



Faculty of Information Technologies

Basic concepts of Business Process Modeling



Outline

- Terminology
- A Brief History on Business Activity
- The basic approaches to Business Process Modeling
 - Formal x informal methods of specification and analysis
 - Verification of the informal defined methods
- Software tools for Business Process specification and analysis (ARIS, BP Studio, JARP, Woflan, Flowmake, Oracle BPEL Process Manager, etc.)
- Standards of BPM in context of BP Challenges
 - BPEL, BPMN, BPML, BPQL, BPSM, XPDL, XLANG, WSCI etc.
- Mapping of UML Business Processes to BPEL4WS
- Conclusions



Terminology of BPM

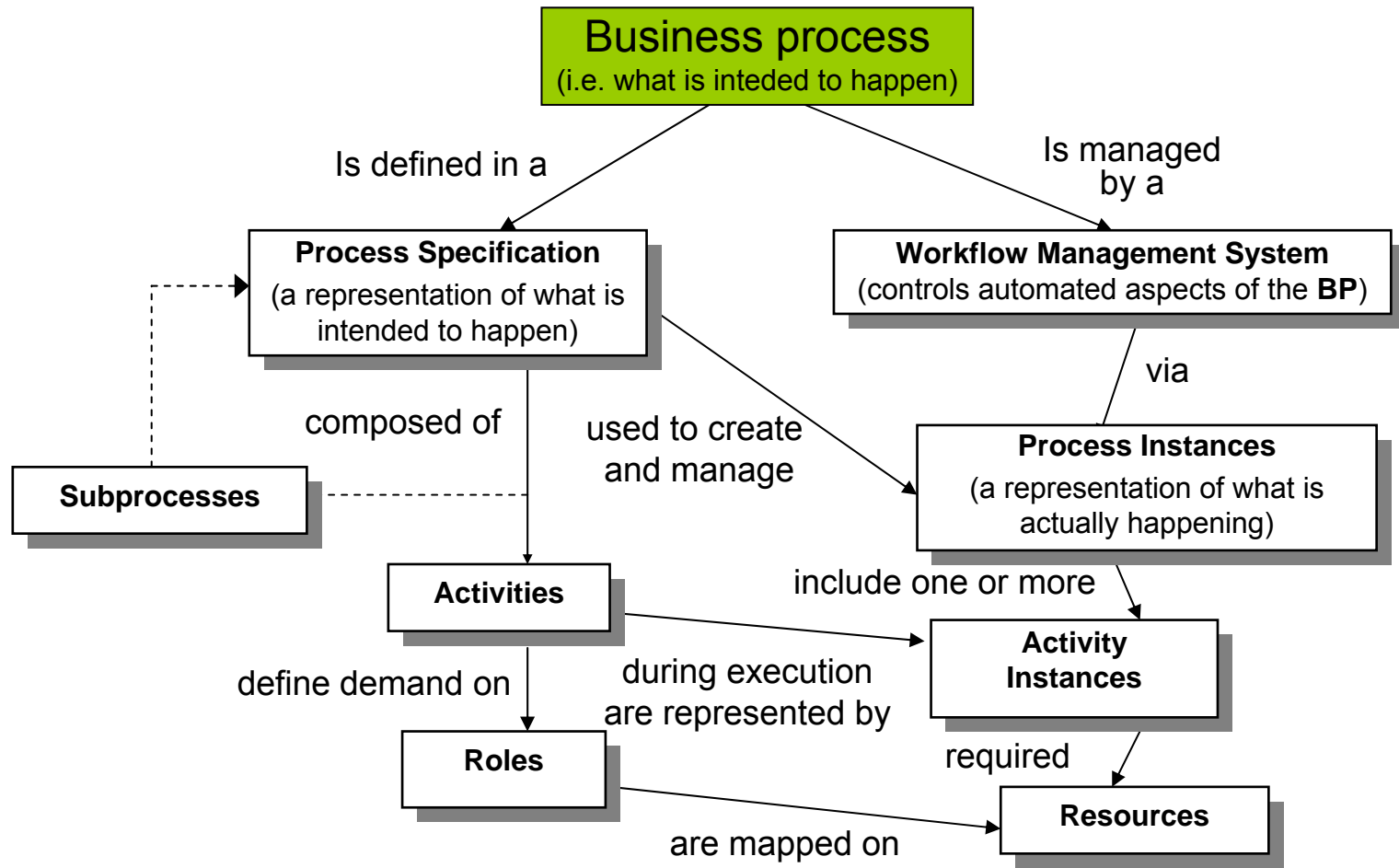
- **Business Process** is a set of one or more linked procedures or activities which collectively realize a business objective or policy goal, normally within the context of an organizational structure defining functional roles and relationships.
- **Business process model** – is the representation of a business process in a form which supports automated manipulation, such as modeling or enactment. The process definition consists of a network of activities and their relationships, criteria to indicate the start and termination of the process, and information about the individual activities, such as participants, associated data, etc.
- **Workflow** – is the automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules.



Terminology of BPM

- **Method** is well-considered (sophisticated) system of doing or arranging something.
- **Activity** – description of an activity representing 1 atomic step in process performance (manual, automated)
- **Process instance** – individual case of process performing
- **Activity instance** – represents an activity, realized with process performing, that means under the process instance
- **Role** – a set of mutual complementary skills assigned to single activities to provide their fulfilment
- **Ontology** – a formal specification of a shared conceptualization
- **Ontology of Process engineering** – definition of terms and their mutual relations describing concrete domain

Ontology of process engineering





Purpose of Business Modeling

- **Business Process Re-engineering (BPR)** - methods that support activities by which an enterprise reexamines its goals and how it achieves them, followed by a disciplined approach of business process redesign.
- **Enterprise Resource Planning (ERP)** - an information system that integrates all manufacturing and related applications for an entire enterprise. Business modeling is the first step in the software process of the ERP implementation.
- **Workflow Management (WFM)** – generic software systems used for definition, management, enactment and control of business processes.

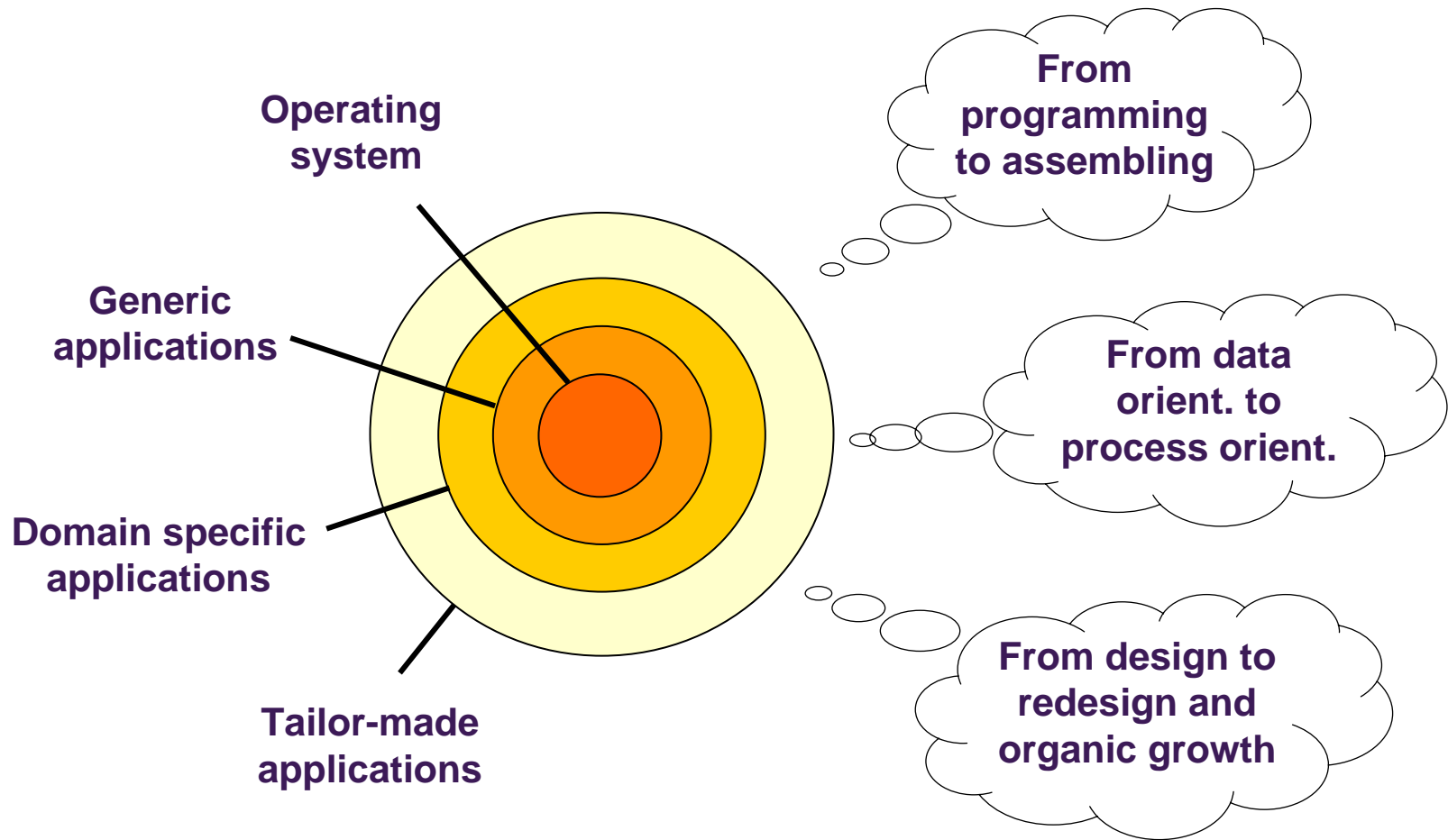


A Brief History on Business Activity

- 60's - Industry looked for quantity
- 70's - Production cost minimization
- 80's - Quality in production
- 90's - Quicker production (lead time)
- 21st century - Offers more services

Trends relevant for BPM

(Trends in Information systems)





Business Process Modeling

(Basic approaches)

- Abstract framework of model specification
- Functional specification (IDEF)
 - Process Specification by EPC using
 - Structural modeling by OOP – UML using
 - ISAC, DFD, SADT etc.
- Formal methods of specification and design
 - Petri Nets
 - WF-Nets
- Formalization a verification of informal methods
 - EPC Formalization
 - EPC Verification



Business Process Modeling

(Basic approaches)

- *Functional approach* – is focused on functions, their structuralization, inputs and outputs - IDEF
- *Behaviour specification approach* – is focused on managing aspects of process performance by way of events and conditions assignment, under that they can be individual activities realized - EPC
- *Structural approach* – is focused on static aspect of process – diagrams of UML method



Integration DEFinition

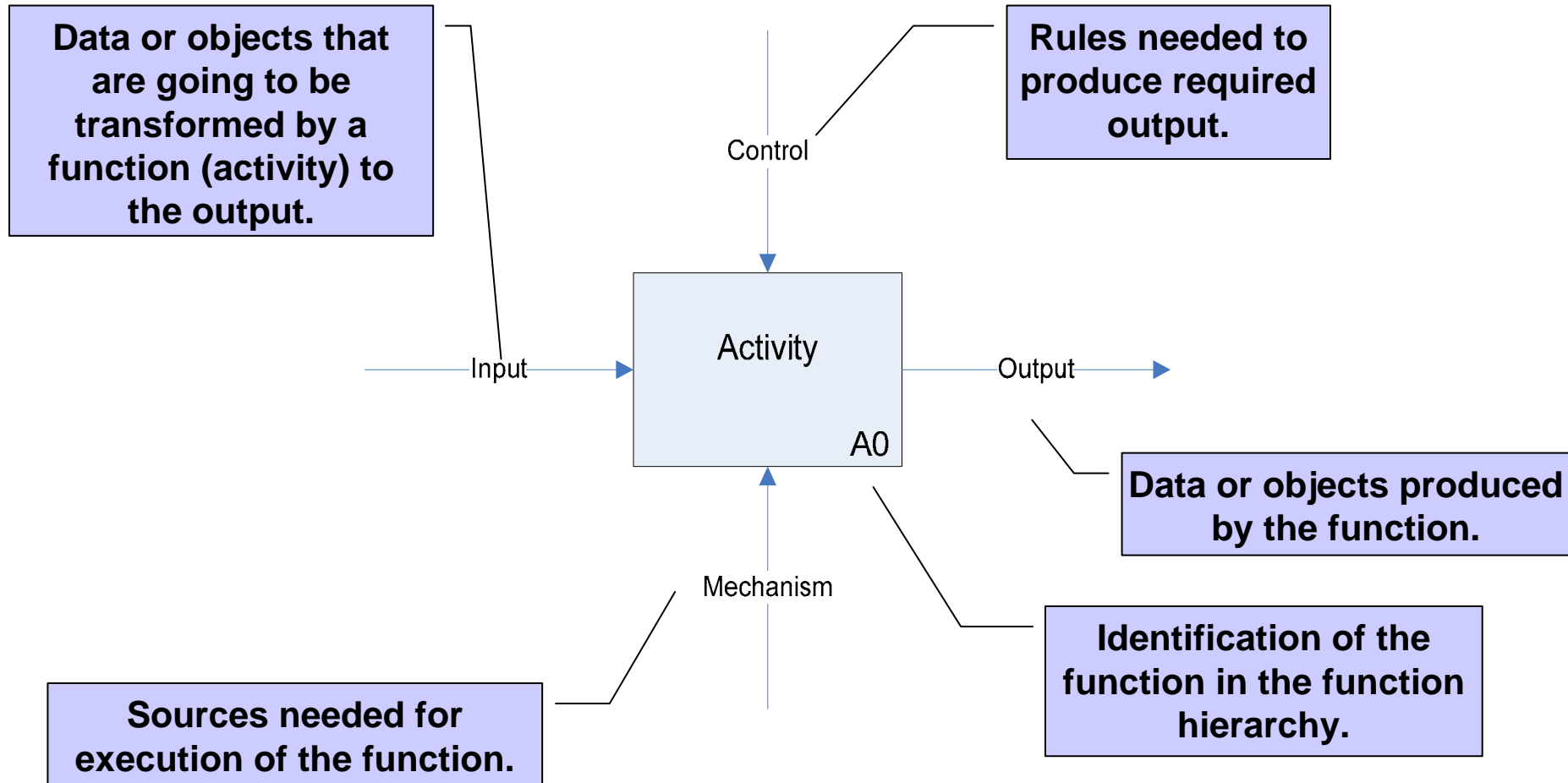
- *IDEF* (ICAM DEFinition, becoming Integrated DEFinition) is a software methodology and diagramming system developed by the US Department of Defense.
- the common name referring to classes of enterprise modeling languages (for modeling activities necessary to support system analysis, design, improvement or integration).
- used to produce a "*function model*". A function model is a structured representation of the functions, activities or processes within the modeled system or subject area.
- based on SADT (Structured Analysis and Design Technique). In its original form, IDEF0 includes both a definition of a graphical model. language (syntax/semantics) and a description of a comprehensive methodology for developing models.



IDEF Family

- IDEF Family of Methods:
 - *IDEF0*: for Function Modeling (purpose:description)
 - *IDEF1*: for Information Modeling (purpose:description)
 - *IDEF1x*: for Data Modeling (purpose:design)
 - *IDEF3*: for Process Modeling (purpose:description)
 - *IDEF4*: for Object-Oriented Design (purpose:design)
 - *IDEF5*: for Ontology Description Capture (purpose:description)
- ⇒ The complete list of IDEF method goes from IDEF0 to IDEF14 (from IDEF5 upwards mostly in development and IDEF7 missing from the list)

Basic IDEF0 Syntax





IDEF0 Pros and Cons

- **Positive aspects**
 - Method is well formalized. The syntax and semantics is well defined.
 - Function specification enables to analyze even complex processes.
 - Method is standardized by National Institute of Standards and Technology (USA).
- **Negative aspects**
 - IDEF0 is focused on functions and their decomposition. The time ordering is not explicitly expressed.
 - Complete specification of the process requires to employ other methods like IDEF1, 2 ... This issue makes resulting specification too complex.



Event-driven Process Chain

- EPC enables us to define how can be realized functions of a system by using of activities (and what sequences of these activities).
- EPC is based on connecting events and action to the sequences which collectively realize a business objective.
- Event is the precondition for the activity. New event (post condition) is generated when the activity is finished. It means that events defines the beginning and end of each activity.
- EPC diagrams are used in SAP R/3 (ERP/WFM) and ARIS (BPR).



EPC Diagram Elements

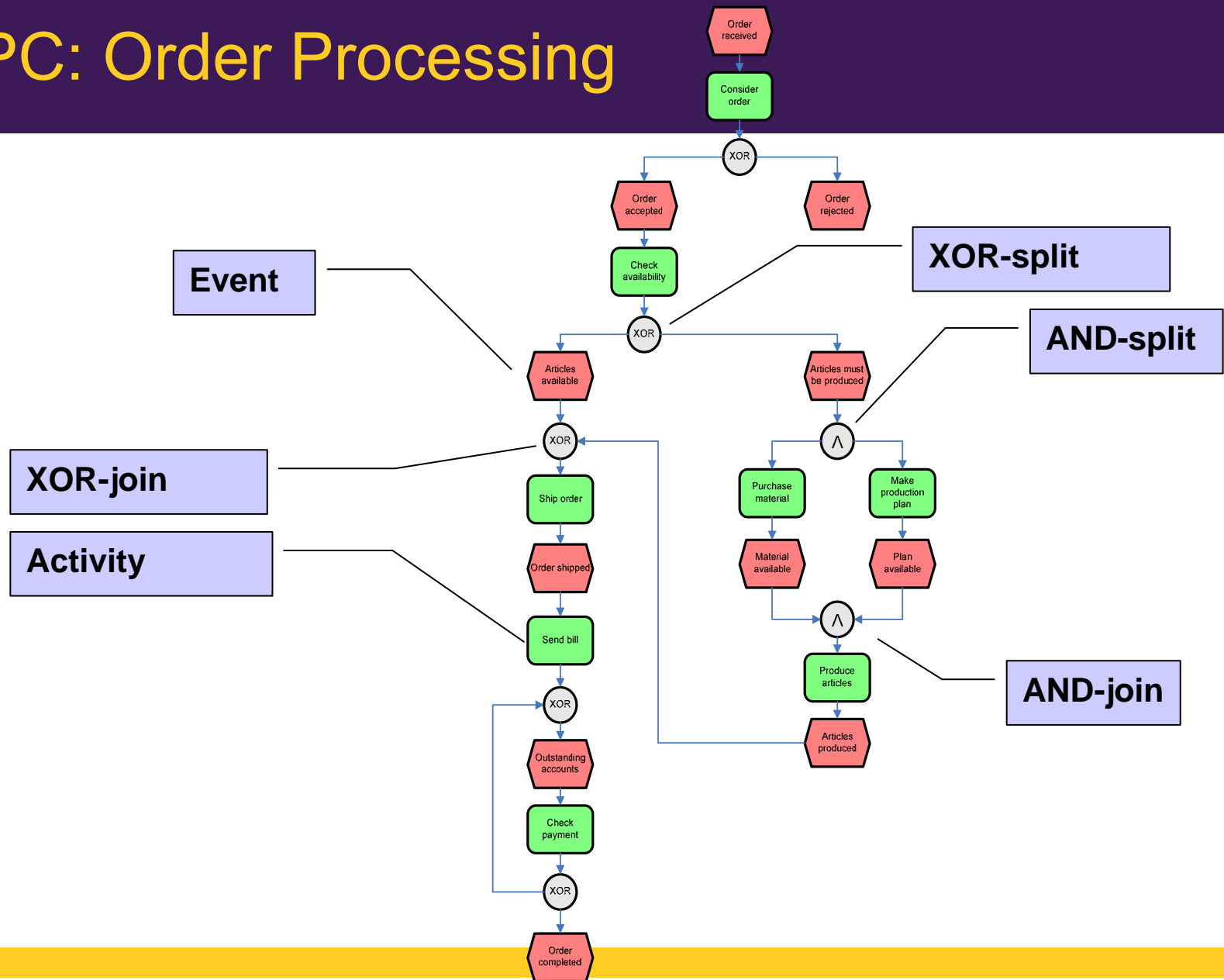
- **Activities** are the basic building blocks that define what should be completed within the process execution.
- **Events** specify situations before and/or after the activity is executed. It means that event may represent an output condition of the one activity and an input condition for the other activity at the same time.
- **Connectors** are used to link together activities and events. This is a way how the flow of control is defined. EPC uses the following three types of connectors: \wedge (AND), \vee (OR) and XOR (exclusive OR).



Semantics of Connectors

- **AND** is used either for *splitting* of the process to at least two concurrent process threads of execution or *joining* of concurrent threads to the one (synchronization point).
- **XOR** splits the process to just one optional thread of many possible ones.
- **OR** is used to split process to one, second or both possible threads of execution.

EPC: Order Processing





EPC Pros and Cons

- **Positive aspects**

- Method provides simple but powerful abstraction based on chaining of event and activities that enables to model complex processes.
- EPC is a part of widely accepted system like SAP and ARIS

- **Negative aspects**

- Language for EPC diagrams is not formally defines. Syntax and semantics is not precise enough (e.g. OR has no obvious semantics assigned).
- Missing formalism complicates portability of EPC between various software tools.



Structural modeling (OOM)

Application of the Object-oriented method to Business Process Modeling, especially of the Class diagram, that is introduced by UML.

Class diagram shows a collection of declarative (static) model elements, such as classes, types, and their contents and relationships.

The main elements:

- Classes describing active (*worker*) and passive (*entity*) objects
- Relations between classes specifying the communication way
- Relationship: Association, Aggregation, Generalization

The next possibilities of UML using:

- Use Case diagram – functional view
- Activity diagram – behavioral view



UML Pros and Cons

- **Positive aspects**

- UML provides a large number of diagrams enabling to capture every aspects of the system being modeled.
- The notation of UML is standardized and it is used by many SW tools dedicated to software system design.
- Since the primary focus of UML is to write software system blueprints it easy and straightforward to interconnect business modeling with the specification of information system.

- **Negative aspects**

- UML is considered as a semi-formal method. The semantics is not precisely defined. It might be a problem to verify complex processes.



Formal Methods

Formal methods include partly the specification domain, partly analysis of this domain and its verification. The main goal is not to describe a system, but to verify its desired attributes, too.

- Formal methods has not reached of massive enlargement (problems with applicability in extensive systems)
- The main domains of its using: systems needed errorless operation in critical situations (aviation, medicine..)

Categorization of formal methods of specification:

- **Function** – system describing from sight of desired behaviour by way of system model, it means by „abstract machine“ enabling simulate this behaviour.
- **Descriptive** – defining of desired system attributes only by declarative manner.



Formal Methods

- Formal methods include
 - Formal specification
 - Specification analysis and proof
 - Transformational development
 - Program verification
- Language for formal specification has to have precise and unambiguous syntax and semantics.

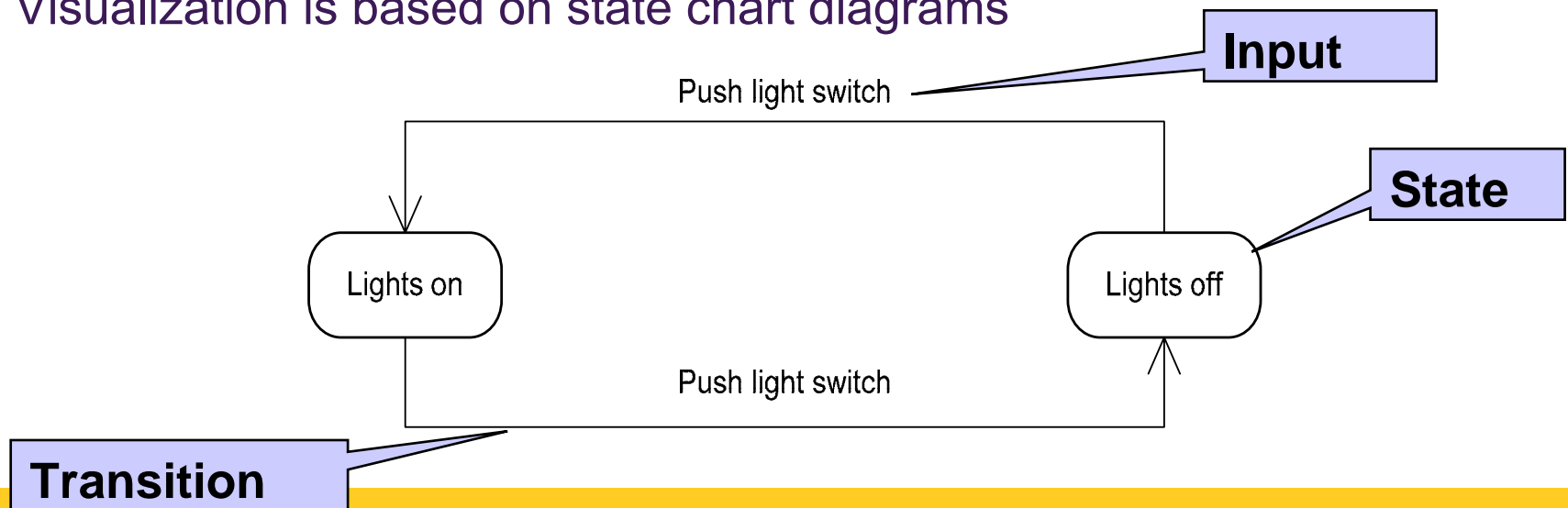


Mathematical Representation of Systems

- *Formal specifications* are expressed in a mathematical notation with precisely defined vocabulary, syntax and semantics.
- *Algebraic approach*
 - The system is specified in terms of its operations and their relationships.
- *Model-based approach*
 - The system is specified in terms of a state model that is constructed using mathematical constructs such as sets and sequences.

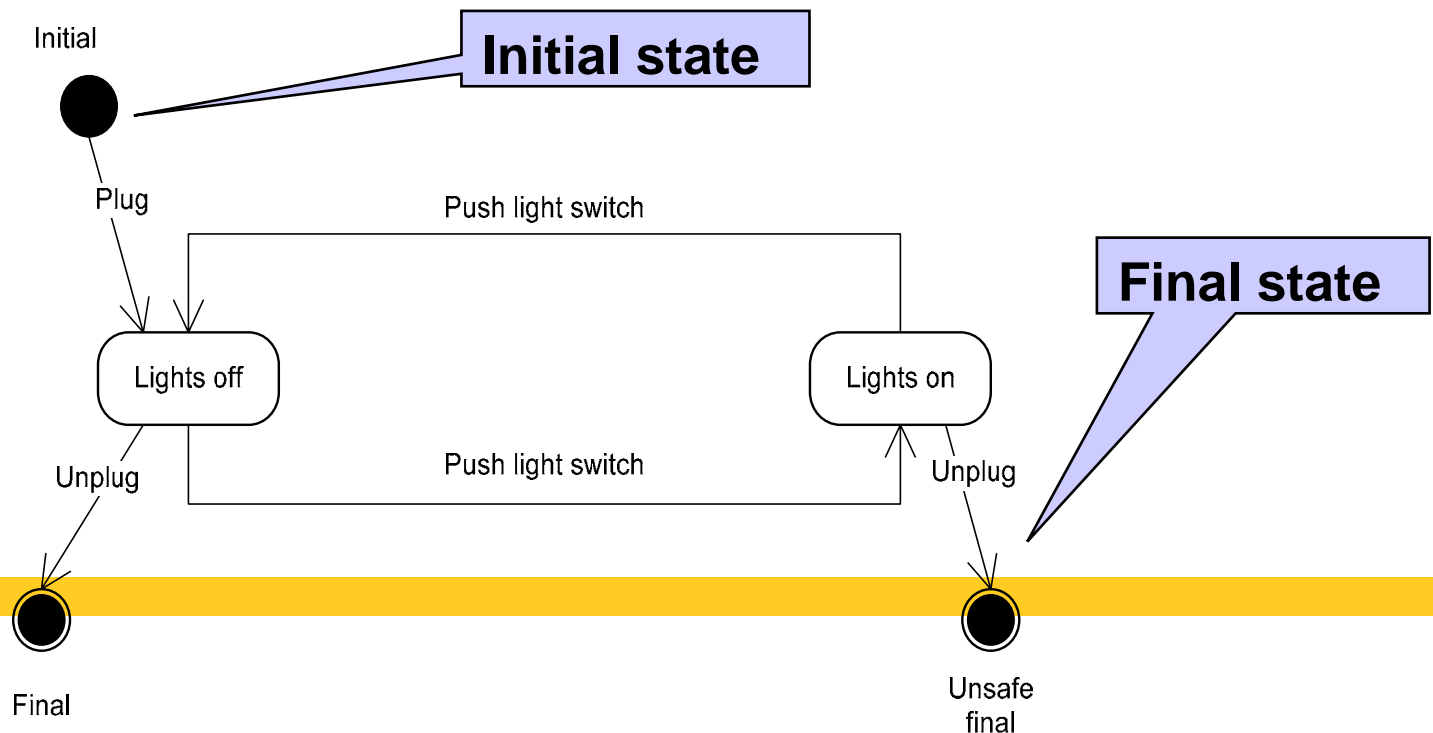
Finite Automata

- One of the most well-known approaches to describe systems behaviour are *The Finite automata*.
- Finite automata are defined by a five-tuple $FA = (Q, I, \delta, q_0, F)$ where:
 - Q – finite set of states
 - I – finite set of inputs
 - $\delta: Q \times I \rightarrow Q$ – state transition function
- Visualization is based on state chart diagrams

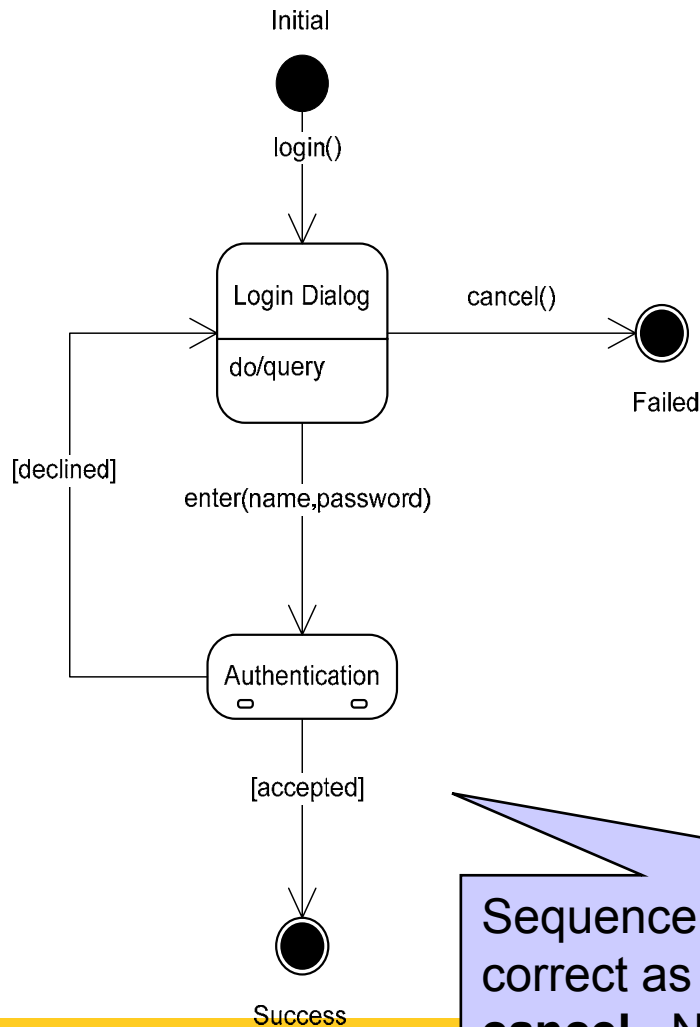


Finite Automata

- Initial and final states
 - q_0 - initial state
 - $F \subseteq Q$ - set of final states
- Sequence of inputs is accepted by automaton when a final state is reached from the initial state.



Example 1: Login behaviour



- $Q = \{Initial, Login\ Dialog, Authentication, Success, Failed\}$
- $I = \{login, enter, cancel, accepted, declined\}$
- $\delta_1(Initial, login) = Login\ Dialog$
 $\delta_2(Login\ Dialog, cancel) = Failed$
 $\delta_3(Login\ Dialog, enter) = Authentication$
 $\delta_4(Authentication, accepted) = Success$
 $\delta_5(Authentication, declined) = Login\ Dialog$
- $q_0 = Initial$
- $F = \{Success, Failed\}$

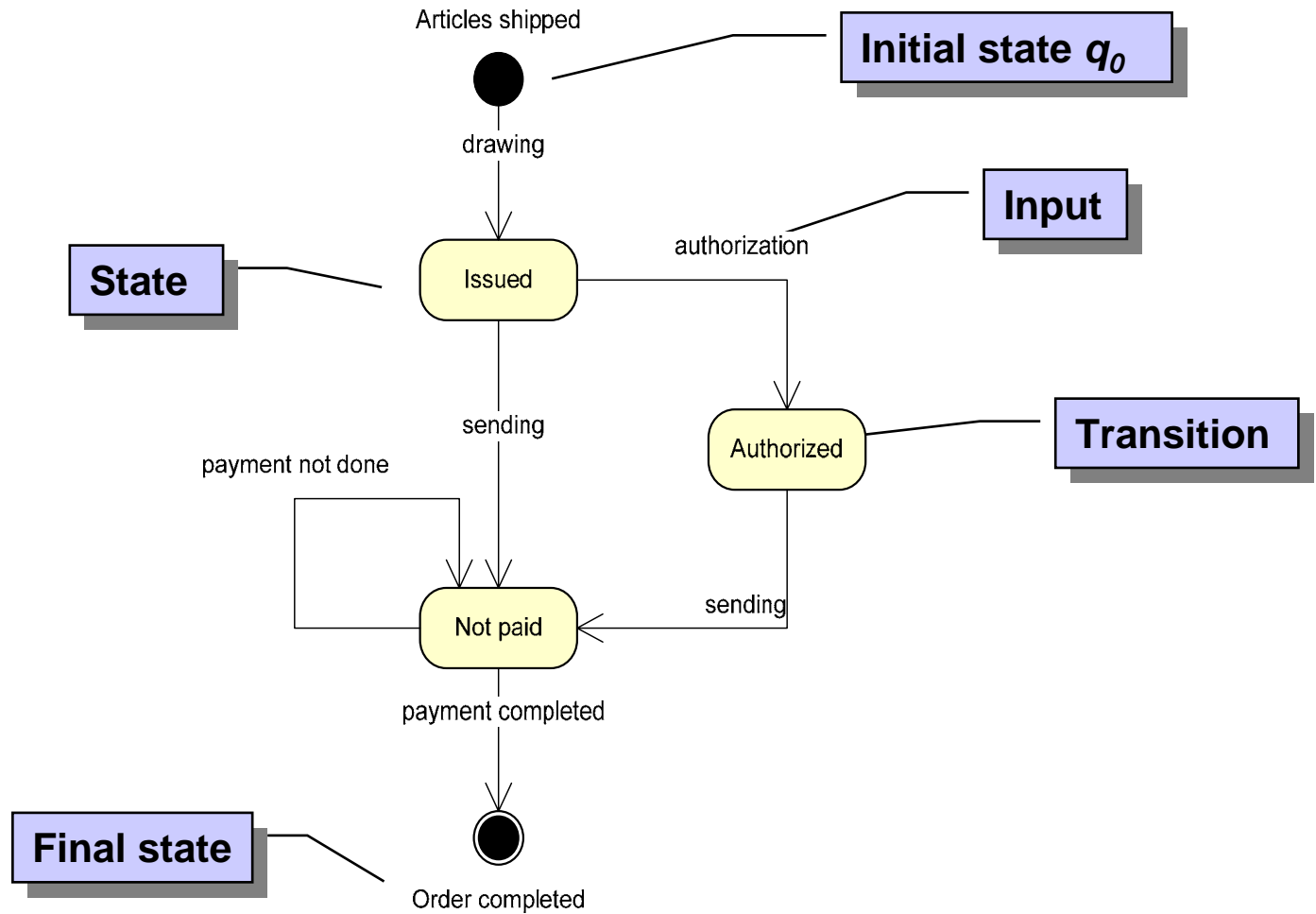
Sequence of inputs **login, enter, accepted** is correct as well as **login, enter, declined, cancel**. Not acceptable is **login, enter, cancel**.



Example 2: Invoice State

- $Q = \{\text{Articles shipped, Issued, Authorized, Not paid, Order completed}\}.$
- $I = \{\text{drawing, authorization, sending, payment not done, payment completed}\}.$
- $\delta(\text{Articles shipped, drawing}) = \text{Issued},$
 $\delta(\text{Issued, authorization}) = A,$
 $\delta(\text{Issued, sending}) = \text{Not paid},$
 $\delta(\text{Authorized, sending}) = \text{Not paid},$
 $\delta(\text{Not paid, payment not done}) = \text{Not paid},$
 $\delta(\text{Not paid, payment completed}) = \text{Order completed}.$
- $q_0 = \text{Articles shipped}.$
- $F = \{\text{Order completed}\}.$

State Diagram: Invoicing

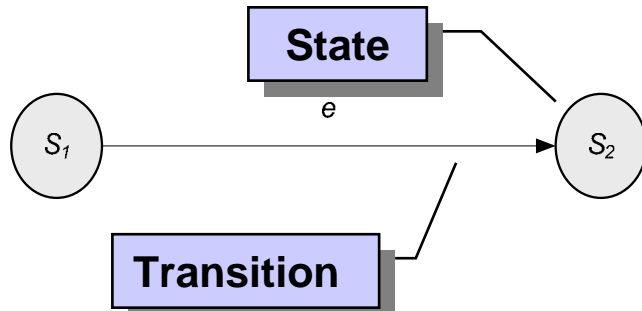




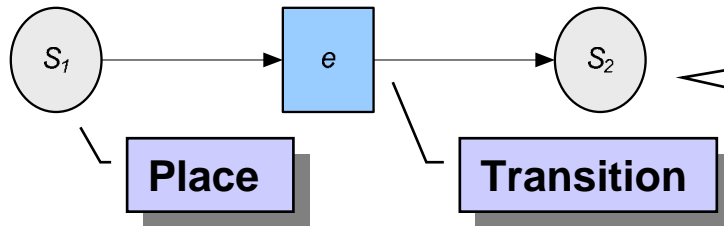
Petri Nets

- The Petri nets have resulted for purpose of enlargement of modelling possibilities of Finite Automata.
- It is a tool intended to processes modelling and analyzing.
- States are described by *places* and events by *transitions*.

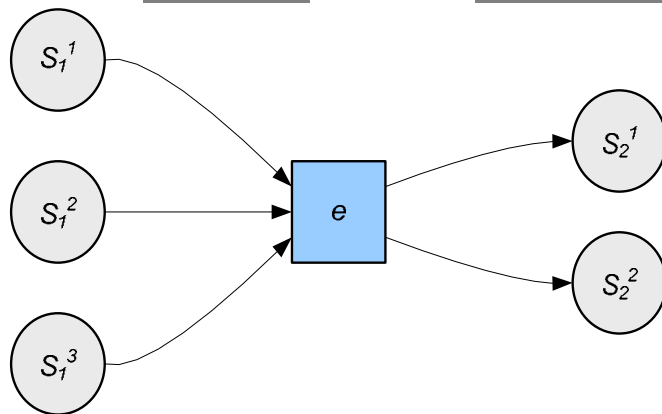
Petri Nets



Change of state modeled by Finite Automata



Change of state modeled by Petri Nets



Partial states modeled by Petri Nets

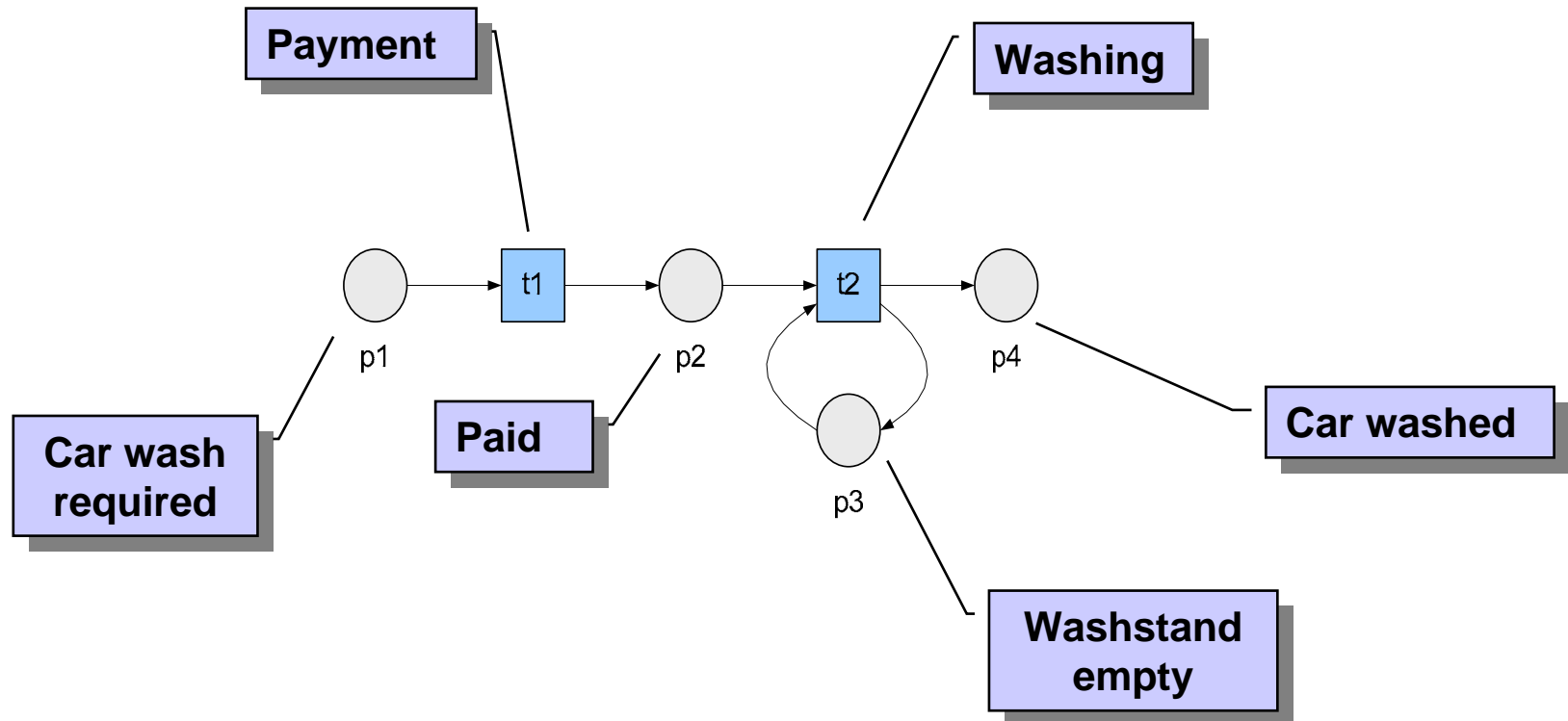


Formal definition of Petri Nets

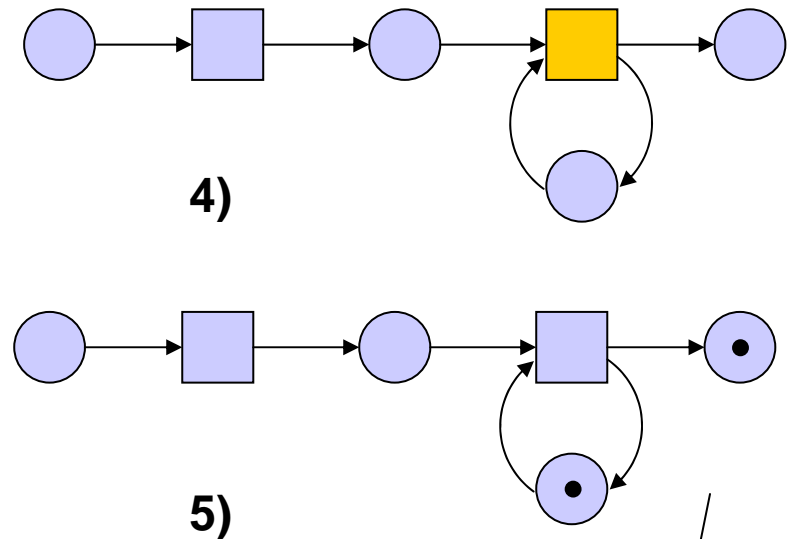
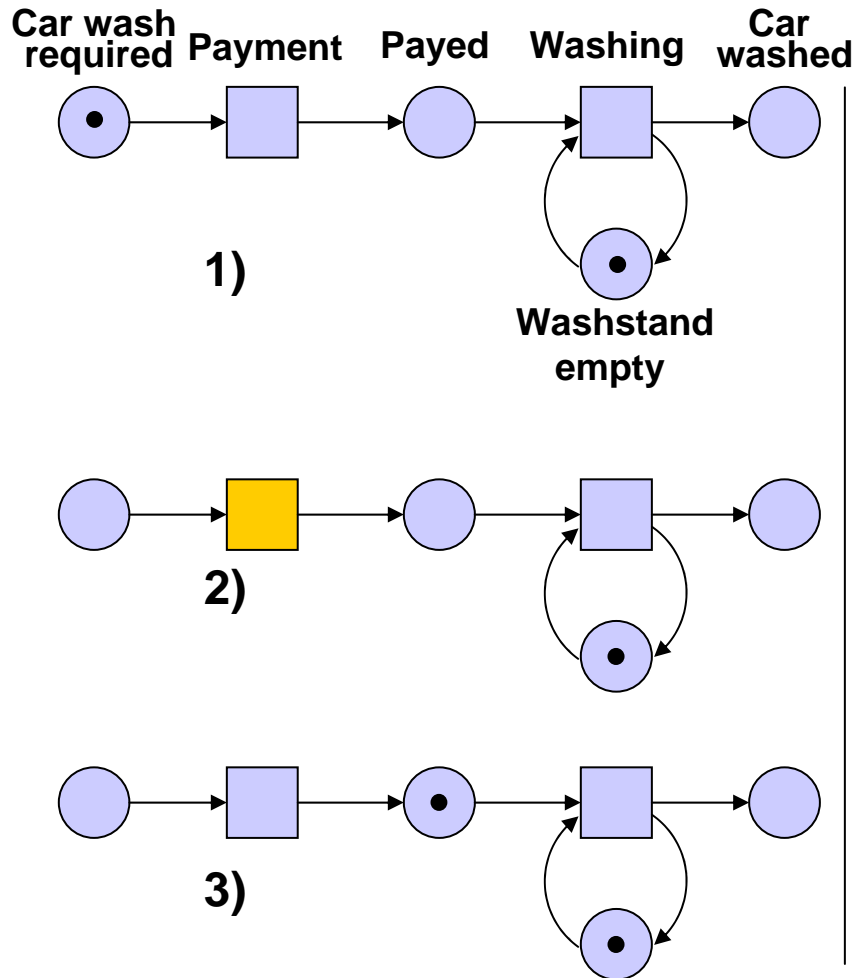
A **Petri net** is a triplet (P, T, F) :

- P is a finite set of places,
- T is a finite set of transitions ($P \cap T = \emptyset$)
- $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs (flow relation)
- A place p is called an *input place* of a transition t if there exists a directed arc from p to t .
- Place p is called an *output place* of transition t if there exists a directed arc from t to p .
- A **marking** of a $PN = (P, T, F)$ denoted by $M: P \rightarrow \mathbb{N}$ is a mapping which assigns a non-negative integer number of tokens to each place of the net.
- A marking M (distribution of tokens over places) is often referred as the state of a given Petri Net.
- The notations $\bullet t$ denotes the set of input places for a transition t .
- The notations $t\bullet$, $\bullet p$, $p\bullet$ have similar meanings.

Petri Net Model of Car wash



Simulation of Car washing:



Whole process is finished. A wash request was carried out and car wash is again empty.



Formal Specification of Wash-stand

- Wash-stand Petri Net
 - $P = \{p1, p2, p3, p4\}$.
 - $T = \{t1, t2\}$.
 - $F = \{\langle p1, t1 \rangle, \langle p2, t2 \rangle, \langle p3, t2 \rangle, \langle t1, p2 \rangle, \langle t2, p3 \rangle, \langle t2, p4 \rangle\}$.
- Dynamic behavior – states reached during process execution
 1. $p1+p3$
 2. $p2+p3$
 3. $p3+p4$.



WF-nets

- WF nets are based on the classical Petri-net for the purpose of business process modelling and has one input place and one output place.
- The *squares* are the active parts of the model and correspond to tasks. The *circles* are the passive parts of the model and are used to represent states.
- In the classical Petri net, the squares are named transitions and the circles are named places. A workflow net models the life-cycle of one case.
- Examples of cases are insurance claims, tax declarations, and traffic violations. Cases are represented by tokens and in this case the token in *start* corresponds to an order.



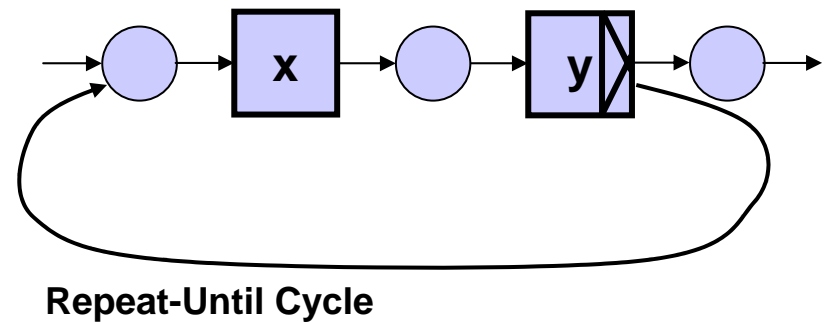
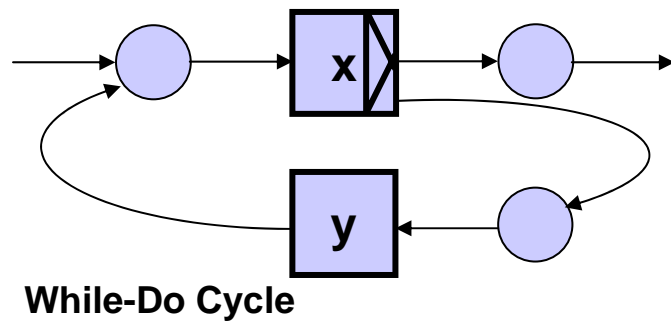
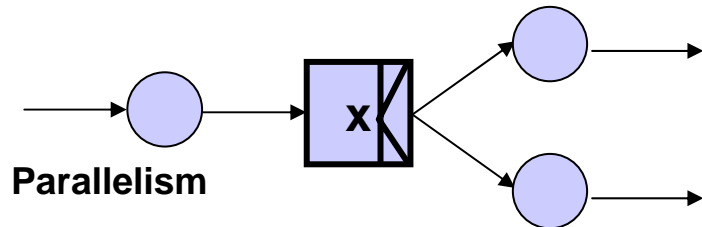
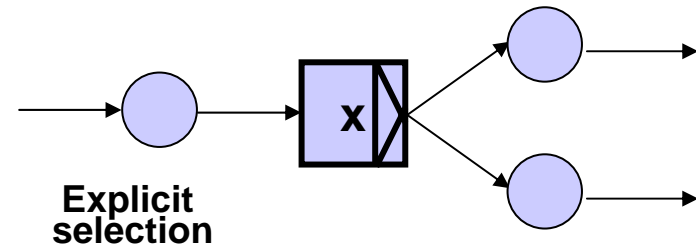
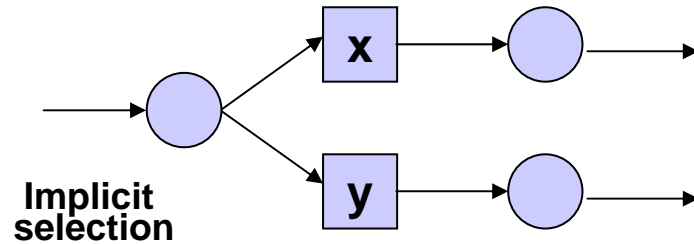
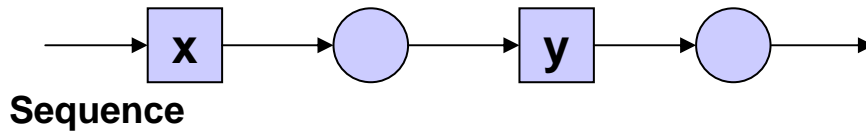
Formal definition of WF-Nets

Path: Given a Petri Net $PN = (P, T, F)$. Path from node $u_1 \in P \cup T$ to node $u_n \in P \cup T$ is sequence (u_1, u_2, \dots, u_n) such that $(u_i, u_{i+1}) \in F$ for $1 \leq i \leq n-1$.

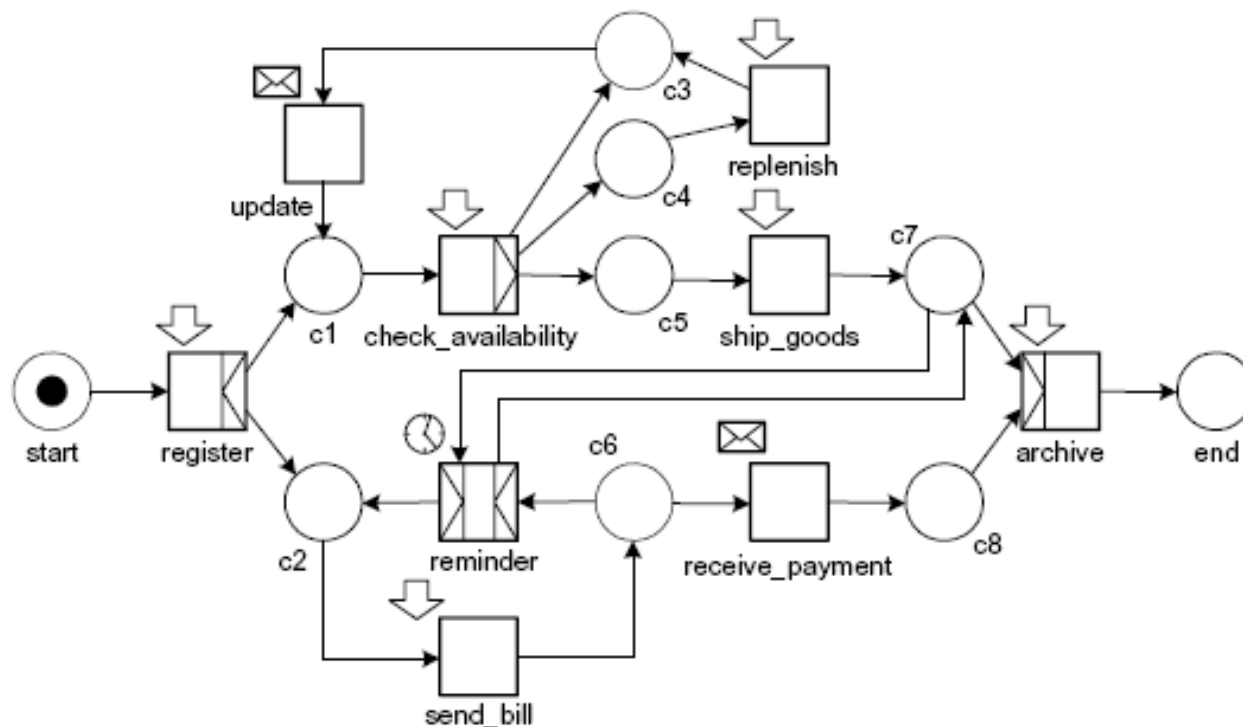
WF-net: a Petri Net $PN = (P, T, F)$ is a WF-net, if and only if:

- There is one source place $i \in P$ such that $\bullet i = \emptyset$.
- There is one sink place $o \in P$ such that $i \bullet = \emptyset$.
- Every node $u \in P \cup T$ is on a path from i to o .

Workflow structure



Example of WF-net





Key Issues

- Formal methods reduce number of errors in software but the mathematical representation requires more time to market software product.
- Formal methods are hard to scale up to large systems.
- The main area of their applicability is critical systems. In this area the use of formal methods seems to be cost-effective.
- The idea is to combine formal methods with diagrammatic languages like EPC or UML. The formal methods specify what cannot be captured by diagrams.



Formalization of BPM

The main characteristics of BPM:

- one of modelling techniques for definition and modelling business processes based on High-level Petri Nets
- a graphical business process specification language which is extended by formalized specification
- was developed for bringing the modern modeling methodology closer to the real business processes
- enables detailed describing and defining of all aspects of analyzed process parties (roles of individual objects, their relations and association to another objects and much more..)



Formalization of BPM

A “*Business Process Modelling*” is a five-tuple (O, A, P, E, S) :

- O (objects) is a finite set of objects,
- A (activities) is a finite set of atomic activities,
- P (processes) is a finite set of sub-processes or processes of the whole business process model,
- E (edges), $E \subseteq (O \times A) \cup (O \times P) \cup (A \times O) \cup (P \times O)$ is a finite set of arcs describe the connections between object, activities and processes,
- S (scenarios) is a finite set of scenarios for individual atomic activities.

Also some functions are defined:

- *activity_scenarios*: $A \rightarrow S$ is a function which maps each activity onto a set of scenarios joined with a given activity.



Formalization of EPC

EPC: Event-driven Process Chain *is a five-tuple* (E, F, C, T, A) :

- E is a finite set of events,
 - F is a finite set of functions,
 - C is a finite set of logic connectors,
 - $T: C \rightarrow \{\wedge, XOR, \vee\}$ is a function which maps connectors onto a connector type \mathbf{a}
 - $A \subseteq (E \times F) \cup (F \times E) \cup (E \times C) \cup (C \times E) \cup (F \times C) \cup (C \times F) \cup (C \times C)$ is a finite set of arcs
- Next step is to *define rules* of how the EPC model can be transformed to WF-Net.
 - Resulting WF-Net can be verified using standard methods and tools applicable to Petri Nets.
 - Building of EPC diagrams can employ well-structured components as well as constructing WF-Nets.



Verification of EPC models

Regularity: *EPC* is *regular*, if and only if:

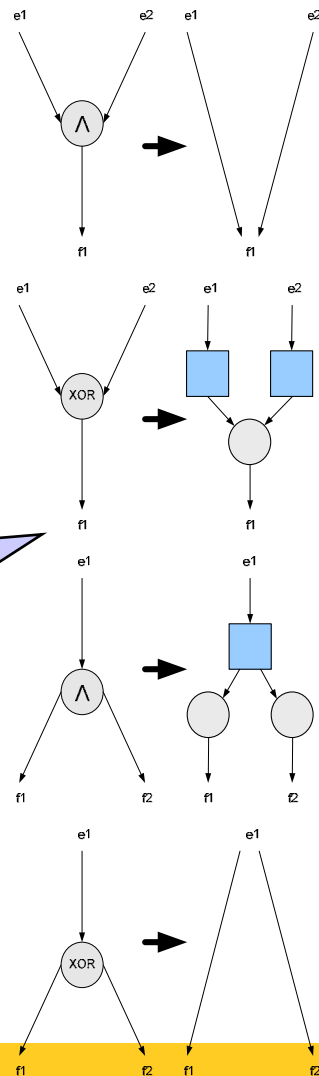
- *EPC* has just two special events e_{start} and e_{final} . Event e_{start} is an initial node, for that is correct $\bullet e_{start} = \emptyset$. Event e_{final} is a finite node, for that is correct $e_{final} \bullet = \emptyset$.
- Each node $n \in N$ is situated on path from e_{start} into e_{final} .

Reliability: Regular *EPC* is *reliable*, if and only if:

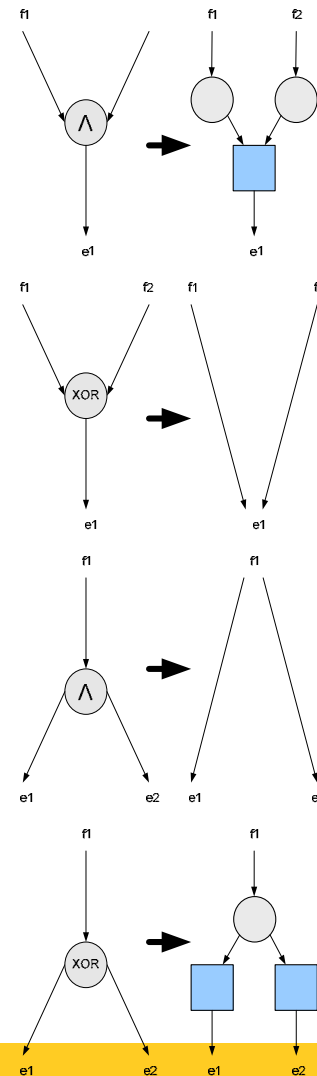
- For each state M reachable from *initial state* (only event e_{start} is correct) exists concatenation of event and functions, that go to finite state (only event e_{final} is correct).
- Finite state is the only state reachable from initial state, when is correct only event e_{final} .
- It does not exist any accessible function, it means for each function $f \in F$, exists concatenation of events and functions, that go to f .

Mapping Connectors on Petri Nets

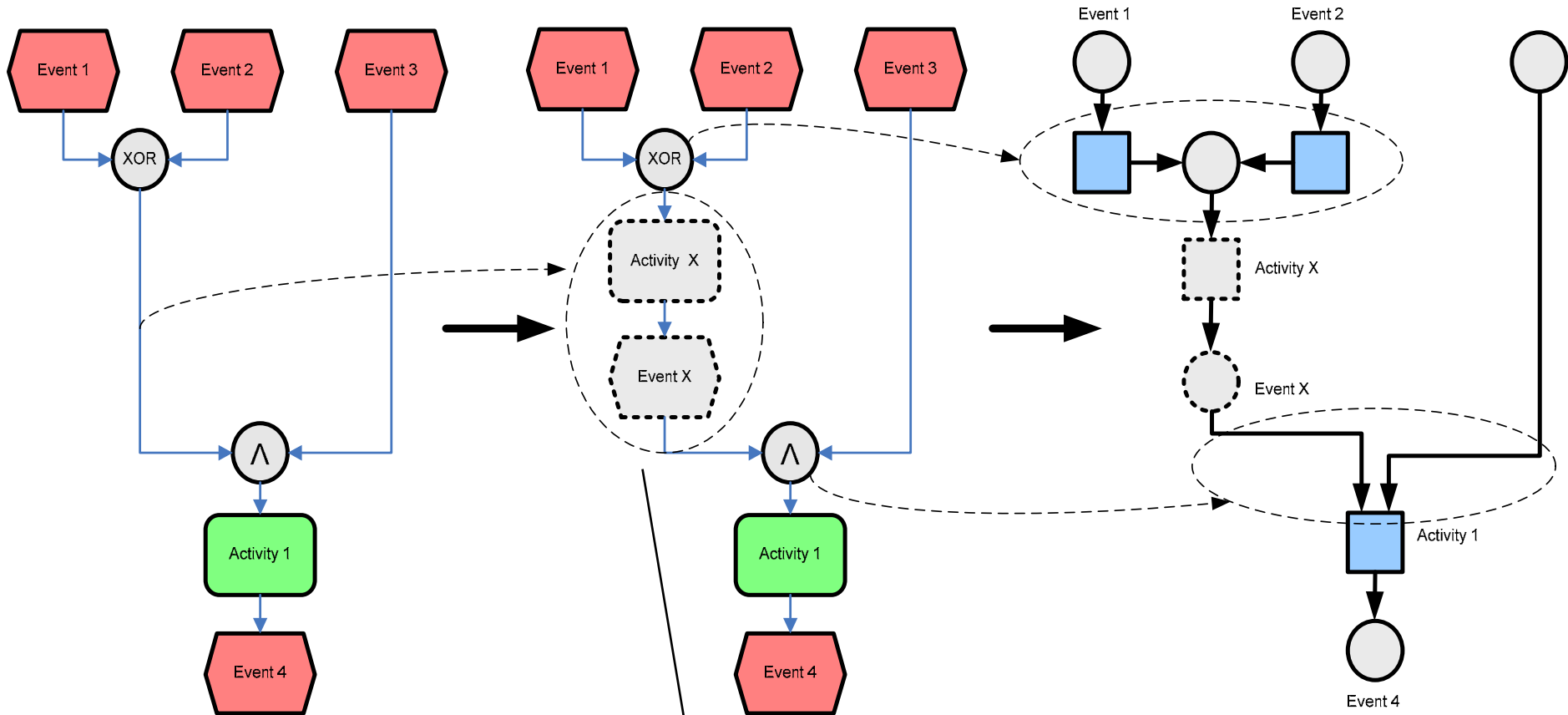
**Connecting
Events to
Activities**



**Connecting
Activities to
Events**



Transformation of EPC Model to Petri Net



Arcs between two connectors must be replaced by events and functions before the EPC is mapped onto a Petri Net.



SW tools for BPM

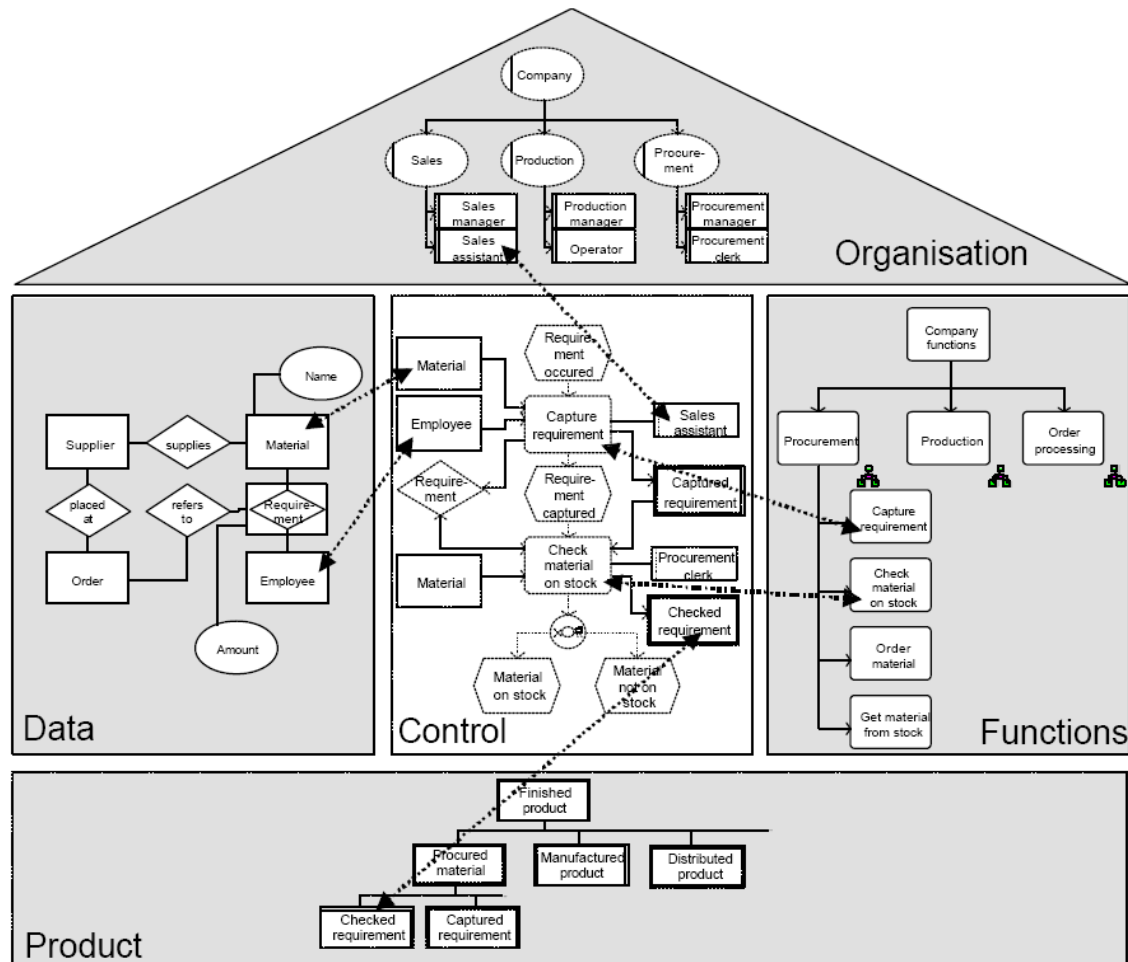
- Tools oriented on EPC diagrams:
 - **ARIS** (ARchitecture of Integrated Systems)
- Tools oriented on UML:
 - **Rational Rose**
 - **Poseidon for UML**
- Tools oriented on Petri Nets:
 - **BP-Studio, Tina, PNtalk, P3, F-net, JFern..**
 - <http://www.informatik.uni-hamburg.de/TGI/PetriNets/>
- General modelling tools:
 - **QPR Process Guide**



ARIS

- **ARIS** (Architecture of Integrated Systems) is focused on business process specification from different points of view and in whole its wideness.
- **Conceptual Framework** that helps to describe organizations: their organizational structure, their processes and their resources in terms of people and information systems.
- **Software tool** that helps to apply the conceptual framework. The software tool helps to electronically describe organizations in a consistent manner and analyze them in some respects.
- More info: <http://www.ids-scheer.com>

ARIS Framework





UML oriented tools

UML is standardized, which enables model portability created in UML among different SW tools. The basic element is a quality meta-model, respected by SW producers for modelling in UML.

Advantages

- this models it is furthermore possible in the same tool transform for needs of software specifications, it does not get to unnecessary losses and mistakes by a translation of business models in models of software work,
- It is common modelling language.

Disadvantages

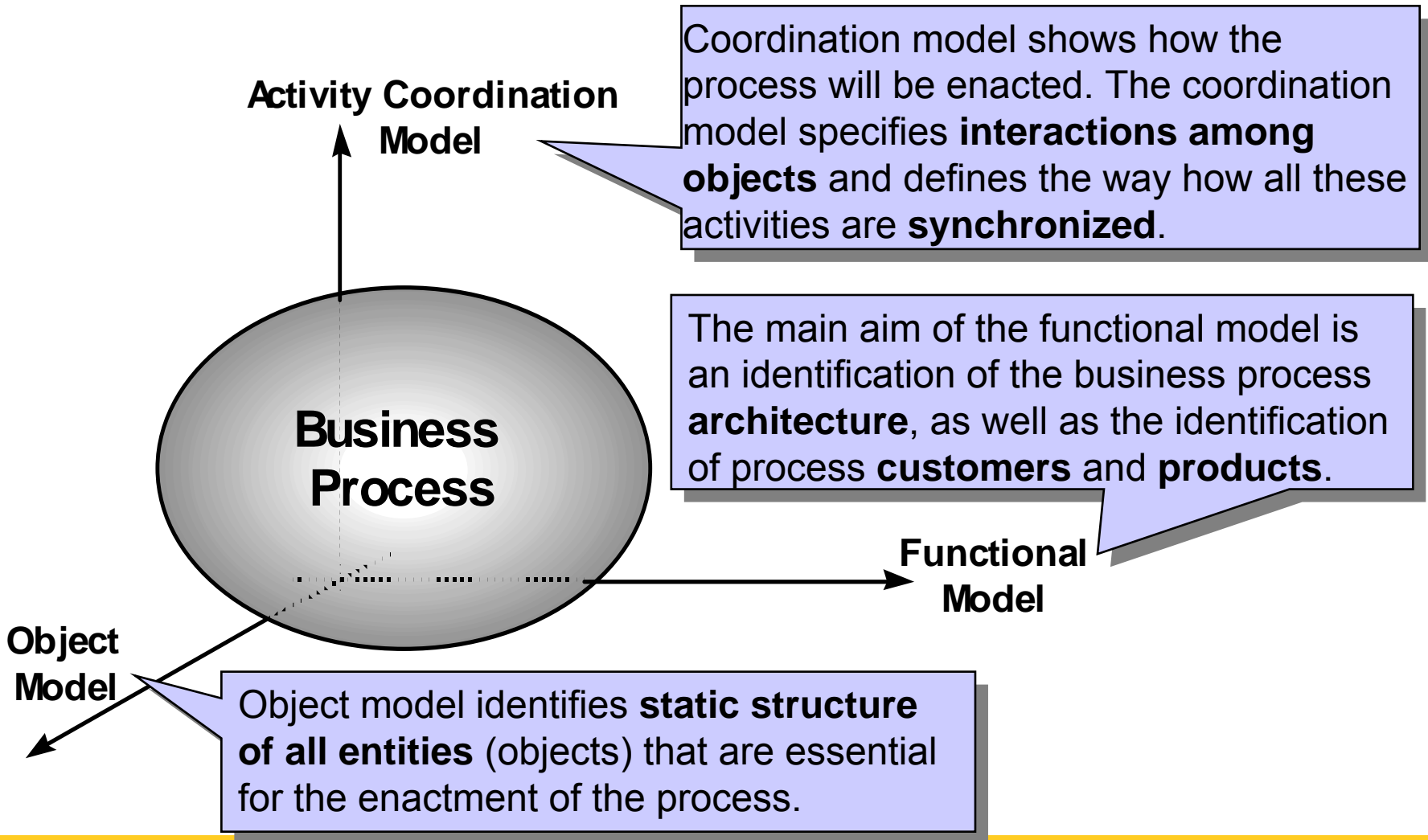
- business modelling forms a subset of functionality, that SW tools offer ⇒ this systems are noticeably more difficulty from the point of view of application and they require a demanding induction.
- It is not effective to invest considerable financial tools, when it is not supposed other use then is just business modelling



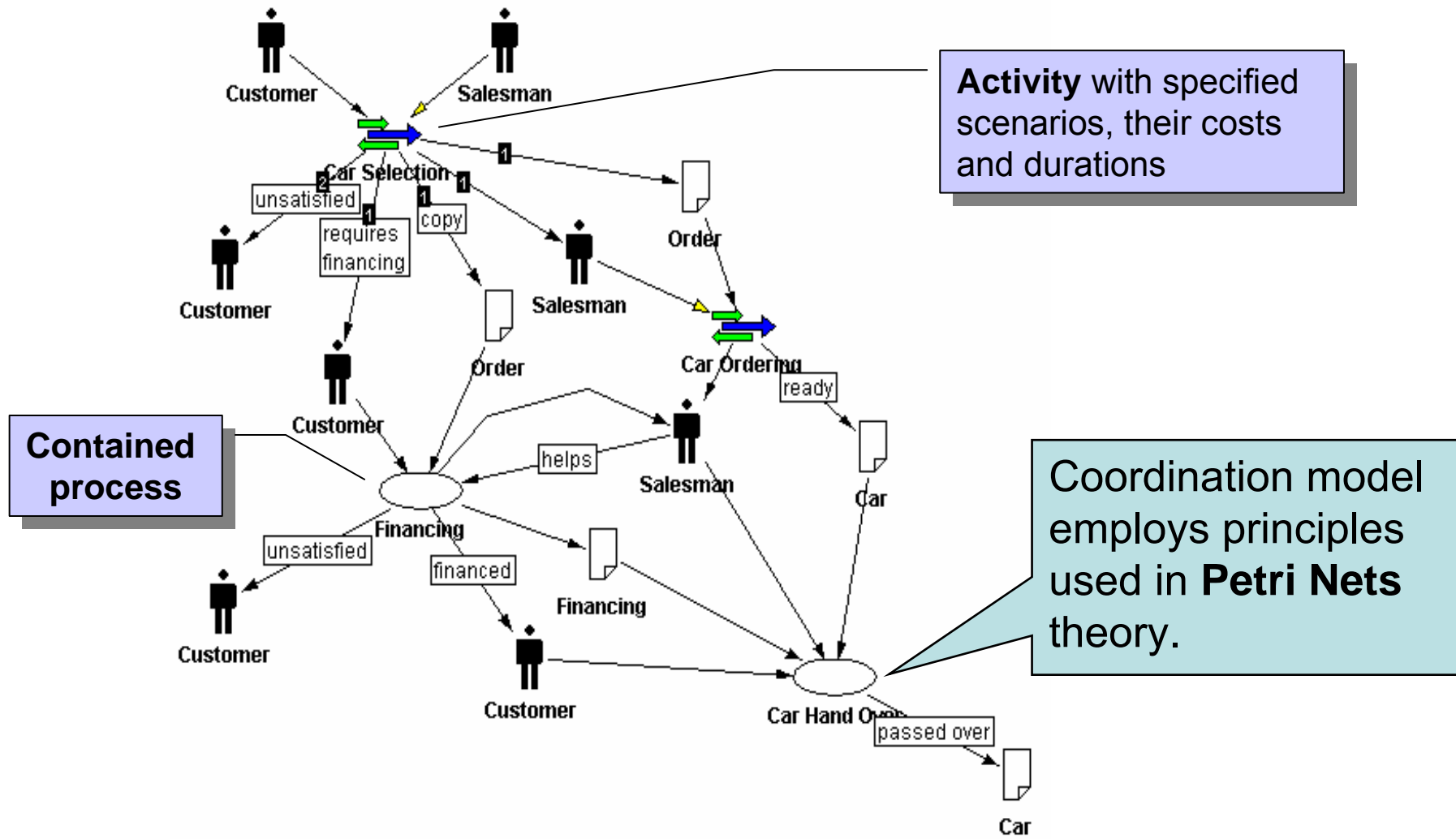
Business Process Studio

- user friendly, so domain experts without a special knowledge of information technology can use it.
- based on a formal approach (Petri Nets) that enables analysis, simulation, and later execution of a built model (workflow engine).
- focused on concurrency as a primary and inherent property of any business process
- Download ...
<http://vondrak.cs.vsb.cz/download.html>

Method BPM



Coordination Model: Car Sale





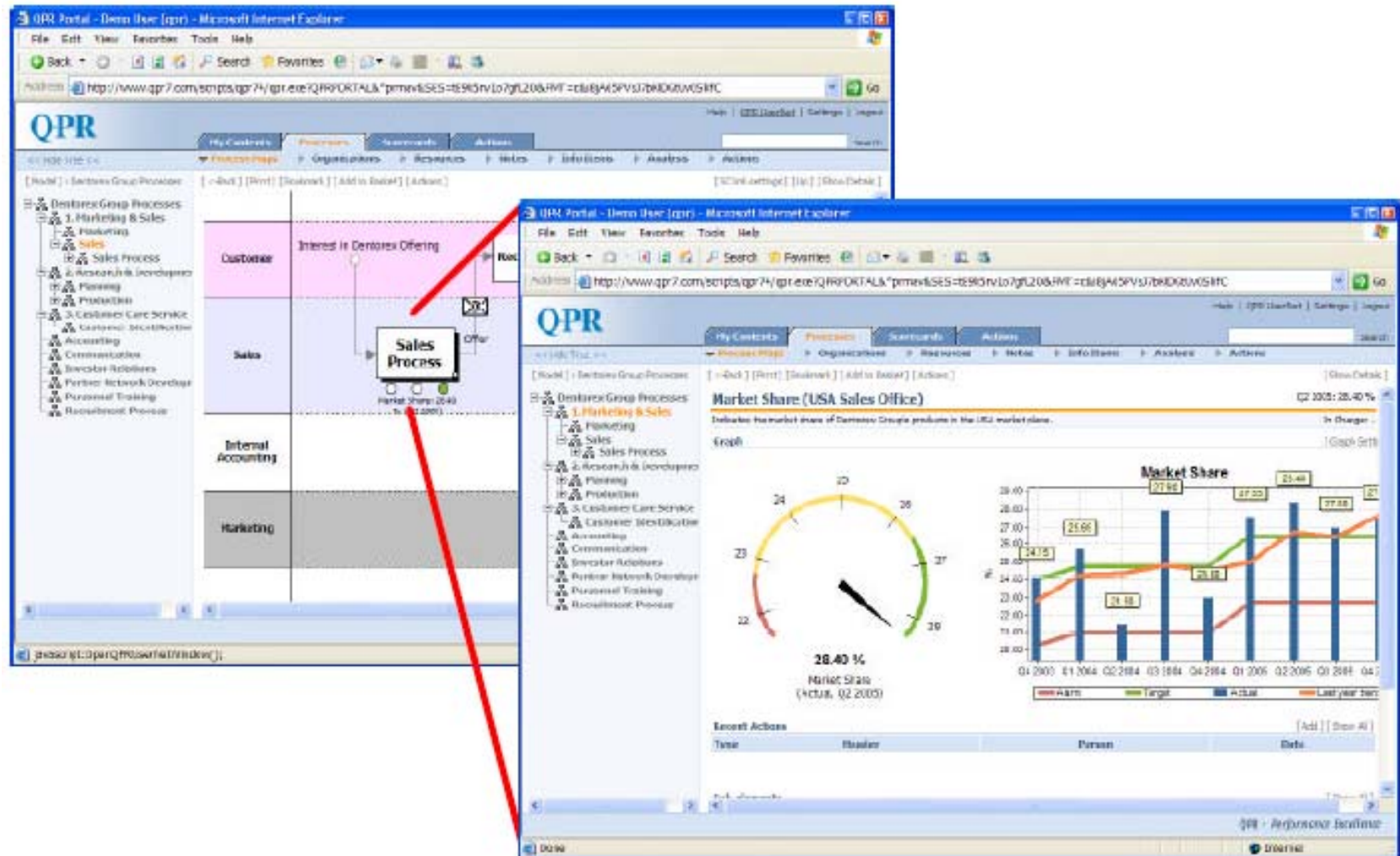
QPR ProcessGuide

***ProcessGuide** is an interactive tool for planning, implementing, communicating and committing people to BP improvement. It enables organizations to do process management in order to achieve the main organizational goals.*

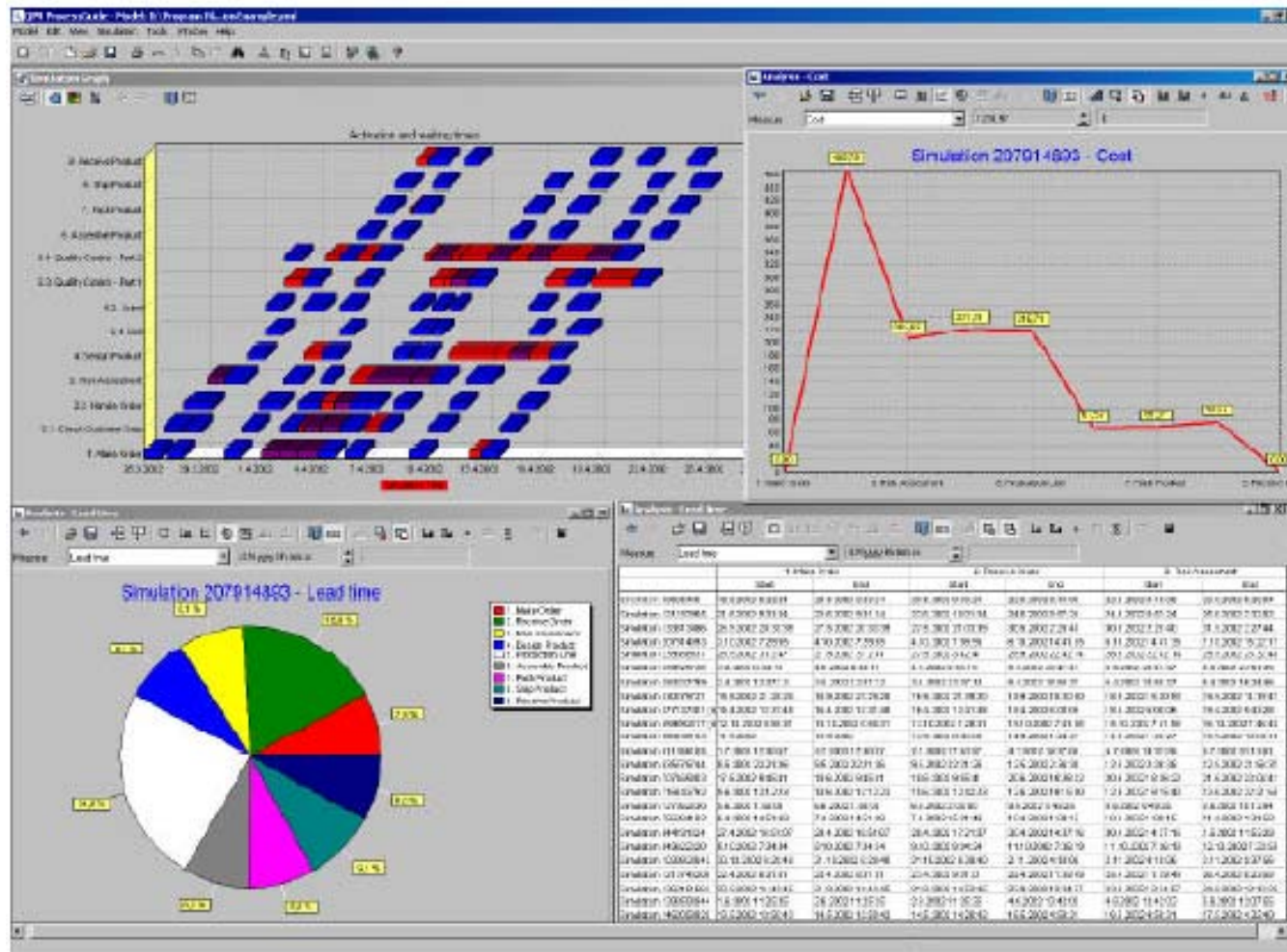
Key features:

- Process modelling, a hierarchical model (different styles, scales..).
- Visual presentation with easy-to-manipulate flowcharts.
- Communication with web publishing, providing personalized access to processes.
- Advanced modelling possibilities.
- Support for simulation and analysis of processes.
- Integration possibilities, allowing linkage to any external data or application and importing/exporting processes in XML.
- Technical architecture that minimizes IT costs of usage.
- Quality systems, knowledge management, SCM, CRM, Different change management methods (6 Sigma etc.).

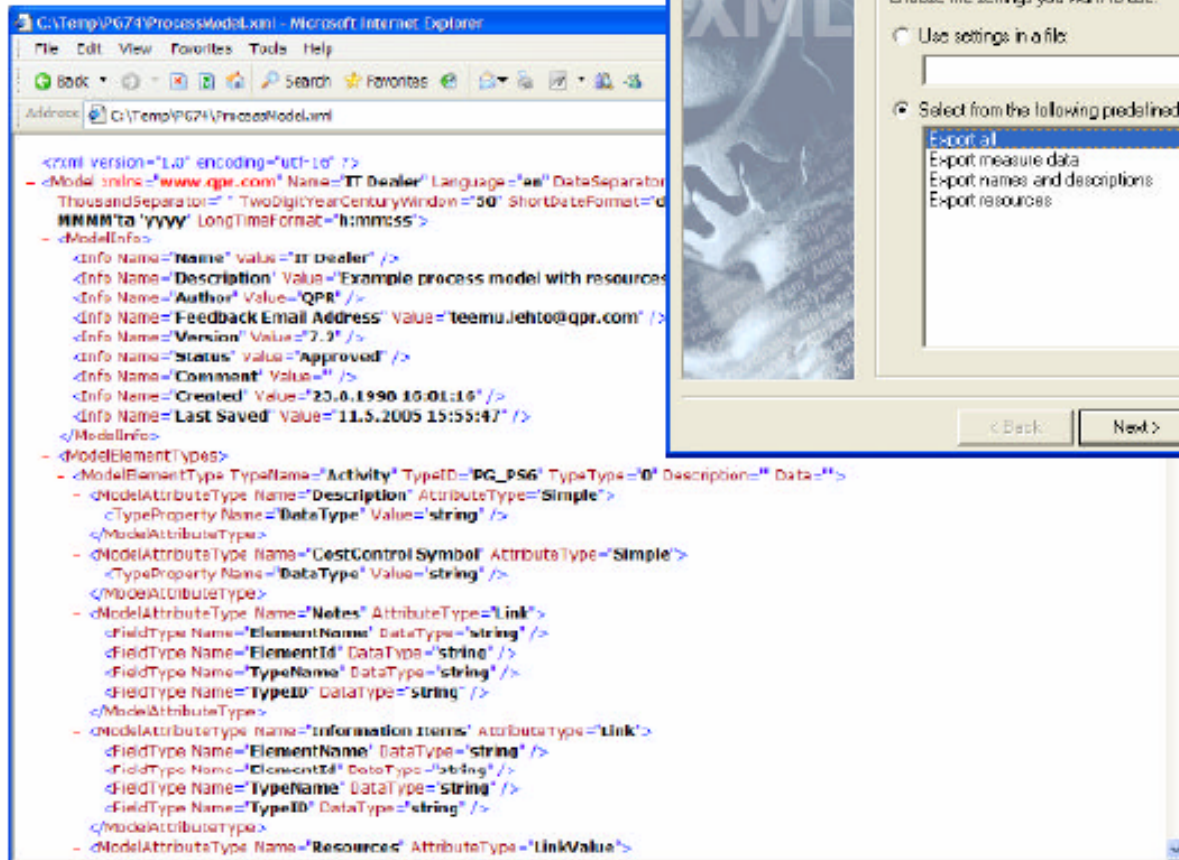
Measure Business Process Performance



Analyse, Improve and Optimize Processes



Information Exports & Imports Using XML



The screenshot shows a Microsoft Internet Explorer window with the address bar pointing to `C:\Temp\PG74\ProcessModel.xml`. The XML content is as follows:

```
<?xml version="1.0" encoding="utf-16" ?>
<Model xmlns="www.qpr.com" Name="IT Dealer" Language="en" DataSeparator="
ThousandSeparator=" " TwoDigitYearCenturyWindow="30" ShortDateFormat="d
MMMM'yy" LongTimeFormat="h:mm:ss">
  <ModelInfo>
    <Info Name="Name" Value="IT Dealer" />
    <Info Name="Description" Value="Example process model with resources" />
    <Info Name="Author" Value="QPR" />
    <Info Name="Feedback Email Address" Value="teemu.lehto@qpr.com" />
    <Info Name="Version" Value="7.9" />
    <Info Name="Status" Value="Approved" />
    <Info Name="Comment" Value="" />
    <Info Name="Created" Value="20.6.1990 10:01:16" />
    <Info Name="Last Saved" Value="11.5.2005 15:55:47" />
  </ModelInfo>
  <ModelElementType>
    <ModelElementType Type="Activity" TypeID="PG_PSG" TypeType="0" Description="" Data="">
      <ModelAttributeType Name="Description" AttributeType="Simple">
        <TypeProperty Name="DataType" Value="string" />
      </ModelAttributeType>
      <ModelAttributeType Name="CostControl Symbol" AttributeType="Simple">
        <TypeProperty Name="DataType" Value="string" />
      </ModelAttributeType>
      <ModelAttributeType Name="Notes" AttributeType="Link">
        <FieldType Name="ElementName" DataType="string" />
        <FieldType Name="ElementId" DataType="string" />
        <FieldType Name="TypeName" DataType="string" />
        <FieldType Name="TypeID" DataType="string" />
      </ModelAttributeType>
      <ModelAttributeType Name="Information Items" AttributeType="Link">
        <FieldType Name="ElementName" DataType="string" />
        <FieldType Name="ElementId" DataType="string" />
        <FieldType Name="TypeName" DataType="string" />
        <FieldType Name="TypeID" DataType="string" />
      </ModelAttributeType>
      <ModelAttributeType Name="Resources" AttributeType="LinkValue">

```

XML Export Wizard: 1/2

This Wizard exports data from the QPR ProcessGuide model into an XML file.

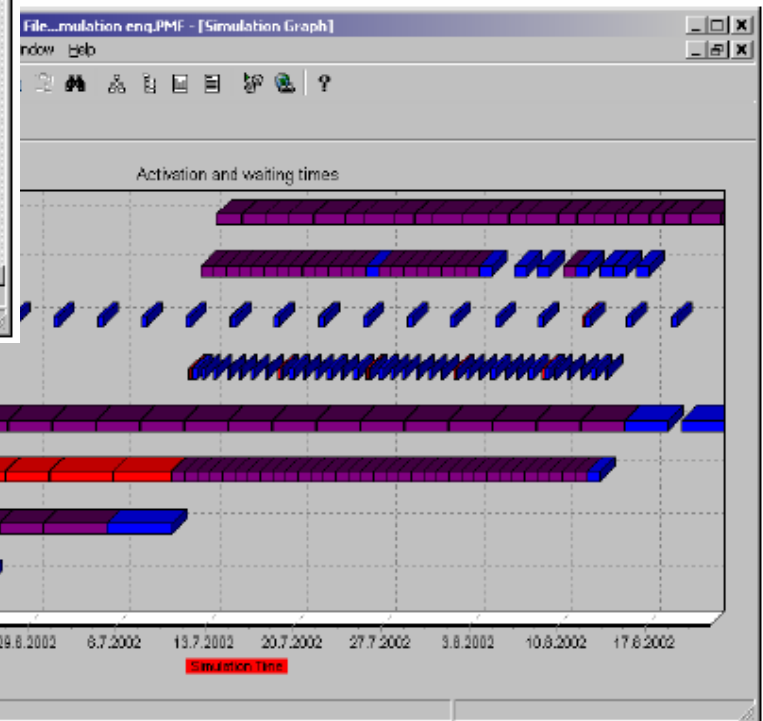
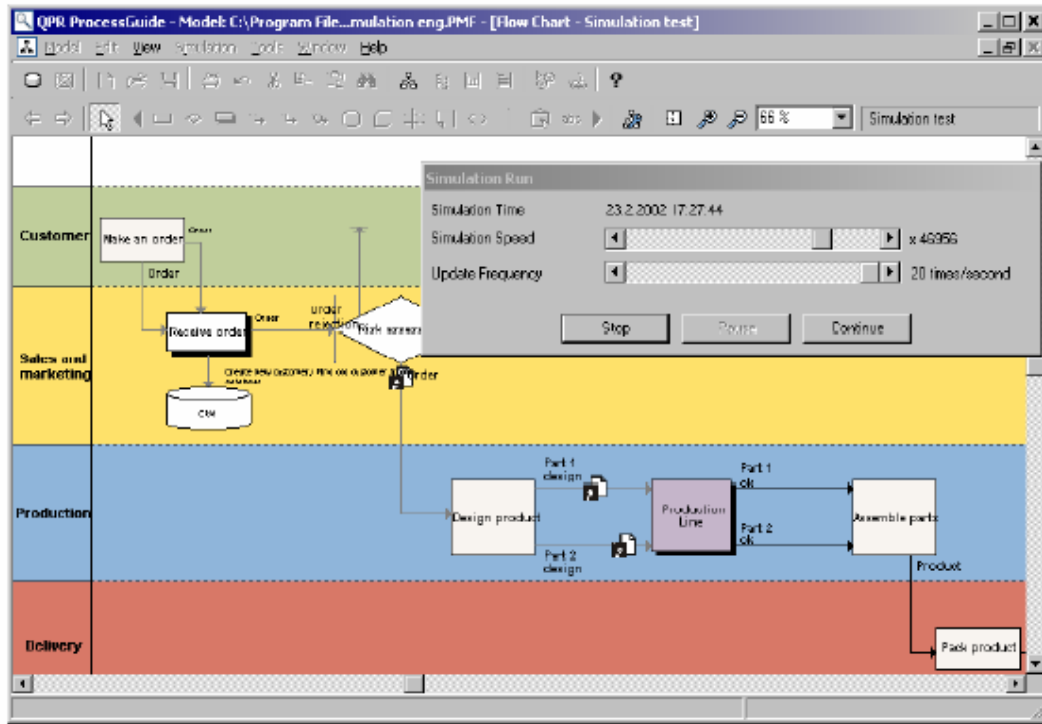
Choose the settings you want to use:

☐ Use settings in a file:

☒ Select from the following predefined settings:

- Export all**
- Export measure data
- Export names and descriptions
- Export resources

Simulation running and graphical presentation of simulation results





Standards of BPM

- **BPML** (Business Process Modeling Language)
- **BPEL4WS** (Business Process Execution Language for Web Services)
- **ebXML** (Electronic Business using eXtensible Markup Language)
- **XPDL** (XML Process Definition Language)
- **BPDM** (Business Process Definition Metamodel)
- **BPMN** (The Business Process Modeling Notation)
- **EPML** (The Event-Driven Process Chain Markup Language)
- **PNML** (The PN Markup Language)
- **UML 2 Activity Diagram, Workflow Reference model**
- **BPRI** (Business Process Runtime Interface)
- **WSCL** (Web Services Conversation Language)
- **WSCDL** (Web Service Choreography Description Language)
- **XLANG** (Web Services for Business Process Design)
- **WSCI, WSCL, PSL, OWL-S, BPXL, BPSS, BPSM, BPQL, WAPI, WFXML, WSFL..**



Business Process Challenges

(in context of BPEL)

- Coordinate asynchronous communication between service
- Correlate message exchanges between parties
- Implement parallel processing of activities
- Manipulate/transform data between partner interactions
- Support for long running business transactions and activities
- Provide consistent exception handling



Orchestration & Choreography

Orchestration

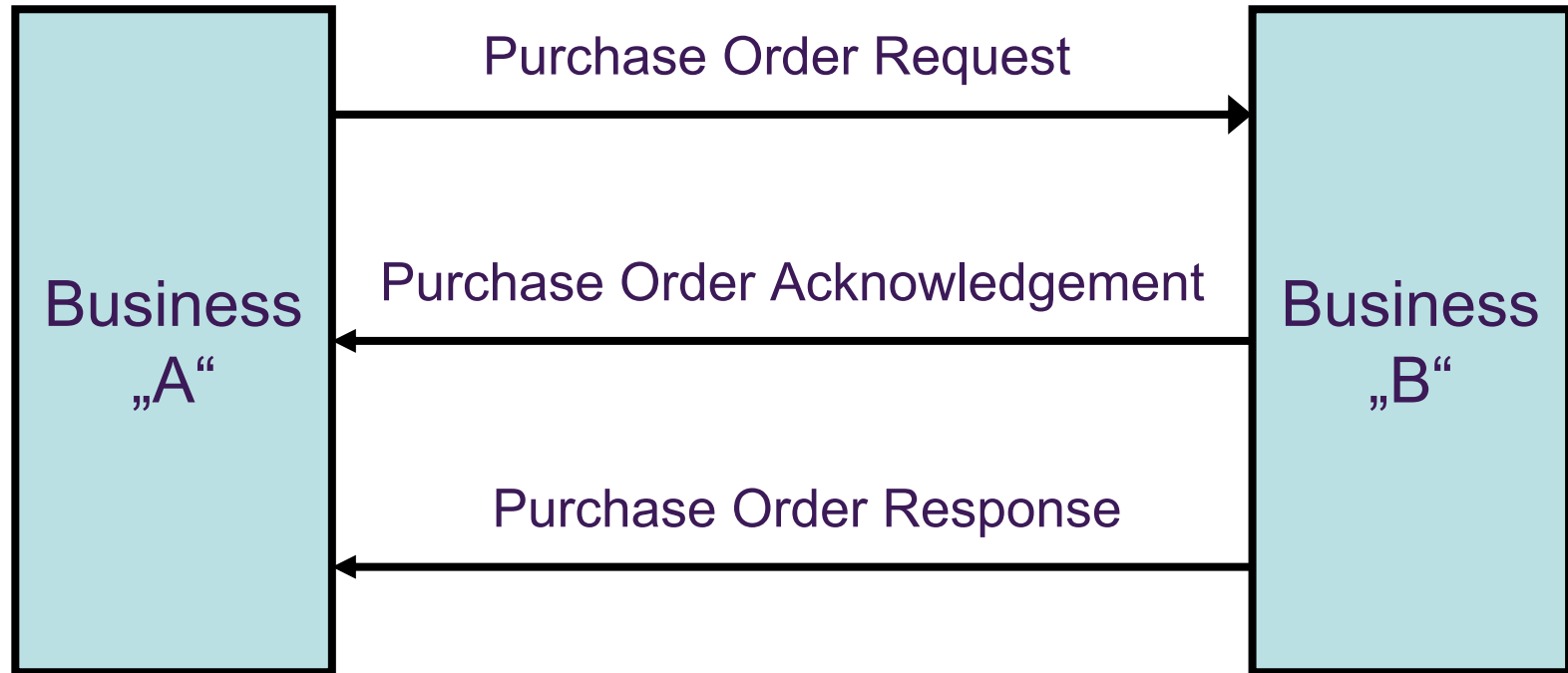
- An executable business process describing a flow from the perspective and under control of a single endpoint (commonly: Workflow)

Choreography

- The observable public exchange of messages, rules of interaction and agreements between two or more business process endpoints



Sample Business Process: Purchase Order





Motivation

- Application integration is a key problem facing businesses
 - Intra enterprise integration (Enterprise Application Integration)
 - Integrating with partners (Business Process Integration)
- Web services → move towards service-oriented computing
 - Applications are viewed as “services”
 - Loosely coupled, dynamic interactions
 - Heterogeneous platforms
 - No single party has complete control
- Service composition
 - How do you compose services in this domain?



Process Model Requirements

- Portability and Interoperability
- Flexible Integration
 - Rich, and easily adaptable to changes in the services it is interacting with
- Recursive, type-based composition, enables ...
 - third-party composition of existing services
 - providing different views on a composition to different parties
 - inter-workflow interaction
 - increased scalability and reuse
- Separation and composability of concerns
 - Decoupled from the supporting mechanisms (quality of service, messaging frameworks)
- Stateful conversations and lifecycle management
 - Can carry multiple stateful long-running conversations
- Recoverability
 - B-processes, and in particular long running ones, need a way to build-in fault handling and compensation mechanisms to handle and recover from errors



BPEL4WS

(Business Process Execution Language for Web Services)

- Portable, interoperable process model for long running processes
- provides an XML notation and semantics for specifying
- is defined in terms of its interactions with partners
- Flexible integration of Web services
 - WSDL abstract interfaces used to define composition
 - *Enables two levels of adaptive behavior:*
 - Abstract partners can be bound to actual services at runtime,
 - The process can choose a protocol for communicating with the service at runtime
 - Services whose data definitions do not match can be composed
 - Data transformations can be in-lined in process definition

WS-BPEL in the WS-* Stack

|| You are here →

WS-BPEL

WSDL, Policy, UDDI, Inspection

Security

Reliable
Messaging

Transactions

Coordination

SOAP (Logical Messaging)

Other protocols

XML, Encoding

Other services

Business
Processes

Description

Quality
Of
Service

Transport
and
Encoding

Getting the Players Together



BPEL4WS 1.1



(*) Organization for the Advancement of Structured Information Standards



WS-BPEL Adoption: Products

- Active Endpoints ActiveWebflow Server
- ActiveBPEL Engine (open source)
- bexee BPEL Execution Engine (open source)
- Cape Clear Orchestrator
- FiveSight PXE
- IBM WebSphere Business Integration – Server Foundation 5.1
- IBM WebSphere Process Server 6.0
- OpenLink Virtuoso Universal Server
- OpenStorm ChoreoServer
- **Oracle BPEL Process Manager**
- Parasoft BPEL Maestro
- SeeBeyond eInsight BPM
- Twister (open source)



WS-BPEL Application Areas

- Business Process Design
- Autonomic Computing
- Grid Computing
- Semantic Web

Mapping of UML Automated Business Processes to BPEL4WS

UML to BPEL4WS

- UML is a widely used, standard modelling language for software design with a visual notation.
- BPEL4WS is a language for specifying business processes which can be executed on a BPEL4WS runtime
- **Goal: Support automated mapping from (a profile of) UML to BPEL4WS**

↙
A profile is a customization of UML for modelling in a particular context.



IBM WebSphere
Software



**BPEL4WS,
WSDL, XSD**

An example of a BPEL program

A BPEL 'program'

state

interaction
points

behaviour

```
<process name="purchaseOrderProcess" ...>

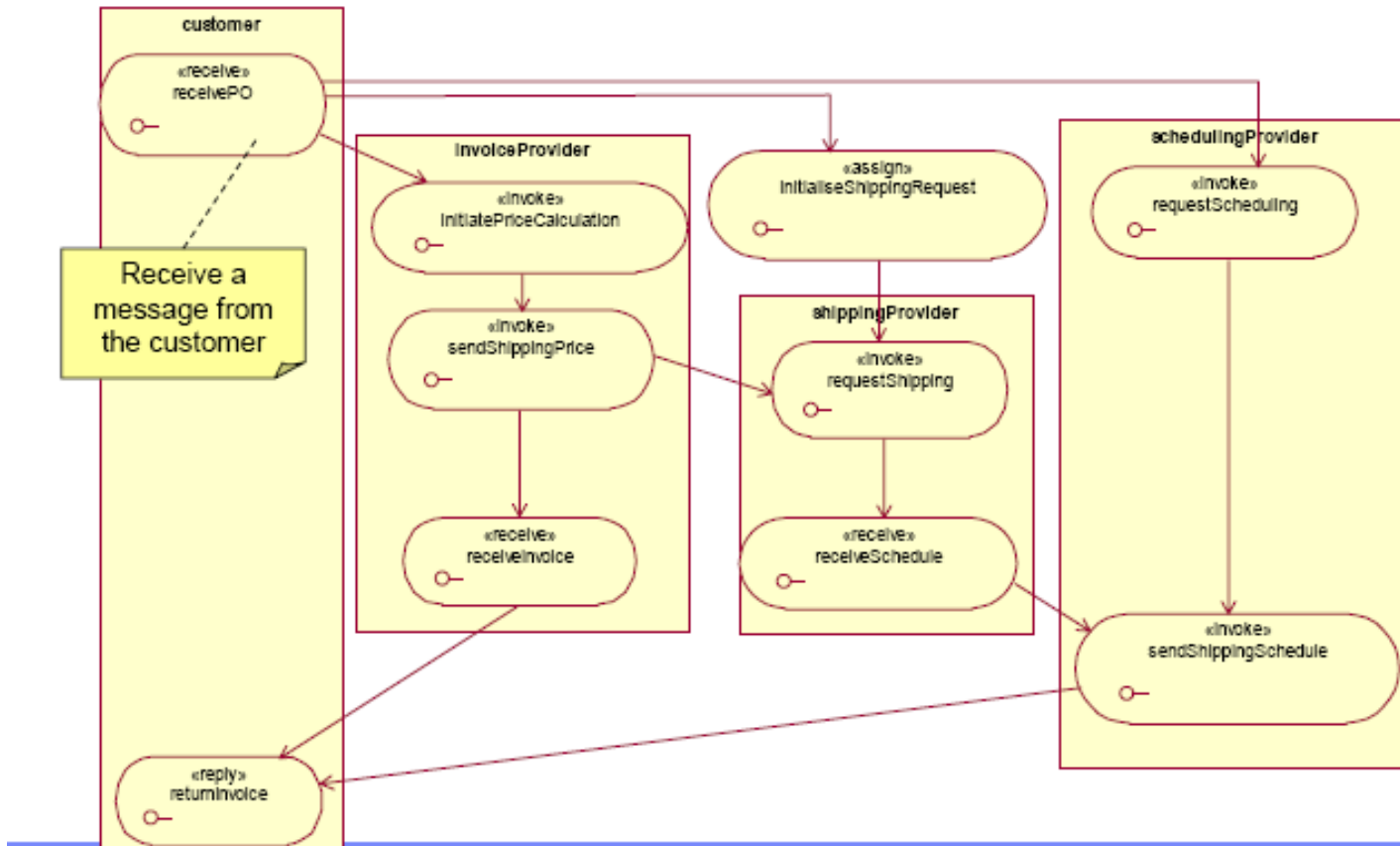
  <containers>
    <container name="PO" messageType="lns:POMessage"/>
    <container name="Invoice" messageType="lns:InvMessage"/>
    ...
  </containers>

  <partners>
    <partner name="customer" serviceLinkType="lns:purchaseLT"
      myRole="purchaseService"/>
    ...
  </partners>

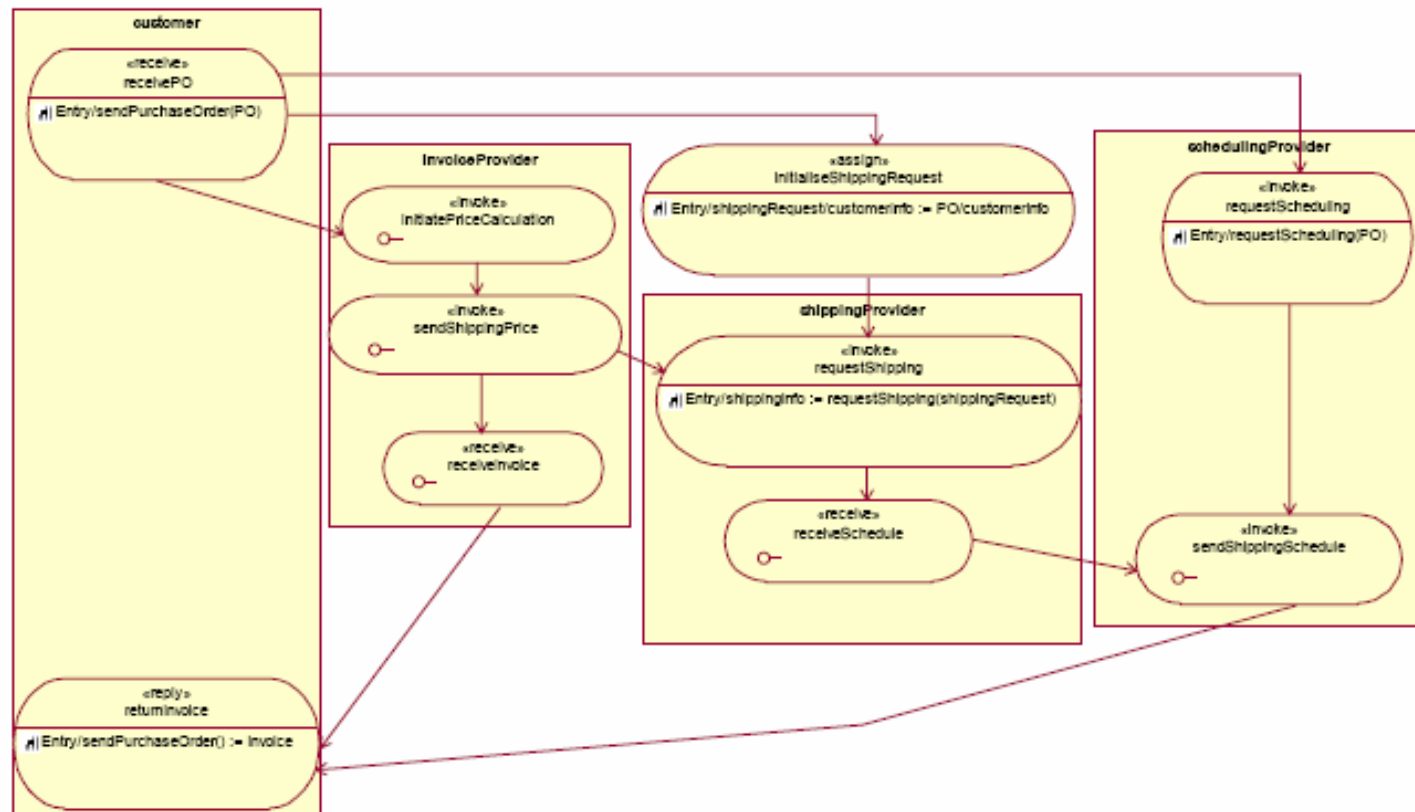
  <sequence>
    <receive partner="customer"
      portType="lns:purchaseOrderPT"
      operation="sendPurchaseOrder"
      container="PO">
    </receive>
    ...
    <reply partner="customer" portType="lns:purchasePT"
      operation="sendPurchaseOrder"
      container="Invoice"/>
    </sequence>

</process>
```


Example: Purchase Order Process



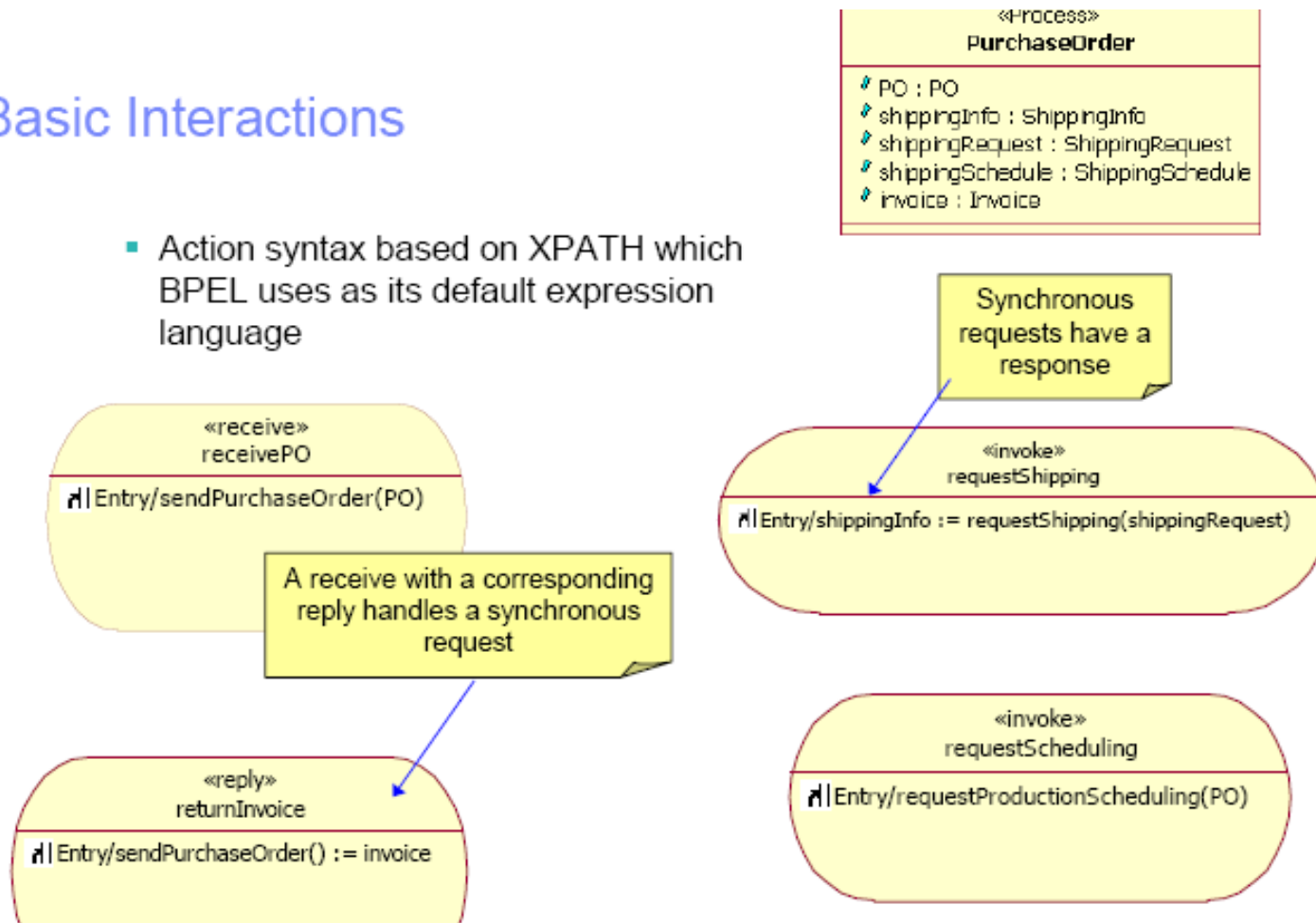
Purchase Order Process: Detailed Behaviour



Basic Interactions

Basic Interactions

- Action syntax based on XPATH which BPEL uses as its default expression language





BPEL Mapping Overview

<<process>> class	BPEL process definition
Activity graph on a <<process>> class	BPEL activity hierarchy
<<port>> associations	BPEL partner declarations
<<process>> class attributes	BPEL containers
Hierarchical structure and control flow	BPEL sequence and flow activities
Decision nodes	BPEL switch activities and transition conditions
<<receive>>, <<reply>>, <<invoke>> activities	BPEL receive, reply, invoke activities
<<protocol>> package with <<role>> classes	BPEL service links types and roles



Conclusions

- The experiences of modelling with UML can be applied to the development of systems that will be deployed using emerging web services standards
- It is possible to specify a profile of UML with sufficient detail that it can be translated automatically to a language such as BPEL4WS
- The approach provides an integration mechanism for multiple standards and specifications which need to be used to build a complex solution

This is especially relevant in business integration scenarios





Future research

- Comparison of actual used tools for BPM
- Evaluation their abilities to catch all important aspects of business processes
- Identification and analysis of the main imperfections of this tools and
- A proposal of an universal and platform independent tool, that enables a connection cross different business systems from different application domains, data formats, languages, etc...



References

1. <http://www.qpr.com>
2. <http://vondrak.cs.vsb.cz/>
3. <http://www-106.ibm.com/developerworks/webservices/library/ws-bpel>
4. Mayer, R.J., Painter, M.: IDEF Family of Methods, Technical Report, Knowledge Based Systems, Inc., College Station, TX, 1991
5. Booch, G., Jacobson, I., Rumbaugh, J.: The Unified Modeling Language User Guide, Addison Wesley Longman, Inc., 1999
6. Vondrak, I., Szturc, R., Kruzel, M.: Company Driven by Process Models, European Concurrent Engineering Conference ECEC '99, SCS, Erlangen-Nuremberg, Germany, pp. 188-193, 1999
7. Wil van der Aalst. Formalization and Verification of Event-driven Process Chains. Information and Software Technology, 41(10):639-650, 1999.
8. Wil van der Aalst. Workflow Verification: Finding Control-Flow Errors using Petri-net-based Techniques. In Business Process Management: Models,
9. Wil van der Aalst, Kees van Hee: Workflow Management, Models, Methods, and Systems. MIT Press, 2002
10. Češka, M.: Petriho sítě, Akademické nakladatelství CERM Brno, 1994
11. Petri Nets for Systems Engineering, Springer-Verlag Berlin, 2003



BPM Standards

- **BPDM:** OMG's Business Process Definition Metamodel - is MOF compliant. The respective BPDM interchange format will rely on XMI production rules. It is subset of UML 2 metamodel focused on modeling business processes. It contains: UML 2.0, Profile for BPD, UML 1.5 Profile for BPEL4WS, Mapping from BPD Metamodel to BPEL4WS, from EDOC to BPD Metamodel and from BPMN to BPD Metamodel.
- **BPEL4WS:** Business Process Execution Language for Web Services - has moved from a consortium of major software vendors to OASIS. BPEL is specified as an interchange format only via an XML Schema. BPEL models tasks as calls to Web Services whose input and output are specified by messages and whose address is identified via Uniform Resource Identifiers (URI) of WSDL port types. SOAP is used as the communication protocol.
- **BPML:** The Business Process Modeling Language proposed by BPMI is similar to BPEL. As the main difference BPML allows to specify multiple processes in one XML document and related communication between those processes. BPML is not tied to WSDL. The communication protocol is left to a BPML compliant implementation.



BPM Standards

- **BPMN** (The Business Process Modeling Notation) developed by BPMI wants to unify the different graphical notations for business processes. The specification provides a mapping to BPEL. Additional specifications will define a BPMN metamodel based on MOF. This will permit serialization with XMI production rules for XML interchange.
- **BPSS** (The Business Process Specification Schema) is part of OASIS and UN/CEFACT's work on ebXML. It includes a metamodel and XML Schema for Web Service choreography. It supports the definition of roles, exceptions, and transactions in an inter-organizational message exchange.
- **BPQL** (Business Process Query Language) is a management interface to a business process management infrastructure that includes a process execution facility (process server) and a process deployment facility (process repository),
- **BPSS** (Business Process Specification Schema) is a standard framework by which business systems may be configured to support execution of business collaborations consisting of business transactions



BPM Standards

- **EPML** (The Event-Driven Process Chain EPC Markup Language) captures the control flow elements of EPCs. Further aspects can be defined via extensions. As EPML aims to facilitate graphical model interchange it includes graphical position information for each EPC object.
- **FBPML** (Fundamentals Business Process Modeling Language) is a visual modeling language, merges IDEF3 and PSL. This language is designed to support both software and workflow system development. It offers precise semantics and can express business processes in logical sentence.
- **OWL-S** (OWL-Services) is a proposal for a service metamodel represented in OWL. It builds on an (input-output-preconditioneffects) quadruple to describe services. OWL permits the definition of so-called groundings which is similar to a WSDL binding to a protocol and related endpoints.
- **PNML** (The PN Markup Language) is a proposal for an XML interchange format for PN models. It supports the basic PN syntax elements and can be extended to represent arbitrary PN types.
- **PSL** (Process Specification Language) defines a neutral representation for manufacturing processes.



BPM Standards

- **UBL** (Universal Business Language) is the product of an international effort to define a royalty-free library of standard electronic XML business documents such as purchase orders and invoices. UBL is designed to plug directly into existing business, legal, auditing, and records management practices, eliminating the re-keying of data in existing fax- and paper-based supply chains and providing an entry point into electronic commerce for small and medium-sized businesses.
- **UML 2 Activity Diagram**: Activity Diagrams of Unified Modeling Language can be exchanged using XML. Their metamodel includes concepts to model input and output of tasks, control flow, data handling, roles, exceptions, and graphical information.
- **WSCDL** (Web Service Choreography Description Language) is only available as a working draft. It builds on WSDL and SOAP and provides different algebraic control flow primitives. It supports data handling, role definition, as well as exception and transaction modelling.
- **WSCI** (W3C's Web Service Choreography Interface) provides a set of extensions to WSDL in order to describe process behavior of message exchanges. Beyond input and output message types, WSDL bindings, and correlation WSCI also supports roles, exception handling, and transactions.



BPM Standards

- **WSCL** (Hewlett-Packard's Web Service Choreography Language) defines a minimal set of concepts in order to describe Web Service choreographies including message types, protocol, and service location. The specification contains a metamodel and a related XML Schema.
- **WSFL** (Web Services Flow Language) is one of the predecessors of BPEL. It includes most of the concepts despite transaction support, graphical position information, and statistical data.
- **XLANG**: is the second predecessor of BPEL. It defines WSDL extensions to describe process behavior of a Web Service similar to WSCI. It provides means for defining message correlation, roles, event and exception handling as well as transaction declaration.
- **XPDL** (XML Process Definition Language) is a standardized interchange format for BP models proposed by WfMC. It includes various concepts like task input/output and address, control flow, data handling, roles, events, and exceptions.
- **xCBL** (XML Common Business Library) is a set of XML building blocks and a document framework that allows the creation of robust, reusable, XML documents to facilitate global trading. It creates interoperability by providing one language that all participants in e-commerce can understand.