^1$

$s_1^1 \rightarrow [r]\,(moved_1 \wedge s_1^0 \vee \neg\, moved_1 \wedge s_1^0)$

$F_1 \equiv s_1^0$

$s_2^0 \rightarrow \neg\, s_2^1$

$s_2^0 \rightarrow [b]\,(moved_2 \wedge s_2^1 \vee \neg\, moved_2 \wedge s_2^0)$

$s_2^0 \rightarrow [r]\neg\, moved_2 \wedge s_2^0$

$s_2^0 \rightarrow [a]\neg\, moved_2 \wedge s_2^0$

$s_2^1 \rightarrow [b]\neg\, moved_2 \wedge s_2^1$

$s_2^1 \rightarrow [a]\neg\, moved_2 \wedge s_2^1$

$s_2^1 \rightarrow [r]\,(moved_2 \wedge s_2^0 \vee \neg\, moved_2 \wedge s_2^0)$

$F_2 \equiv s_2^0$

...

Example (4)

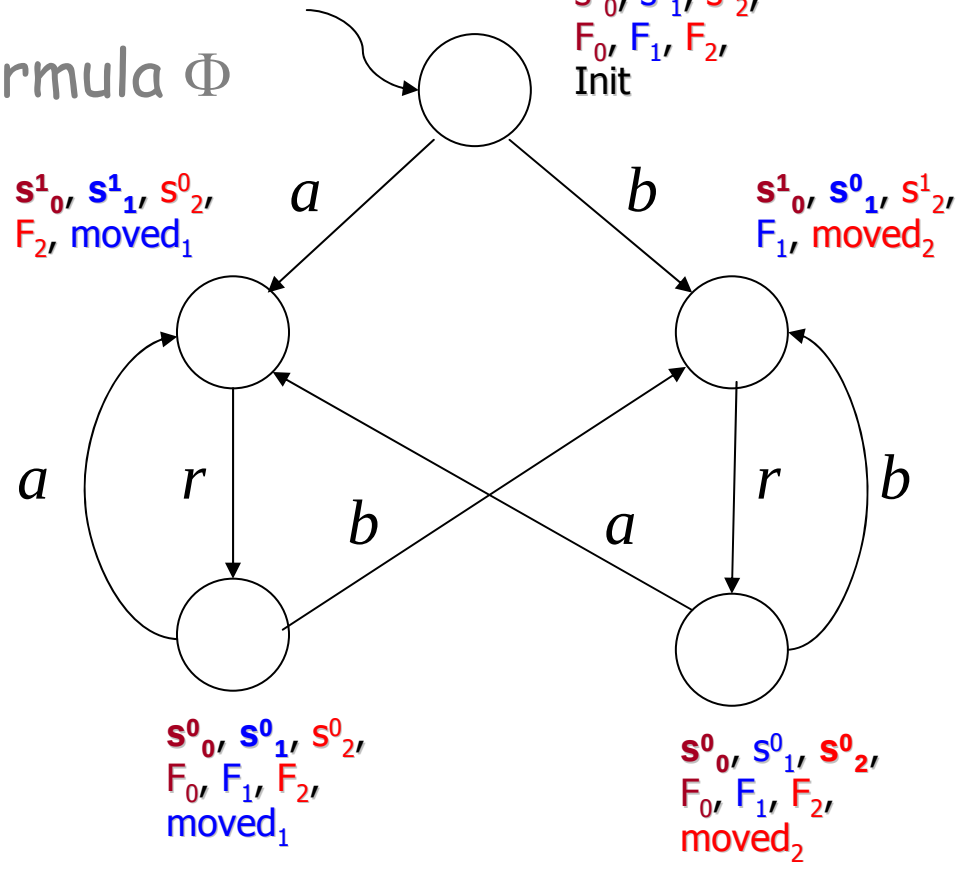
Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

Check: run SAT on PDL formula Φ

Example

Check: run SAT on PDL formula Φ

Yes $\Rightarrow$ (small) model

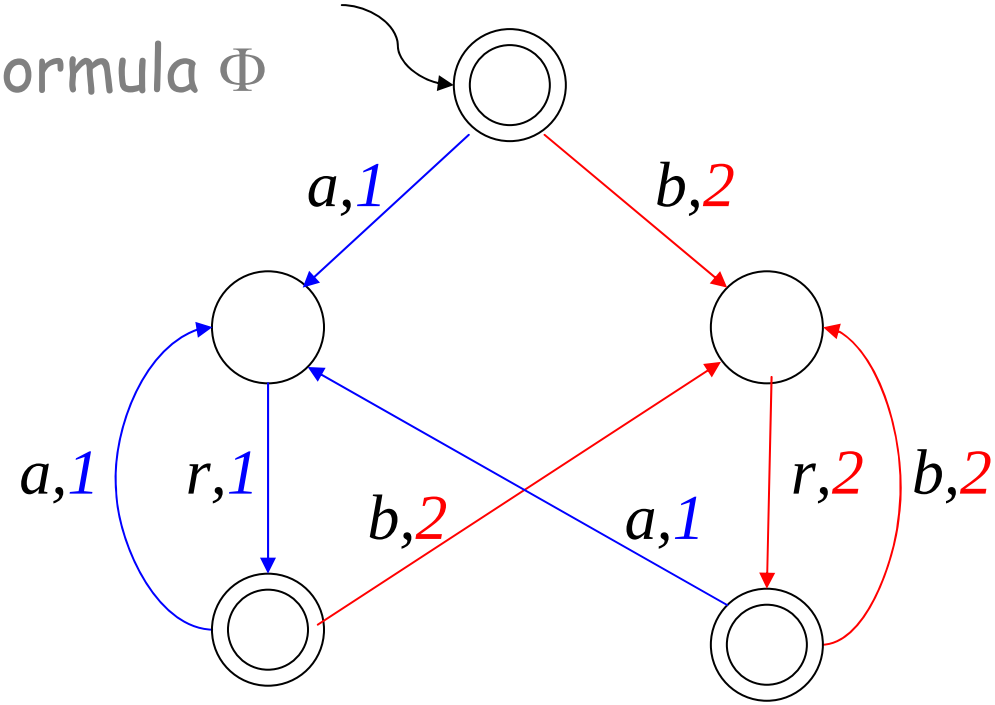


Example

Check: run SAT on PDL formula Φ

Yes $\Rightarrow$ (small) model

$\Rightarrow$ extract finite TS



Example

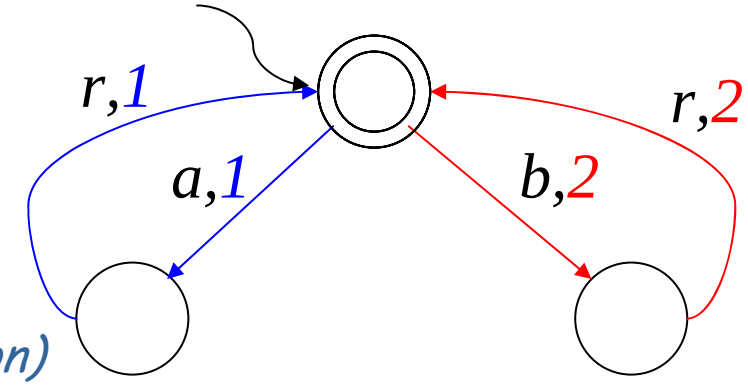
Check: run SAT on PDL formula Φ

Yes $\Rightarrow$ (small) model

$\Rightarrow$ extract finite TS

$\Rightarrow$ minimize finite TS

(similar to Mealy machine minimization)



Results on Synthesizing Composition

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS

Università di Roma "La Sapienza"

- Using PDL reasoning algorithms based on model construction (cf. tableaux), build a (small) model

  Exponential in the size of the PDL encoding/services finite TS

  Note: SitCalc, etc. can compactly represent finite TS, PDL encoding can preserve compactness of representation

- From this model extract a corresponding finite TS

  Polynomial in the size of the model

- Minimize such a finite TS using standard techniques (opt.)

  Polynomial in the size of the TS

  Note: finite TS extracted from the model is not minimal because encodes output in properties of individuals/states

Tools for Synthesizing Composition

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- In fact we use only a fragment of PDL in particular we use fixpoint (transitive closure) only to get the universal modality …
- … thanks to a tight correspondence between PDLs and Description Logics (DLs), we can use current highly optimized DL reasoning systems to do synthesis …
- … when the ability or returning models will be added …
- … meanwhile we have developed a prototype tool on this idea (see ESC – E-Service Composer: **http://sourceforge.net/projects/paride**)

Extensions (1)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- **Loose specification of target service**
  - target service "under-specified"
  - the client delegates the composer to resolve certain choices

  Note: angelic nondeterminism

- **Interaction between services without client involvement**
  - A component service may initiate an interaction with another component service to get to a certain state
  - Client specifies when during the computation this is allowed and to what extent

  Note: again angelic nondeterminism

See ICSOC04

Extensions (2)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Università di Roma "La Sapienza"

- Incomplete specification of services of the community
  - Component services export their behavior as nondeterministic TS (also related to **data acquisition**) ...

    Note: diabolic nondeterminism
  - ...the transitions are not completely under the control of the client nor of the orchestrator ...
  - ... composition has to work in spite of choices made by component services
  - Two possible assumptions
    1. Orchestrator **can observe** state reached by component service
    2. Orchestrator **cannot fully observe** state reached by component service

    For 1, variants of the technique shown here still apply - work in progress

    For 2, most likely techniques based on automata on infinite trees are needed – see Vardi's work on process synthesis

References

Roman Group

- *See above*

Process Synthesis

[Vardi&Kupferman ICTL97] M. Vardi, O. Kupferman: Synthesis with Incomplete Information (full version of ICTL'97).

[Vardi CAV95] M. Vardi, An Automata-Theoretic Approach to Fair Realizability and Synthesis. CAV 1995.

[Vardi&Kupferman MFCS2000] M. Vardi, O. Kupferman: Mu-calculus Synthesis. MFCS 2000

[Kupferman &Vardi LCS01] Orna Kupferman, Moshe Y. Vardi: Synthesizing Distributed Systems. Logic in Computer Science, 2001.

Description Logics and relationship with Logic of Programs

[BCMNS2003] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, P. F. Patel-Schneider: The Description Logic Handbook: Theory, Implementation, and Applications Cambridge University Press 2003

[Calvanese etal HAR01] D. Calvanese, G. De Giacomo, M. Lenzerini, D. Nardi: Reasoning in Expressive Description Logics. Handbook of Automated Reasoning 2001: 1581-1634

Recent evolution of the Roman Approach: COLOMBO Project (joint work with Rick Hull)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS

Università di Roma "La Sapienza"

- Integrates ideas from
  - Hull's Conversation Model: Messages
  - OWL-S: "Semantics" & Data
  - Roman Model: Composition by "Synthesis"

into a single approach.

COLOMBO Project (1)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- A model that combines
  - Impact on "real world": OWL-S atomic process
    - Incorporate explicit database manipulations
  - Message passing: WSDL and other standards
  - Process model inside web service: FSM-based, as in Roman and Conversation models
- Composition synthesis results, similar in spirit to the Roman results
  - For our first results, willing to impose many restrictions
  - Hopefully, can generalize the results in subsequent research

COLOMBO Project (2)

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Real world modeled by keyed relations
- Atomic Processes modeled after OWL-S
  - Atomic Process can read/write selected world relations
- Web Service: Guarded automaton
  - Local store (with scalars)
  - Transitions have conditions based on local store
  - Transitions have actions:
    - Atomic Process
    - Send message
    - Receive message
  - Ports (reminiscent of WSDL)
- System: family of web services, with linkage
  - Instantaneous description, $(id, I) \vdash (id', I')$
  - Execution tree
- Focus on
  - Client request is the realization of a given target service
  - Web services drawn from pre-existing set (UDDI directory)
  - Client-tailored approach

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

Conclusions

Summary

Dipartimento di Informatica e Sistemistica "Antonio Ruberti" DIS
Università di Roma "La Sapienza"

- Review of relevant technologies
- Distinction between orchestration and choreography
- Relevance of transition systems for abstracting over technologies
- State of the art in service composition
- Automatic composition: a basic research view
- Transition systems and formal systems for verification and synthesis
- In-depth analysis of a particular approach

What to Bring Home

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
DIS
Università di Roma "La Sapienza"

- Composition is not Choreography

  Composition: we are starting to understand
  Choreography: still needs to be developed

- Relevance of Transition Systems

  Don't confuse with language-theoretic automata

- Relevance of Data Integration

  If you are involved with information-based services, study data integration literature
  If you are not involved, look at it anyway, e.g., GAV and LAV

- Relevance of Planning

  If you can afford to have clients not driving the control flow, planning has a lot to say
  If not, planning doesn't directly apply, but has a lot to say anyway (it is a rich area)

- Relevance of Process Verification and Synthesis

  Full potential still largely unexplored

Having clients driving the control flow of the synthesized process is a new kind of need that is completely specific of a service composition