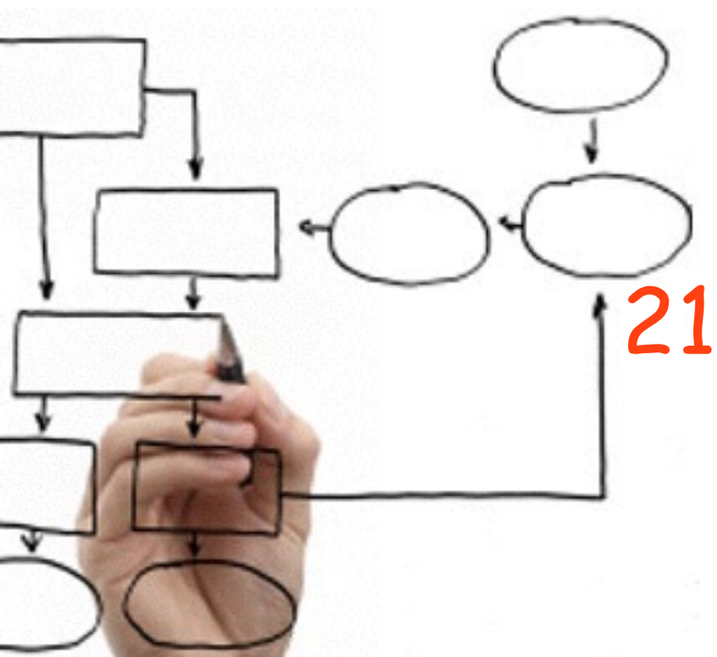
Methods for the specification and verification of business processes

MPB (6 cfu, 295AA)



Roberto Bruni

http://www.di.unipi.it/~bruni

21 - Business process execution language

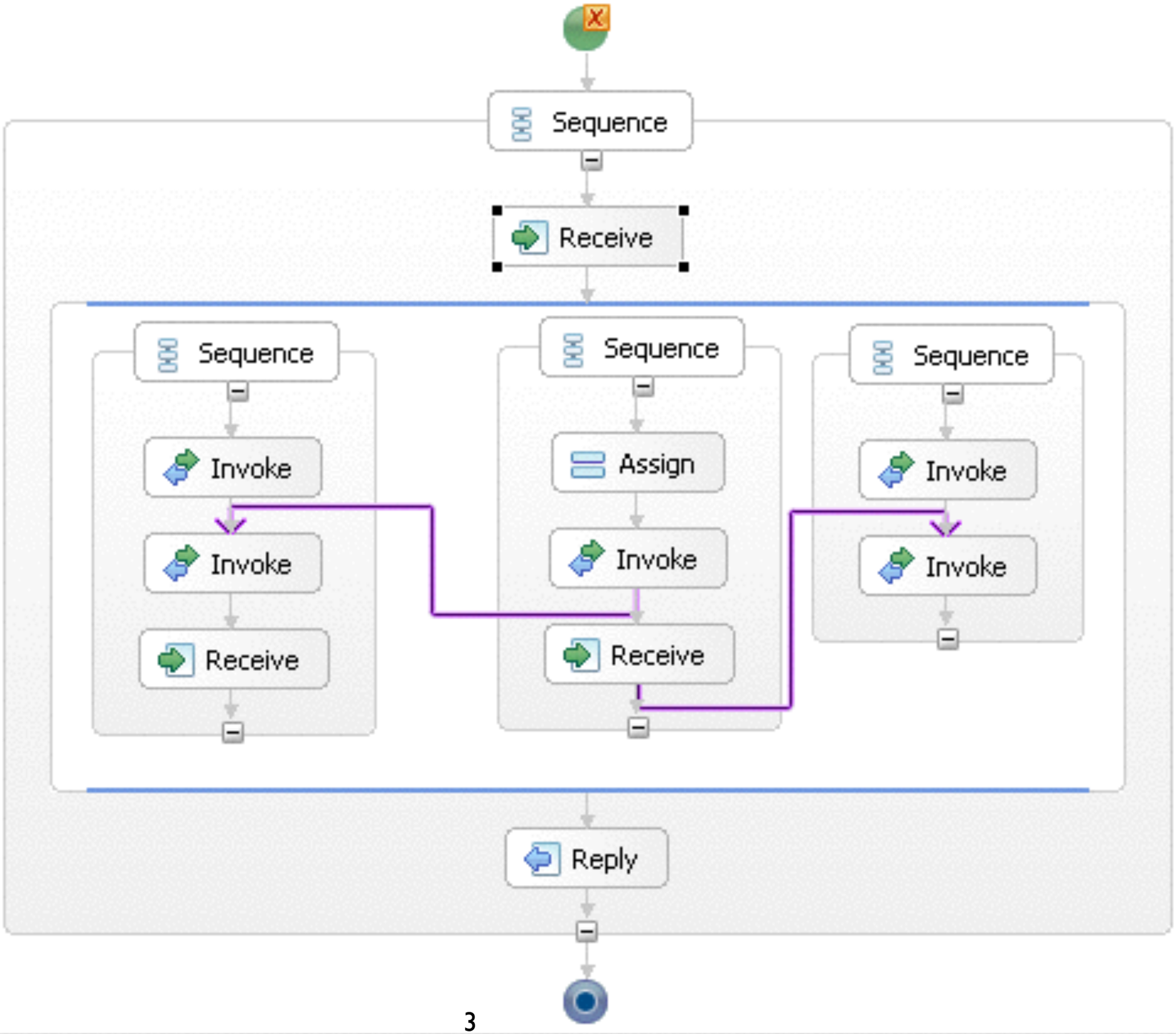
Object

We overview the key features of BPEL

Material inspired in part to Antonio Brogi's slides on Software Services: thanks!

purchaseOrderProcess

Selection Tool
Marquee Tool
Actions
Empty
Invoke
Receive
Reply
Assign
Control
Switch
Pick
While
Repeat Until
Wait
Faults
Exit
Throw
Rethrow
Compensate
Zoom In
Zoom Out



BPEL

Business process execution language

Also known as:
Web Services Business Process Execution Language (WS-BPEL)
Business Process Execution Language for Web Services (BPEL4WS)

it is a standard executable language for orchestrating the use of Web Service within business processes

it deals with import / export information, remote invocation, correlation, fault handling, compensation

Web services

Web services fix a standard for interoperability between heterogeneous, loosely coupled, remote software applications
(separately developed, running on different platforms)
over (not only) the HTTP protocol

Informally:
web services are for software
what web sites are for human

WS basics

Services must be made available on the web
(need a server)

Services must be advertised over the web
(need some repositories)

Service repositories must be queried
(need service description)

Services must be invoked
(need standard communication format)

XMLification

WSFL, BPEL, ...

Service composition

UDDI

Service discovery

Service publication

WSDL

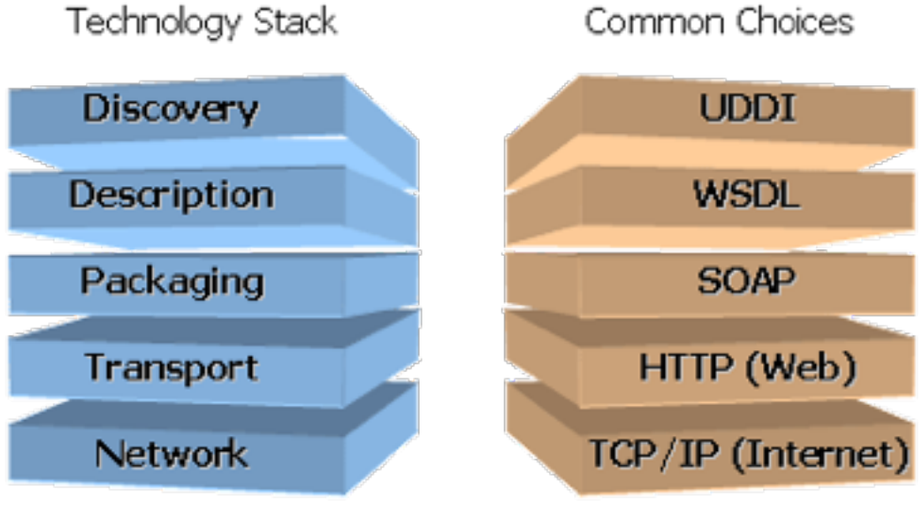
Service description

SOAP

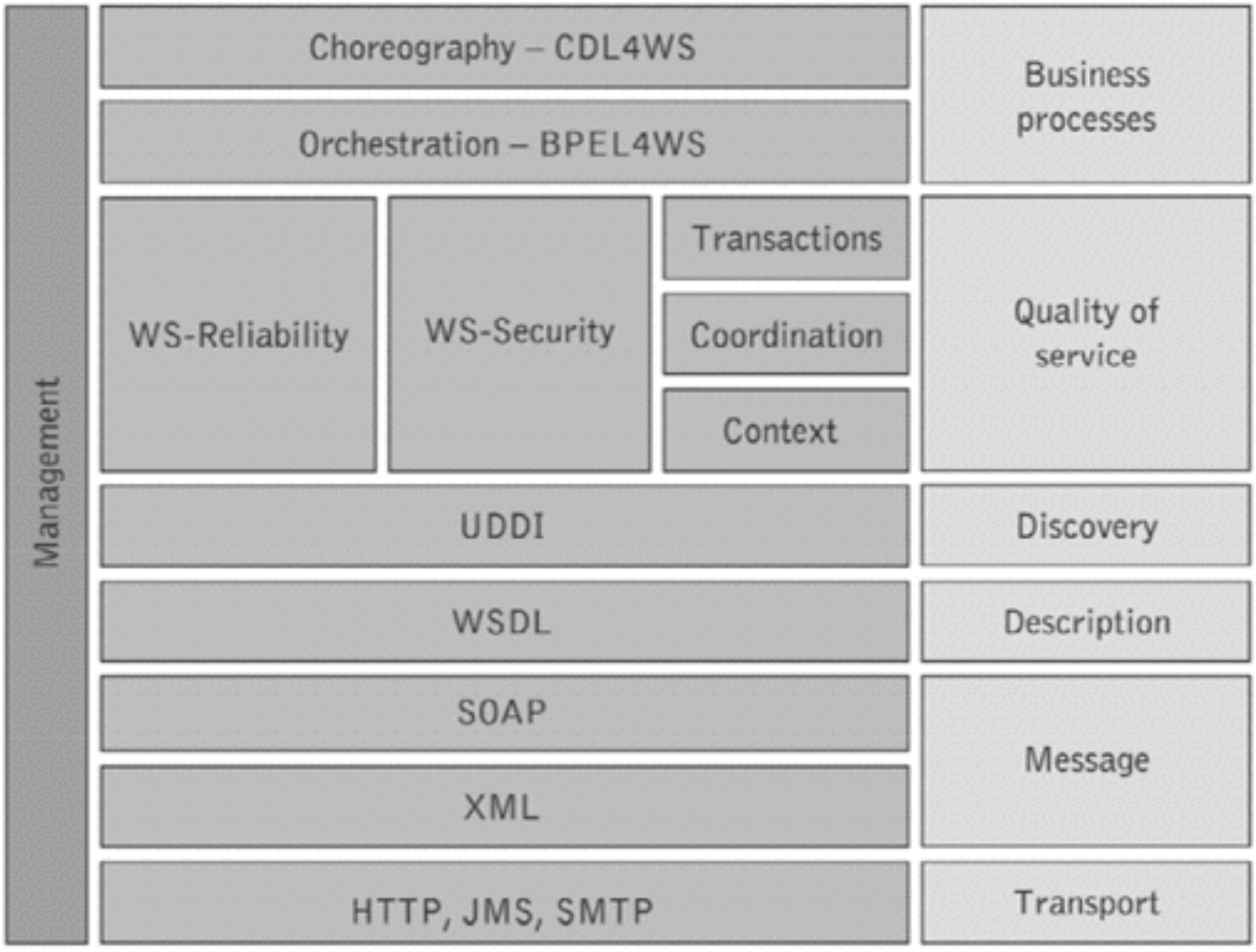
XML based messaging

HTTP, HTTPS, SMTP

Network



WS-*


Choreography – CDL4WS
Orchestration – BPEL4WS
Business processes
WS-Reliability
WS-Security
Transactions
Coordination
Context
Quality of service
UDDI
Discovery
WSDL
Description
SOAP
XML
Message
HTTP, JMS, SMTP
Transport
Management

Web Services Standards Overview

Interoperability Issues

- Basic Profile 1.1 · WS-I · Final Specification
- Basic Profile 1.2 · WS-I · Working Group Draft
- Basic Profile 2.0 · WS-I · Working Group Draft
- Attachments Profile 1.0 · WS-I · Final Specification
- Simple SOAP Binding Profile 1.0 · WS-I · Final Specification
- Basic Security Profile 1.0 · WS-I · Board Approval Draft
- REL Token Profile 1.0 · WS-I · Working Group Draft
- SAML Token Profile 1.0 · WS-I · Working Group Draft
- Conformance Claim Attachment Mechanism (CCAM) 1.0 · WS-I · Final Specification
- Reliable Asynchronous Messaging Profile (RAMP) 1.0 · WS-I · Working Draft

Business Process Specifications

- Business Process Execution Language for Web Services 1.1 (BPEL4WS) · 1.1 · BEA Systems, IBM, Microsoft, SAP, Siebel Systems · OASIS-Standard
- WS-Choreography Model Overview · 1.0 · W3C · Working Draft
- Web Service Choreography Interface (WSCI) · 1.0 · W3C · Sun Microsystems, SAP, BEA Systems and Intalio · Note
- Web Service Choreography Description Language (CDL4WS) · 1.0 · W3C · Candidate Recommendation
- Business Process Execution Language for Web Services 2.0 (BPEL4WS) · 2.0 · OASIS, BEA Systems, IBM, Microsoft, SAP, Siebel Systems · Committee Draft
- Business Process Management Language (BPML) · 1.1 · BPMI.org · Final Draft
- XML Process Definition Language (XPDL) · 2.0 · Final

Management Specifications

- Management Using Web Services (WSDM-MUWS) · 1.0 · OASIS · OASIS-Standard
- Management Of Web Services (WSDM-MOWS) · 1.0 · OASIS · OASIS-Standard
- WS-Management · AMD, Dell, Intel, Microsoft and Sun Microsystems · Published Specification
- Service Modeling Language · IBM, BEA, BMC, Cisco, Dell, HP, Intel, Microsoft, Sun · Draft Specification

Presentation Specifications

- Web Services for Remote Portlets (WSRP) · 2.0 · OASIS · Committee Draft

Metadata Specifications

- WS-Policy · 1.5 · W3C · Working Draft
- WS-PolicyAssertions · 1.1 · BEA Systems, IBM, Microsoft, SAP · Public Draft
- WS-PolicyAttachment · 1.2 · W3C · W3C Member Submission
- WS-Discovery · Microsoft, BEA Systems, Canon, Intel and webMethods · Draft
- WS-MetadataExchange · 1.1 · BEA Systems, Computer Associates, IBM, Microsoft, SAP, Sun Microsystems and webMethods · Public Draft
- Universal Description, Discovery and Integration (UDDI) · 3.0.2 · OASIS · OASIS-Standard
- Web Service Description Language 2.0 SOAP Binding · 2.0 · W3C · Working Draft
- Web Service Description Language 2.0 Core · 2.0 · W3C · Candidate Recommendation
- Web Service Description Language 1.1 · 1.1 · W3C · Note

Reliability Specifications

- WS-ReliableMessaging · 1.1 · OASIS · Committee Draft
- WS-Reliable Messaging Policy Assertion (WS-RM Policy) · 1.1 · OASIS · Committee Draft
- WS-Reliability · 1.1 · OASIS · OASIS-Standard

Security Specifications

- WS-Security · 1.1 · OASIS · OASIS-Standard
- WS-SecurityPolicy · 1.1 · IBM, Microsoft, RSA Security, VeriSign · Public Draft
- WS-Security: SOAP Message Security · 1.1 · OASIS · Public Review Draft
- WS-Security: Username Token Profile · 1.1 · OASIS · Public Review Draft
- WS-Security: Kerberos Binding · 1.0 · Microsoft, IBM, OASIS · Working Draft
- WS-Federation · 1.0 · IBM, Microsoft, BEA Systems, RSA Security, and VeriSign · Initial Draft
- WS-Security: SAML Token Profile · 1.1 · OASIS · Public Review Draft
- WS-Trust · 1.0 · BEA Systems, Computer Associates, IBM, Layer 7 Technologies, Microsoft, Netegrity, Oblix, OpenNetwork, Ping Identity Corporation, Reactivity, RSA Security, VeriSign and Westbridge Technology · Initial Draft
- WS-Security: X.509 Certificate Token Profile · 1.1 · OASIS · Public Review Draft
- WS-SecureConversation · BEA Systems, Computer Associates, IBM, Layer 7 Technologies, Microsoft, Netegrity, Oblix, OpenNetwork, Ping Identity Corporation, Reactivity, RSA Security, VeriSign and Westbridge Technology · Public Draft

Transaction Specifications

- WS-Coordination · 1.1 · OASIS · Working Draft
- WS-Business Activity · 1.1 · OASIS · Working Draft
- WS-Atomic Transaction · 1.1 · OASIS · Committee Draft
- WS-Composite Application Framework (WS-CAF) · 1.0 · Arjuna Technologies, Fujitsu, IONA, Oracle and Sun Microsystems · Committee Draft
- WS-Context (WS-CTX) · 1.0 · Arjuna Technologies, Fujitsu, IONA, Oracle and Sun Microsystems · Committee Draft
- WS-Coordination Framework (WS-CF) · 1.0 · Arjuna Technologies, Fujitsu, IONA, Oracle and Sun Microsystems · Committee Draft
- WS-Transaction Management (WS-TXM) · 1.0 · Arjuna Technologies, Fujitsu, IONA, Oracle and Sun Microsystems · Committee Draft

Resource Specifications

- Web Services Resource Framework (WSRF) · 1.2 · OASIS · OASIS-Standard
- WS-BaseFaults (WSRF) · 1.2 · OASIS · Working Draft
- WS-ServiceGroup (WSRF) · 1.2 · OASIS · Working Draft
- WS-ResourceProperties · 1.2 · OASIS · Working Draft
- WS-ResourceLifetime · 1.2 · OASIS · Working Draft
- WS-Transfer · W3C · W3C Member Submission
- Resource Representation SOAP Header Block (RRSHB) · W3C · Recommendation

Messaging Specifications

- WS-Notification · 1.3 · OASIS · OASIS-Standard
- WS-BrokeredNotification · 1.3 · OASIS · OASIS-Standard
- WS-BaseNotification · 1.3 · OASIS · OASIS-Standard
- WS-Eventing · W3C · Public Draft
- WS-Addressing – Core · 1.0 · W3C · Recommendation
- WS-Enumeration · Systinet, Microsoft, Sonic Software, BEA Systems and Computer Associates · Public Draft
- WS-Topics · 1.3 · OASIS · OASIS-Standard
- WS-Addressing – WSDL Binding · 1.0 · W3C · Candidate Recommendation
- WS-Addressing – SOAP Binding · 1.0 · W3C · Recommendation

SOAP

- SOAP · 1.2 · W3C · Recommendation
- SOAP Message Transmission Optimization Mechanism (MTOM) · 1.0 · W3C · Recommendation
- SOAP · 1.1 · W3C · Note

XML Specifications

- XML 1.1 · 1.1 · W3C · Recommendation
- XML 1.0 · 1.0 · W3C · Recommendation
- Namespaces in XML · 1.1 · W3C · Recommendation
- XML Information Set · 1.0 · W3C · Recommendation
- XML Schema · 1.1 · W3C · Working Draft
- XML binary Optimized Packaging (XOP) · 1.0 · W3C · Recommendation
- Describing Media Content of Binary Data in XML (DMCBDX) · W3C · Note

Standards Bodies

- OASIS – The Organization for the Advancement of Structured Information Standards (OASIS) is a not-for-profit, international consortium that drives the development, convergence, and adoption of e-business standards.
- W3C – The World Wide Web Consortium (W3C) was created in October 1994 to lead the World Wide Web to its full potential by developing common protocols that promote its evolution and ensure its interoperability.
- WS-I – The Web Services Interoperability Organization (WS-I) is an open industry organization chartered to promote Web services interoperability across platforms, operating systems and programming languages.
- IETF – The Internet Engineering Task Force (IETF) is a large open international community of network designers, operators, vendors, and researchers concerned with the evolution of the Internet architecture and the smooth operation of the Internet.

Dependencies

Messaging Specifications
SOAP 1.1 · SOAP 1.2 · SOAP Message Transmission Optimization Mechanism · WS-Notification · WS-BaseNotification · WS-Topics · WS-BrokeredNotification · WS-Addressing – Core · WS-Addressing – WSDL Binding · WS-Addressing – SOAP Binding · WS-Eventing · WS-Enumeration
(Resource · Security · Metadata)

Metadata Specifications
WS-Policy · WS-PolicyAssertions · WS-PolicyAttachment · WS-Discovery · WS-MetadataExchange · Universal Description, Discovery and Integration · Web Service Description Language 1.1 · Web Service Description Language 2.0 Core · Web Service Description Language 2.0 SOAP Binding
(Security · Messaging)

Security Specifications
WS-Security · WS-Security: SOAP Message Security · WS-Security: Kerberos Binding · WS-Security: SAML Token Profile · WS-Security: X.509 Certificate Token Profile · WS-Security: Username Token Profile · WS-SecurityPolicy · WS-Trust · WS-Federation · WS-SecureConversation
(Reliability · Messaging · Metadata)

Reliability Specifications
WS-ReliableMessaging · WS-Reliability · WS-Reliable Messaging Policy Assertion
(Transaction · Basic Profile · Security · Metadata)

Resource Specifications
Web Service Resource Framework · WS-BaseFaults · WS-ServiceGroup · WS-ResourceProperties · WS-ResourceLifetime · WS-Transfer · Resource Representation SOAP Header Block (RRSHB)
(Transaction · Security · Messaging)

Management Specifications
WS-Management · Management Of Web Services · Management Using Web Services · Service Modeling Language
(Resource · Meta · Security · Messaging)

Business Process Specifications
Business Process Execution Language for Web Services · Web Service Choreography Description Language · Web Service Choreography Interface · WS-Choreography Model Overview · Business Process Management Language · Business Process Execution Language for Web Serv. 2.0 · XML Process Definition Language
(Reliability · Security · Messaging · Transaction)

Transaction Specifications
WS-Business Activity · WS-Atomic Transaction · WS-Coordination · WS-Composite Application Framework · WS-Transaction Management · WS-Context · WS-Coordination Framework
(Metadata · Messaging · Reliability · Security)

Presentation Specifications
Web Services for Remote Portlets
(Reliab. · Secur. · Mess.)

innoQ

innoQ Deutschland GmbH
Halskestraße 17
D-40880 Ratingen
Phone +49 21 02 77 162-100
info@innoq.com · www.innoq.com

innoQ Schweiz GmbH
Gewerbestrasse 11
CH-6330 Cham
Phone +41 41 743 01 11

Version 3.0 · February 2007

Birth of BPEL

IBM was pushing for a standard called WSFL

Microsoft was pushing for a technology called XLANG

Intalio was pushing for BPML

IBM and Microsoft merged their efforts and pushed together for BPEL (a hybrid WSFL+XLANG)
and BPEL was soon widely adopted

Life of BPEL

BPEL4WS 1.0 (2002) by BEA, IBM, Microsoft

SAP + Siebel joined the effort
BPEL 1.1 (2003)
submitted to OASIS

Adobe + HP + NEC + Oracle + Sun + many more joined
WS-BPEL 2.0 (2005)

The problem with BPEL

BPEL is not a graphical language

BPEL is an XML dialect

Machines like XML
Humans being should not like XML

A typical BPEL tutorial

Turn to page 4 of any BPEL tutorial
(the first couple of pages are just a verbal introduction)
and you find the first small example...

... of about two pages of formatted XML code

```
<process name="purchaseOrderProcess" targetNamespace="http://example.com/ws-bp/purchase" xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable" xmlns:lns="http://manufacturing.org/wsdl/purchase">

   <documentation xml:lang="EN">
      A simple example of a WS-BPEL process for handling a purchase order.
   </documentation>

   <partnerLinks>
      <partnerLink name="purchasing" partnerLinkType="lns:purchasingLT" myRole="purchaseService" />
      <partnerLink name="invoicing" partnerLinkType="lns:invoicingLT" myRole="invoiceRequester" partnerRole="invoiceService" />
      <partnerLink name="shipping" partnerLinkType="lns:shippingLT" myRole="shippingRequester" partnerRole="shippingService" />
      <partnerLink name="scheduling" partnerLinkType="lns:schedulingLT" partnerRole="schedulingService" />
   </partnerLinks>

   <variables>
      <variable name="PO" messageType="lns:POMessage" />
      <variable name="Invoice" messageType="lns:InvMessage" />
      <variable name="shippingRequest" messageType="lns:shippingRequestMessage" />
      <variable name="shippingInfo" messageType="lns:shippingInfoMessage" />
      <variable name="shippingSchedule" messageType="lns:scheduleMessage" />
   </variables>

   <faultHandlers>
      <catch faultName="lns:cannotCompleteOrder" faultVariable="POFault" faultMessageType="lns:orderFaultType">
         <reply partnerLink="purchasing" portType="lns:purchaseOrderPT" operation="sendPurchaseOrder" variable="POFault" faultName="cannotCompleteOrder" />
      </catch>
   </faultHandlers>

   <sequence>
      <receive partnerLink="purchasing" portType="lns:purchaseOrderPT" operation="sendPurchaseOrder" variable="PO" createInstance="yes">
         <documentation>Receive Purchase Order</documentation>
      </receive>

      <flow>
         <documentation>
            A parallel flow to handle shipping, invoicing and scheduling
         </documentation>
         <links>
            <link name="ship-to-invoice" />
            <link name="ship-to-scheduling" />
         </links>
         <sequence>
            <assign>
               <copy>
                  <from>$PO.customerInfo</from>
                  <to>$shippingRequest.customerInfo</to>
               </copy>
            </assign>
            <invoke partnerLink="shipping" portType="lns:shippingPT" operation="requestShipping" inputVariable="shippingRequest" outputVariable="shippingInfo">
               <documentation>Decide On Shipper</documentation>
               <sources>
                  <source linkName="ship-to-invoice" />
               </sources>
            </invoke>
            <receive partnerLink="shipping" portType="lns:shippingCallbackPT" operation="sendSchedule" variable="shippingSchedule">
               <documentation>Arrange Logistics</documentation>
               <sources>
                  <source linkName="ship-to-scheduling" />
               </sources>
            </receive>
         </sequence>
         <sequence>
            <invoke partnerLink="invoicing" portType="lns:computePricePT" operation="initiatePriceCalculation" inputVariable="PO">
               <documentation>
                  Initial Price Calculation
               </documentation>
            </invoke>
            <invoke partnerLink="invoicing" portType="lns:computePricePT" operation="sendShippingPrice" inputVariable="shippingInfo">
               <documentation>
                  Complete Price Calculation
               </documentation>
               <targets>
                  <target linkName="ship-to-invoice" />
               </targets>
            </invoke>
            <receive partnerLink="invoicing" portType="lns:invoiceCallbackPT" operation="sendInvoice" variable="Invoice" />
         </sequence>
         <sequence>
            <invoke partnerLink="scheduling" portType="lns:schedulingPT" operation="requestProductionScheduling" inputVariable="PO">
               <documentation>
                  Initiate Production Scheduling
               </documentation>
            </invoke>
            <invoke partnerLink="scheduling" portType="lns:schedulingPT" operation="sendShippingSchedule" inputVariable="shippingSchedule">
               <documentation>
                  Complete Production Scheduling
               </documentation>
               <targets>
                  <target linkName="ship-to-scheduling" />
               </targets>
            </invoke>
         </sequence>
      </flow>
      <reply partnerLink="purchasing" portType="lns:purchaseOrderPT" operation="sendPurchaseOrder" variable="Invoice">
         <documentation>Invoice Processing</documentation>
      </reply>
   </sequence>
</process>
```

Learning the syntax

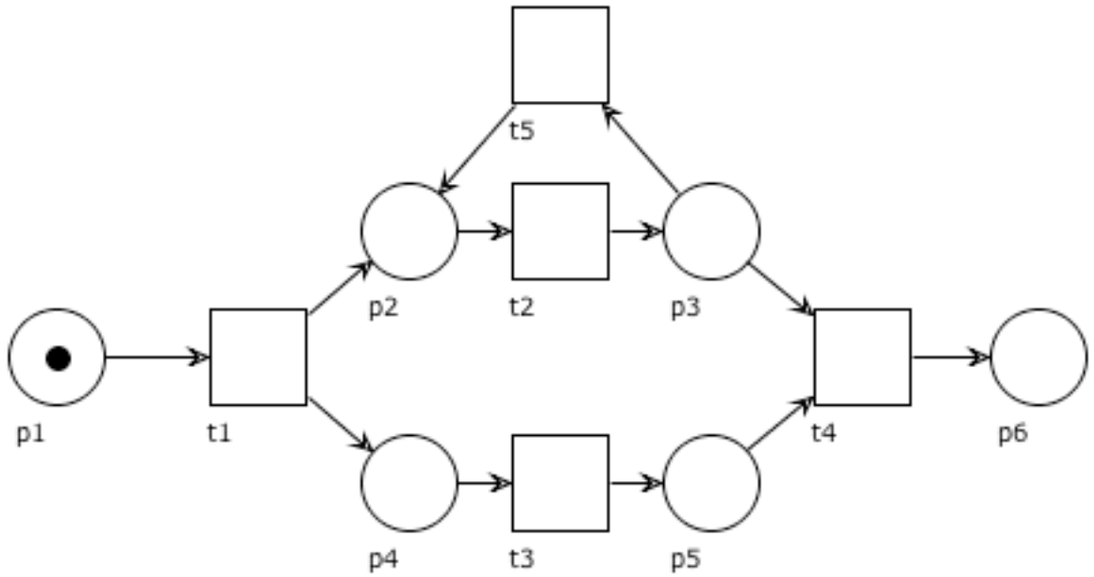
Learning BPEL by looking at XML documents

is like

learning Petri nets by looking at PNML documents

or similar to

learning Java by looking at the bytecode




```
<?xml version="1.0" encoding="UTF-8"?><pnml>
  <net type="http://www.informatik.hu-berlin.de/top/pntd/ptNetb" id="noID">
    <place id="p6">
      <name>
        <text>p6</text>
        <graphics>
          <offset x="430" y="270"/>
        </graphics>
      </name>
      <graphics>
        <position x="430" y="230"/>
        <dimension x="40" y="40"/>
      </graphics>
    </place>
    <place id="p5">
      <name>
        <text>p5</text>
        <graphics>
          <offset x="300" y="320"/>
        </graphics>
      </name>
      <graphics>
        <position x="300" y="280"/>
        <dimension x="40" y="40"/>
      </graphics>
    </place>
    <place id="p4">
      <name>
        <text>p4</text>
        <graphics>
          <offset x="180" y="320"/>
        </graphics>
      </name>
      <graphics>
        <position x="180" y="280"/>
        <dimension x="40" y="40"/>
      </graphics>
    </place>
    <place id="p3">
      <name>
        <text>p3</text>
        <graphics>
          <offset x="300" y="220"/>
        </graphics>
      </name>
      <graphics>
        <position x="300" y="180"/>
        <dimension x="40" y="40"/>
      </graphics>
    </place>
    <place id="p2">
      <name>
        <text>p2</text>
        <graphics>
          <offset x="180" y="220"/>
        </graphics>
      </name>
      <graphics>
        <position x="180" y="180"/>
        <dimension x="40" y="40"/>
      </graphics>
    </place>
    <place id="p1">
      <name>
        <text>p1</text>
        <graphics>
          <offset x="40" y="270"/>
        </graphics>
      </name>
      <graphics>
        <position x="40" y="230"/>
        <dimension x="40" y="40"/>
      </graphics>
      <initialMarking>
        <text>1</text>
      </initialMarking>
    </place>
    <transition id="t3">
      <name>
        <text>t3</text>
        <graphics>
          <offset x="240" y="320"/>
        </graphics>
      </name>
      <graphics>
        <position x="240" y="280"/>
        <dimension x="40" y="40"/>
      </graphics>
      <toolspecific tool="WoPeD" version="1.0">
        <time>0</time>
        <timeUnit>1</timeUnit>
        <orientation>1</orientation>
      </toolspecific>
    </transition>
    <transition id="t2">
      <name>
        <text>t2</text>
        <graphics>
          <offset x="240" y="220"/>
        </graphics>
      </name>
      <graphics>
        <position x="240" y="180"/>
        <dimension x="40" y="40"/>
      </graphics>
      <toolspecific tool="WoPeD" version="1.0">
        <time>0</time>
        <timeUnit>1</timeUnit>
        <orientation>1</orientation>
      </toolspecific>
    </transition>
    <transition id="t1">
      <name>
        <text>t1</text>
        <graphics>
          <offset x="120" y="270"/>
        </graphics>
      </name>
      <graphics>
        <position x="120" y="230"/>
        <dimension x="40" y="40"/>
      </graphics>
      <toolspecific tool="WoPeD" version="1.0">
        <time>0</time>
        <timeUnit>1</timeUnit>
        <orientation>1</orientation>
      </toolspecific>
    </transition>
    <transition id="t4">
      <name>
        <text>t4</text>
        <graphics>
          <offset x="360" y="270"/>
        </graphics>
      </name>
      <graphics>
        <position x="360" y="230"/>
        <dimension x="40" y="40"/>
      </graphics>
      <toolspecific tool="WoPeD" version="1.0">
        <time>0</time>
        <timeUnit>1</timeUnit>
        <orientation>1</orientation>
      </toolspecific>
    </transition>
    <transition id="t5">
      <name>
        <text>t5</text>
        <graphics>
          <offset x="240" y="150"/>
        </graphics>
      </name>
      <graphics>
        <position x="240" y="110"/>
        <dimension x="40" y="40"/>
      </graphics>
      <toolspecific tool="WoPeD" version="1.0">
        <time>0</time>
        <timeUnit>1</timeUnit>
        <orientation>1</orientation>
      </toolspecific>
    </transition>
    <arc id="a9" source="t1" target="p4">
      <inscription>
        <text>1</text>
      </inscription>
      <graphics/>
      <toolspecific tool="WoPeD" version="1.0">
        <probability>1.0</probability>
        <displayProbabilityOn>false</displayProbabilityOn>
        <displayProbabilityPosition x="500.0" y="0.0"/>
      </toolspecific>
    </arc>
    <arc id="a17" source="p3" target="t4">
      <inscription>
        <text>1</text>
      </inscription>
      <graphics/>
      <toolspecific tool="WoPeD" version="1.0">
        <probability>1.0</probability>
        <displayProbabilityOn>false</displayProbabilityOn>
        <displayProbabilityPosition x="500.0" y="0.0"/>
      </toolspecific>
    </arc>
    <arc id="a14" source="p5" target="t4">
      <inscription>
        <text>1</text>
      </inscription>
      <graphics/>
      <toolspecific tool="WoPeD" version="1.0">
        <probability>1.0</probability>
        <displayProbabilityOn>false</displayProbabilityOn>
        <displayProbabilityPosition x="500.0" y="0.0"/>
      </toolspecific>
    </arc>
    <arc id="a21" source="p3" target="t5">
      <inscription>
        <text>1</text>
      </inscription>
      <graphics/>
      <toolspecific tool="WoPeD" version="1.0">
        <probability>1.0</probability>
        <displayProbabilityOn>false</displayProbabilityOn>
        <displayProbabilityPosition x="500.0" y="0.0"/>
      </toolspecific>
    </arc>
    <arc id="a13" source="t3" target="p5">
      <inscription>
        <text>1</text>
      </inscription>
      <graphics/>
      <toolspecific tool="WoPeD" version="1.0">
        <probability>1.0</probability>
        <displayProbabilityOn>false</displayProbabilityOn>
        <displayProbabilityPosition x="500.0" y="0.0"/>
      </toolspecific>
    </arc>
    <arc id="a10" source="p4" target="t3">
      <inscription>
        <text>1</text>
      </inscription>
      <graphics/>
      <toolspecific tool="WoPeD" version="1.0">
        <probability>1.0</probability>
        <displayProbabilityOn>false</displayProbabilityOn>
        <displayProbabilityPosition x="500.0" y="0.0"/>
      </toolspecific>
    </arc>
    <arc id="a24" source="t5" target="p2">
      <inscription>
        <text>1</text>
      </inscription>
      <graphics/>
      <toolspecific tool="WoPeD" version="1.0">
        <probability>1.0</probability>
        <displayProbabilityOn>false</displayProbabilityOn>
        <displayProbabilityPosition x="500.0" y="0.0"/>
      </toolspecific>
    </arc>
    <arc id="a1" source="p1" target="t1">
      <inscription>
        <text>1</text>
      </inscription>
      <graphics/>
      <toolspecific tool="WoPeD" version="1.0">
        <probability>1.0</probability>
        <displayProbabilityOn>false</displayProbabilityOn>
        <displayProbabilityPosition x="500.0" y="0.0"/>
      </toolspecific>
    </arc>
    <arc id="a2" source="t1" target="p2">
      <inscription>
        <text>1</text>
      </inscription>
      <graphics/>
      <toolspecific tool="WoPeD" version="1.0">
        <probability>1.0</probability>
        <displayProbabilityOn>false</displayProbabilityOn>
        <displayProbabilityPosition x="500.0" y="0.0"/>
      </toolspecific>
    </arc>
    <arc id="a5" source="p2" target="t2">
      <inscription>
        <text>1</text>
      </inscription>
      <graphics/>
      <toolspecific tool="WoPeD" version="1.0">
        <probability>1.0</probability>
        <displayProbabilityOn>false</displayProbabilityOn>
        <displayProbabilityPosition x="500.0" y="0.0"/>
      </toolspecific>
    </arc>
    <arc id="a6" source="t2" target="p3">
      <inscription>
        <text>1</text>
      </inscription>
      <graphics/>
      <toolspecific tool="WoPeD" version="1.0">
        <probability>1.0</probability>
        <displayProbabilityOn>false</displayProbabilityOn>
        <displayProbabilityPosition x="500.0" y="0.0"/>
      </toolspecific>
    </arc>
    <arc id="a18" source="t4" target="p6">
      <inscription>
        <text>1</text>
      </inscription>
      <graphics/>
      <toolspecific tool="WoPeD" version="1.0">
        <probability>1.0</probability>
        <displayProbabilityOn>false</displayProbabilityOn>
        <displayProbabilityPosition x="500.0" y="0.0"/>
      </toolspecific>
    </arc>
    <toolspecific tool="WoPeD" version="1.0">
      <bounds>
        <position x="11" y="33"/>
        <dimension x="755" y="490"/>
      </bounds>
      <treeWidth>2</treeWidth>
      <verticalLayout>false</verticalLayout>
      <resources/>
      <simulations/>
      <partnerLinks/>
      <variables/>
    </toolspecific>
  </net>
</pnml>
```

The source of the problem

BPEL is designed to work with WSDL documents
of the services required by the process

A process can itself be exposed as a service
which needs its own WSDL document

For us:
we can forget that WSDL documents are written in XML
we regard them as abstract interface descriptions

BPEL guidelines

Structured control vs free flow

BPEL4WS should provide
both **structured** (hierarchical)
and **graph-like** control regimes,
and allow their usage to be blended
as seamlessly as possible.

About data handling

BPEL4WS provides **limited data handling** functions
that are sufficient for the simple manipulation of data
that is needed to define
process relevant data and control flow.

Abstract vs executable

BPEL4WS should define a set of Web service orchestration concepts that are meant to be used in common by both the external (abstract) and internal (executable) views of a business process.

Such a business process defines the **behavior of a single autonomous entity**, typically operating in interaction with other similar peer entities.

It is recognized that each usage view will require a few specialized extensions, but these extensions are to be kept to a minimum and tested against requirements

WSDL preliminaries

Service

A service can be thought of as a container
for a set of (logically related) operations
that are made available via web-based protocols

Roughly: a remote object

PortType / Interface

The **<portType>** element,
renamed to **<interface>** in WSDL 2.0,
defines a web service,
the operations that can be performed,
and the messages that are used to perform the operation.

Roughly: the type of a remote object

i.e., a remote (abstract) class

Operation

Each operation can be thought of as
a method or function call in some programming language.

Four kinds of operations
(one-way, request-response, notification, solicit-response)
Three kinds of parameters/arguments
(input, output, fault)
(not all combinations allowed)

Roughly: a remote (abstract) method

Port / Endpoint

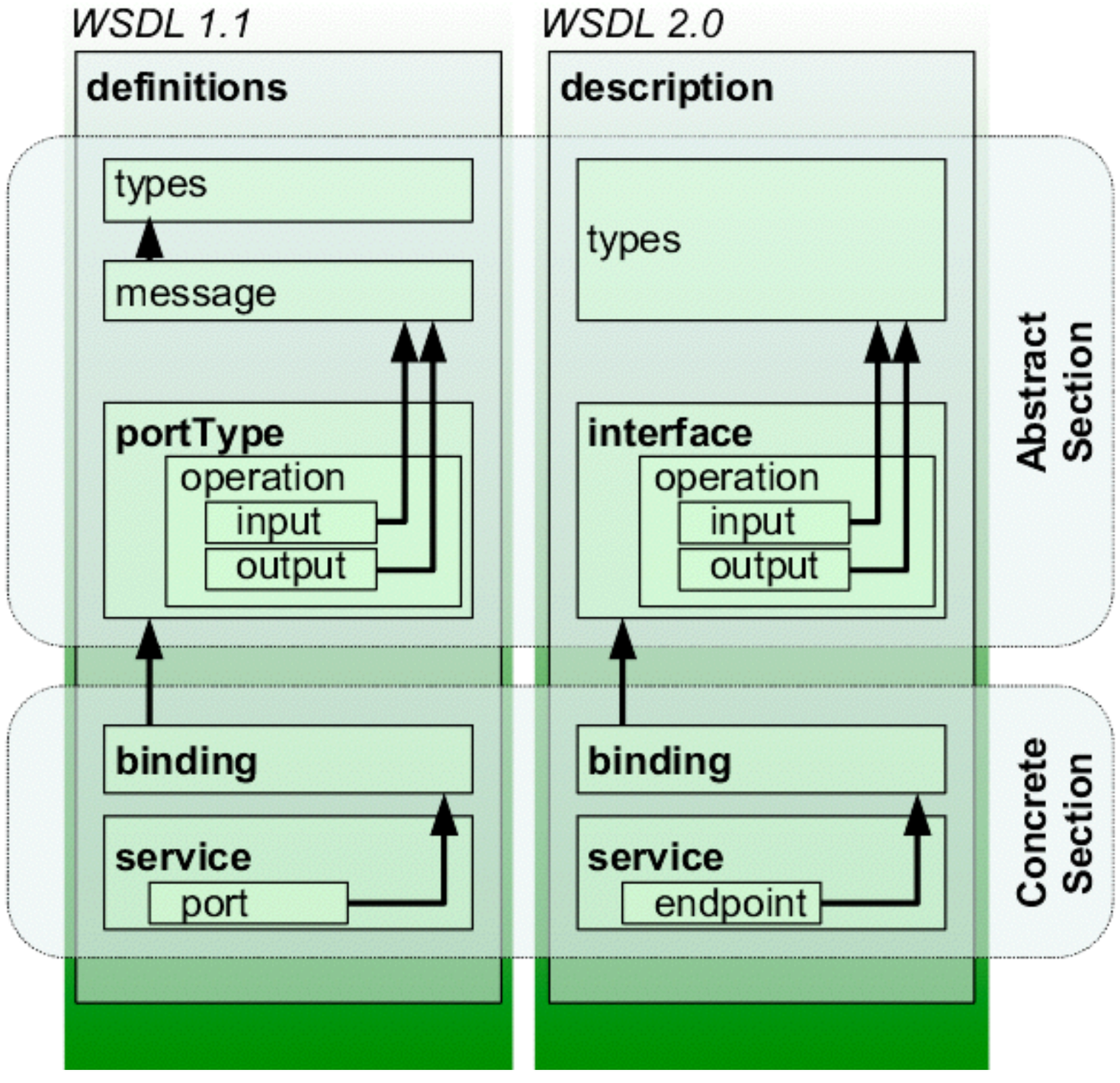
The **<port>** element,
renamed to **<endpoint>** in WSDL 2.0,
declares the address of a web service.

It typically involves a **name**, a **binding** and a **URL**

Binding

The binding specifies the interface as well as the SOAP binding style (message format) and SOAP transport protocol (http / smtp).

WSDL (from wikipedia)



```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="PurchaseExample"
    targetNamespace="http://www.fluidimagination.com/sams/PurchaseExample.wsdl"
    xmlns:tns="http://www.fluidimagination.com/sams/PurchaseExample.wsdl"
    xmlns:soap="http://www.schemas.xmlsoap.org/wsdl/soap/"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">

    <wsdl:types>
        <xsd:schema
                targetNamespace="http://www.fluidimagination.com/sams/productType.wsdl"
                xmlns:xsd="http://www.w3.org/2001/XMLSchema">
            <xsd:complexType name="scannerType">
                <xsd:all>
                    <xsd:element name="upc" type="upcType"/>
                    <xsd:element name="isbn" type="isbnType"/>
                </xsd:all>
            </xsd:complexType>
            <xsd:simpleType name="upcType">
                <xsd:restriction base="xsd:string">
                    <xsd:pattern value="[0-9]{12}"/>
                </xsd:restriction>
            </xsd:simpleType>
            <xsd:simpleType name="isbnType">
                <xsd:restriction base="xsd:string">
                    <xsd:pattern value="([0-9]- ){10}"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:schema>
    </wsdl:types>
    <!-- Adding a message that has two addresses -->
    <wsdl:message name="purchaseMessage">
        <wsdl:part name="productCode" element="tns:scannerType"/>
    </wsdl:message>
    <!--create a port type with one operation -->
    <wsdl:portType name="purchaseType">
        <wsdl:operation name="purchaseOperation">
            <wsdl:input name="tns:purchaseMessage"/>
        </wsdl:operation>
    </wsdl:portType>
    <!--Bind the message to SOAP using HTTP -->
    <wsdl:binding name="purchaseBinding" type="tns:purchaseType">
        <soap:binding style="document"
                transport="http://schemas.xmlsoap.org/soap/http"/>
        <wsdl:operation name="tns:purchaseOperation">
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
        </wsdl:operation>
    </wsdl:binding>
    <!--Bind the message to SOAP over  SMTP -->
    <wsdl:binding name="purchaseBindingSMTP" type="tns:purchaseType">
        <soap:binding style="document"
                transport="http://schemas.xmlsoap.org/soap/smtp"/>
        <wsdl:operation name="tns:purchaseOperation">
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
        </wsdl:operation>
    </wsdl:binding>
</wsdl:definitions>
```

BPEL ingredients

BPEL ingredients

Data flow
(scoped variables)

Partner links and Message correlation

Message flow
(one-way, request-response, notify, solicit-response)

Control flow
(structured activities and synchronization links)

Handling events, faults, compensations

Variable

Variables can be defined (within a local scope)

The activity **<assign>** can be used to copy data (messages, part of messages, service references) between variables

```
<assign>
      <copy>
            <from variable="PO" part="customerInfo"/>
            <to variable="shippingRequest" part="customerInfo"/>
      </copy>
</assign>
```

Stateless services, stateful processes

When a message for (WS-BPEL) service arrives,
it must be delivered either to a new
or to an existing instance of the process

Stateful business processes are instantiated to act according to interaction history

Messages should not only be delivered to the correct port, but also to the correct instance of the business process that provides that port

Message correlation

Message correlation is the way to tie together messages coming from different communications

A correlation set is a set of properties such that all messages having the same values of all properties are part of the same interaction

The partner that first fixes the values of the properties in the correlation set is the **initiator** of the exchange, the other partners are called the **followers**

Message flow

Basic activities are available
to send and receive messages to partners

Activity **`<invoke>`**: asynchronous (one-way) or synchronous (request-response)

Activity **`<receive>`**: a request from a partner to execute one of the (WSDL) operations implemented by the process

Activity **`<reply>`**: to return the result of a **`<receive>`**d synchronous request-response operation

Partner Link

A partner is a service that the process invokes,
or a client that invokes the process

A BPEL process interacts with a partner using a
<partnerLink> a (typed) connector
that the process offers to/requires from its partner
(to be declared in the BPEL document)

```
<partnerLinks>
    <partnerLink name="shipping"
                 partnerLinkType="lns:shippingLT"
                 myRole="shippingRequester"
                 partnerRole="shippingService"/>
    ...
</partnerLinks>
```

Invoke

Needed information: the **<partnerLink>**,
the WSDL **<portType>** of the service to be invoked, and
the name and parameters of the **<operation>**

```
<invoke partnerLink="shipping"
        portType="lns:shippingPT"
        operation="requestShipping"
        inputVariable="shippingRequest"
        outputVariable="shippingInfo">
        <source linkName="ship-to-invoice"/>
</invoke>
```

Receive

Needed information: the **<partnerLink>**, the WSDL **<portType>** of the exposed service, and a **<variable>** where to copy the parameters of the **<operation>**

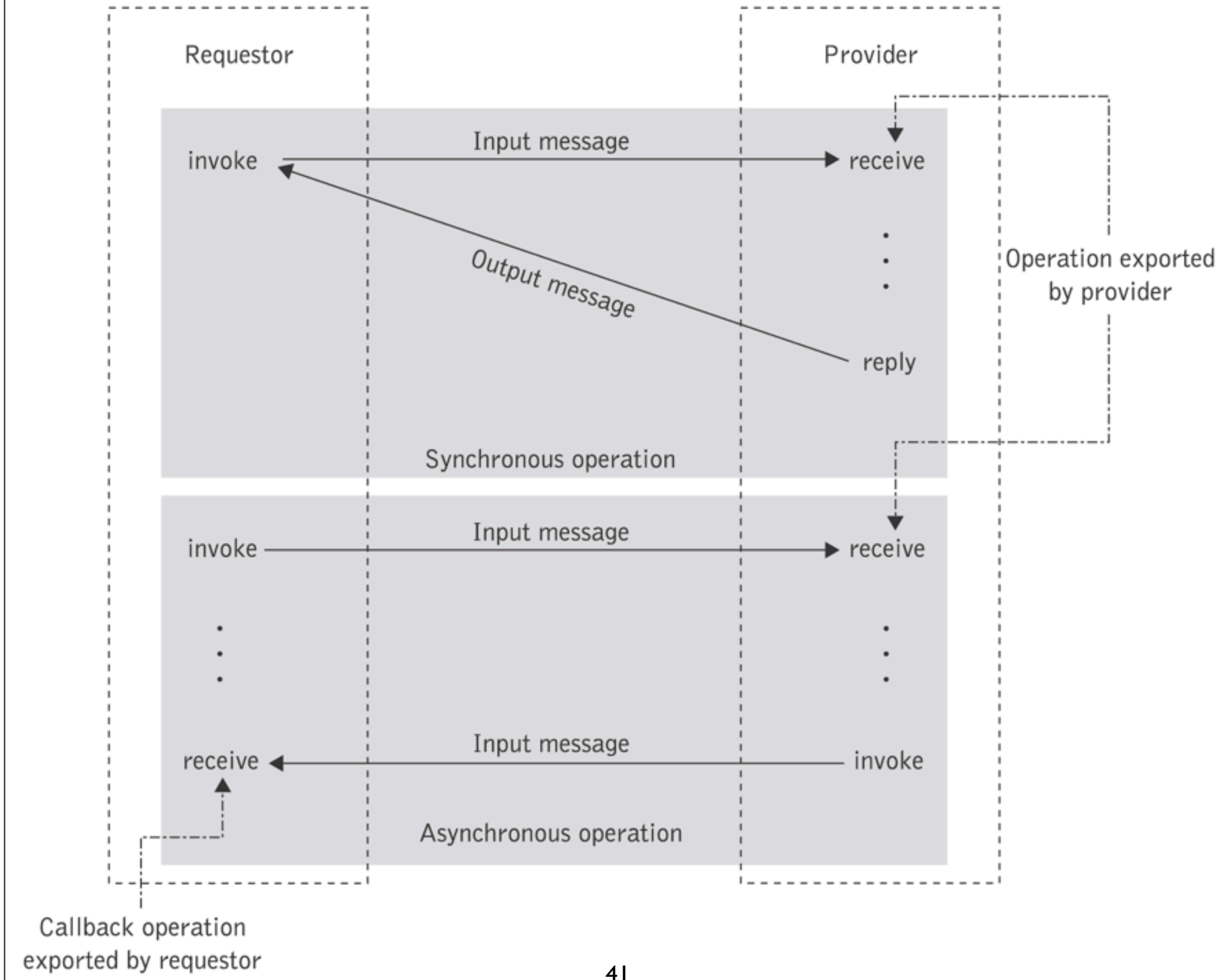
```
<receive partnerLink="purchasing"
         portType="lns:purchaseOrderPT"
         operation="sendPurchaseOrder"
         variable="PO">
</receive>
```

Reply

A process can **<reply>** to a message it **<receive>**d

```
<reply partnerLink="purchasing"
       portType="lns:purchaseOrderPT"
       operation="sendPurchaseOrder"
       variable="Invoice" />
```

Asynchronous operations do not use **<reply>**
If a reply must be sent,
<invoke> is used to call back a client operation


Requestor
Provider
invoke
Input message
receive
Output message
reply
Operation exported
by provider
Synchronous operation
invoke
Input message
receive
receive
Input message
invoke
Asynchronous operation
Callback operation
exported by requestor

Structured activities

<sequence> for specifying sequential compositions

only one branch is selected

<switch> for (local) internal choices
(ordered list of conditional **<case>** branches,
possibly ended by an **<otherwise>** branch)

<pick> for (global) external choices
(set of event handlers of the form event → activity,
<onMessage> arrival of a message or **<onAlarm>** timer)

<flow> for parallel composition

<while> for iterations (guards are XPath expressions)

Control links

Control links are a non-structural element
that introduces control dependencies between activities

Each link carry a predicate, called "**transition condition**"

An activity can be the **source** of many links
(when the activity completes,
the transition conditions of all links are evaluated)

An activity can be the **target** of many links
(it waits for the boolean evaluation of the transition
conditions of incoming links and apply a "**join condition**")

Link

A **`<link>`** expresses synchronisation dependencies among activities in a process

Each **`<link>`** has a name,
one source activity, one target activity, and
it may be associated with a **transition condition**
(a predicate to be evaluated when the source activity ends)

Join condition

Any activity that is the target of one or more links
may have an explicit **<joinCondition>**,
(a predicate on the status values of the incoming links,
to be evaluated once all such values have been determined)

otherwise, the implicit join condition is the OR

If the **<joinCondition>** evaluates to:

TRUE the activity can be executed,

FALSE a **<joinFailure>** fault may be thrown
(depending on the **<suppressJoinFailure>** flag

Join condition failure

If the attribute **`suppressJoinFailure`** is set to **`no`**,
a join failure needs to be thrown,
which triggers a standard fault handling procedure

If the attribute **`suppressJoinFailure`** is set to **`yes`**,
the activity will not be performed,
will end up in the “finished” state,
(the processing of any following activity will not be affected)
and the status of all outgoing links will be set to **`false`**.
This is known as **dead path elimination**
(the false link status is propagated transitively along the paths formed by control links, until a join condition is reached that evaluates to true)

Scope

A scope provides fault and compensation handling capabilities to the activities nested within it

A **<scope>** activity consists of:
a, possibly structured, primary activity,

a set of (optional) fault handlers,

a single (optional) compensation handlers,

a set of (optional) event handlers
(executed concurrently with the process,
they enable a scope to react to messages and alarm events)

Faults

BPEL defines three kinds of faults:

application faults (also **service faults**)
generated by invoked services

process-defined faults
generated by a **`<throw>`** activity

system faults
generated by the process engine, such as join failures

“it is never possible to run more than one fault handler for the same scope, under any circumstances”

Formal Semantics and Analysis of Control Flow in WS-BPEL

(Revised Version)

Chun Ouyang[1], Eric Verbeek[2], Wil M.P. van der Aalst[2,1], Stephen Breutel[1], Marlon Dumas[1], and Arthur H.M. ter Hofstede[1]

[1] Faculty of Information Technology, Queensland University of Technology, GPO Box 2434, Brisbane QLD 4001, Australia
{c.ouyang,sw.breutel,m.dumas,a.terhofstede}@qut.edu.au

[2] Department of Technology Management, Eindhoven University of Technology, GPO Box 513, NL-5600 MB, The Netherlands
{h.m.w.verbeek,w.m.p.v.d.aalst}@tm.tue.nl

Formal semantics of control flow in BPEL

Motivation

BPEL specification:
rigorous XML syntax
English prose semantics (of apparent clarity)

Consequences:
inconsistencies, ambiguities, incompleteness

try to google for “WS BPEL issues list”, e.g.

Issue 32 Link Semantics in Event Handlers (resolved)
Issue 39 Inconsistent syntax for query attribute values in spec examples (resolved)

...

Issue 42 Need for Formalism (resolved) YES

Approaches

Promela (SPIN)

Process algebras

Abstract State Machines

Automata

Weakest preconditions / strongest postconditions

Axiomatic semantics

Petri nets

Goal

Unveil ambiguities in BPEL specification
(reported to BPEL standardization committee)

Complete formalization of all control-flow constructs

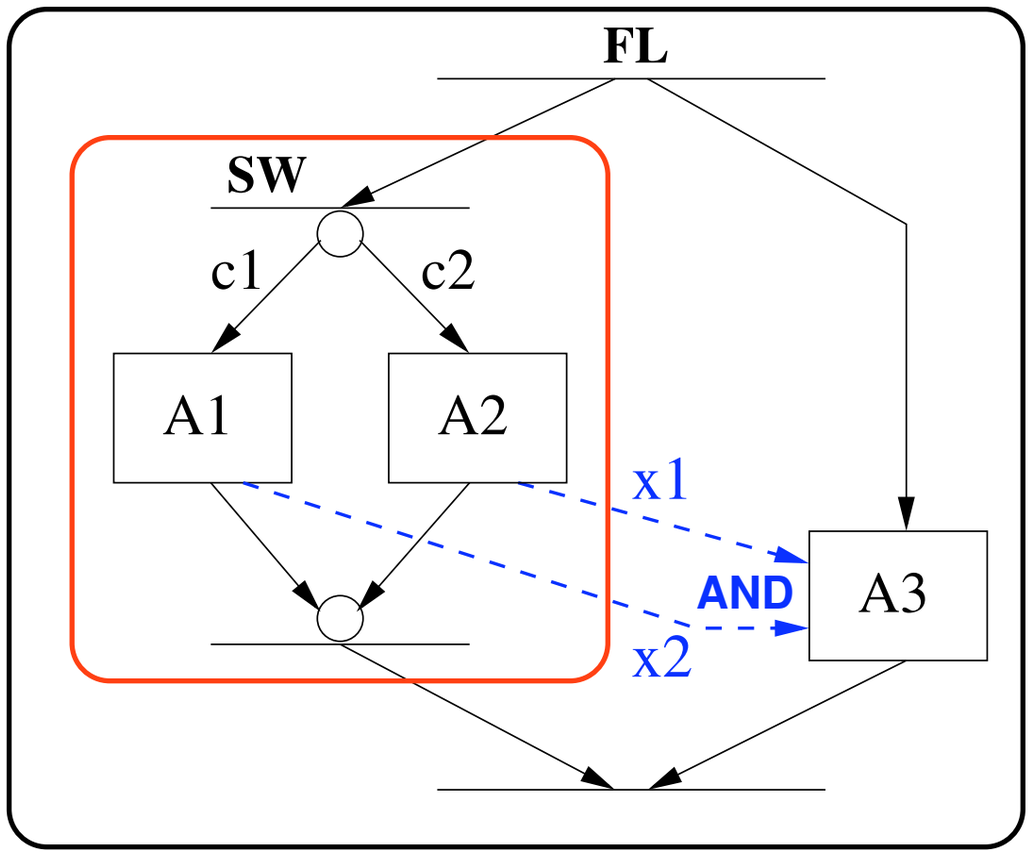
Checking for unreachable activities

Checking for potential conflicting message receipt actions

Determining which messages can be eventually consumed

Example: BPEL with unreachable activity

```
<process name="unreachableTask"
    targetNamespace="http://samples.otn.com"
    suppressJoinFailure="yes"
    xmlns:tns="http://samples.otn.com"
    xmlns:services="http://services.otn.com"
    xmlns="http://schemas.xmlsoap.org/ws/2003/03/business-process/">
  <flow name="FL" suppressJoinFailure="yes">
   <links>
     <link name="x1"/>
     <link name="x2"/>
   </links>
   <switch name="SW">
     <case>
       <invoke name="A1">
        <sources>  <source linkName="x1"/>   </sources>
       </invoke>
     </case>
     <otherwise>
       <invoke name="A2">
        <sources>  <source linkName="x2"/>   </sources>
       </invoke>
     </otherwise>
   </switch>
   <invoke name = "A3">
    <targets>
     <joinCondition>
       bpws:getLinkStatus('x1') and bpws:getLinkStatus('x2')
     </joinCondition>
     <target linkName="x1"/>
     <target linkName="x2"/>
    </targets>
   </invoke>
  </flow>
</process>
```



Basic Activity

Flow

Switch

Control Link

Basic activity X

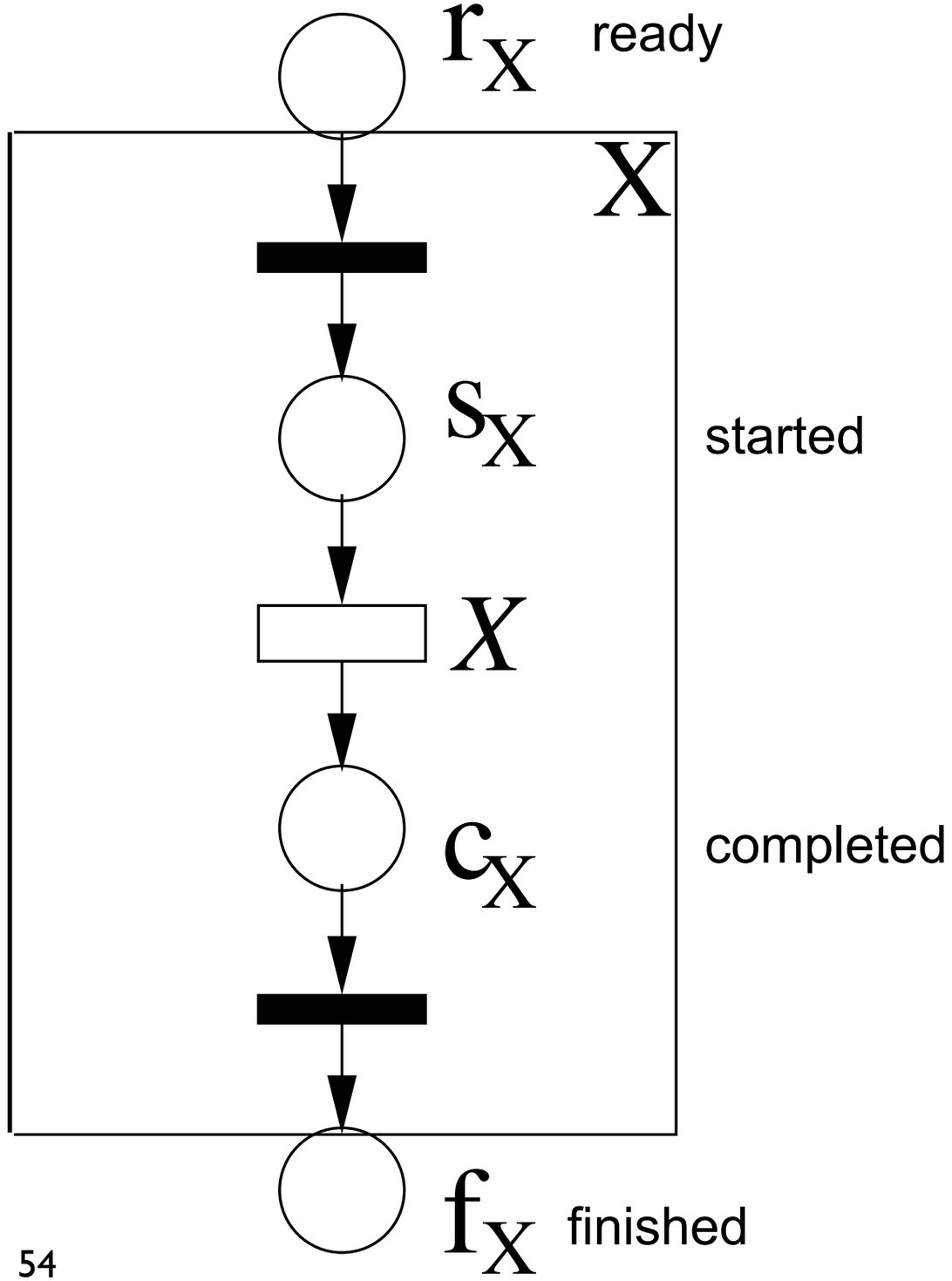
The activity can be

<assign>

<invoke>

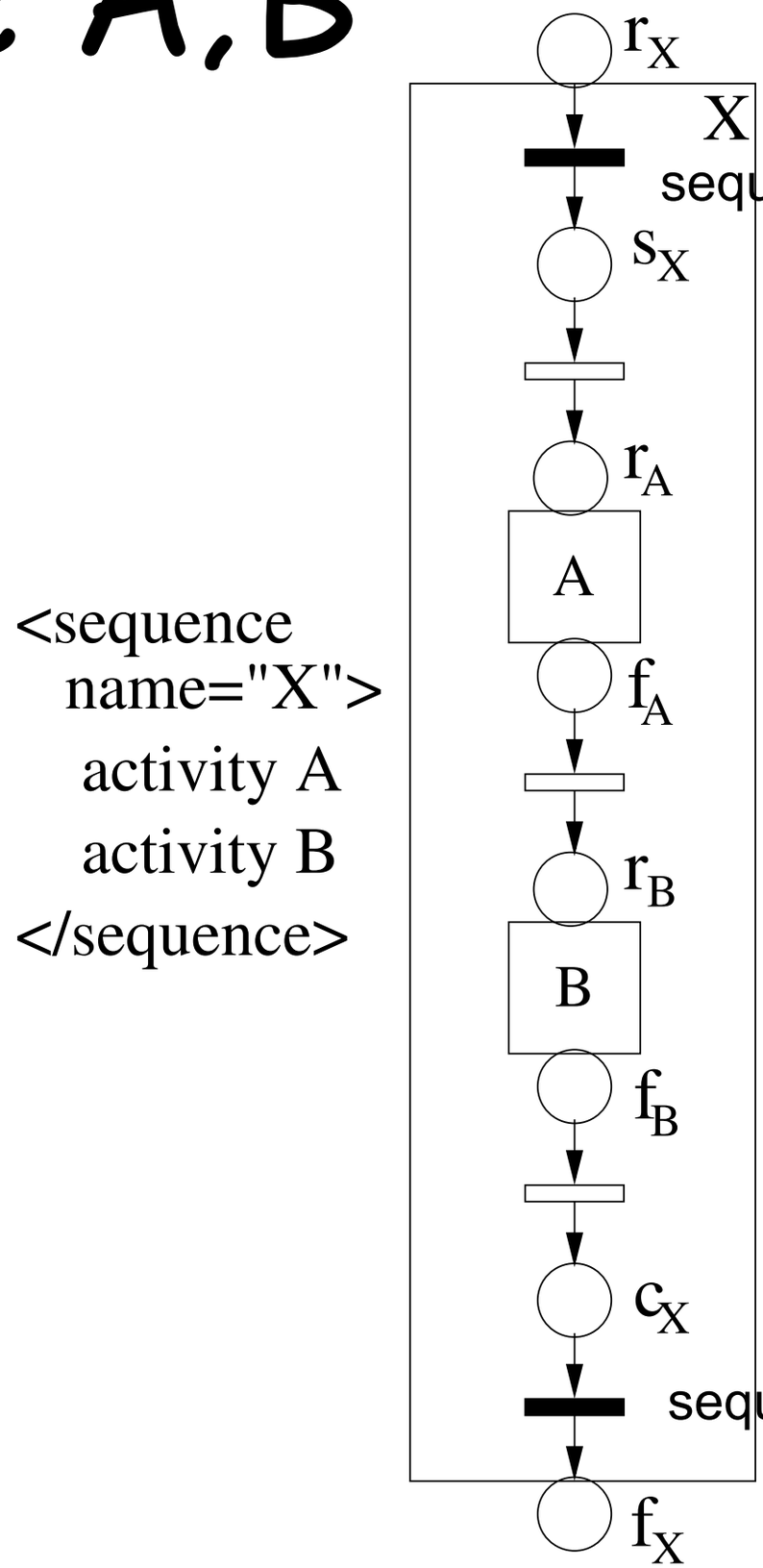
<receive>

<reply>



Sequence A;B

We show the binary version, but it can be generalized to an arbitrary number of activities

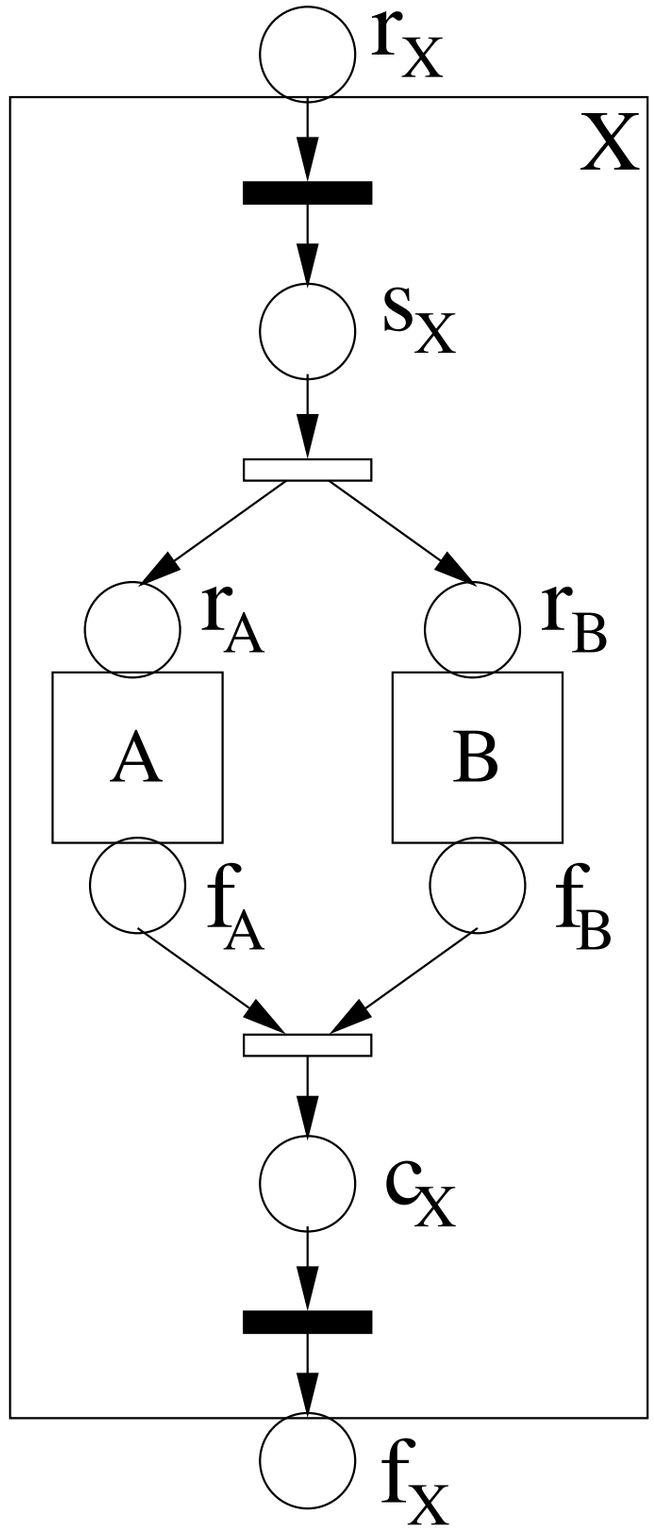



```
<sequence
  name="X">
  activity A
  activity B
</sequence>
```

Flow A|B

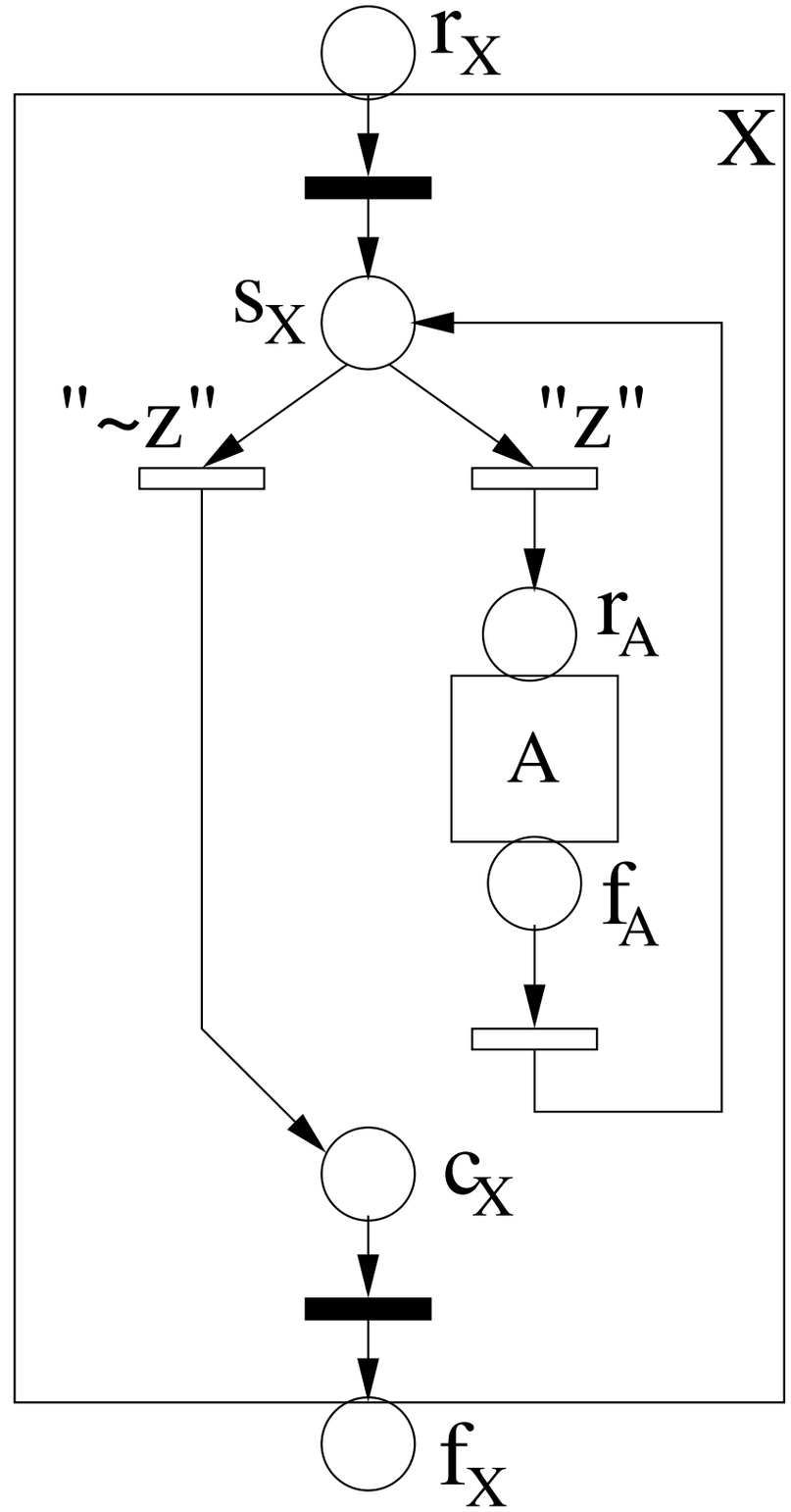
We show the binary version, but it can be generalized to an arbitrary number of activities

```
<flow
   name="X">
    activity A
    activity B
</flow>
```



While z do A

```
<while name="X">
   <condition>
        z
   </condition>
   activity A
</while>
```



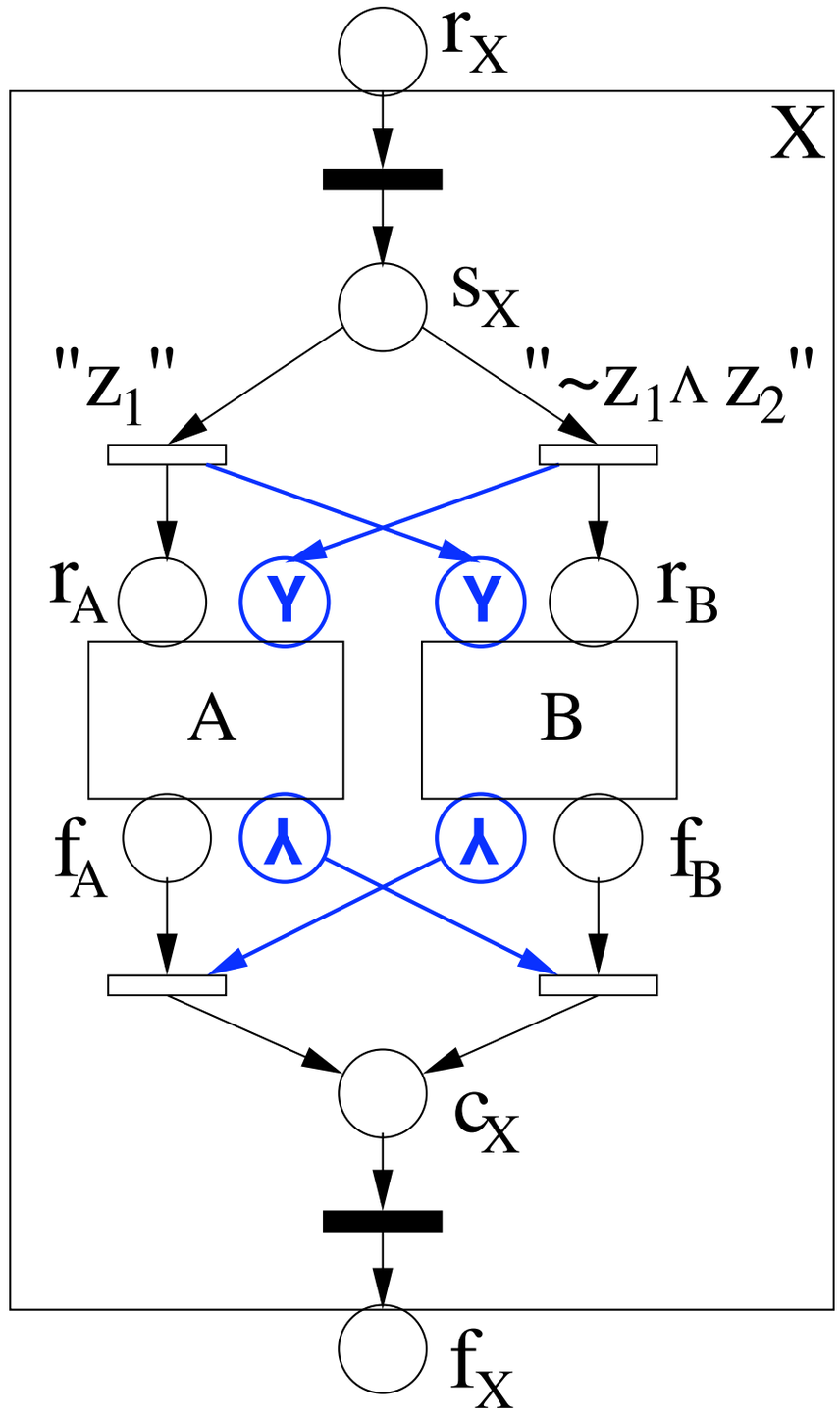
Switch $(z_1)A,(z_2)B$

We show the binary version, but it can be generalized to an arbitrary number of activities

In blue: alternative flow to skip activities (to deal with links and dead-path elim.)

Y ʎ are just decorations

```
<switch name="X">
  <case>
    <condition>
      z1
    </condition>
    activity A
  </case>
  <case>
    <condition>
      z2
    </condition>
    activity B
  </case>
</switch>
```



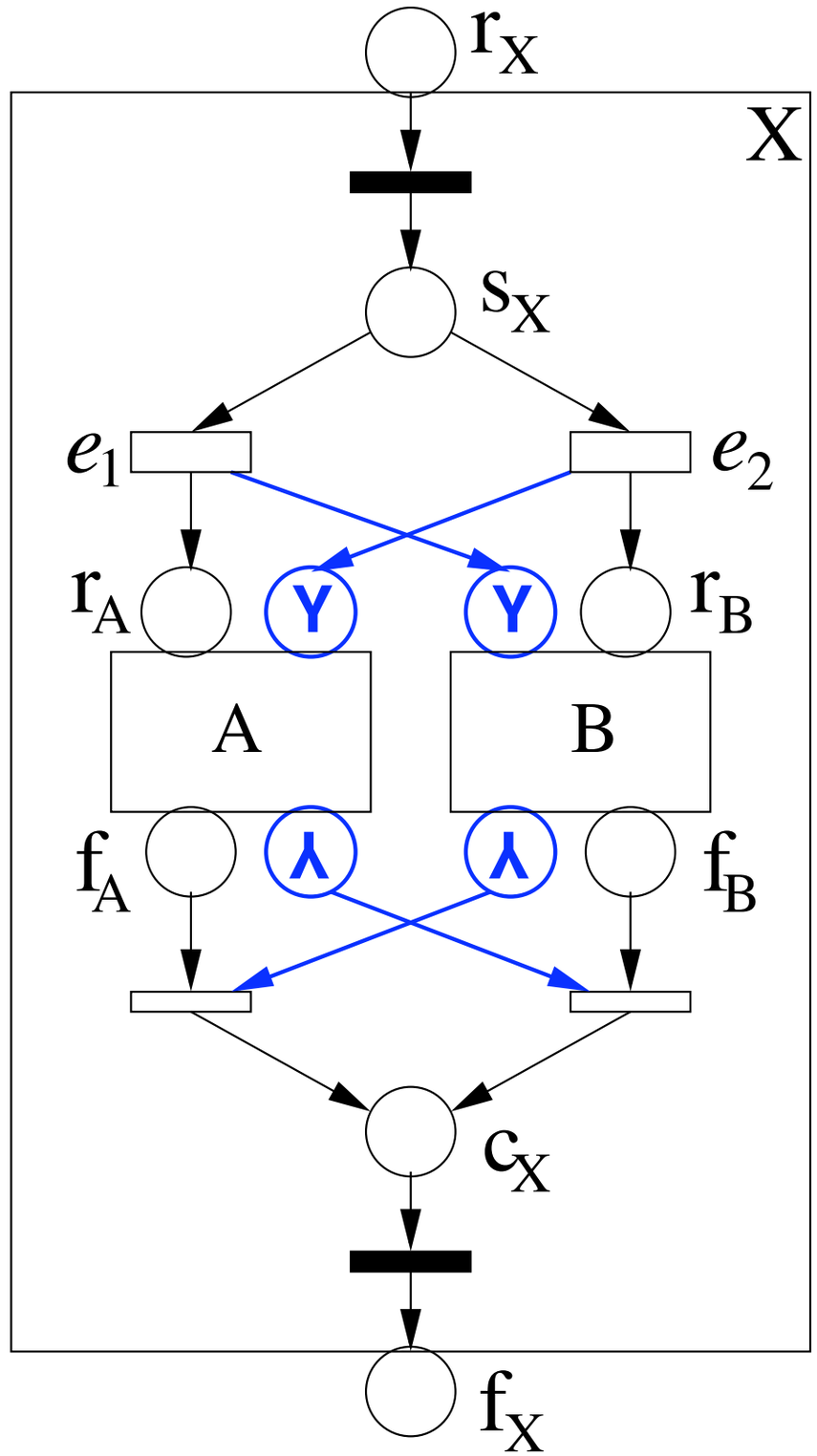
Pick $(e_1)A,(e_2)B$

We show the binary version, but it can be generalized to an arbitrary number of activities

In blue:
alternative flow
to skip activities
(to deal with links
and dead-path elim.)

are just decorations

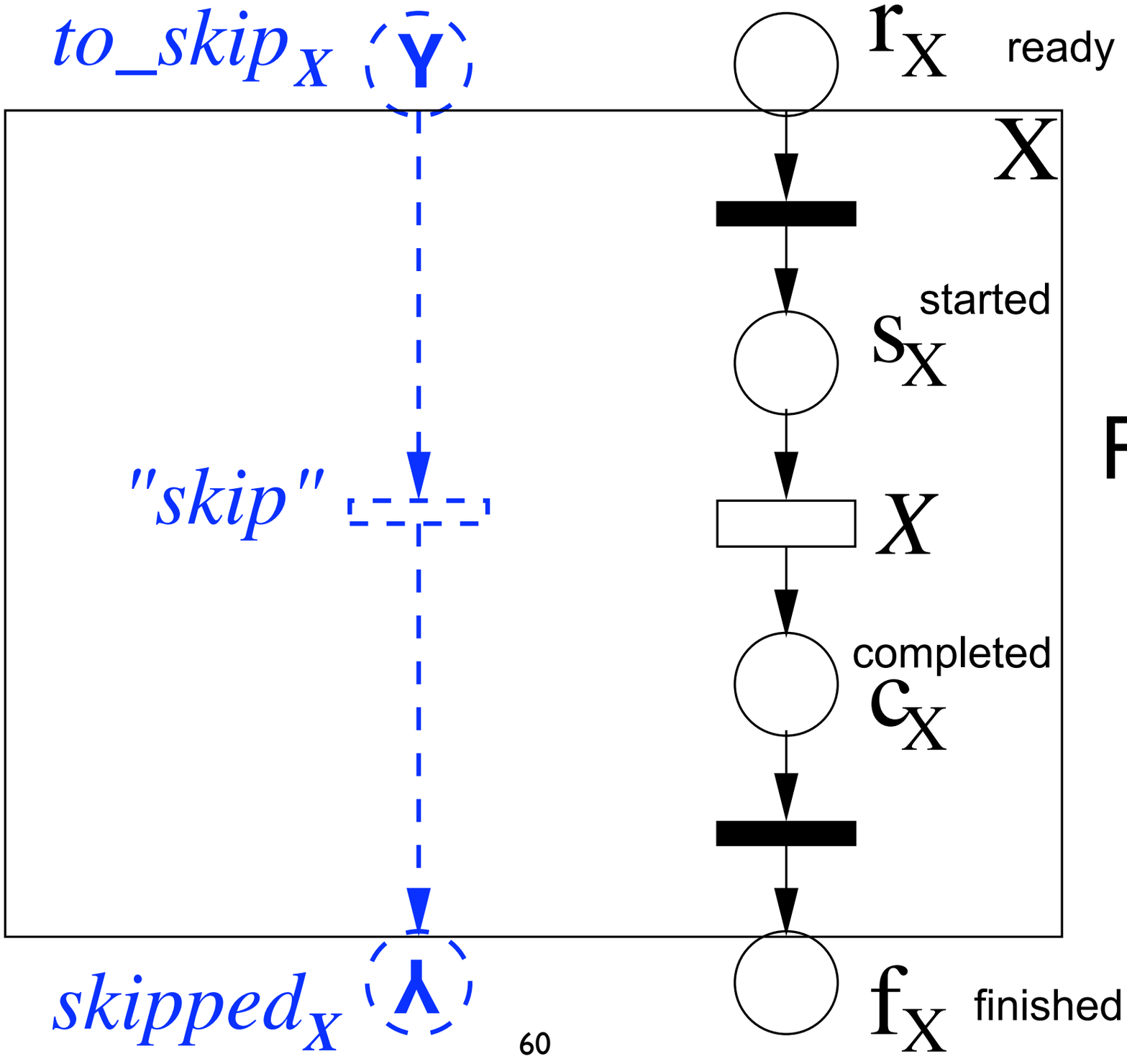
```
<pick name="X">
  <onMessage e1>
      activity A
  </onMessage>
  <onAlarm  e2>
      activity B
  </onAlarm>
</pick>
```



Basic activity + skip

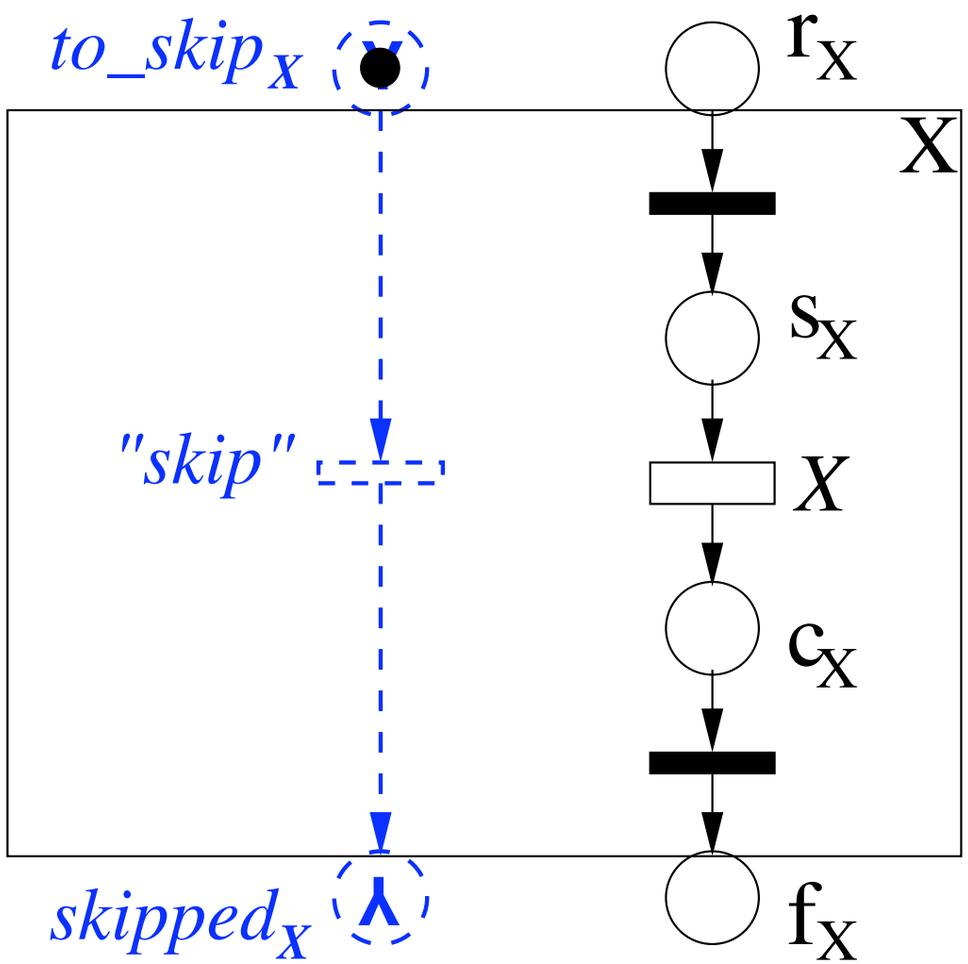
Skip path

Regular flow

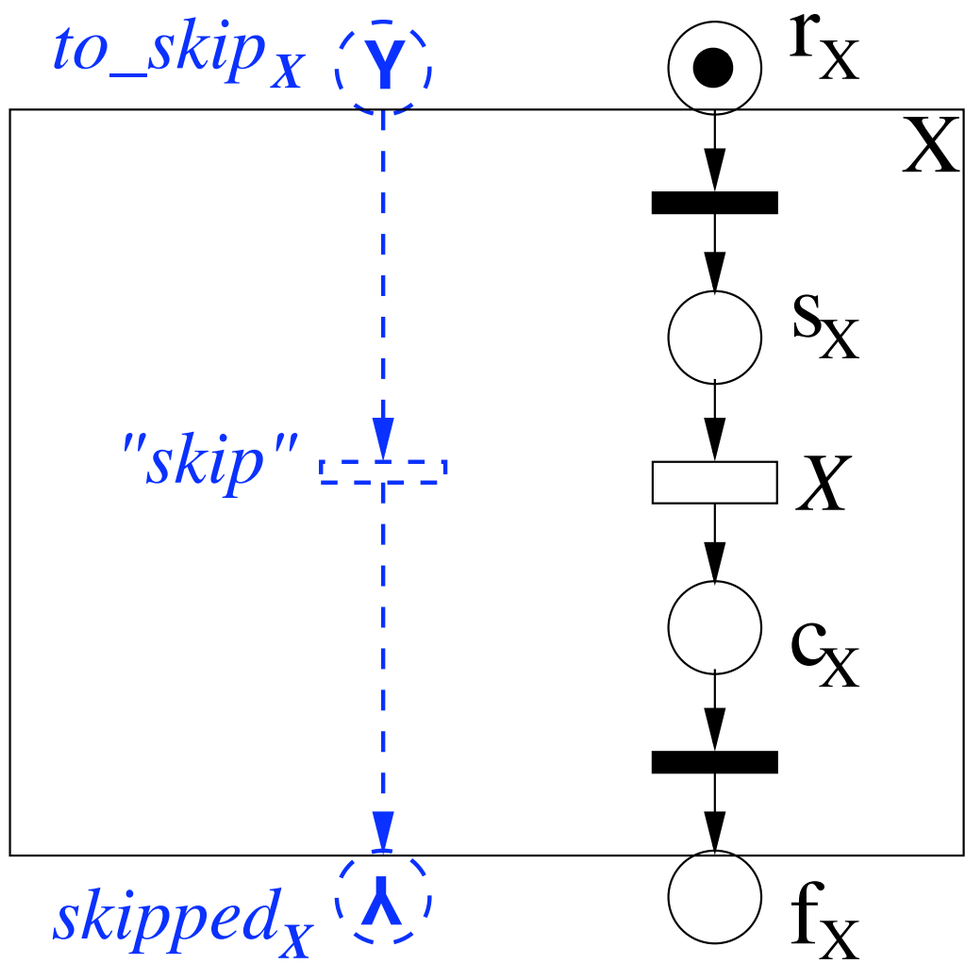


Basic activity + skip

The token arrives either here...



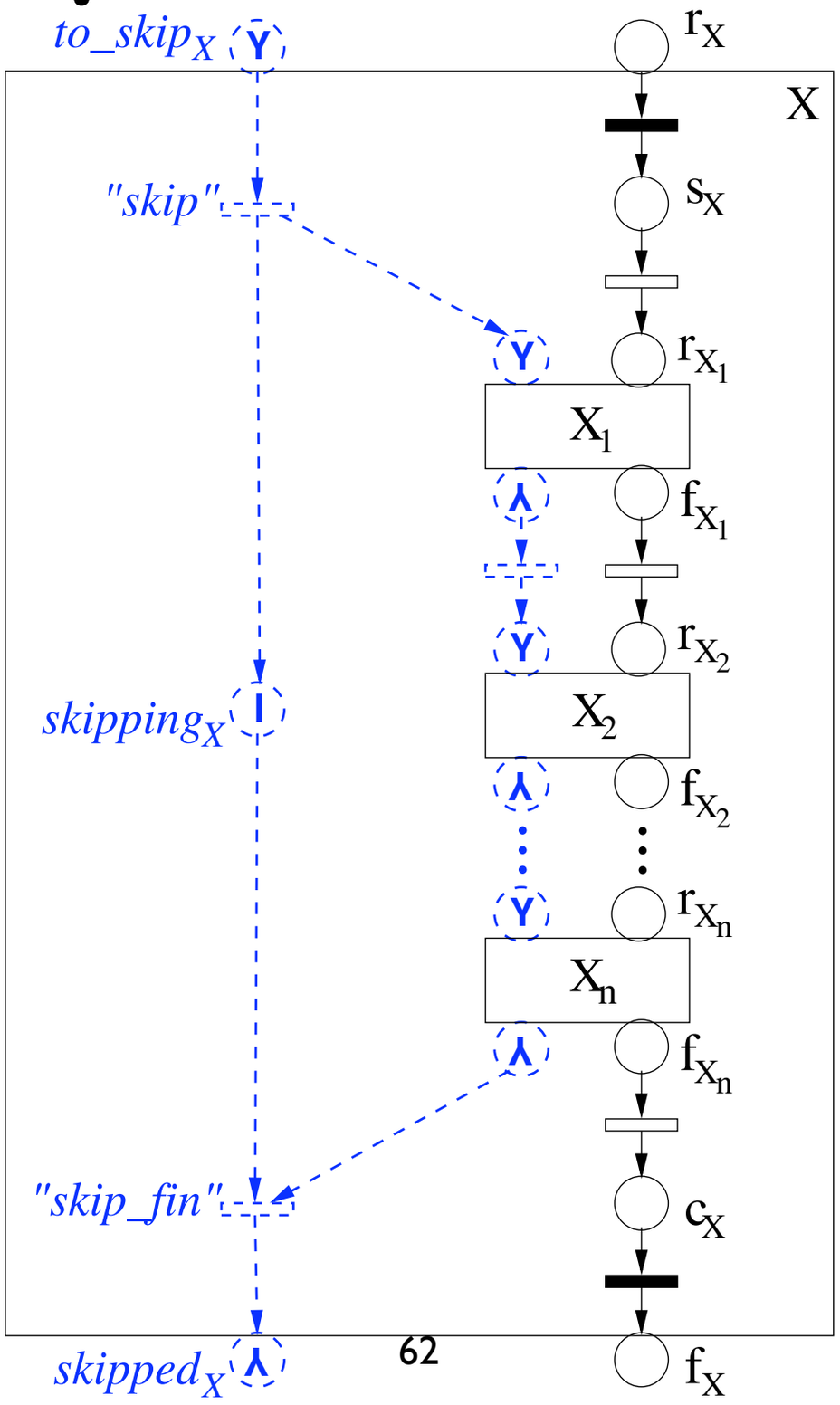
... or here



(but not both)

Sequence + skip

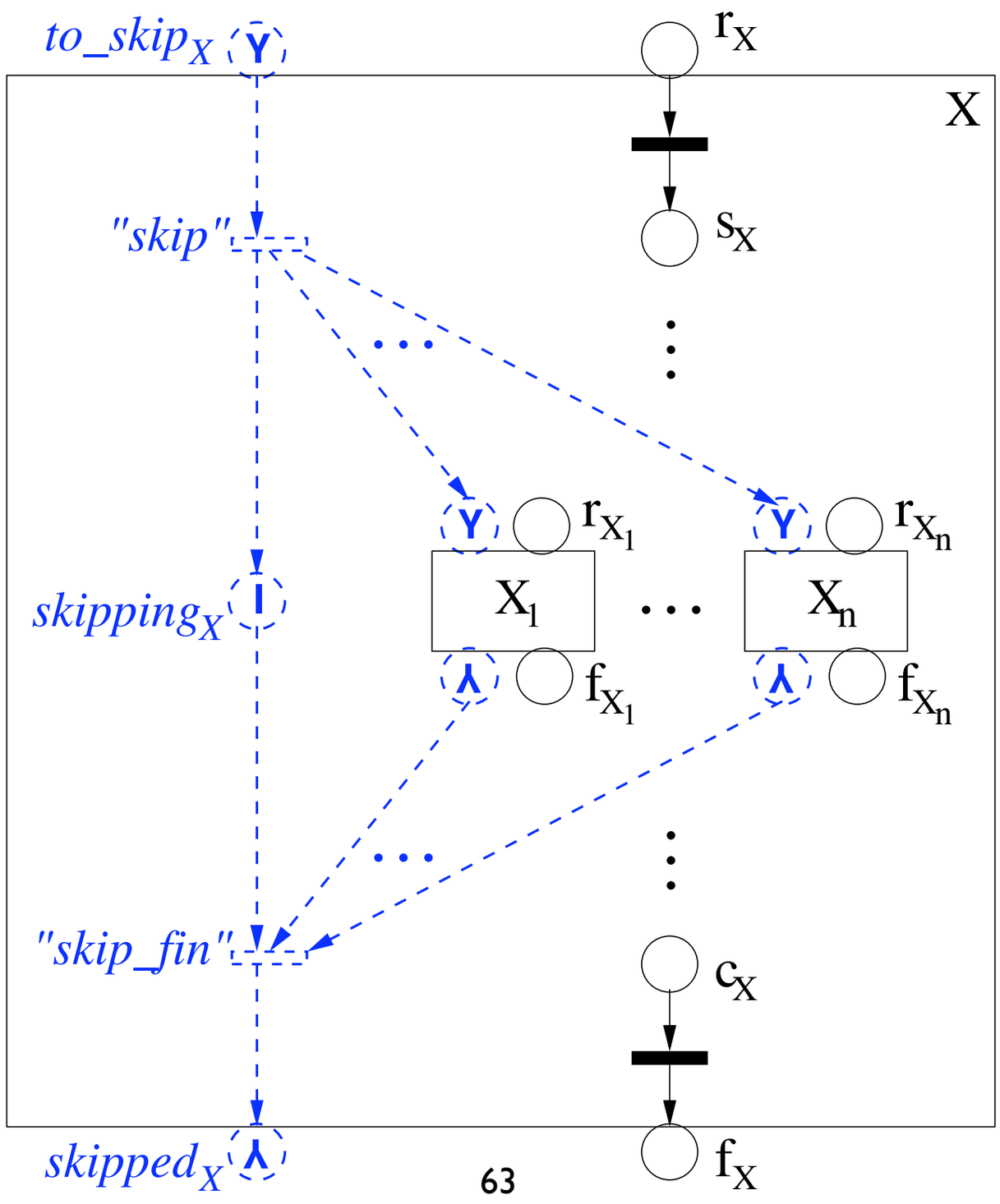
Skip path



Regular flow

Non-sequence + skip

Skip path

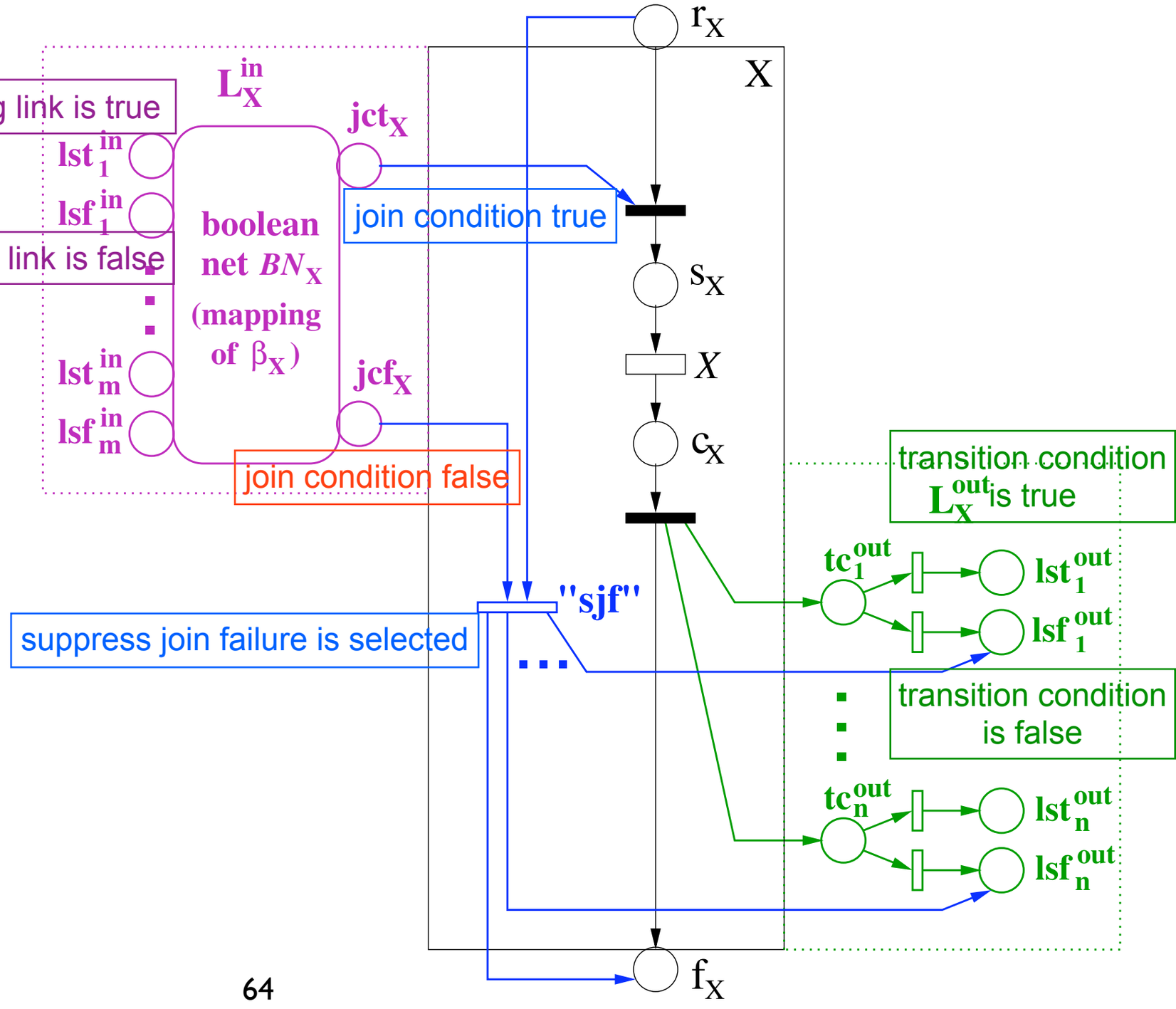


Regular flow

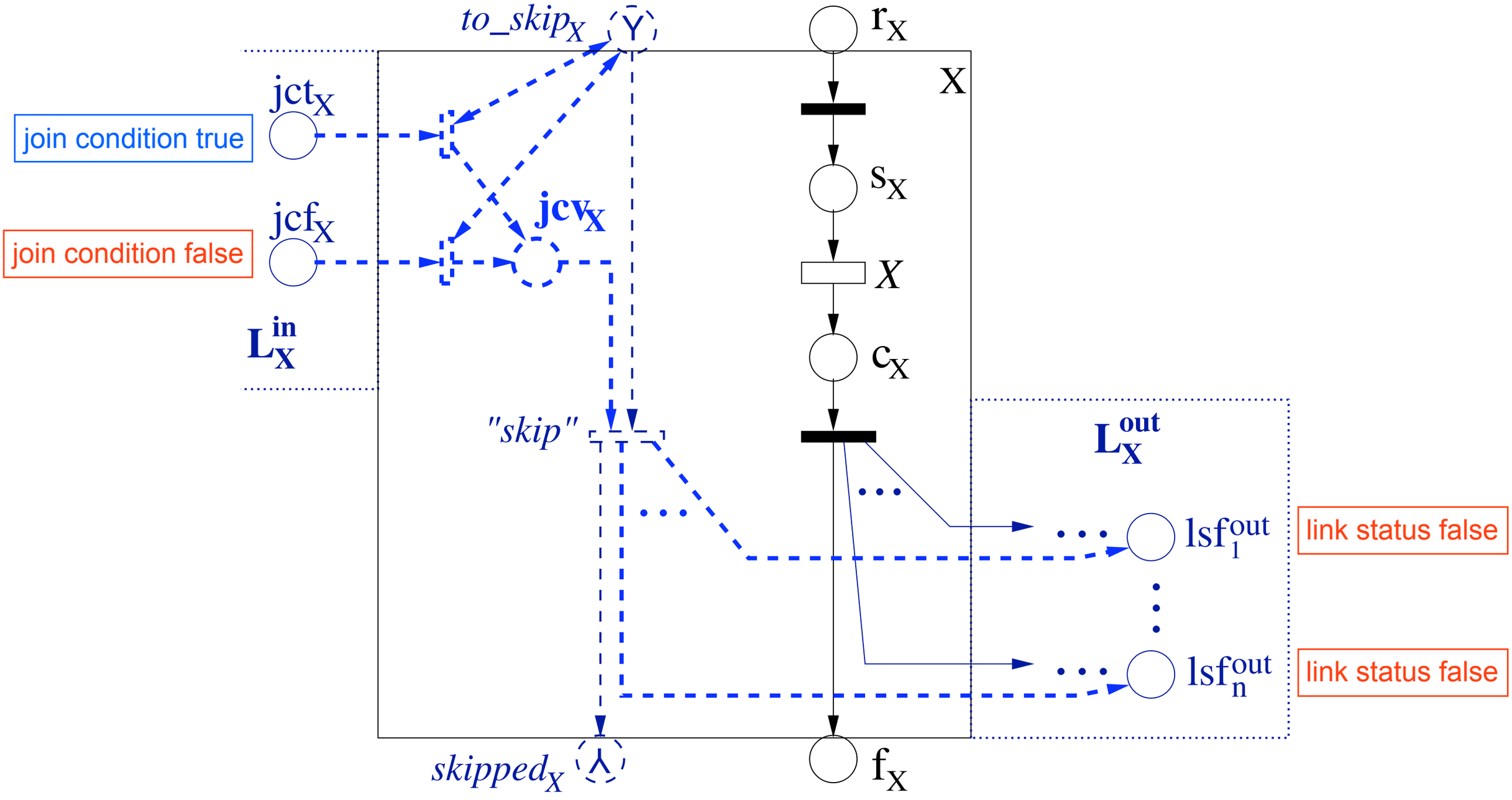
Basic activity with control links

```
<activityX suppressJoinFailure="yes">
    <sources>
        <source linkname="X_1^out">
            <transitionCondition>
                tc_1^out
            </transitionCondition>
        </source>
            ⋮
        <source linkname="X_n^out">
            <transitionCondition>
                tc_n^out
            </transitionCondition>
        </source>
    </sources>
    <targets>
        <joinCondition>
            β_X (ls_1^in , ... , ls_m^in )
        </joinCondition>
        <target linkname="X_1^in">
            ⋮
        <target linkname="X_m^in">
    </targets>
</activityX>
```

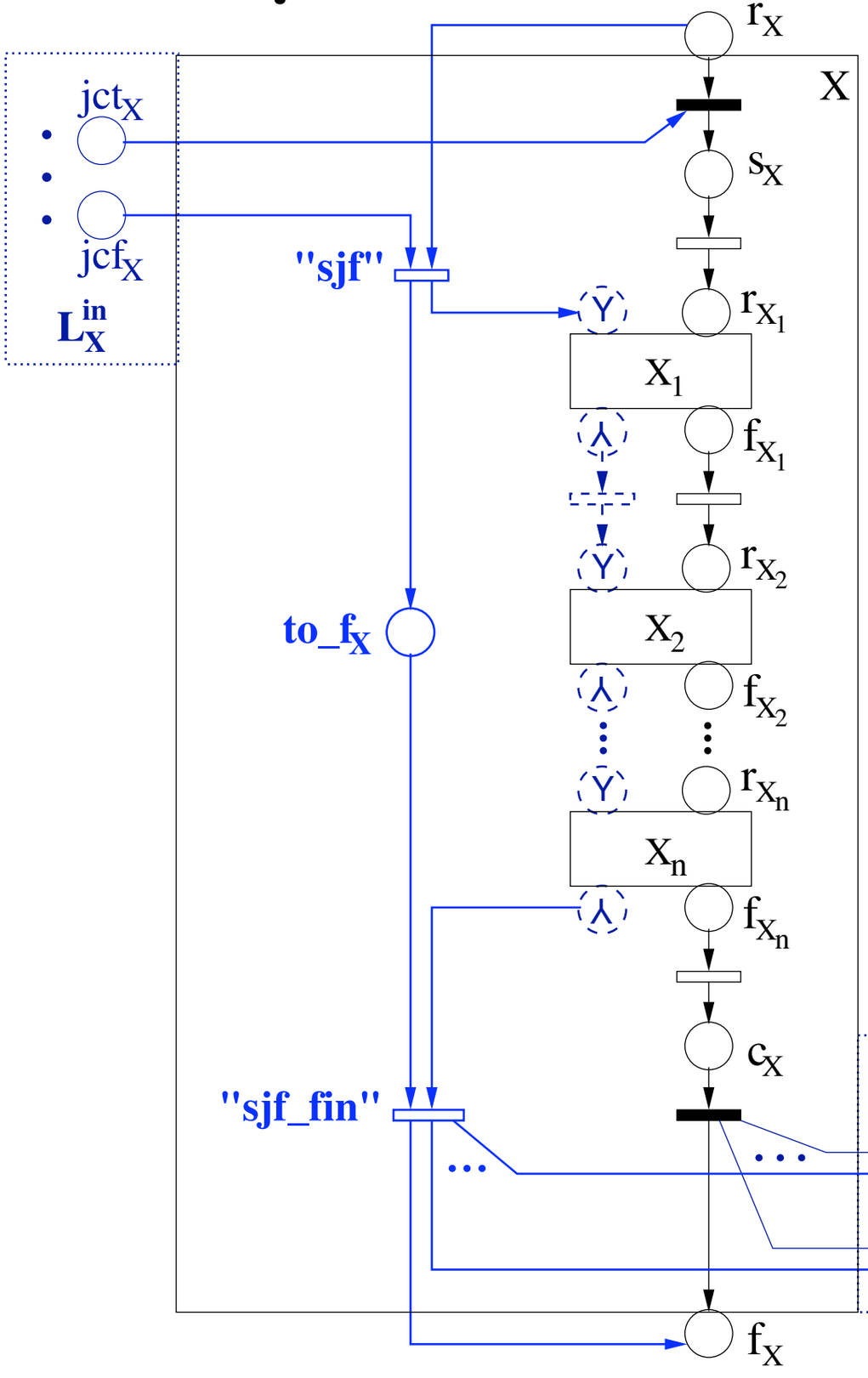
[note] ls_j^{in} *is the status of control link* X_j^{in}, *where j=1, 2, ... , m.*



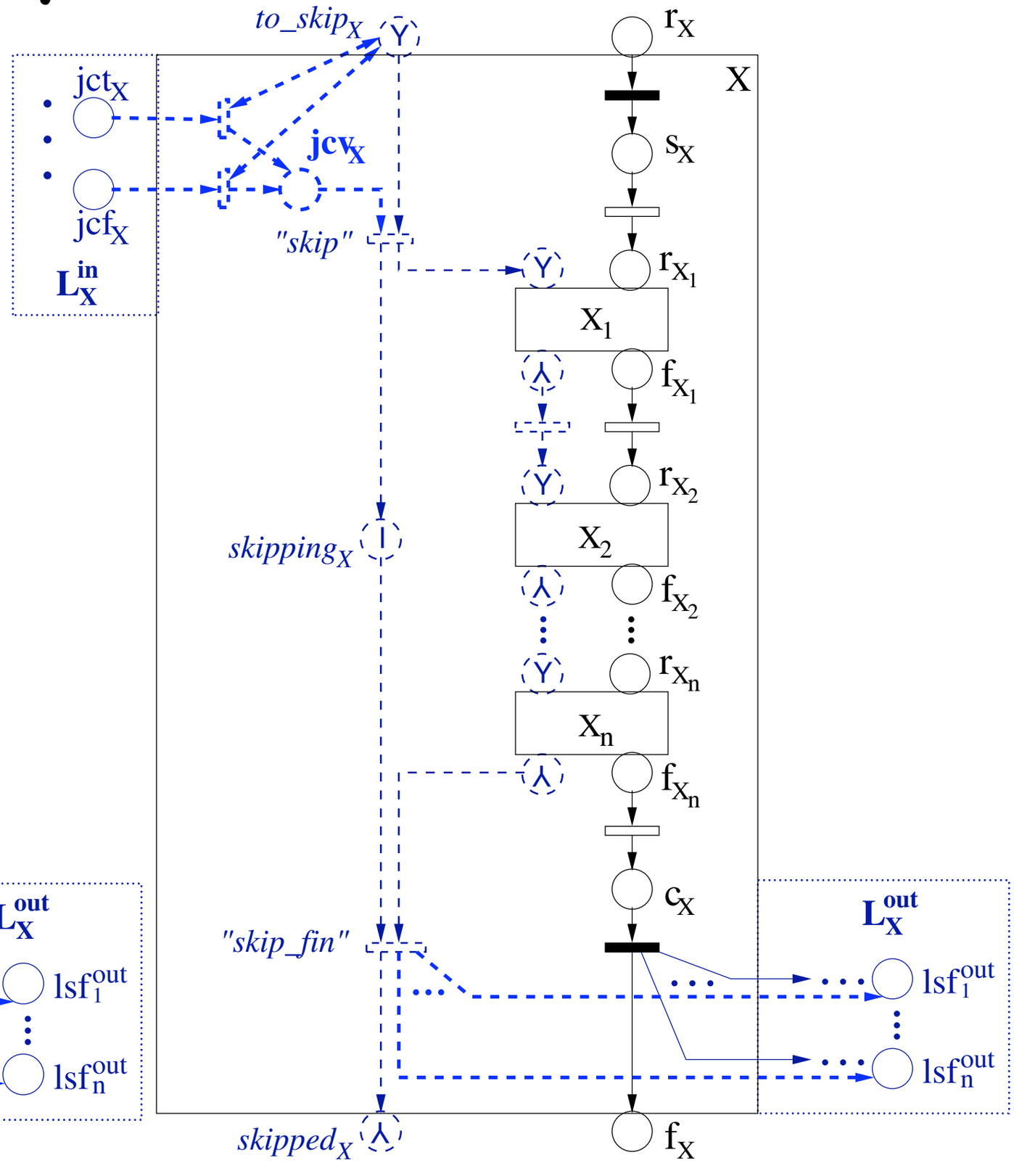
Skipping a basic activity with control links



Sequence activity with control links

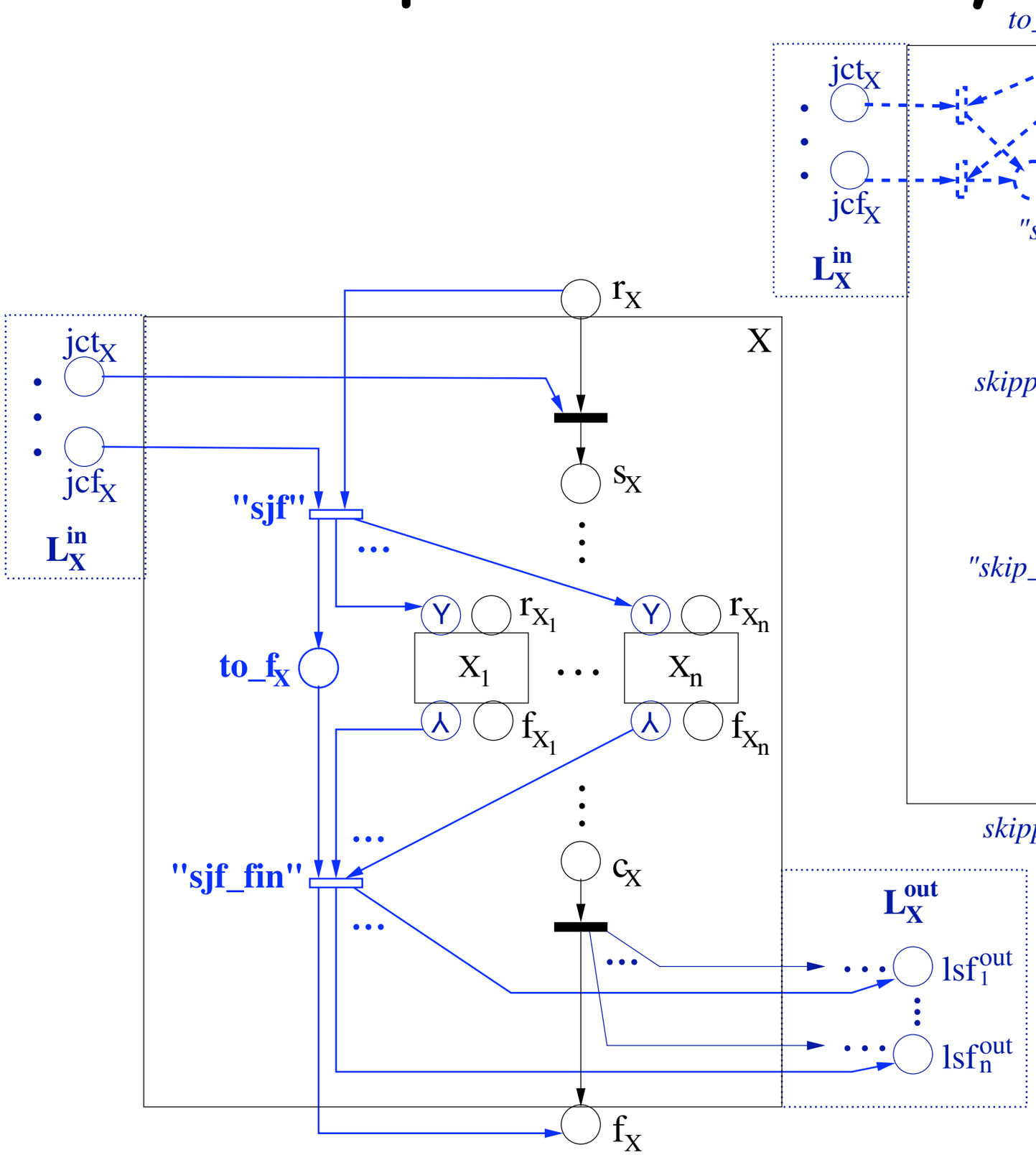


(a) normal behaviour

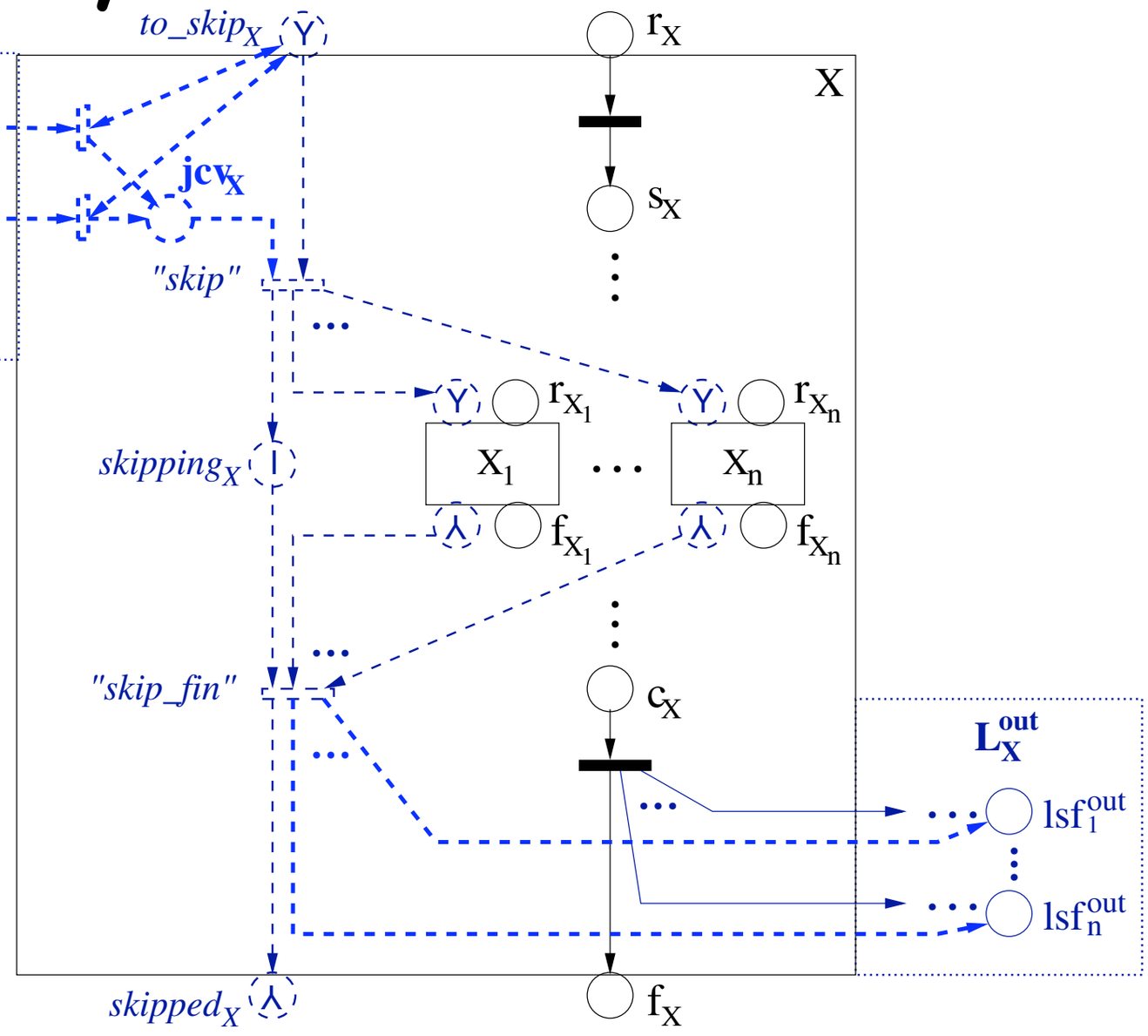


(b) skipping behaviour

Non-sequence activity with control links

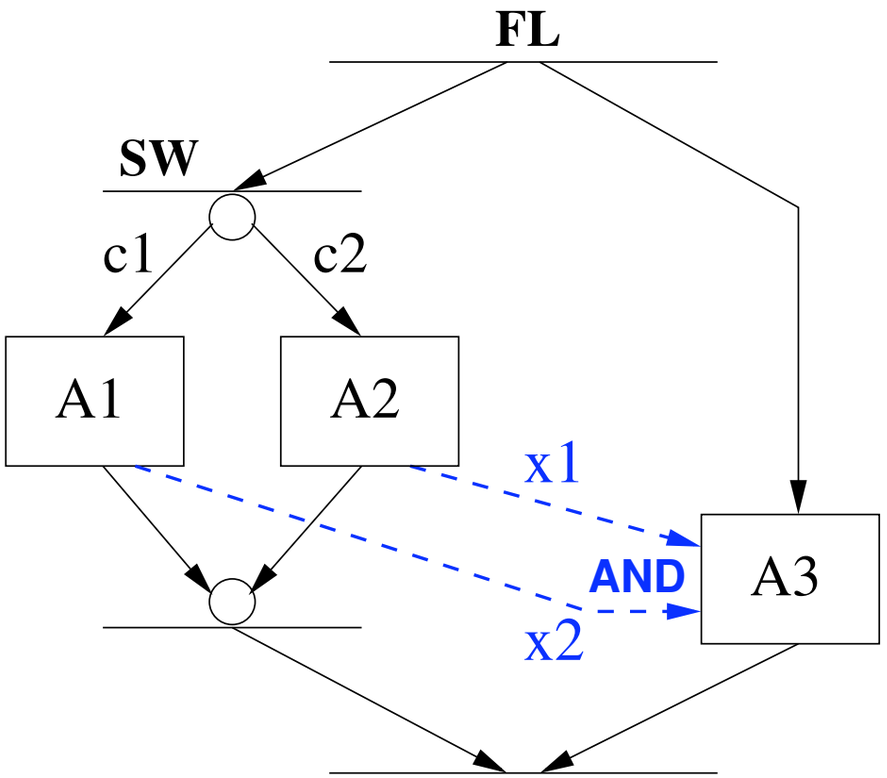


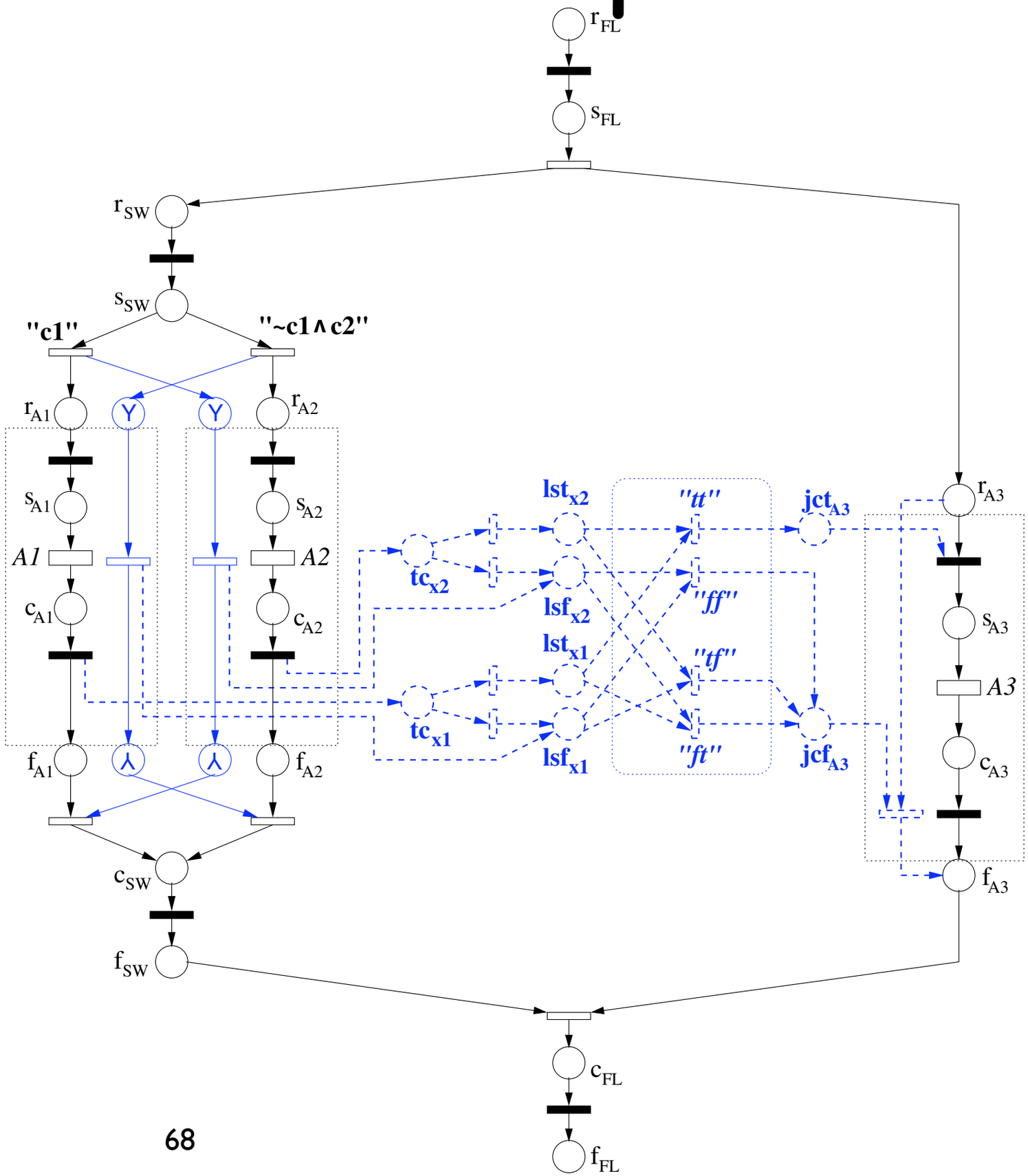
(a) normal behaviour



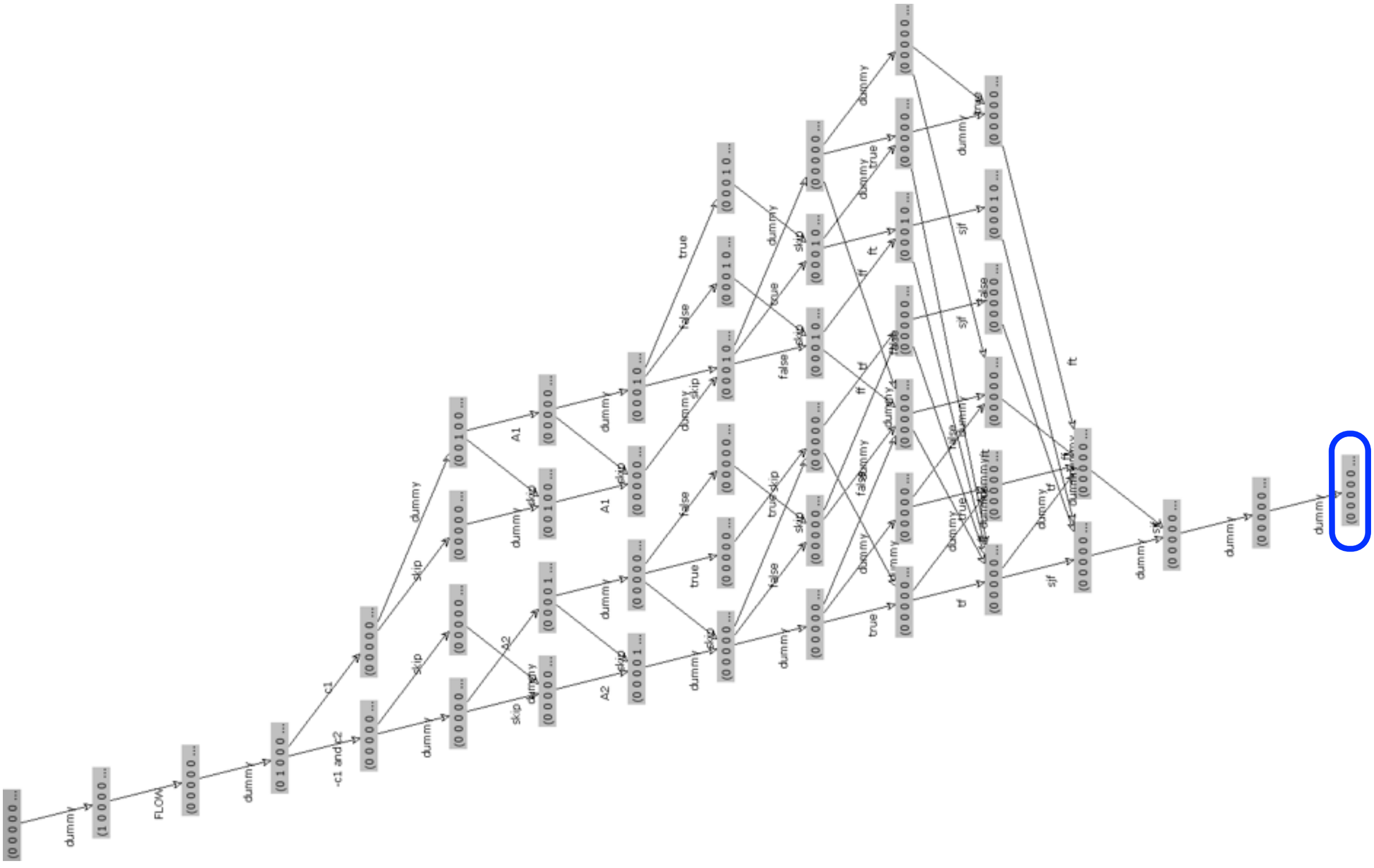
(b) skipping behaviour

The previous example

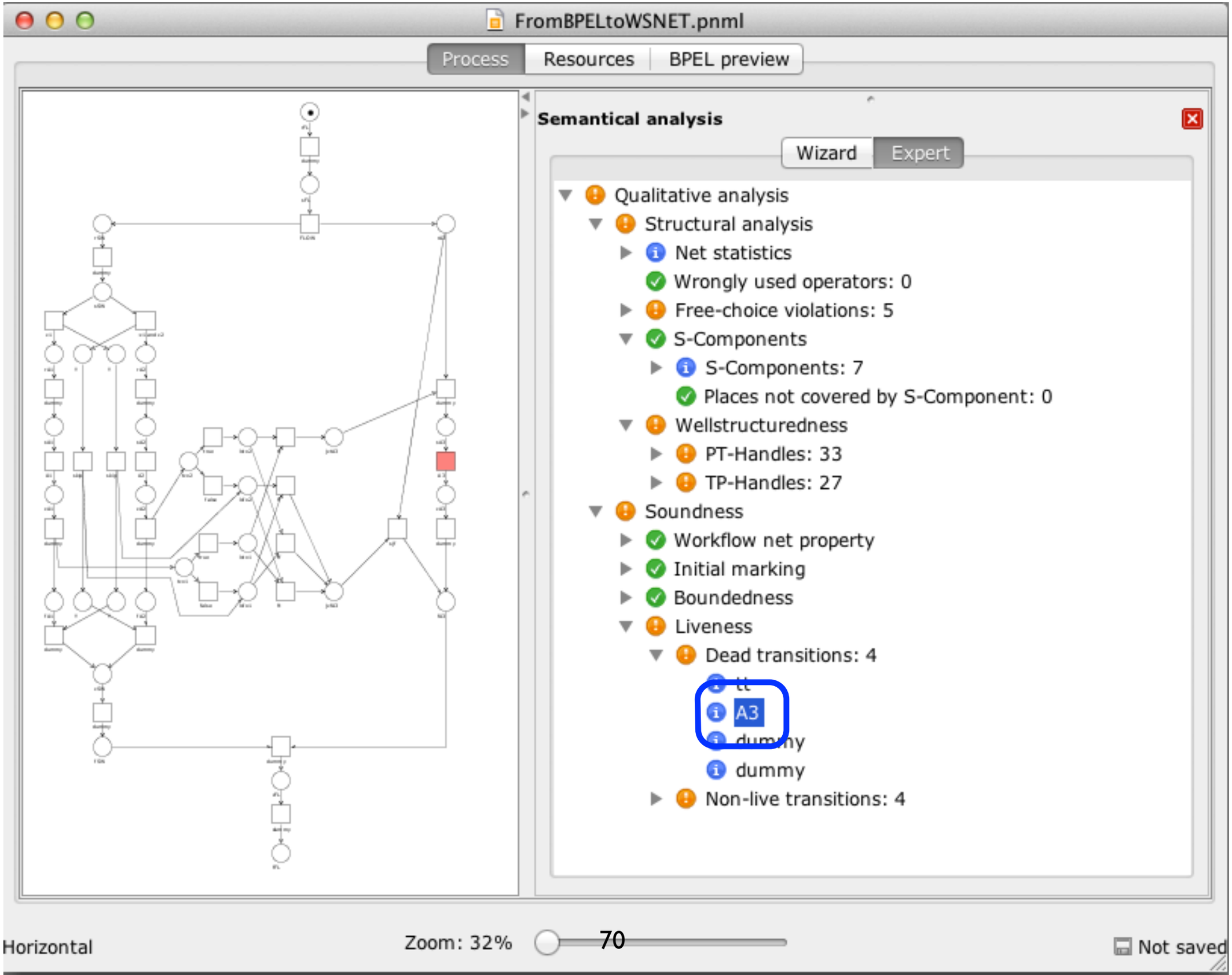




The previous example



The previous example

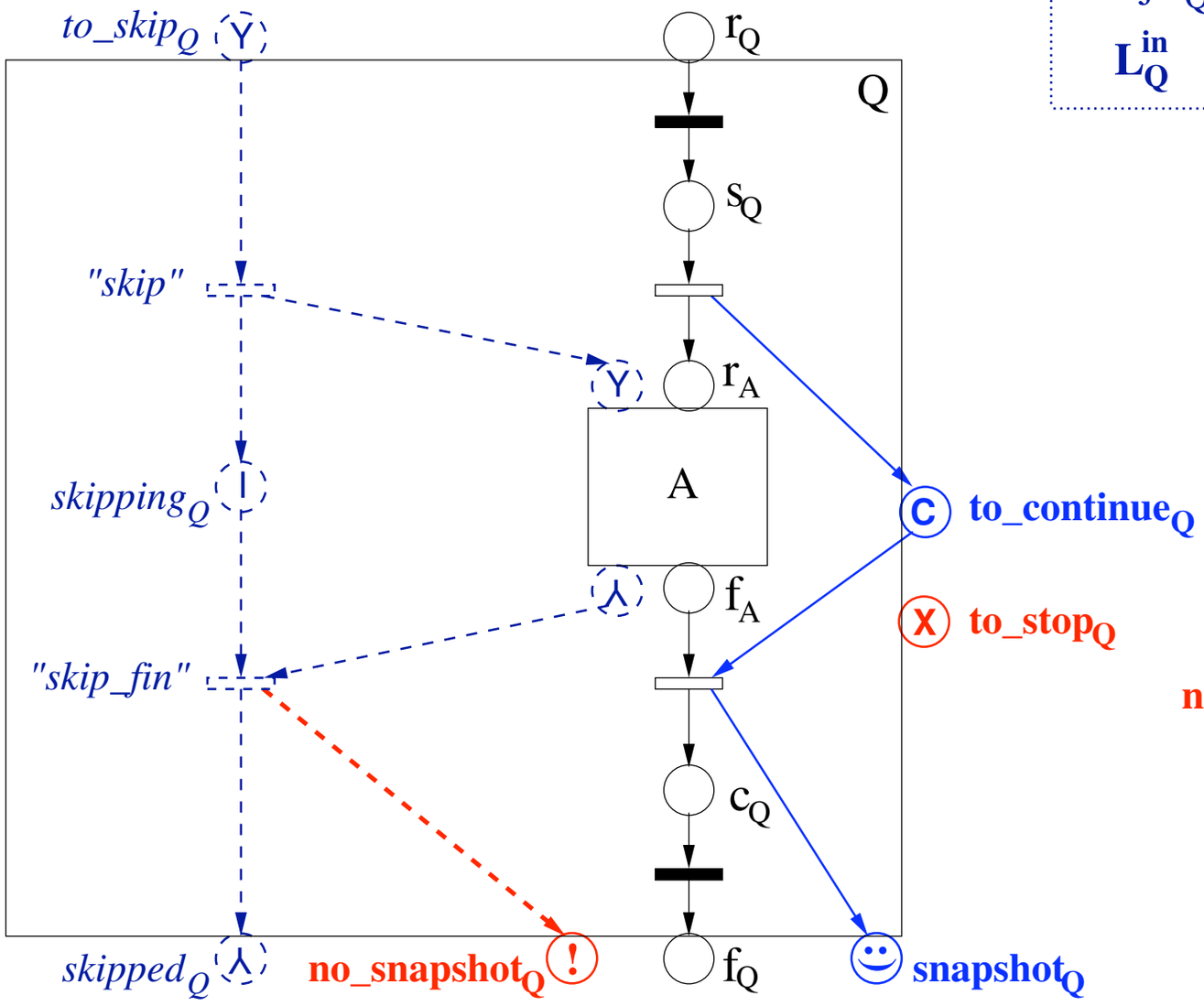


Scope

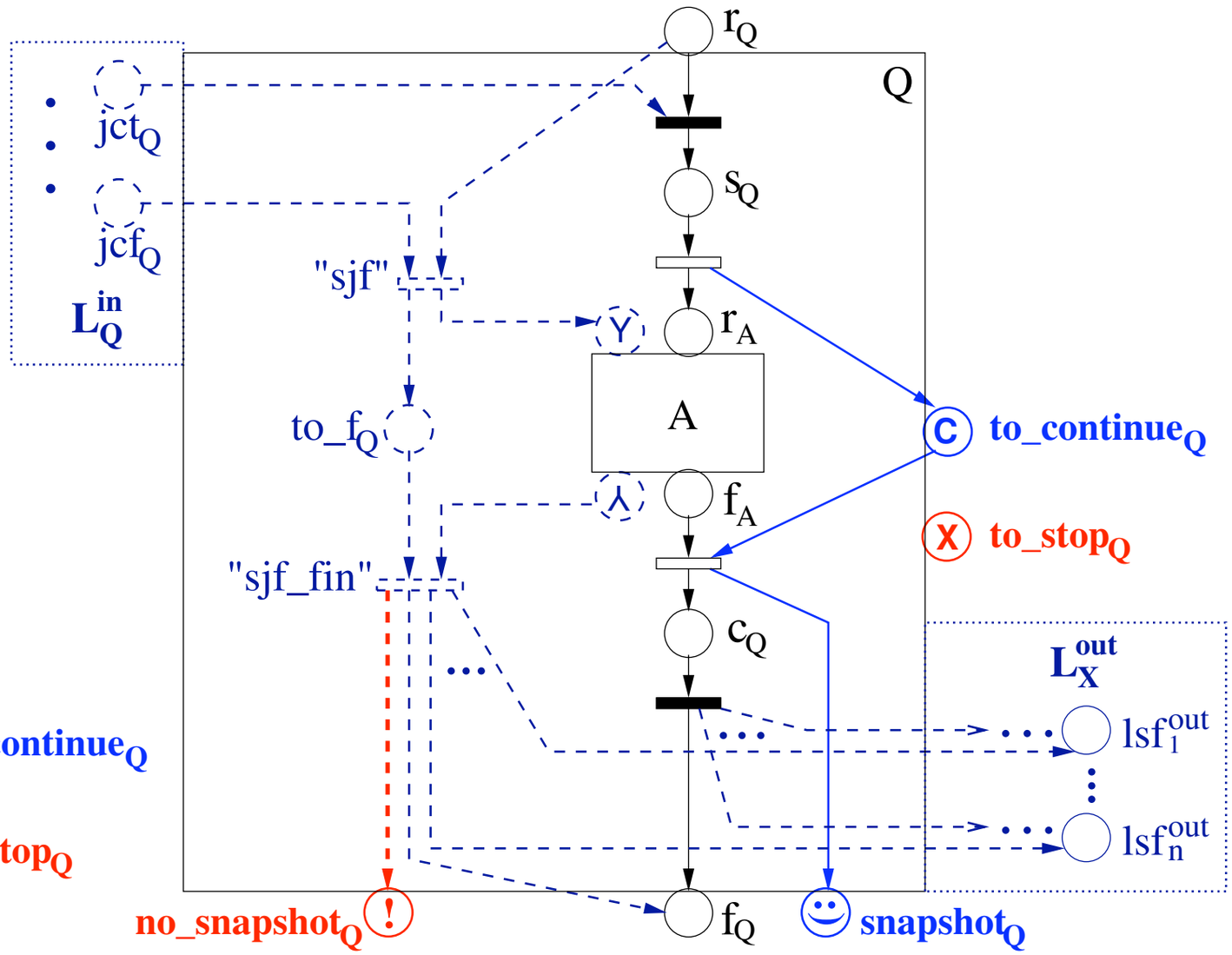
Remind that a scope has a primary activity, and optionally:
a set of fault handlers,
a set of event handlers, and
one compensation handler.

To deal with them, four “flags” are attached to a scope:
`to_continue` (no exception, execution is in progress)
`to_stop` (an error occurred, activities need to stop)
`snapshot` (successfully completed, uncompensated)
`no_snapshot` (no compensation needed)

Scope



(a) skipping behaviour



(b) suppressing join failure

Full translation

The interested reader can find out
more details in the paper by Ouyang *et al.*
and play with the **BPEL2PNML** tool available at
http://www.win.tue.nl/~hverbeek/doku.php?id=projects:prom:plug-ins:conversion:bpel2tpn

An alternative translation is given in the paper
``Transforming BPEL to Petri Nets'' by Hinz, Schmidt, Stahl,
supported by the **BPEL2oWFN** tool available at
http://www.gnu.org/software/bpel2owfn/

The two translations are compared in
``Comparing and Evaluating Petri Net Semantics for BPEL''
by Lohmann, Verbeek, Ouyang, Stahl, van der Aalst