Automated Web Service Composition in Practice:

from Composition Requirements Specification to Process Run.

Annapaola Marconi, Marco Pistore and Paolo Traverso
FBK-irst, Trento, Italy
`(marconi,pistore,traverso)@itc.it`

June 11, 2007 - YR-SOC 07

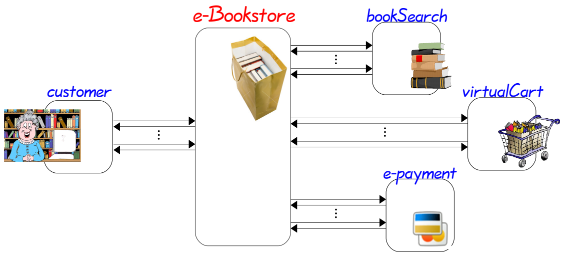
Outline

Web Service Composition

- **Web Services:** software platform-independent applications that export a description of their functionalities and make it available using **standard network technologies**
  - e.g. SOAP, WSDL, UDDI, WS-BPEL, WS-Transaction, ..
- **Web Service Composition:** combine existing services, available on the web, to define higher level functionalities

Web Service Composition

- **Web Services:** software platform-independent applications that export a description of their functionalities and make it available using **standard network technologies**
  - e.g. SOAP, WSDL, UDDI, WS-BPEL, WS-Transaction, ..
- **Web Service Composition:** combine existing services, available on the web, to define higher level functionalities

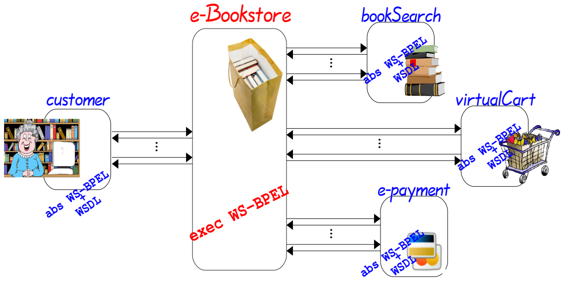


Automated Web Service Composition

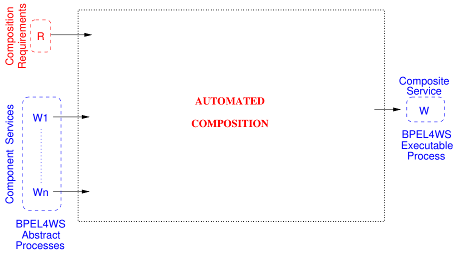
- **Automated Web Service Composition:** automatically synthesise a ready to run executable process that, interacting with a set of component services, satisfies given composition requirements.

Automated Web Service Composition

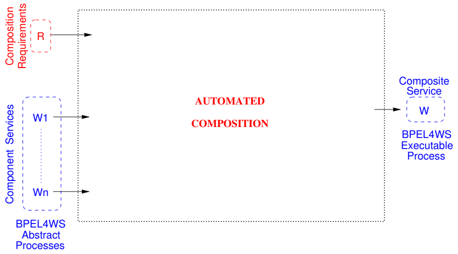
- **Automated Web Service Composition:** automatically synthesise a ready to run executable process that, interacting with a set of component services, satisfies given composition requirements.



The ASTRO Automated Composition Approach



The ASTRO Automated Composition Approach



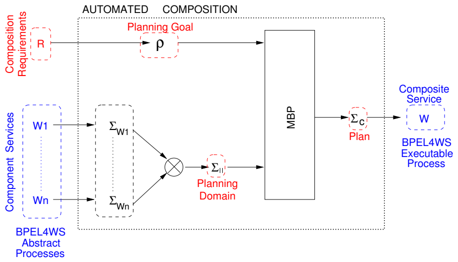
Challenges

⇒ Complex Composition Requirements

- control flow requirements: preferences and recovery conditions
- data flow requirements: constrain data manipulation and exchange

⇒ Component Services as Stateful Business Processes

The ASTRO Automated Composition Approach



- Intuitive and easy-to-define requirements specification languages
- Efficient automated composition techniques
  ⇒ Automated composition as a planning problem (Planning as Model Checking)

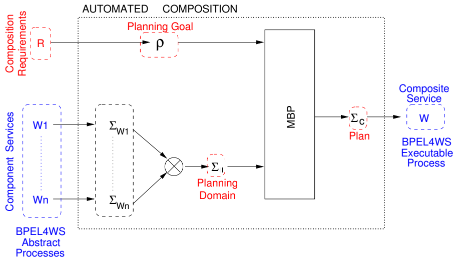
Challenges

⇒ Complex Composition Requirements

- control flow requirements: preferences and recovery conditions
- data flow requirements: constrain data manipulation and exchange

⇒ Component Services as Stateful Business Processes

The ASTRO Automated Composition Approach



- Intuitive and easy-to-define requirements specification languages
- Efficient automated composition techniques
  $\Rightarrow$ Automated composition as a planning problem (Planning as Model Checking)

Challenges

$\Rightarrow$ Complex Composition Requirements

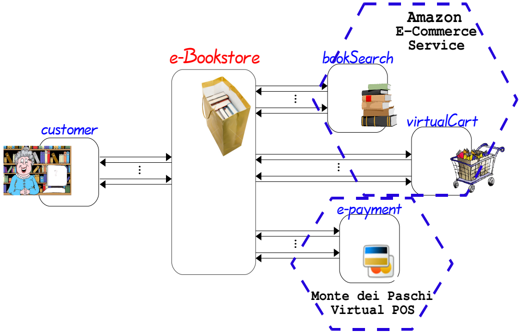
- control flow requirements: preferences and recovery conditions
- data flow requirements: constrain data manipulation and exchange

$\Rightarrow$ Component Services as Stateful Business Processes

$\Rightarrow$ **Practical applicability in real composition scenarios**

The Challenge

Evaluate the feasibility and efficiency of the ASTRO approach on a real composition scenario that entails a high level of complexity.



Outline

Amazon E-Commerce Service (ECS)

ECS aim

Exposes Amazon product information and e-commerce functionalities:

- searching for Amazon products (books, movies, music, restaurant, etc.)
- handling shopping carts
- inspecting customer contents (reviews, wish lists, listmania lists, etc..)
- inspecting vendor contents (customer feedbacks, etc..)

Amazon E-Commerce Service (ECS)

ECS aim

Exposes Amazon product information and e-commerce functionalities:

- searching for Amazon products (books, movies, music, restaurant, etc.)
- handling shopping carts
- inspecting customer contents (reviews, wish lists, listmania lists, etc..)
- inspecting vendor contents (customer feedbacks, etc..)

ECS specification

- WSDL document defining available operations, messages and their data structure
- several documents describing informally (natural language, flow charts, etc.):
  $\Rightarrow$ business workflows
  $\Rightarrow$ failures and non-nominal cases
  $\Rightarrow$ structure of each specific purpose message (movie-Search vs book-Search)

Amazon E-Commerce Service (ECS)

ECS aim

Exposes Amazon product information and e-commerce functionalities:

- searching for Amazon products (books, movies, music, restaurant, etc.)
- handling shopping carts
- inspecting customer contents (reviews, wish lists, listmania lists, etc..)
- inspecting vendor contents (customer feedbacks, etc..)

ECS specification

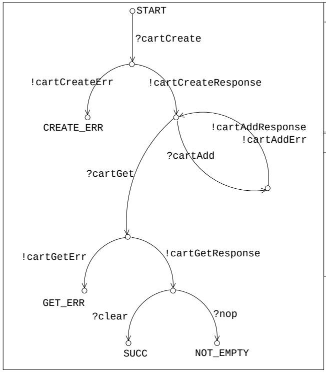
- WSDL document defining available operations, messages and their data structure
- several documents describing informally (natural language, flow charts, etc.):
  - ⇒ business workflows
  - ⇒ failures and non-nominal cases
  - ⇒ structure of each specific purpose message (movie-Search vs book-Search)

⇒ **Need for an explicit and formal specification of each business workflow**

⇒ Amazon Book-Search and Amazon Virtual-Cart

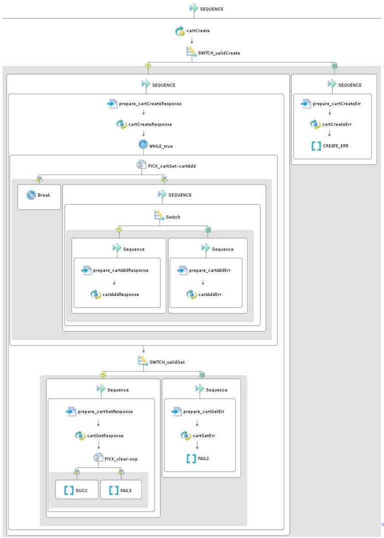
Amazon Virtual-Cart Service

Amazon Virtual Cart



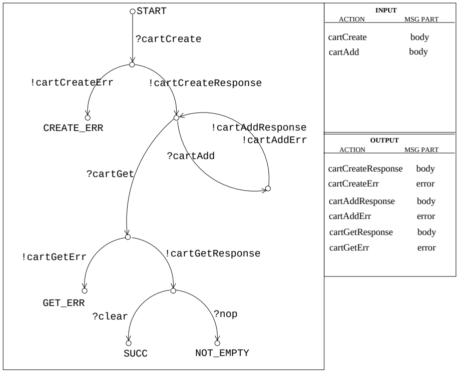
INPUT	
ACTION	MSG PART
cartCreate	body
cartAdd	body

OUTPUT	
ACTION	MSG PART
cartCreateResponse	body
cartCreateErr	error
cartAddResponse	body
cartAddErr	error
cartGetResponse	body
cartGetErr	error



Amazon Virtual-Cart Service

Amazon Virtual Cart

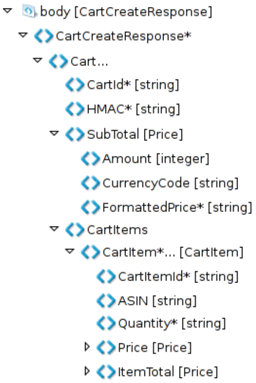


INPUT

ACTION	MSG PART
cartCreate	body
cartAdd	body

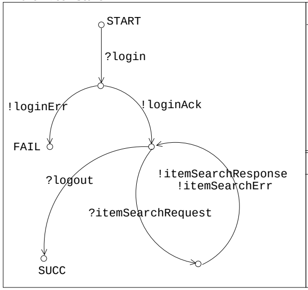
OUTPUT

ACTION	MSG PART
cartCreateResponse	body
cartCreateErr	error
cartAddResponse	body
cartAddErr	error
cartGetResponse	body
cartGetErr	error


```
body [CartCreateResponse]
  CartCreateResponse*
    Cart...
      CartId* [string]
      HMAC* [string]
      SubTotal [Price]
        Amount [integer]
        CurrencyCode [string]
        FormattedPrice* [string]
      CartItems
        CartItem*... [CartItem]
          CartItemId* [string]
          ASIN [string]
          Quantity* [string]
          Price [Price]
          ItemTotal [Price]
```

Amazon Book-Search Service

Amazon Book Search



INPUT

ACTION	MSG PART
login	customerId
itemSearchRequest	body

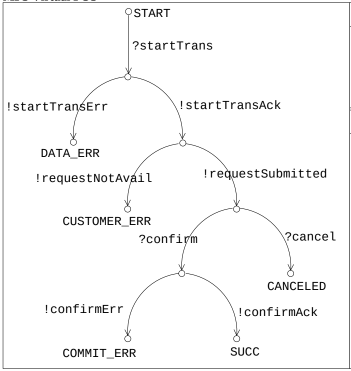
OUTPUT

ACTION	MSG PART
itemSearchResponse	body
itemSearchError	error

MPS Virtual Point of Sale (POS) Service

Models a real on-line payment service offered by an Italian bank (Monte dei Paschi di Siena).

MPS Virtual POS



INPUT

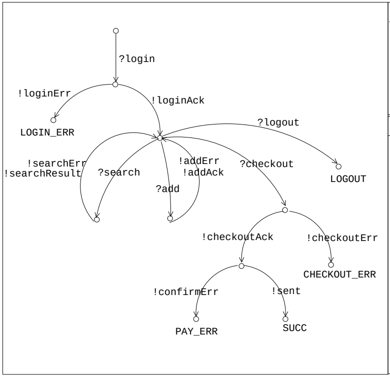
ACTION	MSG PART
startTrans	amount shopID shopEmail

OUTPUT

ACTION	MSG PART
startTransAck	paymentURL
startTransErr	errorDesc
requestSubmitted	transID
requestNotAvail	errorDesc
confirmAck	transDate transTime transAuthorization
confirmErr	errorDesc

e-Bookstore service customer interface

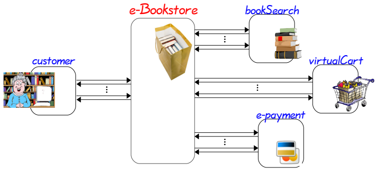
e−Bookstore Client



INPUT	
ACTION	MSG PART
login	customerId
search	keyword author publisher title
add	ASIN quantity

OUTPUT	
ACTION	MSG PART
searchResult	numItems ASIN detailPageURL author title publisher ISBN price
searchErr	error
addErr	error
addAck	subTotal
checkoutErr	error
checkoutAck	subTotal paymentURL
sent	transDate transTime transAuthorization
confirmErr	error

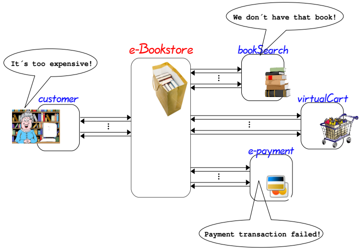
Control Flow Requirements



e-Bookstore goal

SELL BOOKS

Control Flow Requirements



e-Bookstore goal

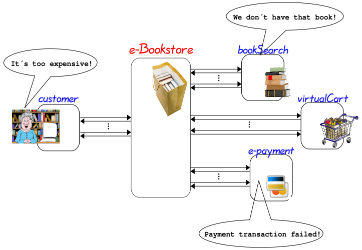
do whatever possible to

SELL BOOKS

if something goes wrong guarantee

NO SINGLE COMMITMENTS

Control Flow Requirements



e-Bookstore goal

do whatever possible to

SELL BOOKS

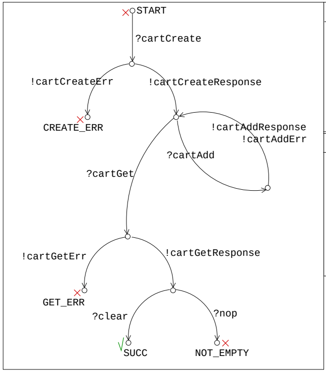
if something goes wrong guarantee

NO SINGLE COMMITMENTS

⇒ **Take into account the transactionality of each service within the overall composition.**

Control Flow Requirements

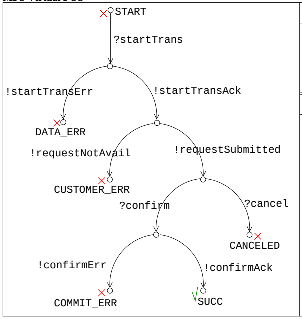
Amazon Virtual Cart



INPUT	
ACTION	MSG PART
cartCreate	body
cartAdd	body

OUTPUT	
ACTION	MSG PART
cartCreateResponse	body
cartCreateErr	error
cartAddResponse	body
cartAddErr	error
cartGetResponse	body
cartGetErr	error

MPS Virtual POS



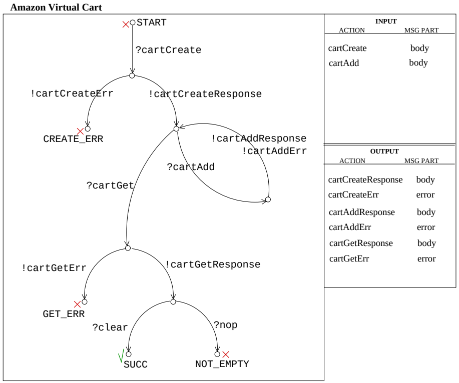
INPUT	
ACTION	MSG PART
startTrans	amount shopID shopEmail

OUTPUT	
ACTION	MSG PART
startTransAck	paymentURL
startTransErr	errorDesc
requestSubmitted	transID
requestNotAvail	errorDesc
confirmAck	transDate transTime transAuthorization
confirmErr	errorDesc

Semantic annotations:

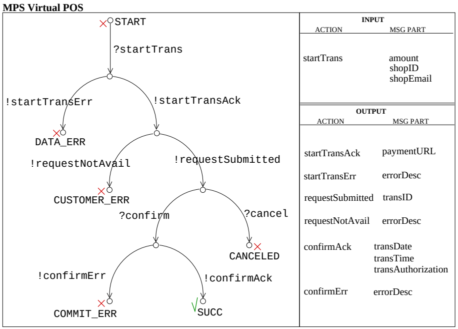
some states are marked as **successful** (✓),
other as **failing** (×).

Control Flow Requirements



INPUT	
ACTION	MSG PART
cartCreate	body
cartAdd	body

OUTPUT	
ACTION	MSG PART
cartCreateResponse	body
cartCreateErr	error
cartAddResponse	body
cartAddErr	error
cartGetResponse	body
cartGetErr	error



INPUT	
ACTION	MSG PART
startTrans	amount shopID shopEmail

OUTPUT	
ACTION	MSG PART
startTransAck	paymentURL
startTransErr	errorDesc
requestSubmitted	transID
requestNotAvail	errorDesc
confirmAck	transDate transTime transAuthorization
confirmErr	errorDesc

	eBS	ABS	AVC	VPOS
Primary	✓	✓	✓	✓
Secondary	×	✓/×	×	×

Specifying Data Flow Requirements

Constraining the flow of data among the Web Services participating in the composition.

Specifying Data Flow Requirements

Constraining the flow of data among the Web Services participating in the composition.

The idea

- Define the valid routings and manipulations of messages that the new composite service can perform
  - How incoming messages must be used, forwarded or manipulated, to obtain outgoing messages

Datanet Modeling Language

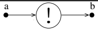
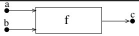
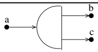
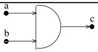
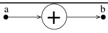
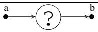
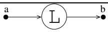
The data flow requirements are collected in a diagram called **data-net**

- **nodes**: sources/target of data on data
- **arcs**: flow or manipulation of data

Datanet Modeling Language

The data flow requirements are collected in a diagram called **data-net**

- **nodes**: sources/target of data on data
- **arcs**: flow or manipulation of data

	*forwarder*: simply forwards data received on the input node to the output node
	*function*: upon receiving data on all input nodes, computes the function result and forwards it to the output node
	*fork*: forwards data received on the input node to all the output nodes
	*merge*: forwards data received on some input node to the output node, preserving temporal order
	*cloner*: forwards, one or more times, data received from the input node to the output node
	*filter*: receives data on the input node and either forwards it to the output node or discards it
	*last*: forwards to the output node the last data received on the input node and discards all previous

e-Bookstore Data Flow Requirements

Amazon Book Search

INPUT
MESSAGE

login
itemSearchRequest

OUTPUT
MESSAGE

itemSearchResponse
itemSearchError

e−Bookstore Client

INPUT
MESSAGE

login
search
add

OUTPUT
MESSAGE

searchResult
searchErr
addErr
addAck
checkoutErr
checkoutAck
sent
confirmErr

Amazon Virtual Cart

INPUT
MESSAGE

cartCreate
cartAdd

OUTPUT
MESSAGE

cartCreateResponse
cartCreateErr
cartAddResponse
cartAddErr
cartGetResponse
cartGetErr

MPS Virtual POS

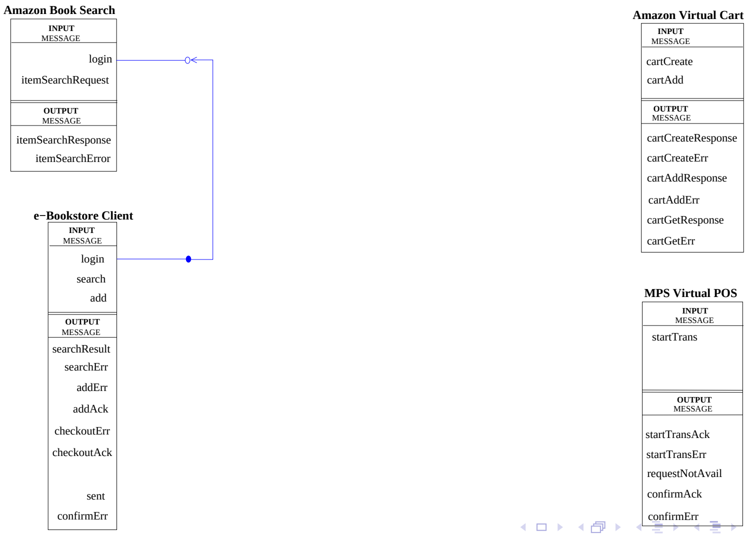
INPUT
MESSAGE

startTrans

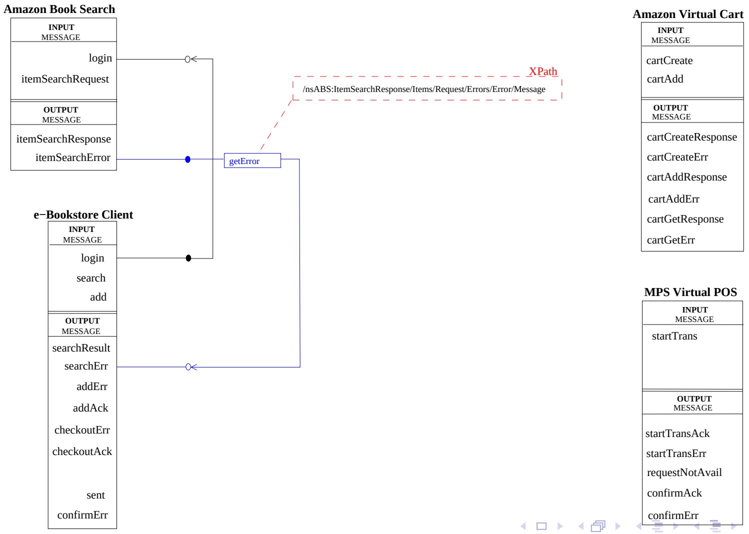
OUTPUT
MESSAGE

startTransAck
startTransErr
requestNotAvail
confirmAck
confirmErr

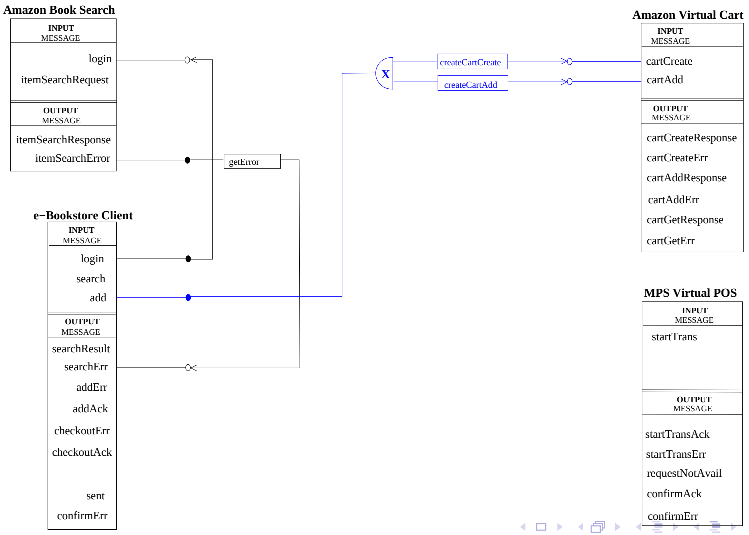
e-Bookstore Data Flow Requirements



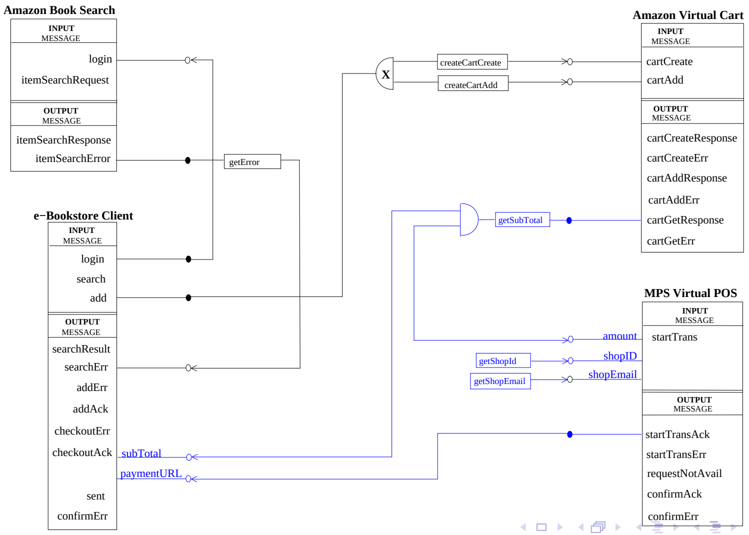
e-Bookstore Data Flow Requirements



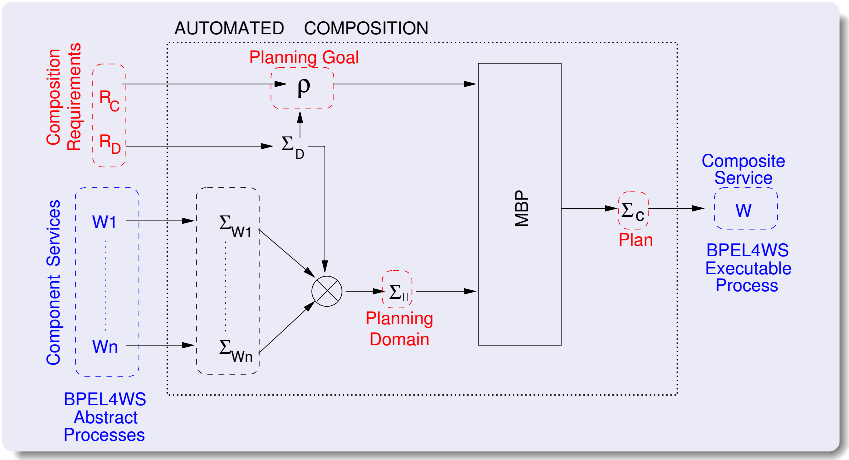
e-Bookstore Data Flow Requirements



e-Bookstore Data Flow Requirements



ASTRO Automated Composition Framework



ASTRO Automated Composition Framework

The e-Bookstore Composite Service

..cannot fit here!

	Time (sec.)		BPEL
	model construction	composition & emission	complex activities
e-Bookstore	2.7	605.2	177

Outline

Lessons Learnt and Future Works

⇒ **WS-BPEL extremely convenient to model component service protocols**

Lessons Learnt and Future Works

⇒ **WS-BPEL extremely convenient to model component service protocols**

⇒ **Efficiency of the automated composition techniques**

- composition techniques can scale up to real world scenarios
- hand writing e-Bookstore code: more than 20 hours
- the synthesised code is readable and easily modifiable

Lessons Learnt and Future Works

⇒ **WS-BPEL extremely convenient to model component service protocols**

⇒ **Efficiency of the automated composition techniques**

- composition techniques can scale up to real world scenarios
- hand writing e-Bookstore code: more than 20 hours
- the synthesised code is readable and easily modifiable

⇒ **Feasibility of the composition requirement specification**

- clear separation between control and data requirements helps a lot
- data and control flow requirements specification: approx. 2 hours

Lessons Learnt and Future Works

⇒ **Need for semantic annotations on data (SA-WSDL)**

- (Future Work) Semantic annotations can be used to automatically obtain (part of) the data-net

Lessons Learnt and Future Works

⇒ **Need for semantic annotations on data (SA-WSDL)**

- (Future Work) Semantic annotations can be used to automatically obtain (part of) the data-net

⇒ **Do we really need to specify the customer interface?**

- (Ongoing Work) Automatically obtain both the customer interface and the composite process
- (Ongoing Work) Iterative composition process: requirements refinement and re-composition

Lessons Learnt and Future Works

⇒ **Need for semantic annotations on data (SA-WSDL)**

- (Future Work) Semantic annotations can be used to automatically obtain (part of) the data-net

⇒ **Do we really need to specify the customer interface?**

- (Ongoing Work) Automatically obtain both the customer interface and the composite process
- (Ongoing Work) Iterative composition process: requirements refinement and re-composition

⇒ **How to deal with "Plan not found" ?**

- (Future Work) Automated requirements relaxation
- (Future Work) Apply verification techniques on the composition domain

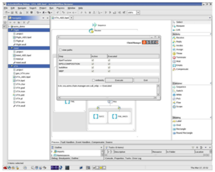
The ASTRO Project

The presented WS composition approach has been implemented within the **ASTRO toolset**

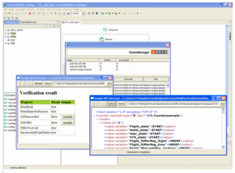
astro Supporting *the* Composition *of* Distributed Business Processes

www.astroproject.org

AUTOMATED COMPOSITION



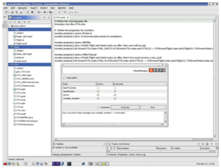
FORMAL VERIFICATION



Verification result

Property	Result	Sample
Deadlock	true	
SimultaneousSuccess	true	
AllSucceeded	false	sample
AllOffer	false	sample
OfferToAvail	true	
InconsistentFinalStates	true	

RUN-TIME MONITORING



The end

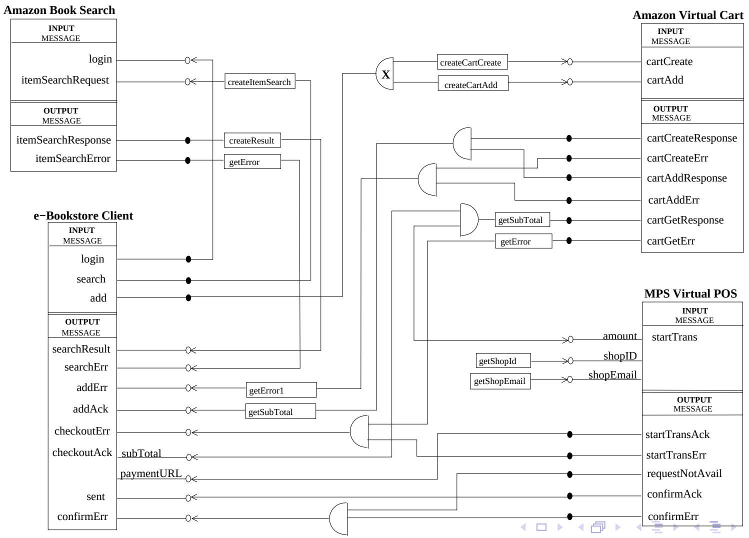
Thank you :)

Questions?

References

- **Automated Composition of Web Services by Planning in Asynchronous Domains.** M. Pistore and P. Traverso and P. Bertoli. (ICAPS 05)
- **Automated Synthesis of Composite BPEL4WS Web Services.** M. Pistore and P. Traverso and P. Bertoli and A.Marconi. (ICWS 05)
- **Automated Composition of Web Services by Planning at the Knowledge Level.** M. Pistore and A. Marconi and P. Traverso and P. Bertoli. (IJCAI 05)
- **Specifying Data-Flow Requirements for the Automated Composition of Web Services.** A. Marconi and M. Pistore and P. Traverso. (SEFM 06)
- **Implicit vs. Explicit Data-Flow Requirements in Web Service Composition Goals.** A. Marconi and M. Pistore and P. Traverso. (ICSOC 06)

e-Bookstore Data Flow Requirements



Data Requirements as STS

A data-net defines constraints on the possible operations that the composite process can perform on messages.

Data Requirements as STS

A data-net defines constraints on the possible operations that the composite process can perform on messages.

We assume that, in the new composite process, there exists a variable for each connection node in the data-net:

Data Requirements as STS

A data-net defines constraints on the possible operations that the composite process can perform on messages.

We assume that, in the new composite process, there exists a variable for each connection node in the data-net:

- variables associated to external connection nodes are those used by the new composite process to store received messages and to prepare the messages to be sent

Data Requirements as STS

A data-net defines constraints on the possible operations that the composite process can perform on messages.

We assume that, in the new composite process, there exists a variable for each connection node in the data-net:

- variables associated to external connection nodes are those used by the new composite process to store received messages and to prepare the messages to be sent
- variables associated to internal connection nodes are those used to manipulate messages by means of internal functions and assignments

Data Requirements as STS

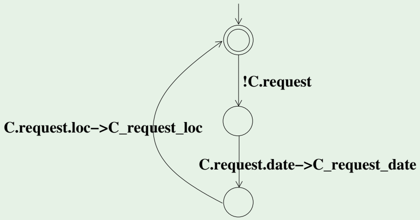
For each output operation of a component service in the data-net we define a STS which represents the sending of the message (as an output action) and the storing of all message parts (as internal actions)

Data Requirements as STS

For each output operation of a component service in the data-net we define a STS which represents the sending of the message (as an output action) and the storing of all message parts (as internal actions)

Example

For the output operation C.request with message parts date and loc we define the following STS:



Data Requirements as STS

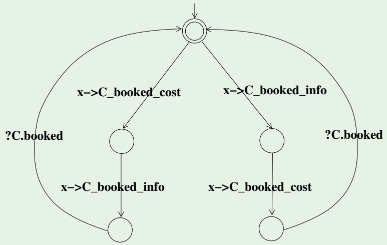
For each input operation of a component service in the data-net we define a STS which represents the storing of all message parts (as internal actions) and the reception of the message (as an input action).

Data Requirements as STS

For each input operation of a component service in the data-net we define a STS which represents the storing of all message parts (as internal actions) and the reception of the message (as an input action).

Example

For the input operation C.booked with message parts info and cost we define the following STS:



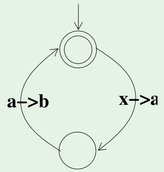
Data Requirements as STS

We define a STS for each data-flow element in the data-net:

Data Requirements as STS

We define a STS for each data-flow element in the data-net:

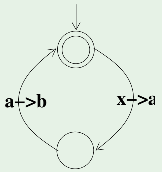
`id(a)(b)`



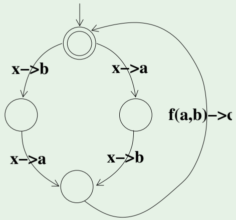
Data Requirements as STS

We define a STS for each data-flow element in the data-net:

`id(a)(b)`



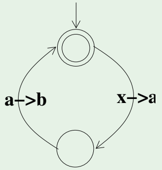
`oper[f](a,b)(c)`



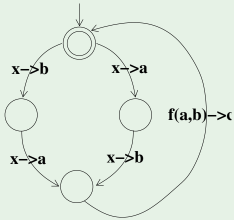
Data Requirements as STS

We define a STS for each data-flow element in the data-net:

`id(a)(b)`



`oper[f](a,b)(c)`



The STS $\Sigma_{\mathcal{D}}$, modeling the data-net, is the synchronized product of all the STSs corresponding to external connection nodes and data-flow elements.