Business Process Management: Concepts, Languages, Architectures

Second Edition

Figures of Chapter 4

Mathias Weske

Legal Notice

- This PowerPoint file contains figures used in
  - Mathias Weske, Business Process Management: Concepts, Languages, Architectures 2012, XIV, 403 p. 300 illus., Hardcover ISBN: 978-3-642-28615-5, eBook ISBN: 978-3-642-28616-2 © Springer-Verlag Berlin Heidelberg 2012, 2007
- Each slide contains one picture in PNG-format with caption and copyright notice. It is illegal to remove the copyright notice from any of the figures.
- Commercial use in any form is not allowed without prior written approval by Springer-Verlag
- Teaching material can be found at BPM Academic Initiative, http://academic.signavio.com

Best regards,

Mathias Weske

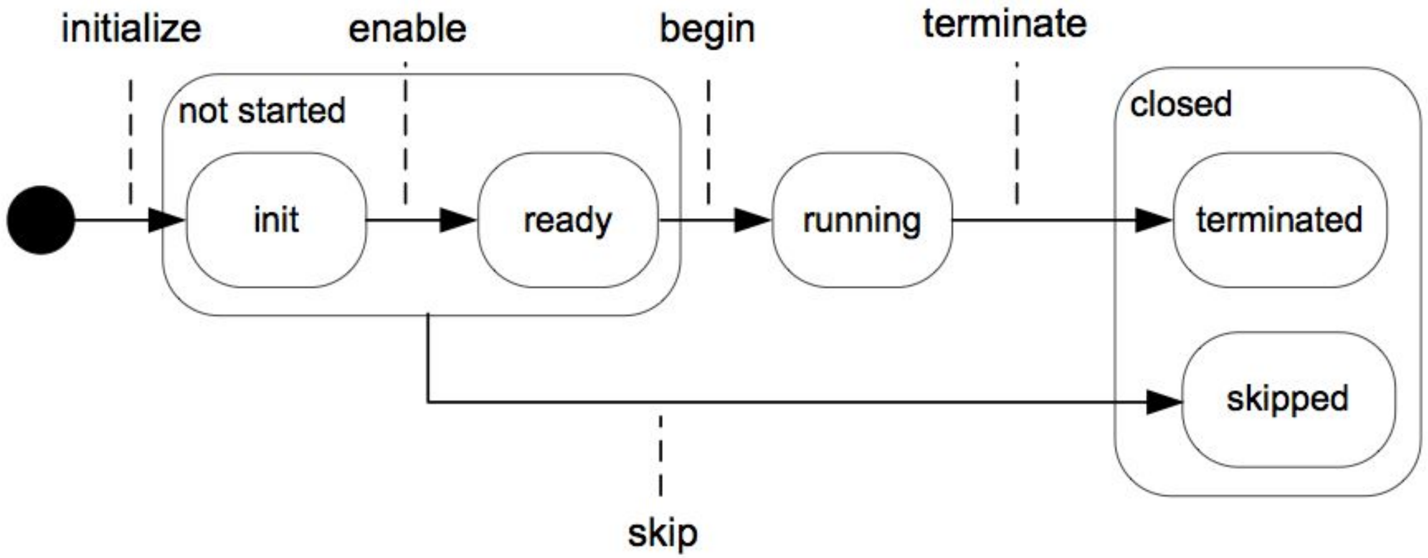


Fig. 4.1. State transition diagram for activity instances

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

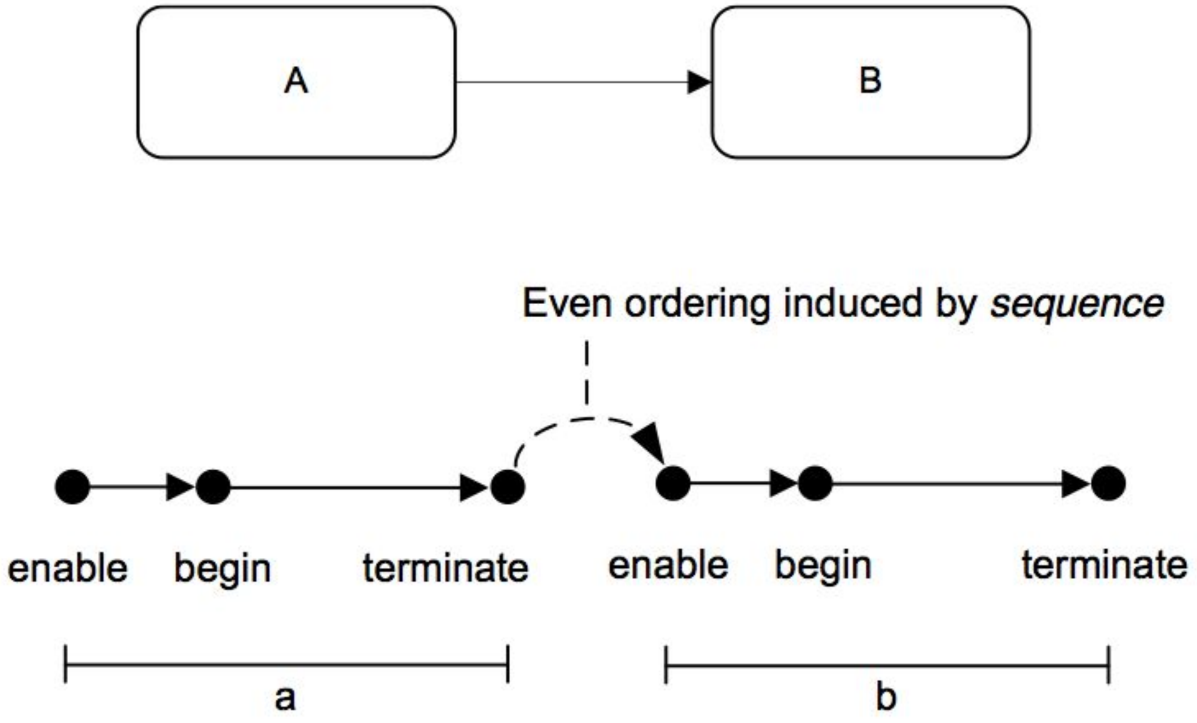


Fig. 4.2. Sequence pattern, with event diagram of process instance

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

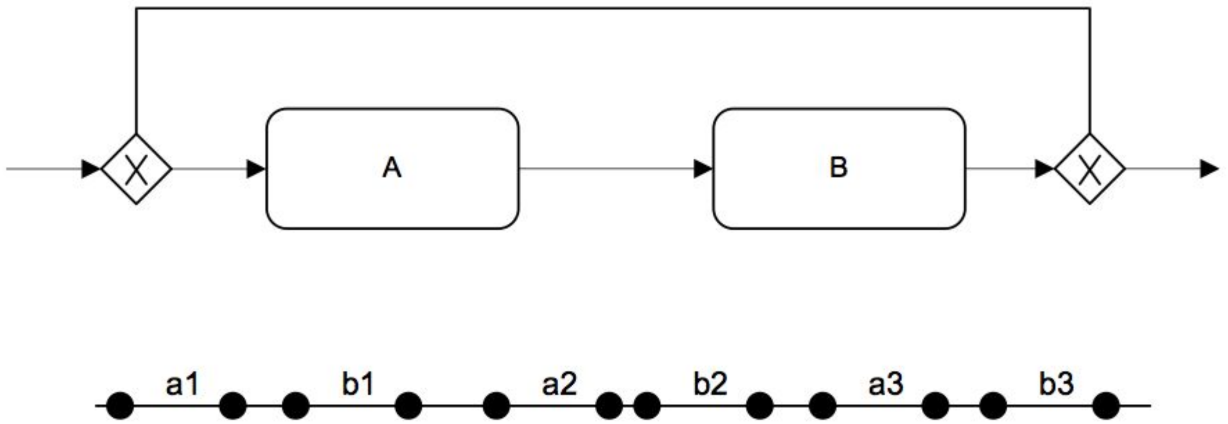


Fig. 4.3. Sequence pattern as part of a loop and event diagram showing three loop iterations

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

HPI Hasso Plattner Institut

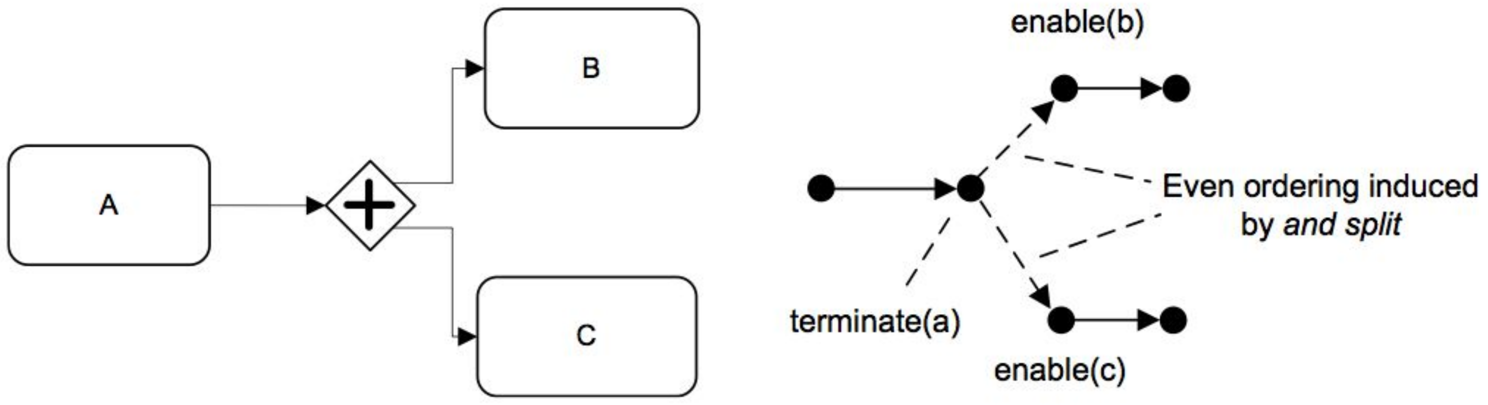


Fig. 4.4. And split pattern

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

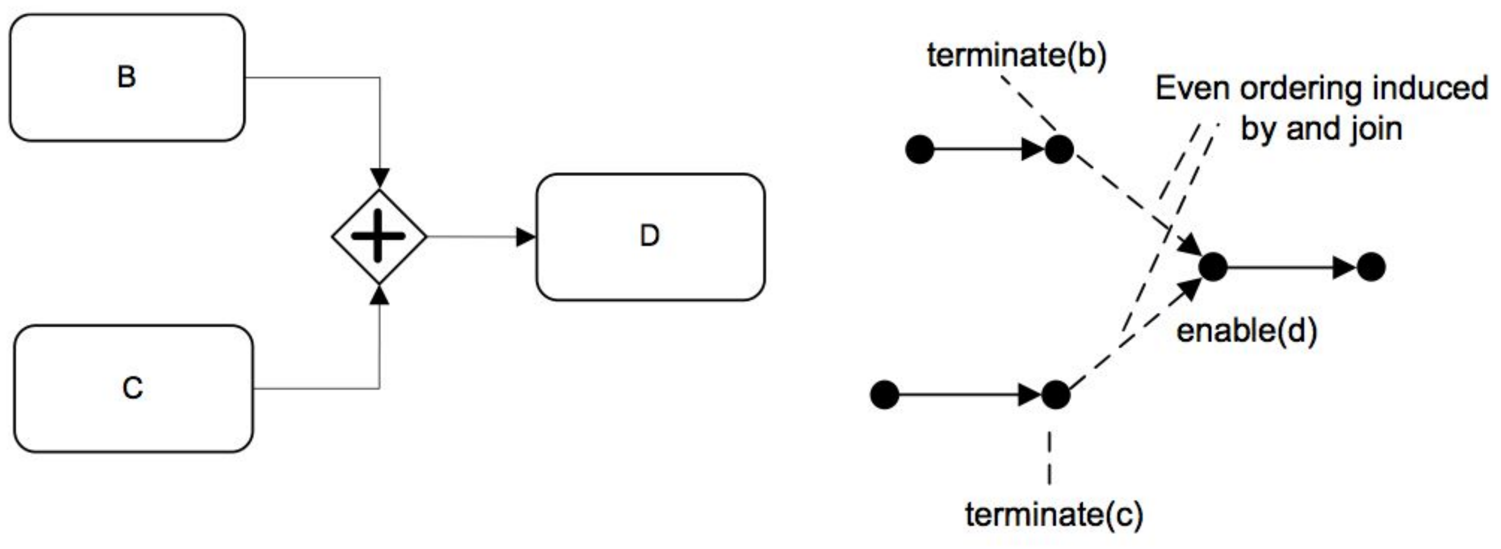


Fig. 4.5. And join pattern

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

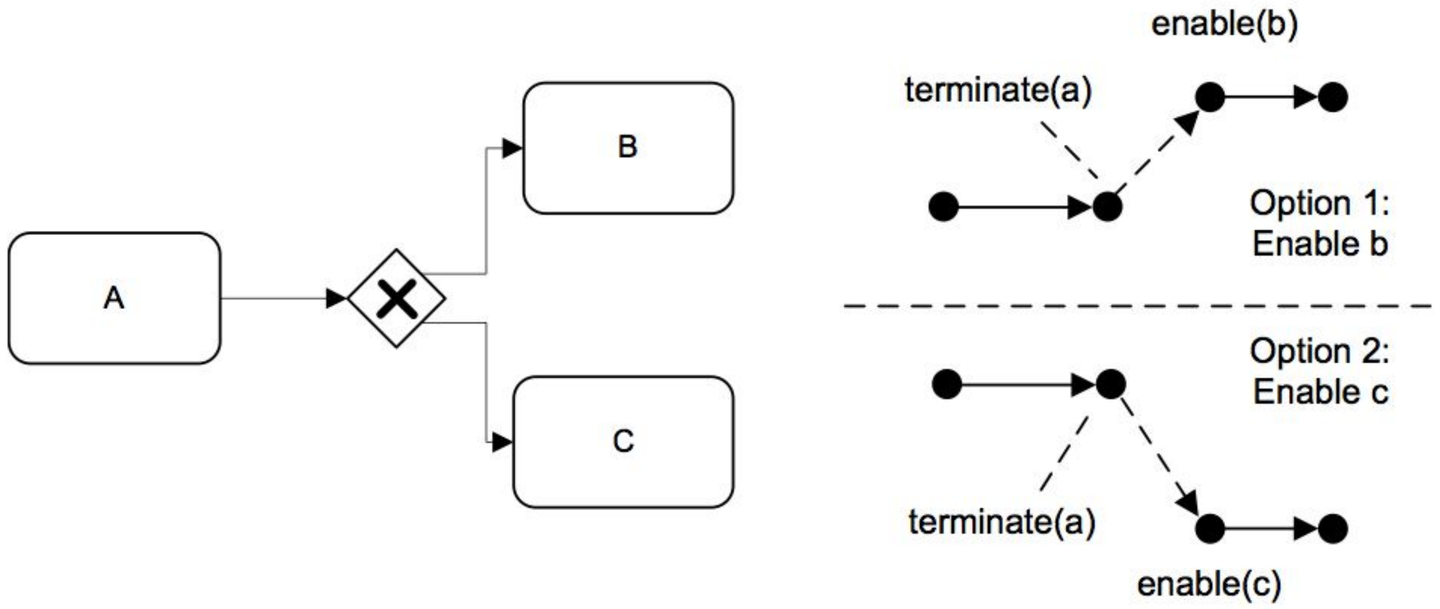


Fig. 4.6. Xor split pattern

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

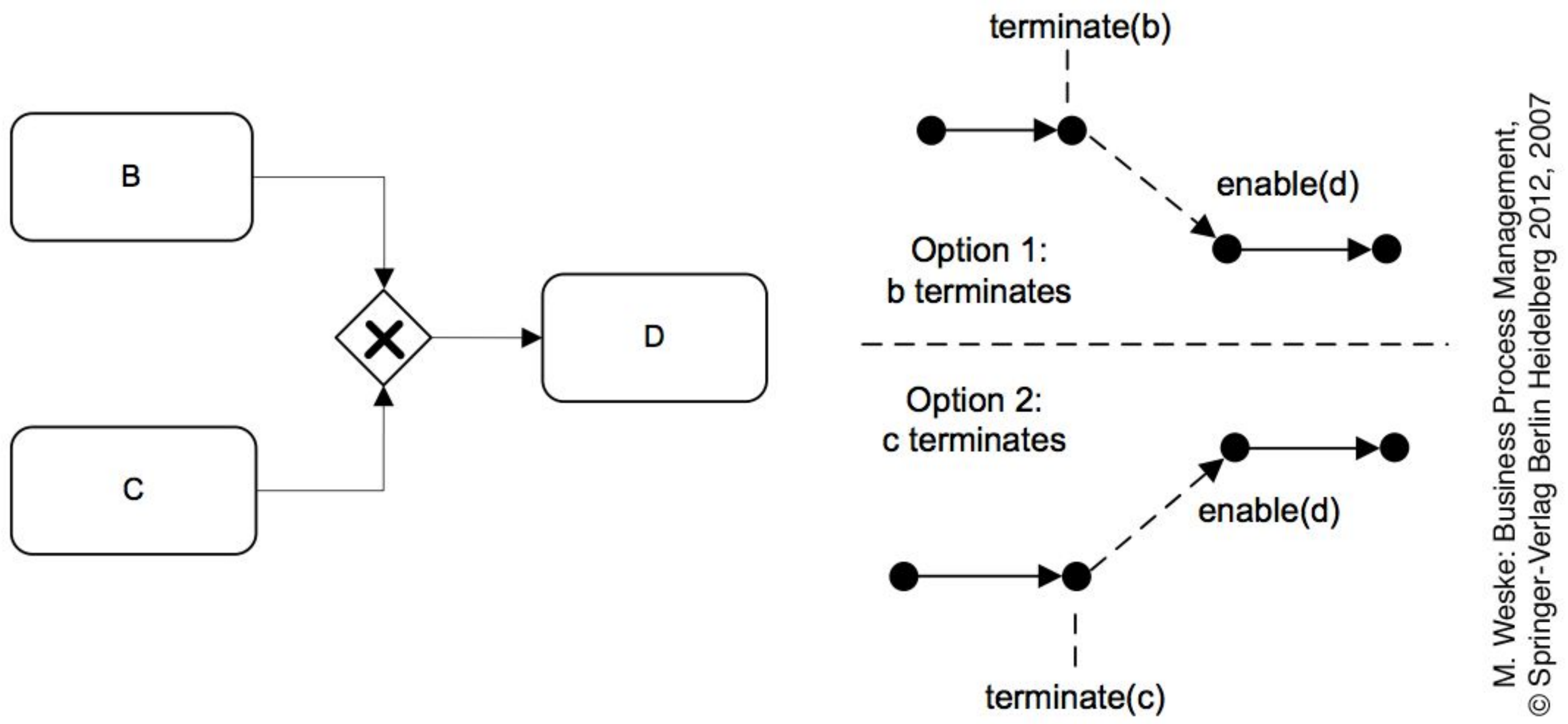


Fig. 4.7. Xor join pattern

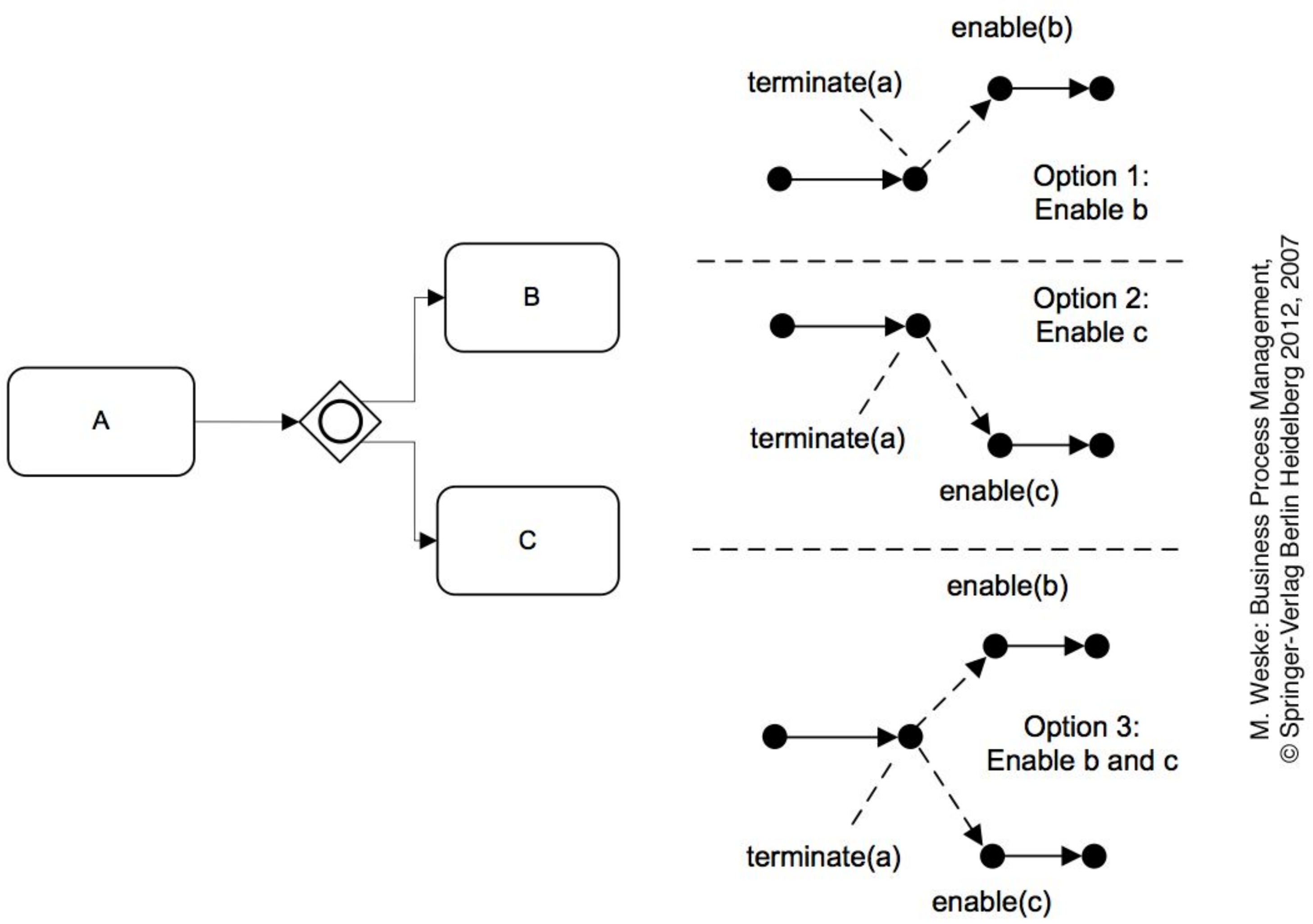


Fig. 4.8. Or split pattern

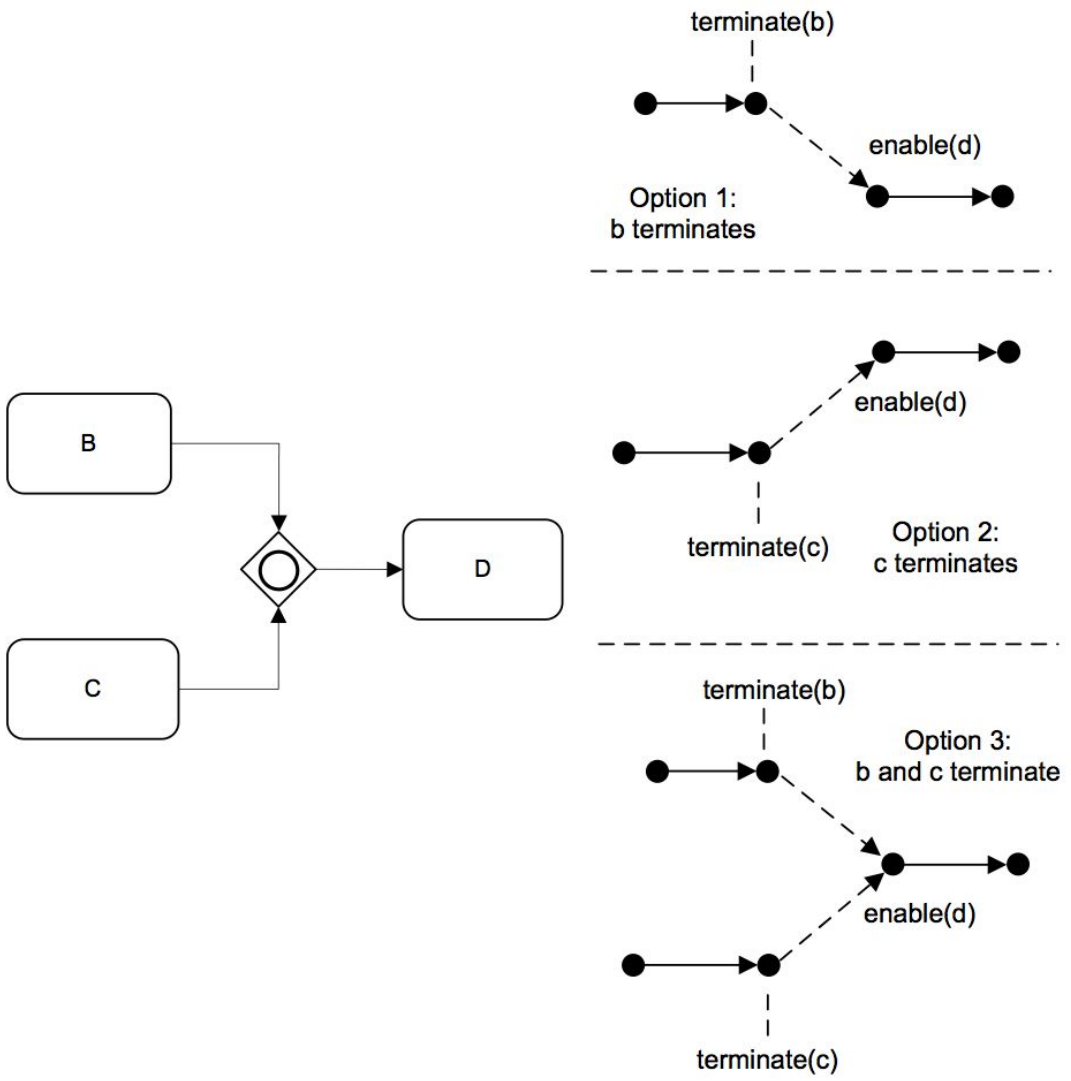
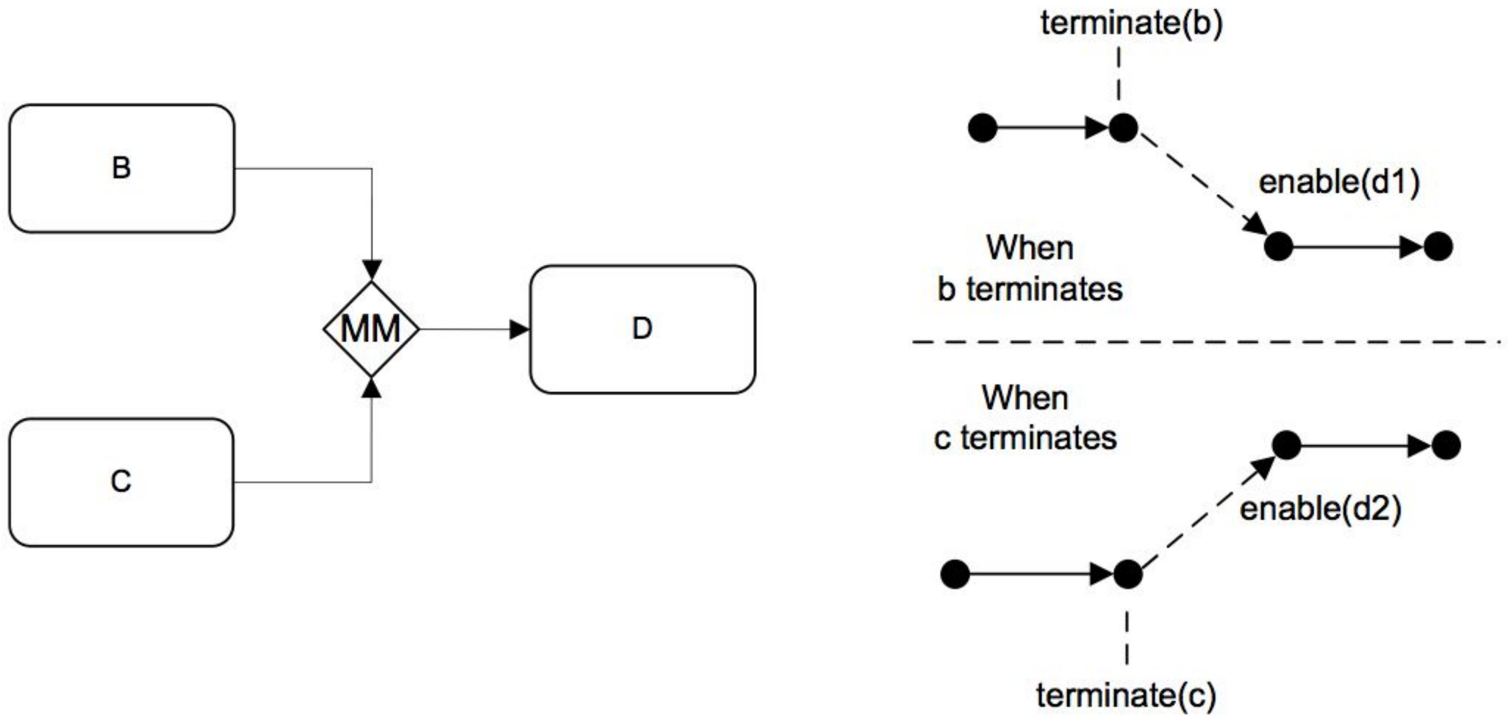


Fig. 4.9. Or join pattern

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007



M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

Fig. 4.10. Multi-merge pattern

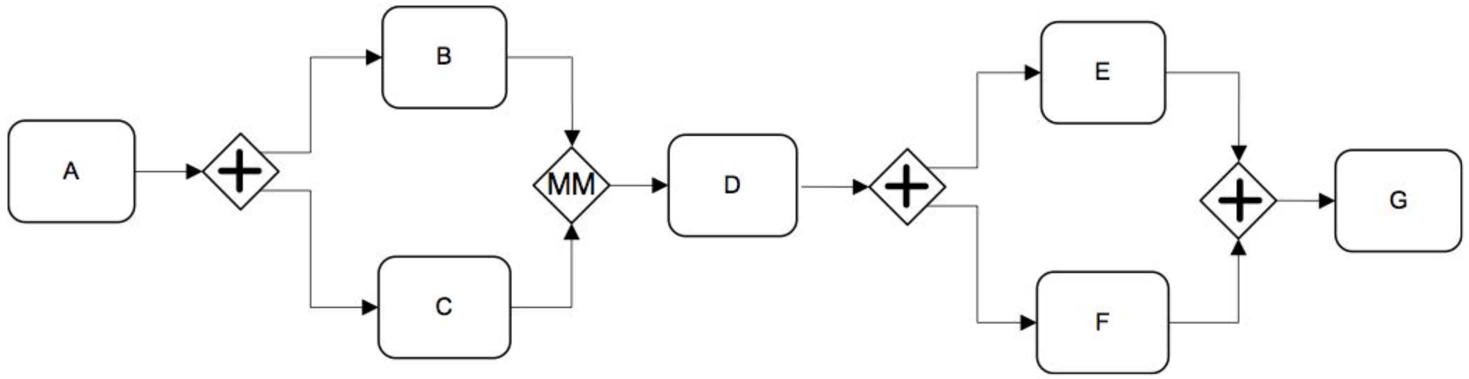


Fig. 4.11. Multi-merge example might lead to incorrect synchronization of branches

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

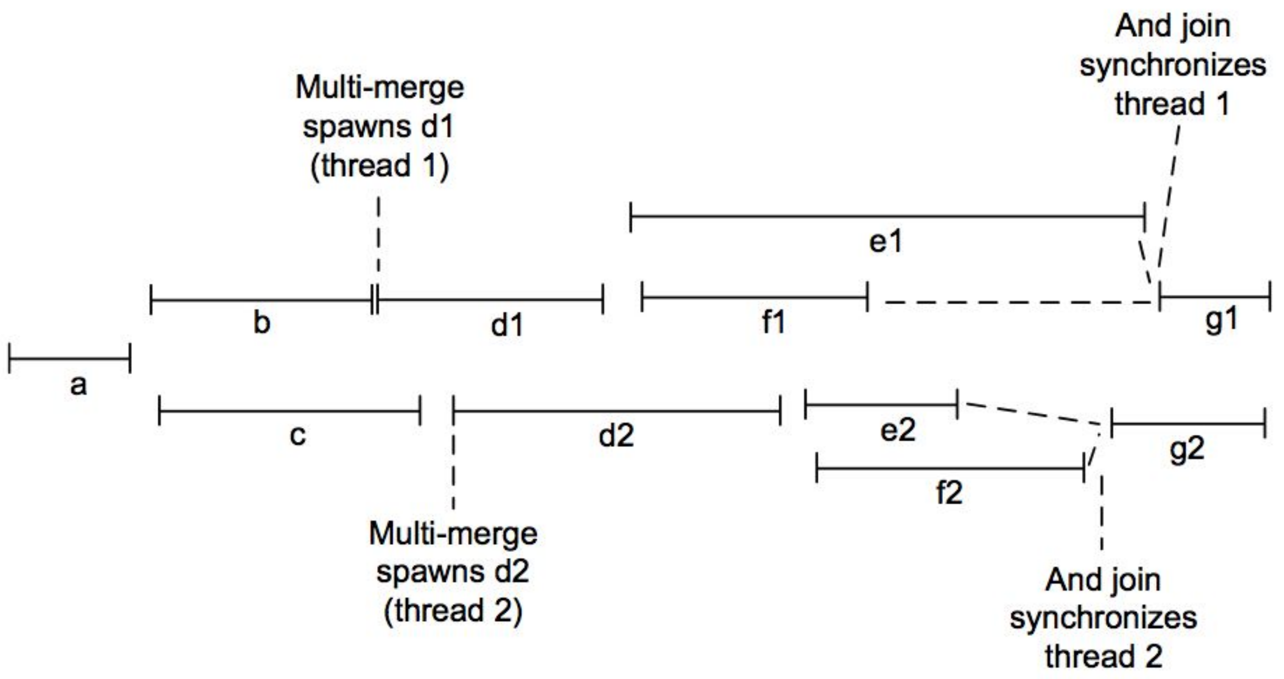

And join synchronizes thread 1
Multi-merge spawns d1 (thread 1)
e1
b
d1
f1
g1
a
c
d2
e2
g2
f2
Multi-merge spawns d2 (thread 2)
And join synchronizes thread 2

Fig. 4.12. Event diagram of a process instance based on the process model shown in Figure 4.11

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

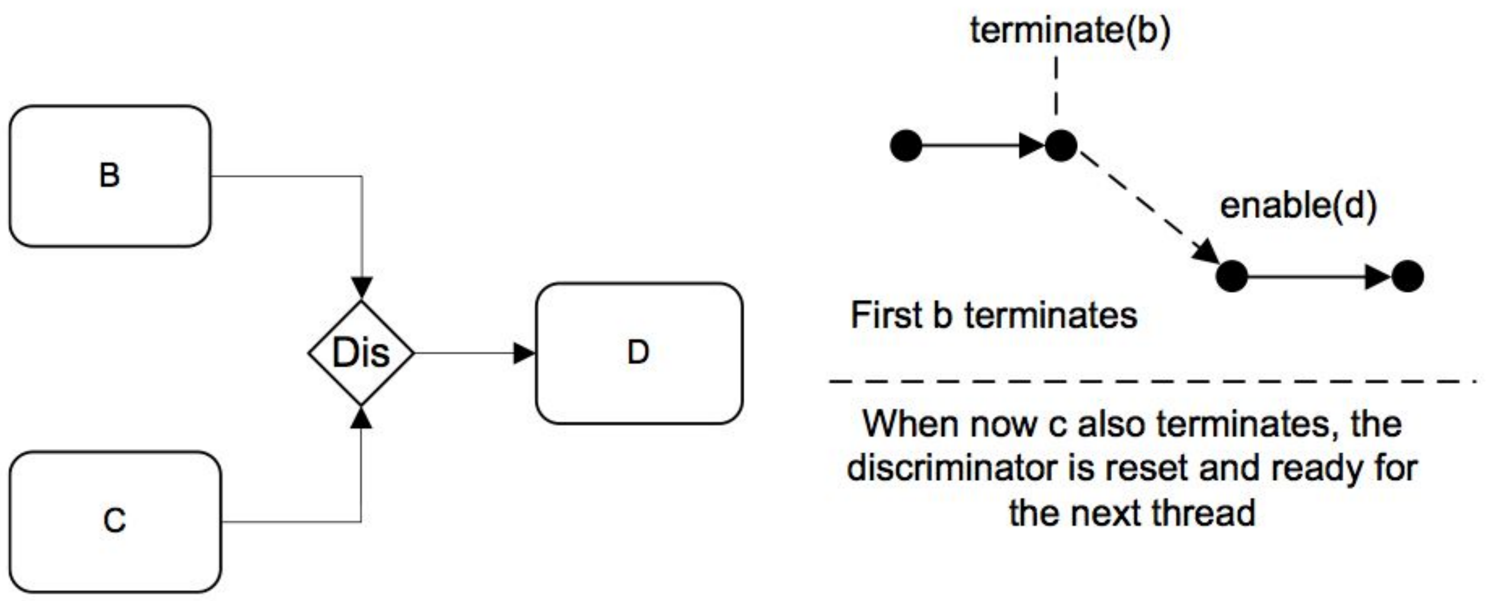

terminate(b)
B
enable(d)
Dis
D
C
First b terminates
When now c also terminates, the discriminator is reset and ready for the next thread

Fig. 4.13. Discriminator pattern

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

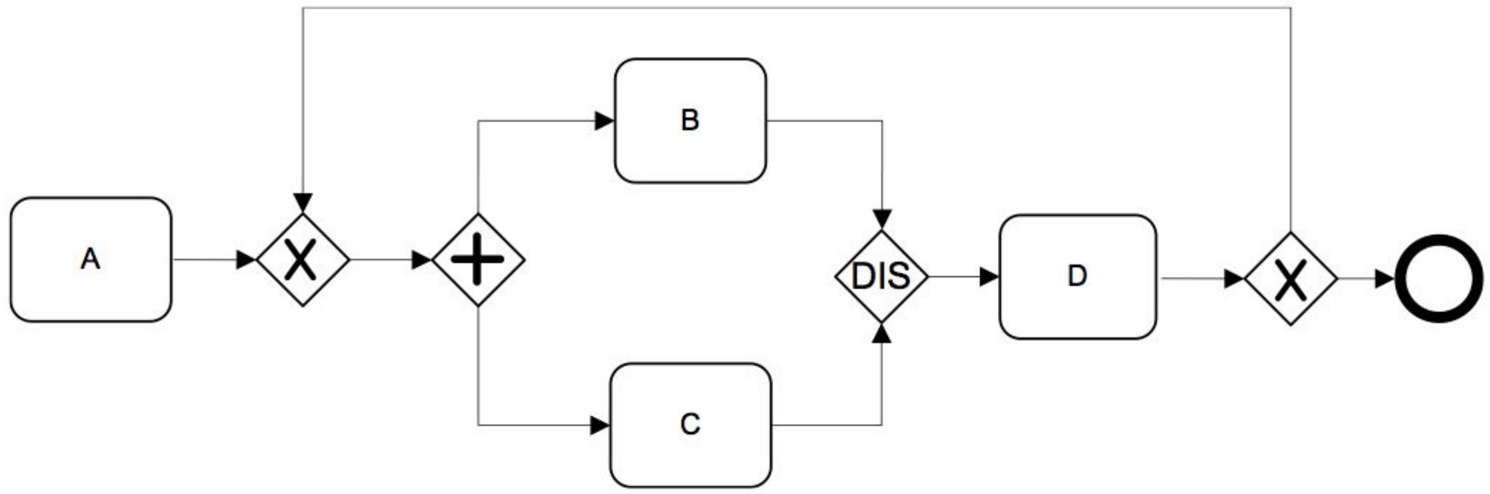


Fig. 4.14. Discriminator example

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

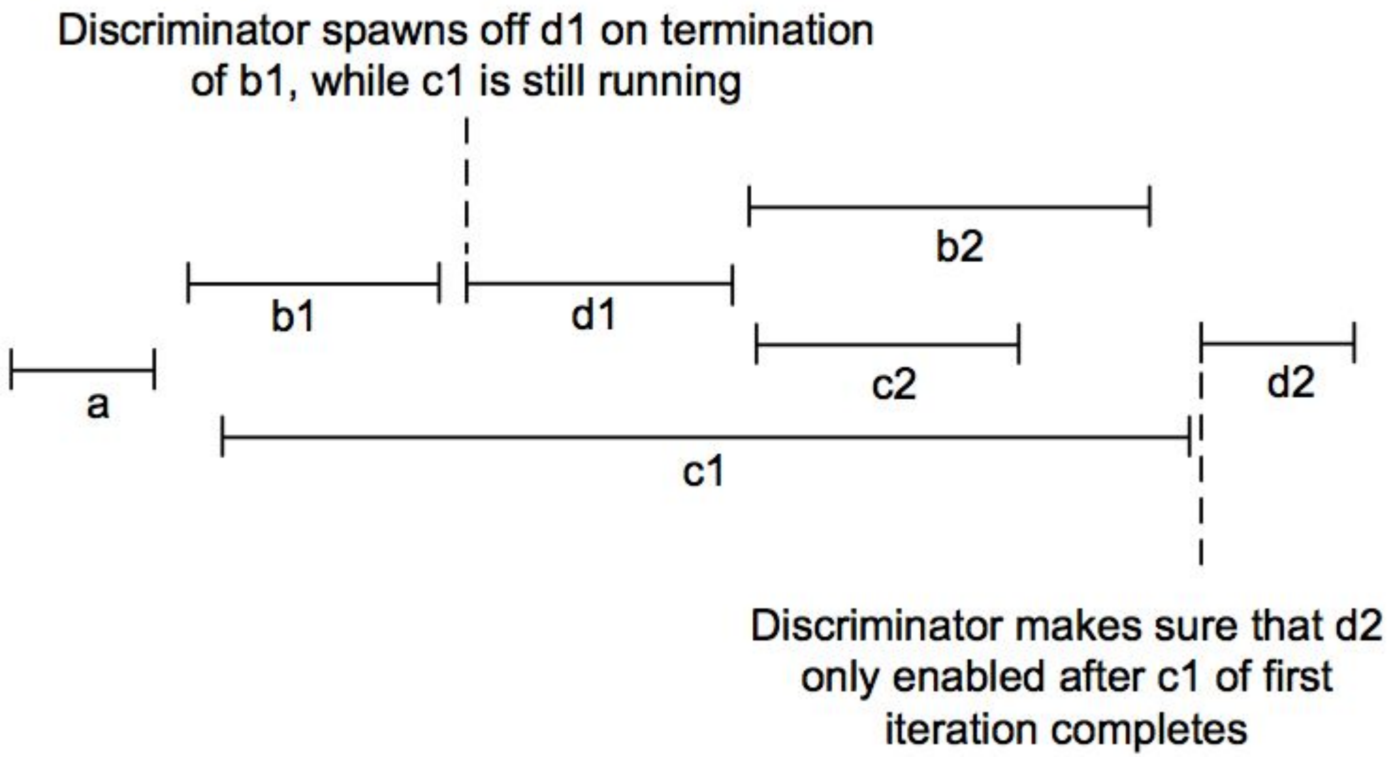
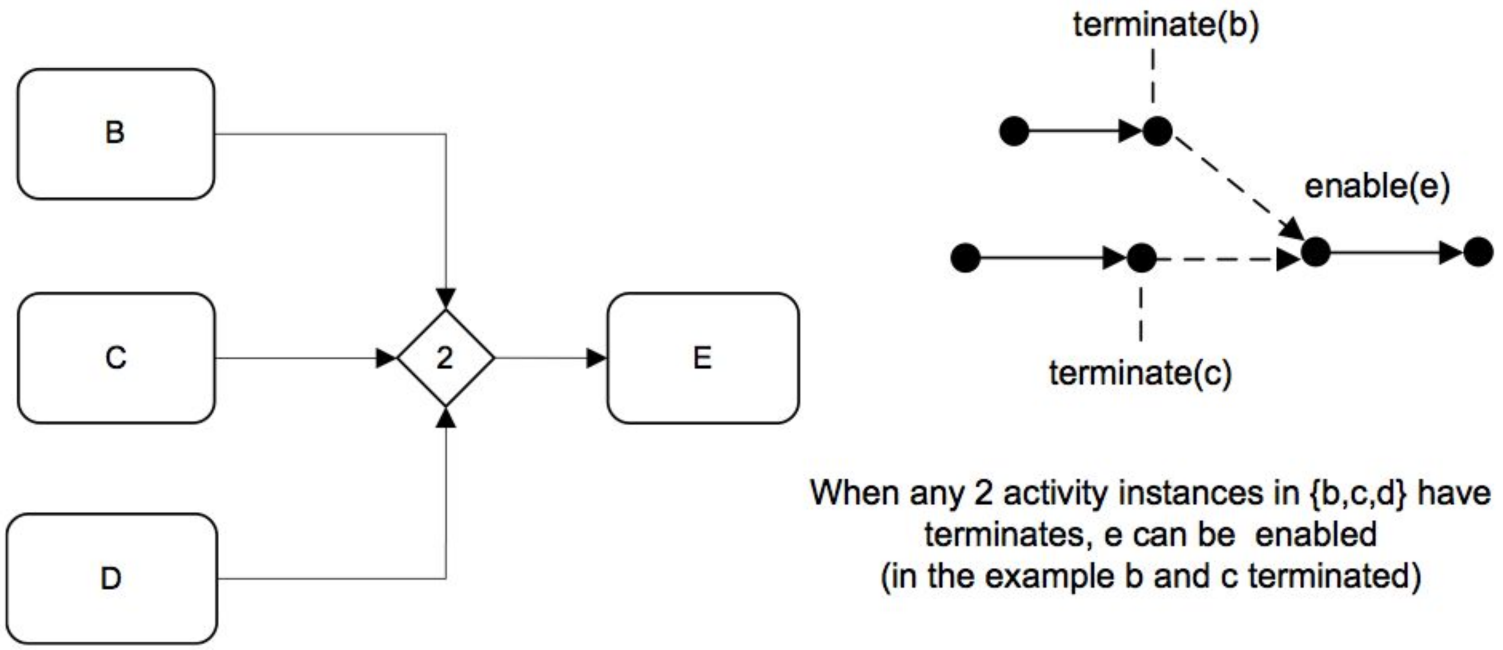

Discriminator spawns off d1 on termination of b1, while c1 is still running
b2
b1
d1
a
c2
d2
c1
Discriminator makes sure that d2 only enabled after c1 of first iteration completes

Fig. 4.15. Event diagram of discriminator example

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007



When any 2 activity instances in {b,c,d} have terminates, e can be enabled (in the example b and c terminated)

Fig. 4.16. N-out-of-M join pattern

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

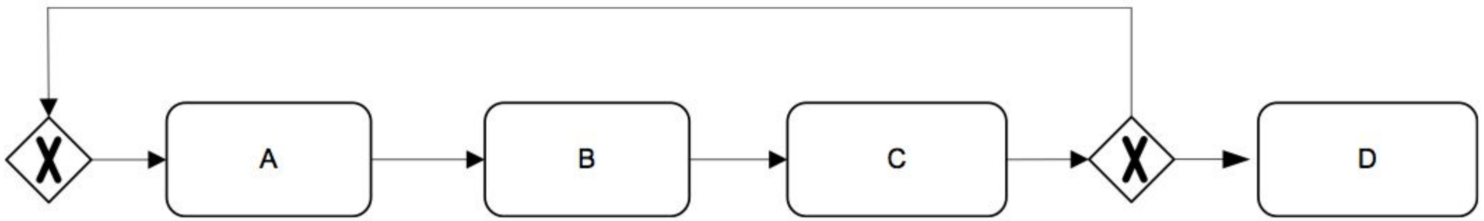


Fig. 4.17. Graphical representation of arbitrary cycles pattern

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

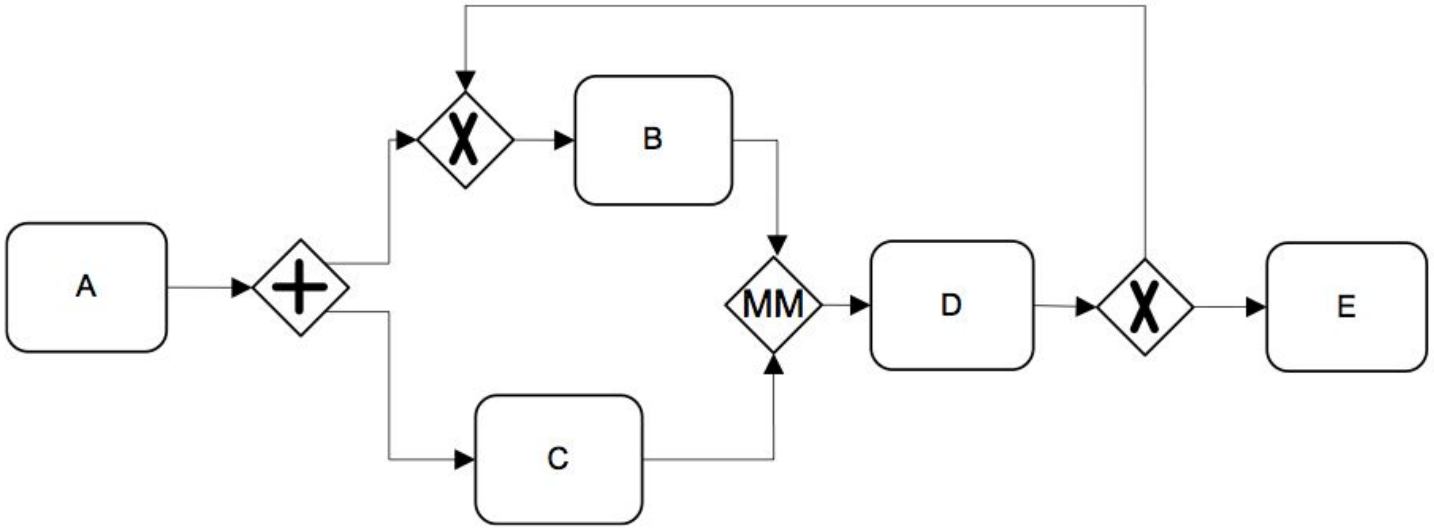


Fig. 4.18. Arbitrary cycles example, using multiple merge pattern

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

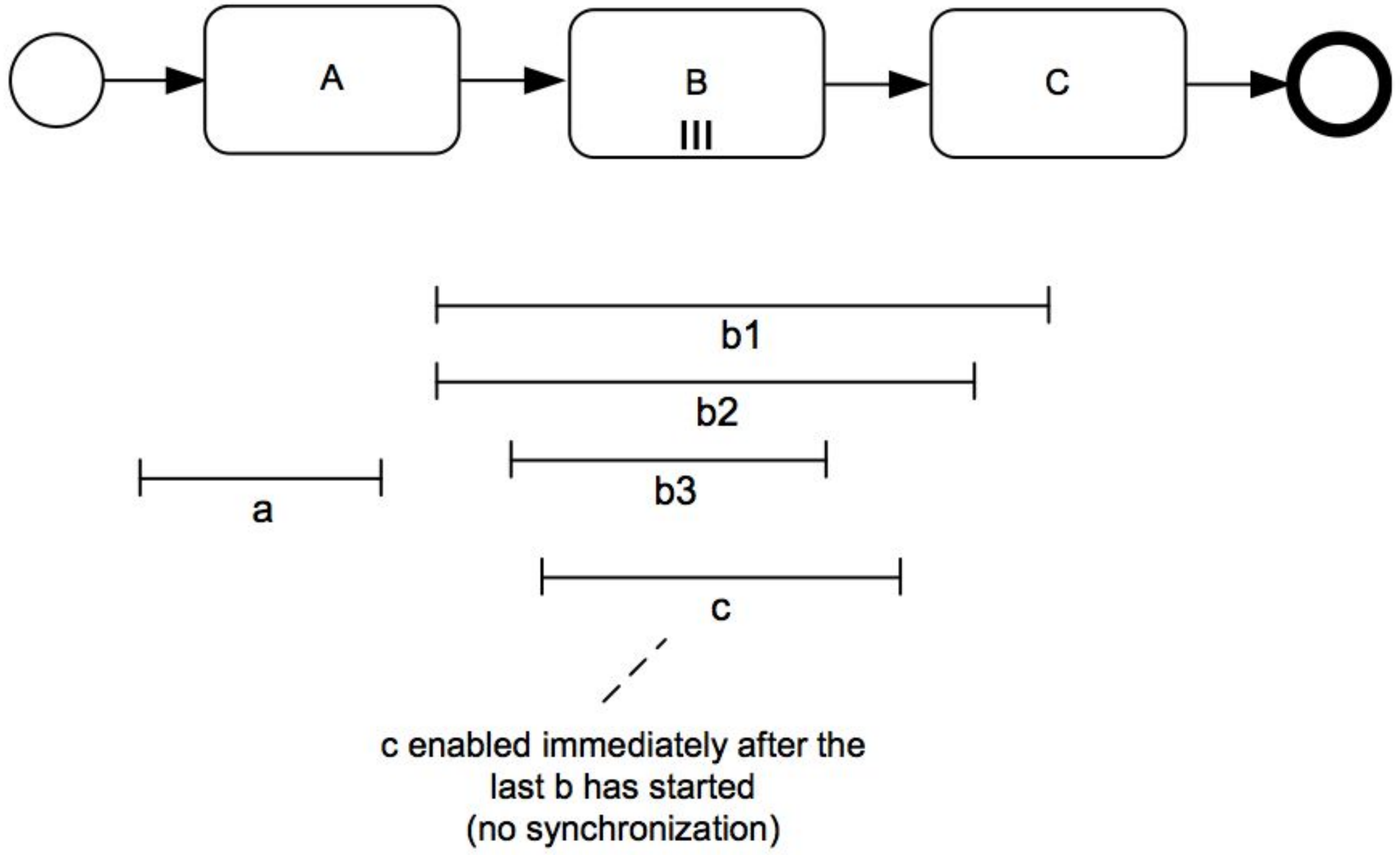


Fig. 4.19. Example for multiple instances without synchronization

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

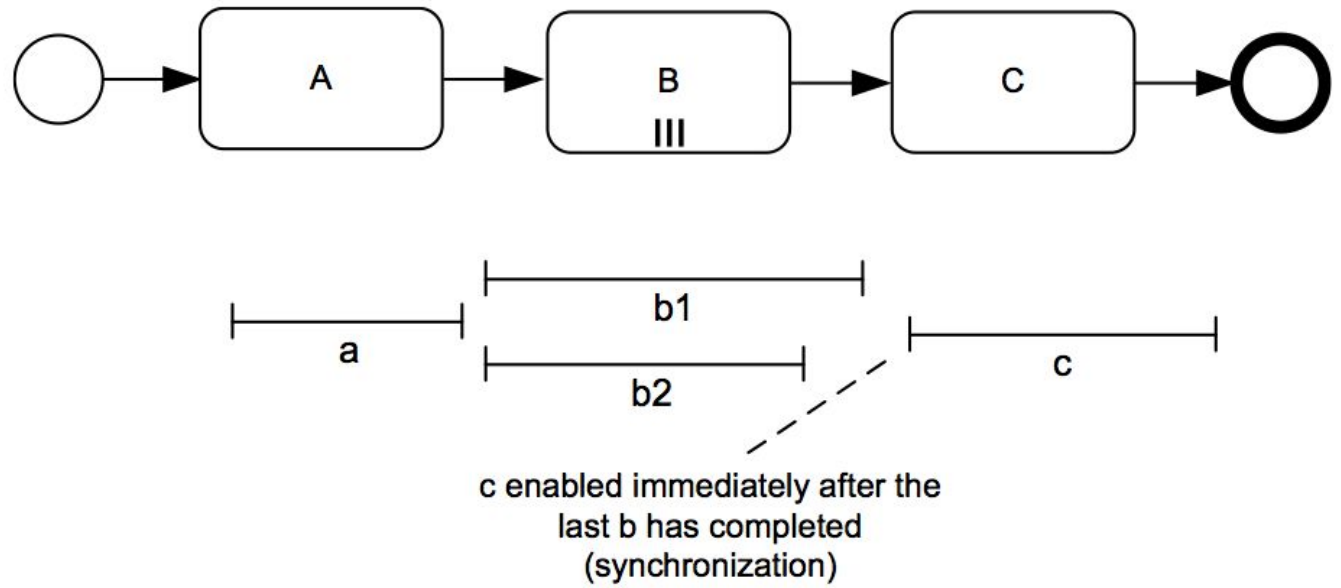

A
B
III
C
a
b1
b2
c
c enabled immediately after the last b has completed (synchronization)

Fig. 4.20. Example for multiple instances with a priori design time knowledge

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

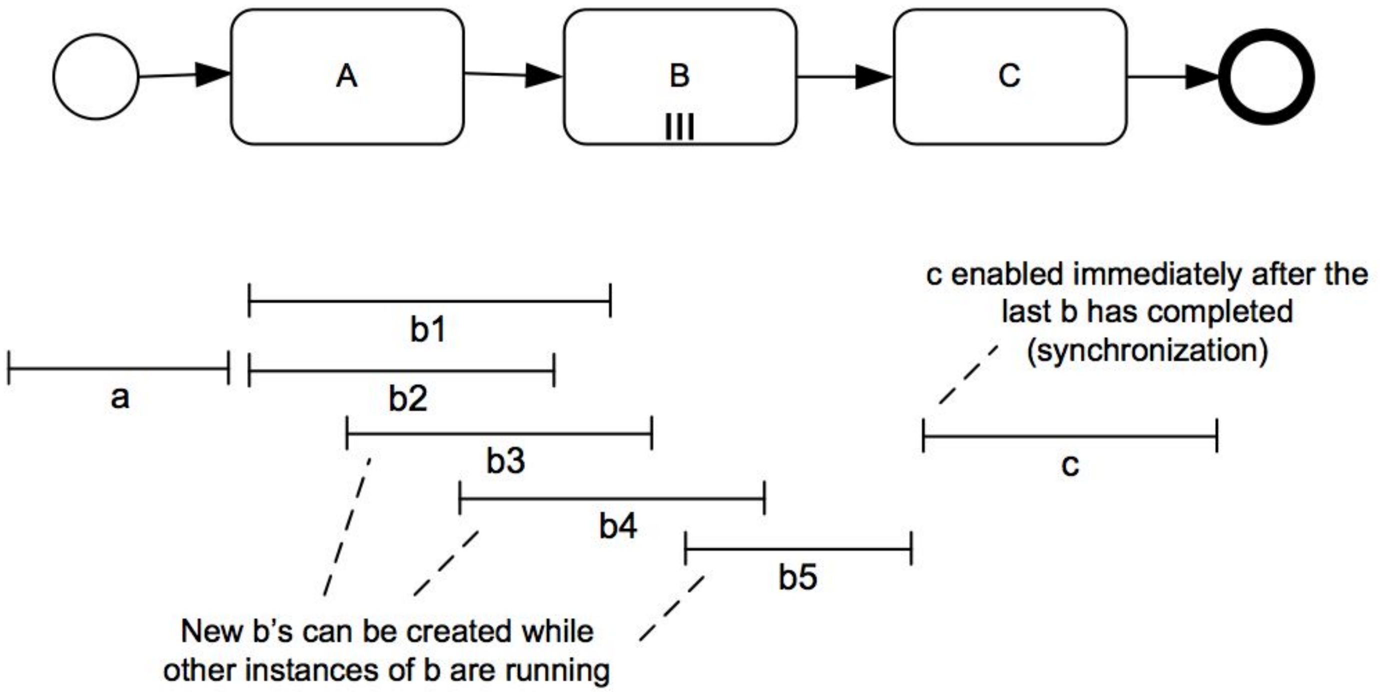

A
B
C
b1
a
b2
b3
b4
b5
c
c enabled immediately after the last b has completed (synchronization)
New b's can be created while other instances of b are running

Fig. 4.21. Example for multiple instances without a priori run time knowledge pattern

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

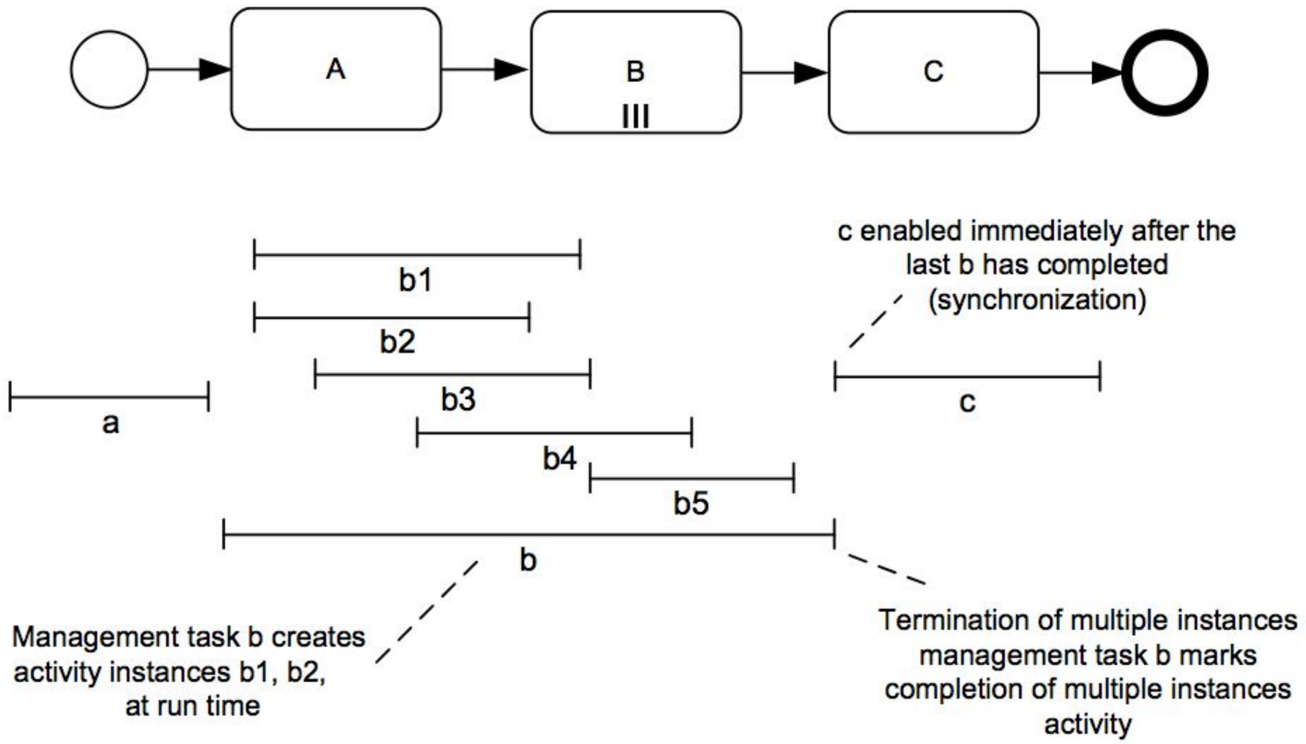

A
B
III
C
b1
b2
b3
b4
b5
a
b
c
c enabled immediately after the last b has completed (synchronization)
Management task b creates activity instances b1, b2, at run time
Termination of multiple instances management task b marks completion of multiple instances activity

Fig. 4.22. Multiple instances without a priori run time knowledge pattern, including management task

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

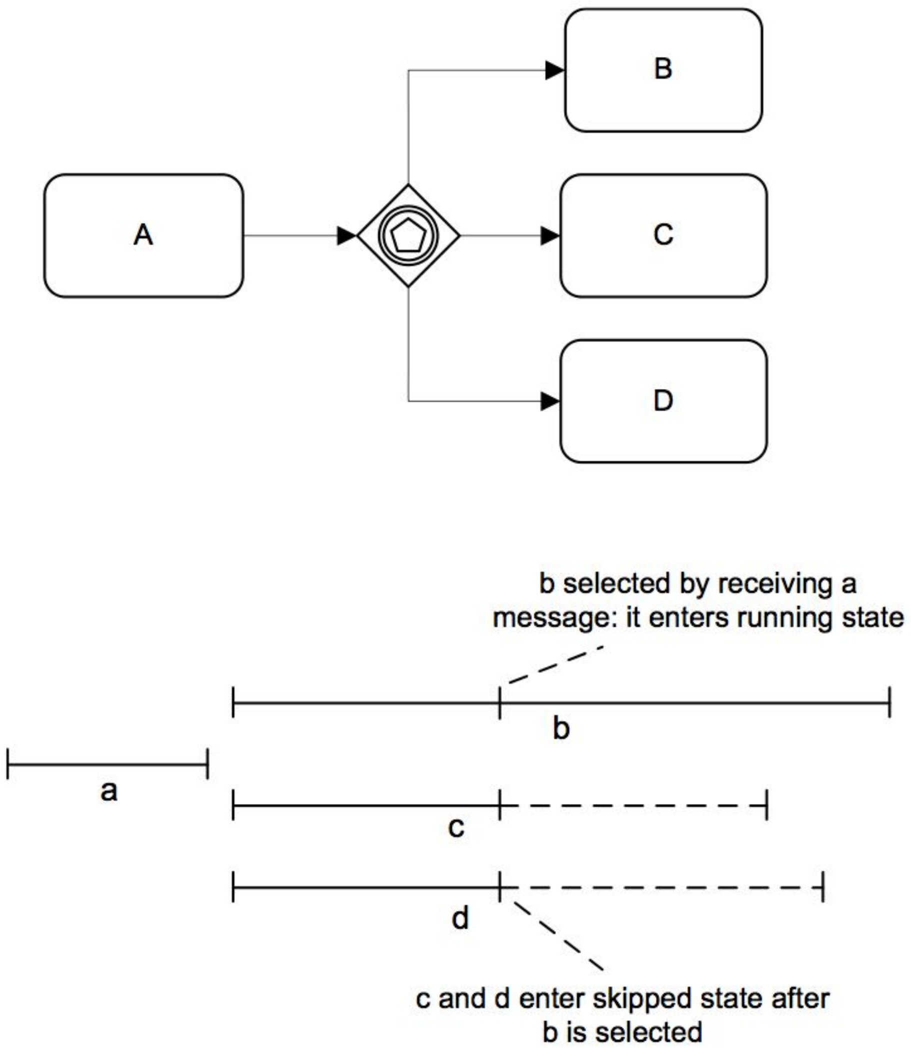

B
A
C
D
b selected by receiving a message: it enters running state
b
a
c
d
c and d enter skipped state after b is selected

Fig. 4.23. Example of deferred choice pattern

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

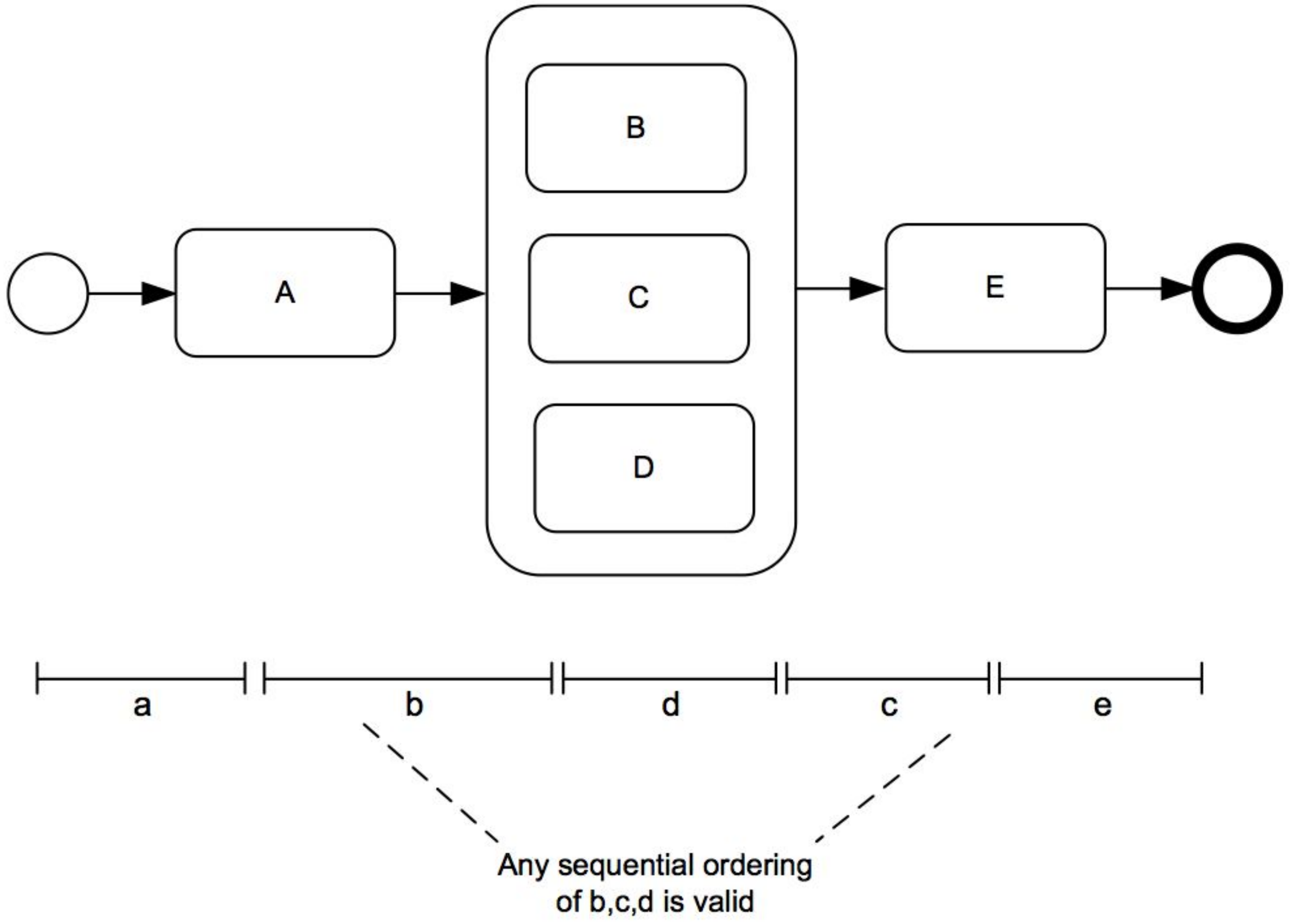
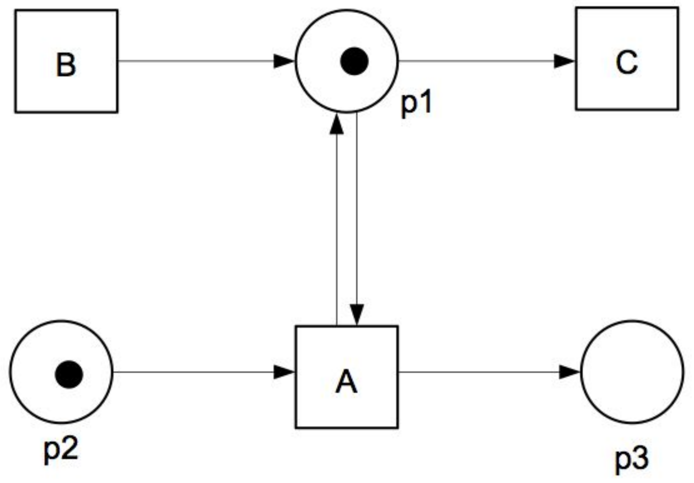
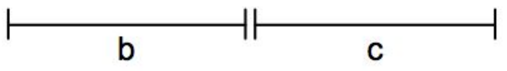


Fig. 4.24. Sequential execution without a priori design time knowledge; any sequential execution ordering of B, C, and D is possible

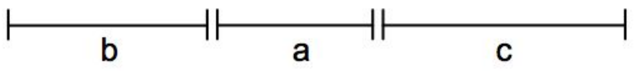
M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007



Option 1: *c* is immediately started after *b* completes, so that *a* cannot be executed



Option 2: *a* is started after *b* completes and before *c* starts



More Options: Multiple *a*'s possible, as long as *c* is not started and there are tokens in *p2*

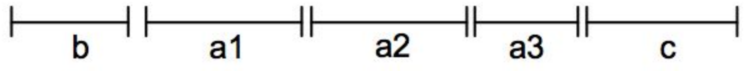


Fig. 4.25. Example of milestone pattern

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

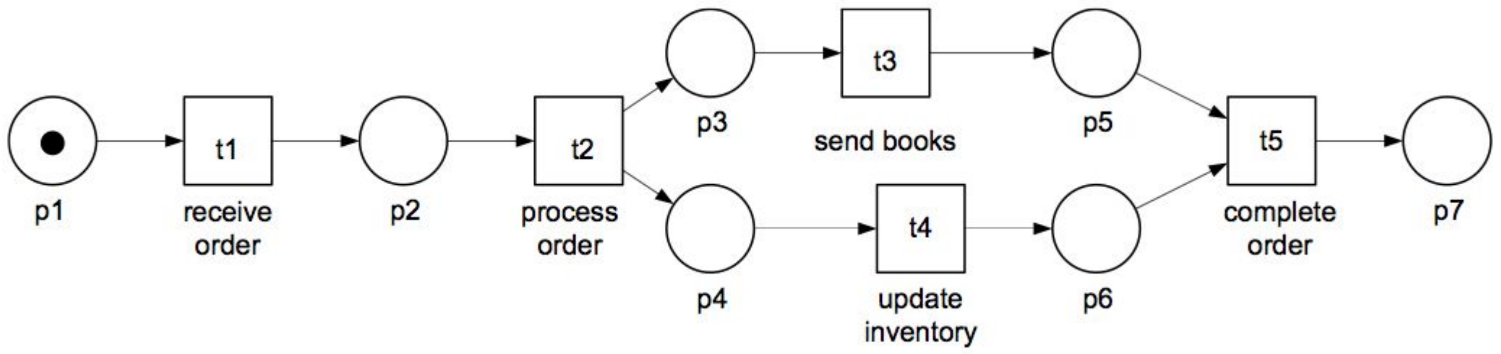


Fig. 4.26. Sample Petri net representing single process instance

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

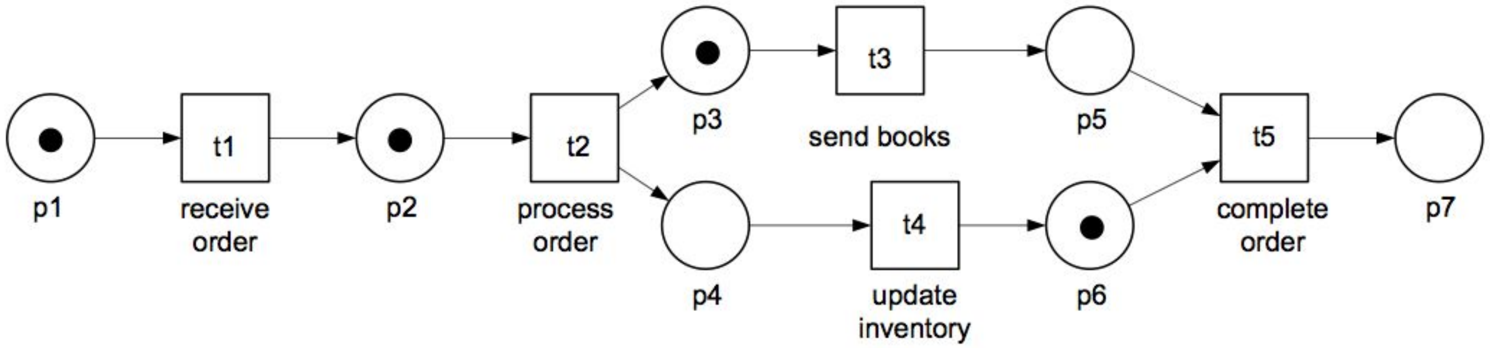


Fig. 4.27. Sample Petri net representing multiple process instances

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

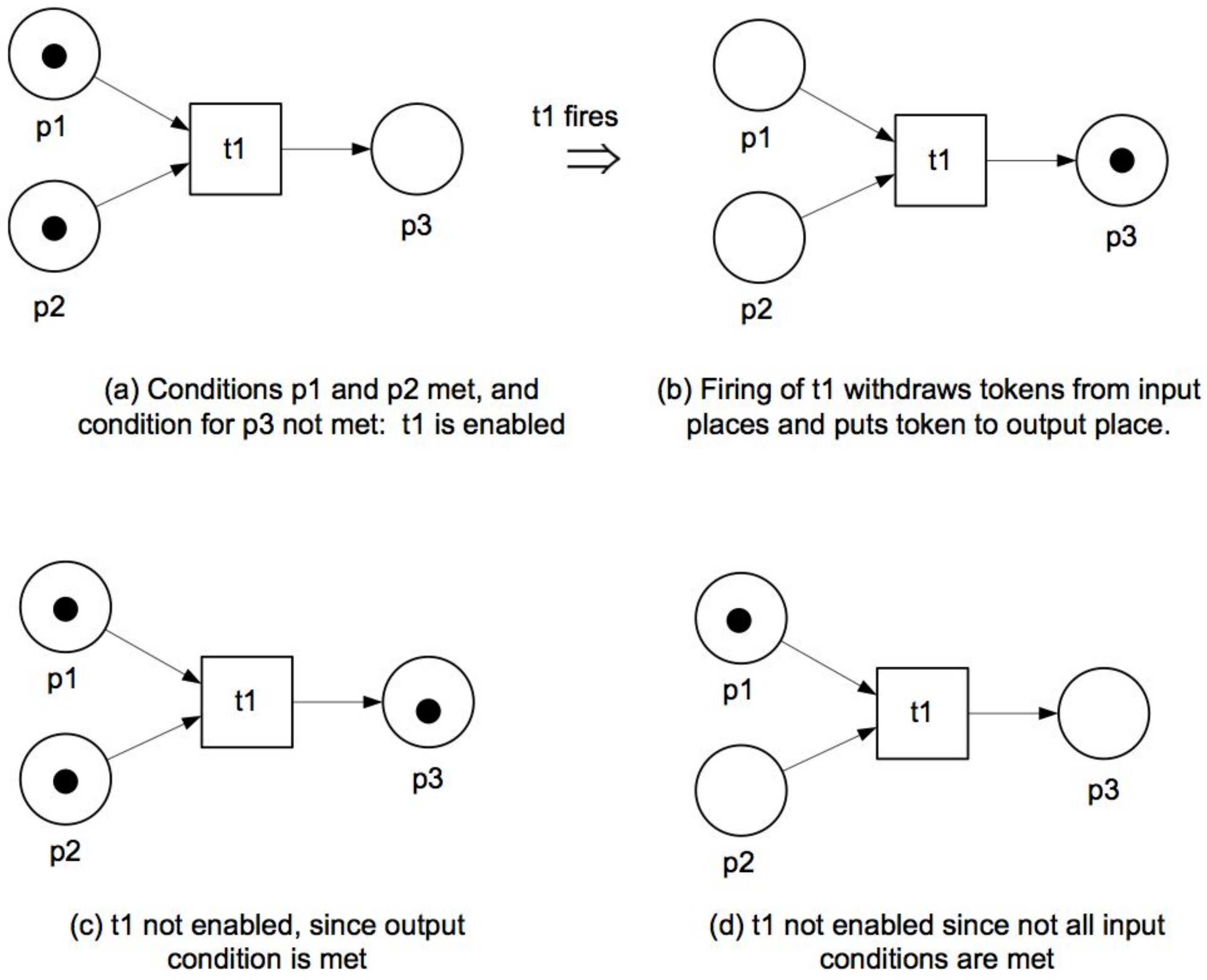

p1
t1
p2
p3
t1 fires
⇒
p1
t1
p2
p3
(a) Conditions p1 and p2 met, and condition for p3 not met: t1 is enabled
(b) Firing of t1 withdraws tokens from input places and puts token to output place.
p1
t1
p2
p3
p1
t1
p2
p3
(c) t1 not enabled, since output condition is met
(d) t1 not enabled since not all input conditions are met

Fig. 4.28. Condition event net

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

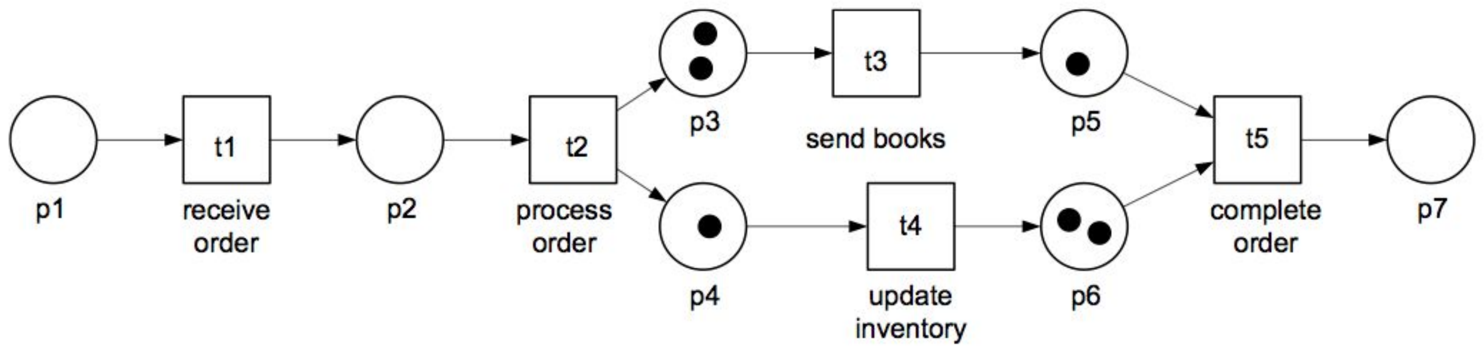


Fig. 4.29. Place transition net with multiple process instances

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

HPI Hasso Plattner Institut

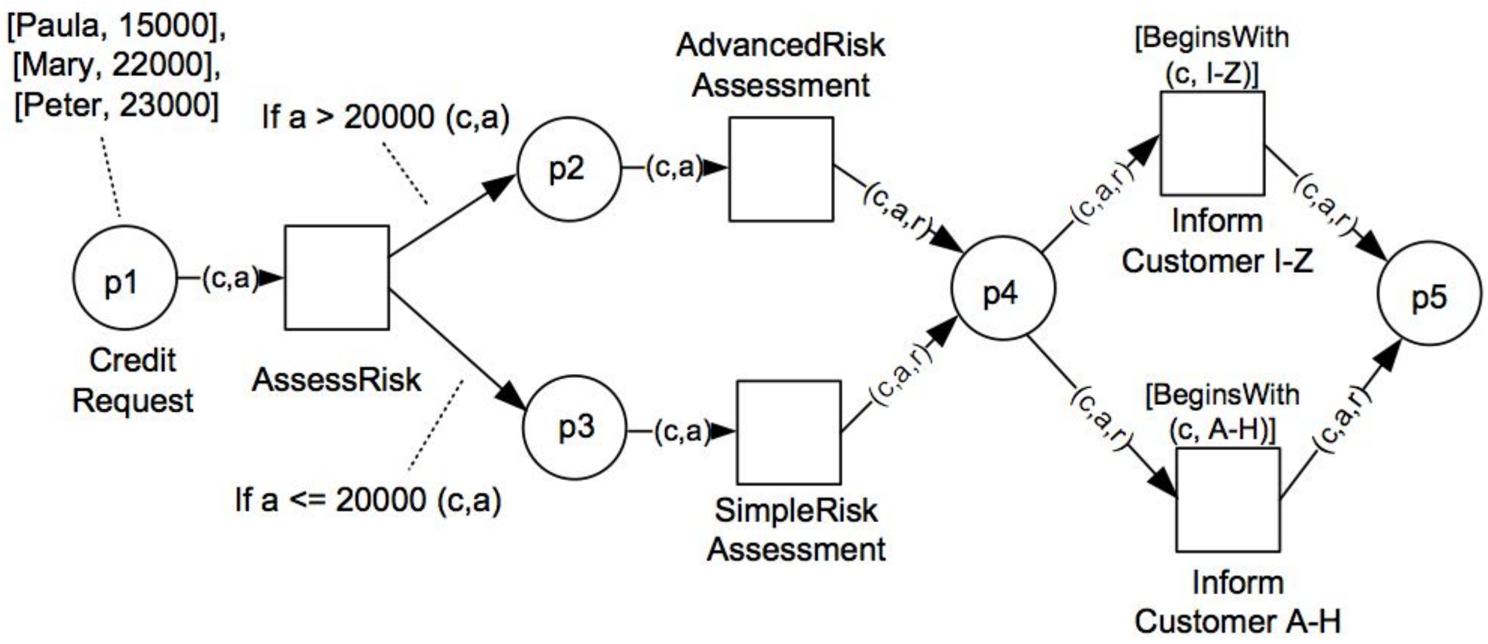


Fig. 4.30. Sample coloured Petri net

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

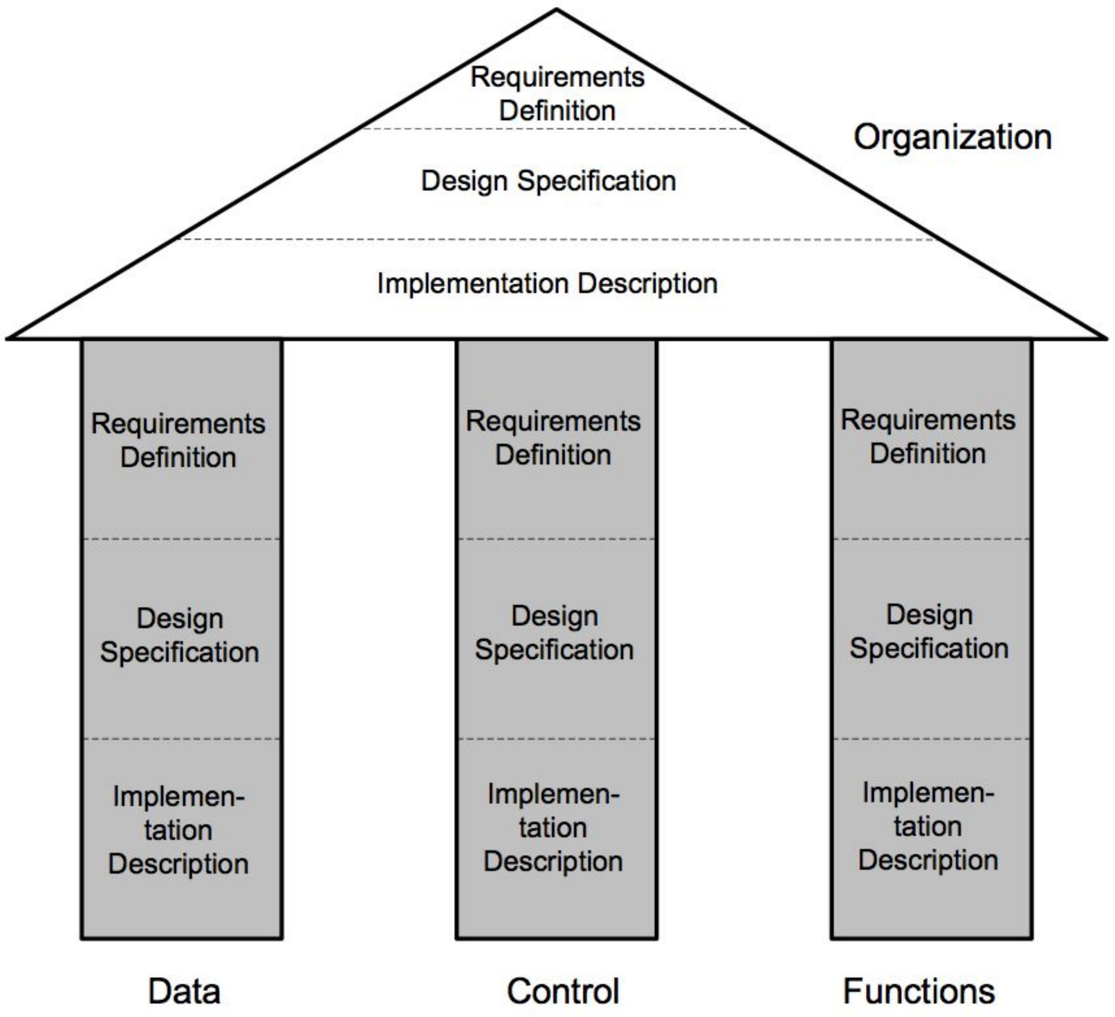


Fig. 4.31. ARIS business process framework, Scheer (2000)

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

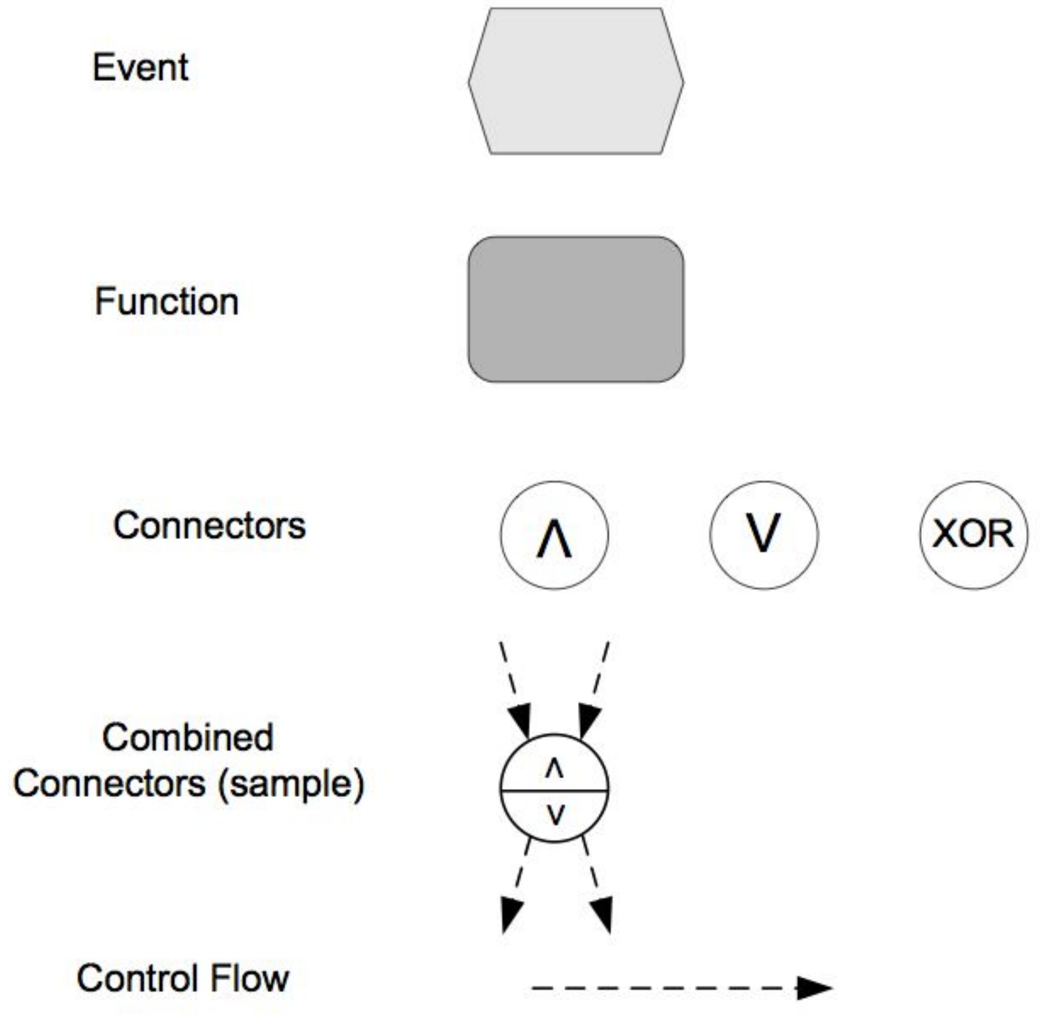


Fig. 4.32. Building blocks of event-driven process chains

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

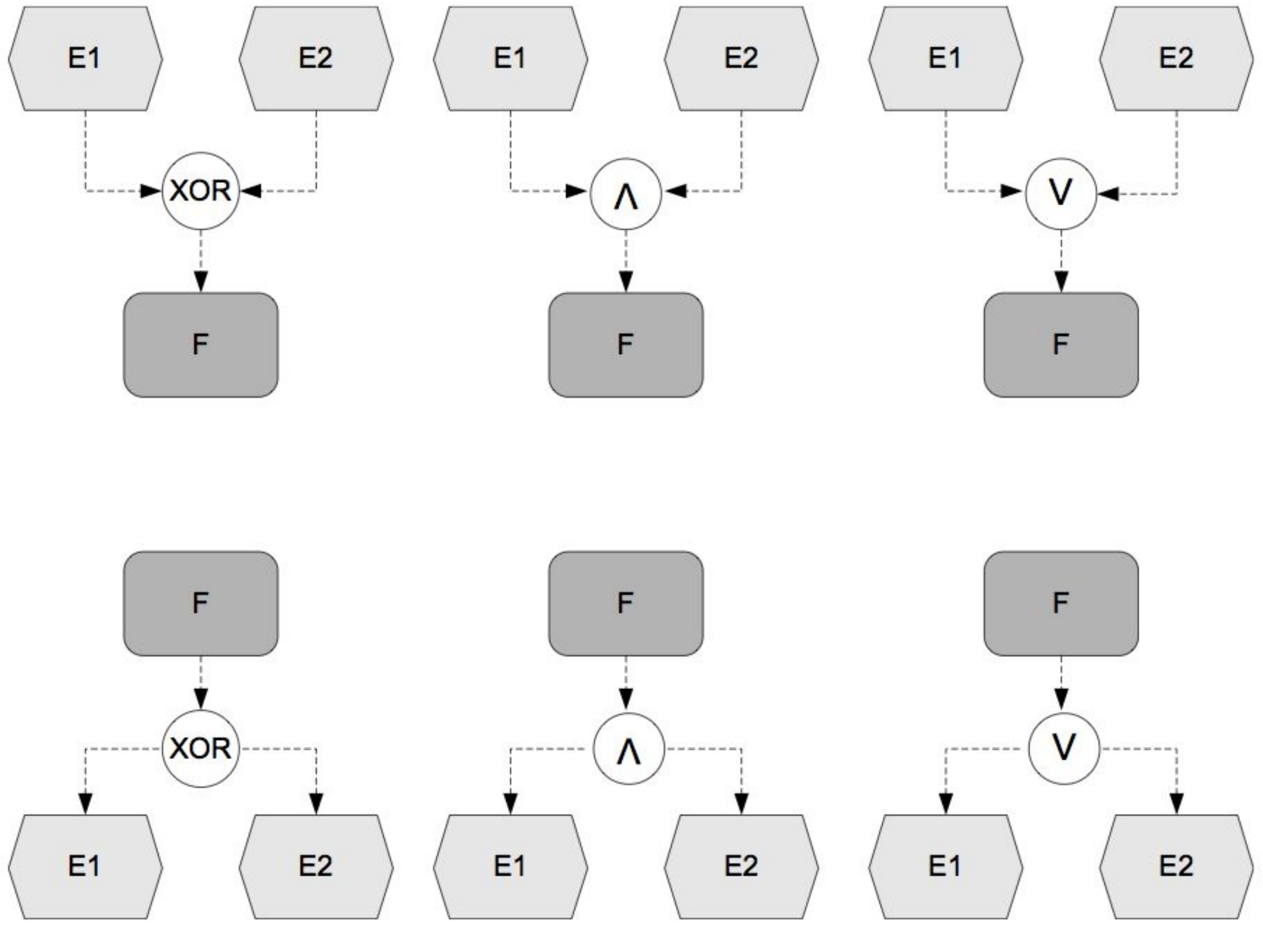


Fig. 4.33. Syntax rules on event-driven process chains: multiple events, single function

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

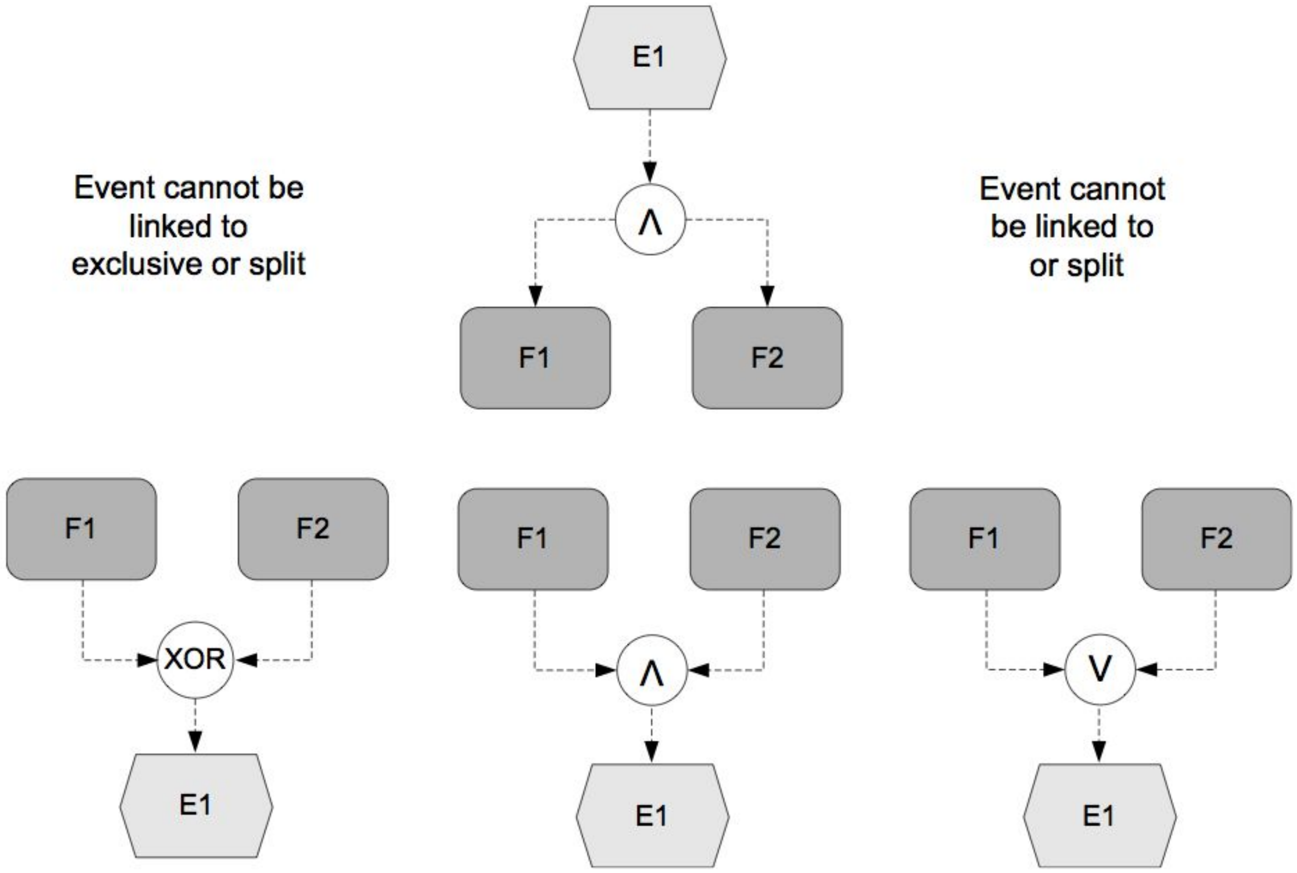

E1
Λ
F1
F2
Event cannot be linked to exclusive or split
Event cannot be linked to or split
F1
F2
XOR
E1
F1
F2
Λ
E1
F1
F2
V
E1

Fig. 4.34. Syntax rules on event-driven process chains: multiple functions, single event

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

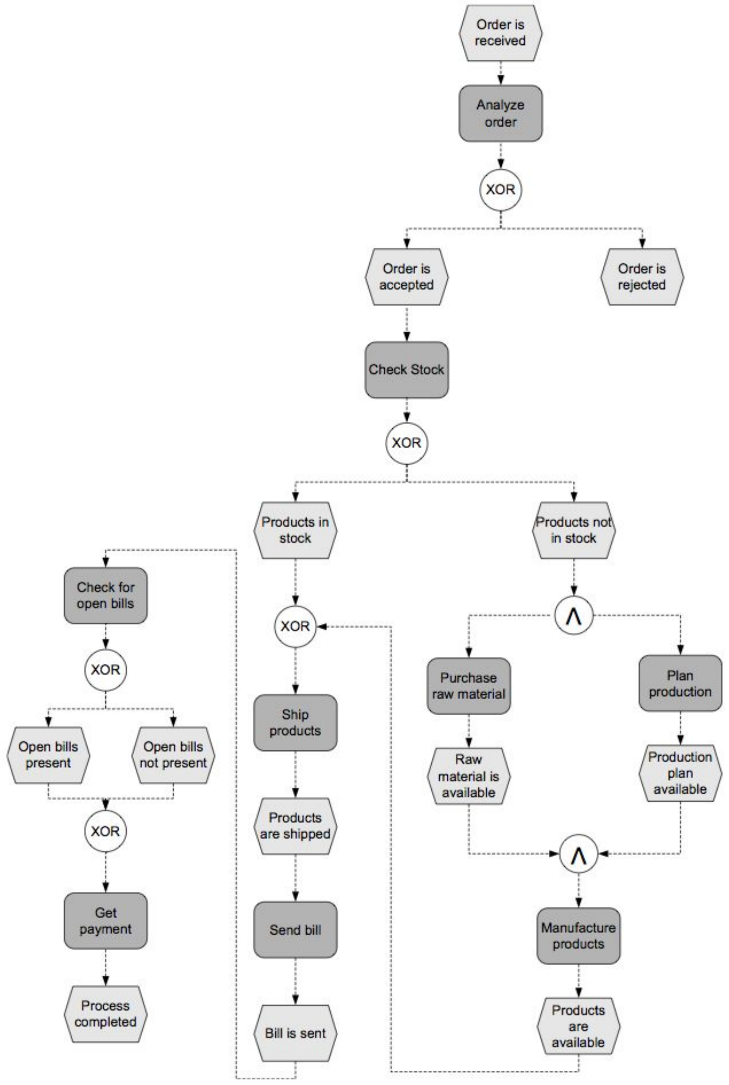


Fig. 4.35. Example event-driven process chain

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

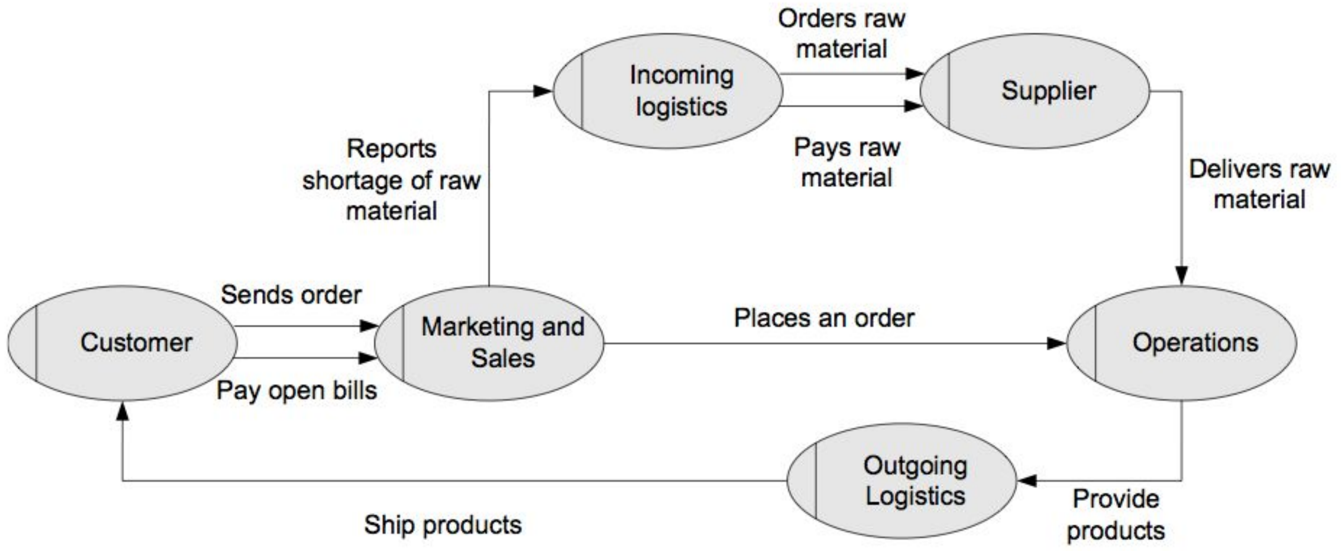


Fig. 4.36. Sample interaction flow diagram, adapted from Scheer et al. (2005)

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

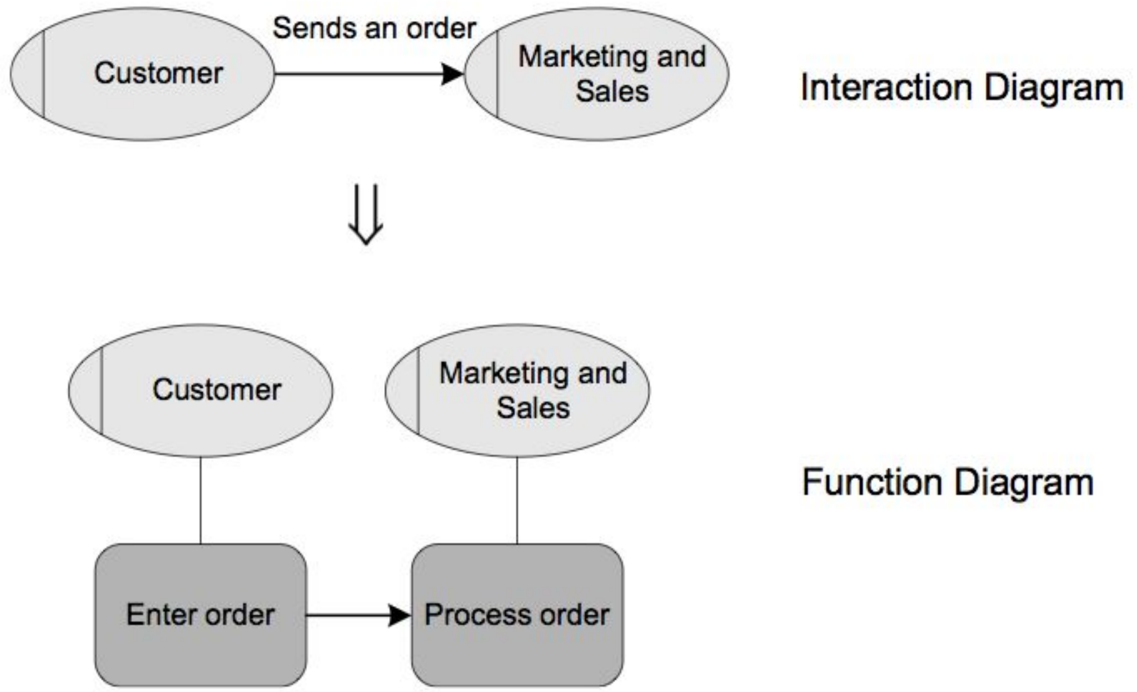


Fig. 4.37. Mapping interactions to relationships between functions

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

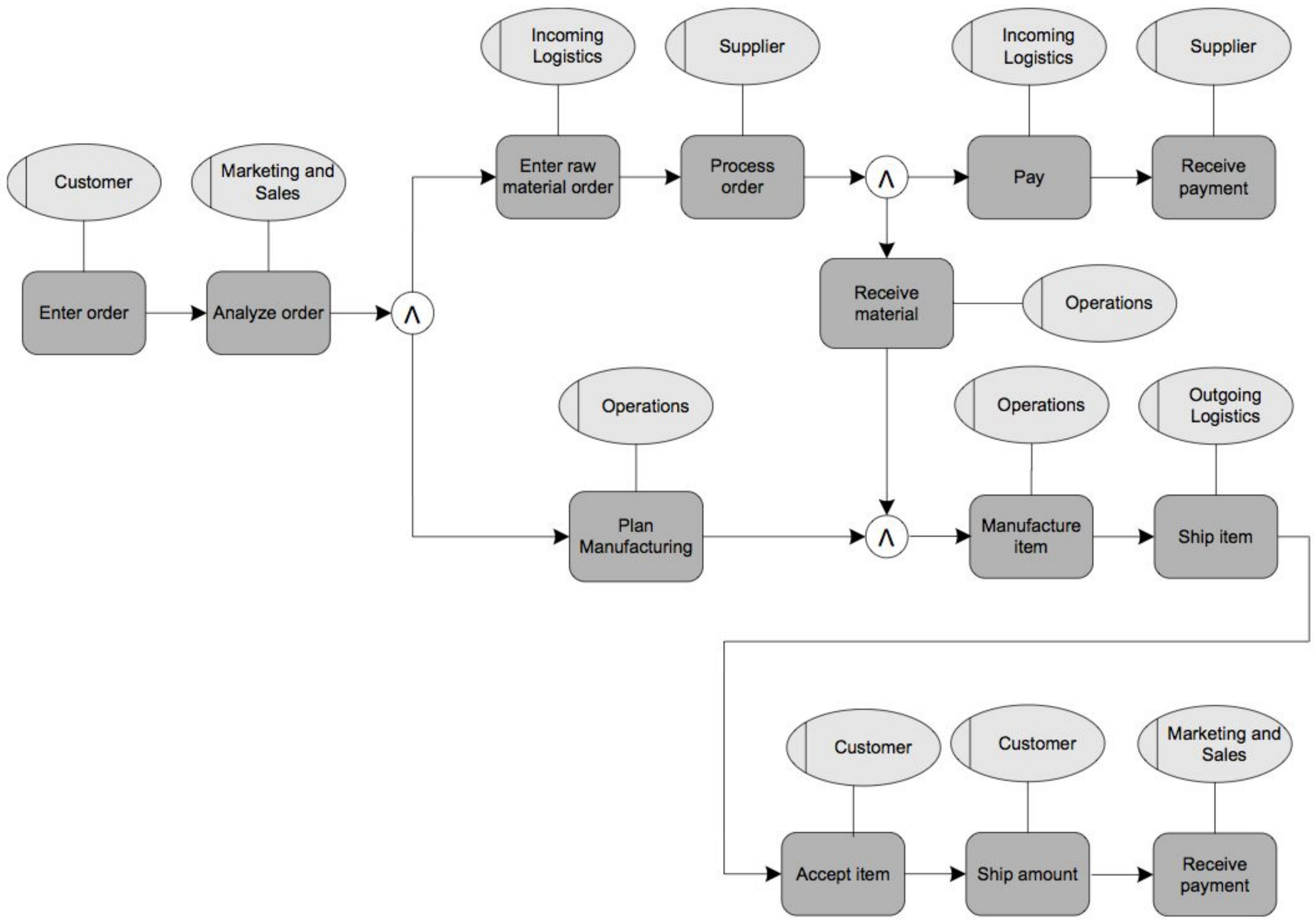


Fig. 4.38. Sample function flow

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

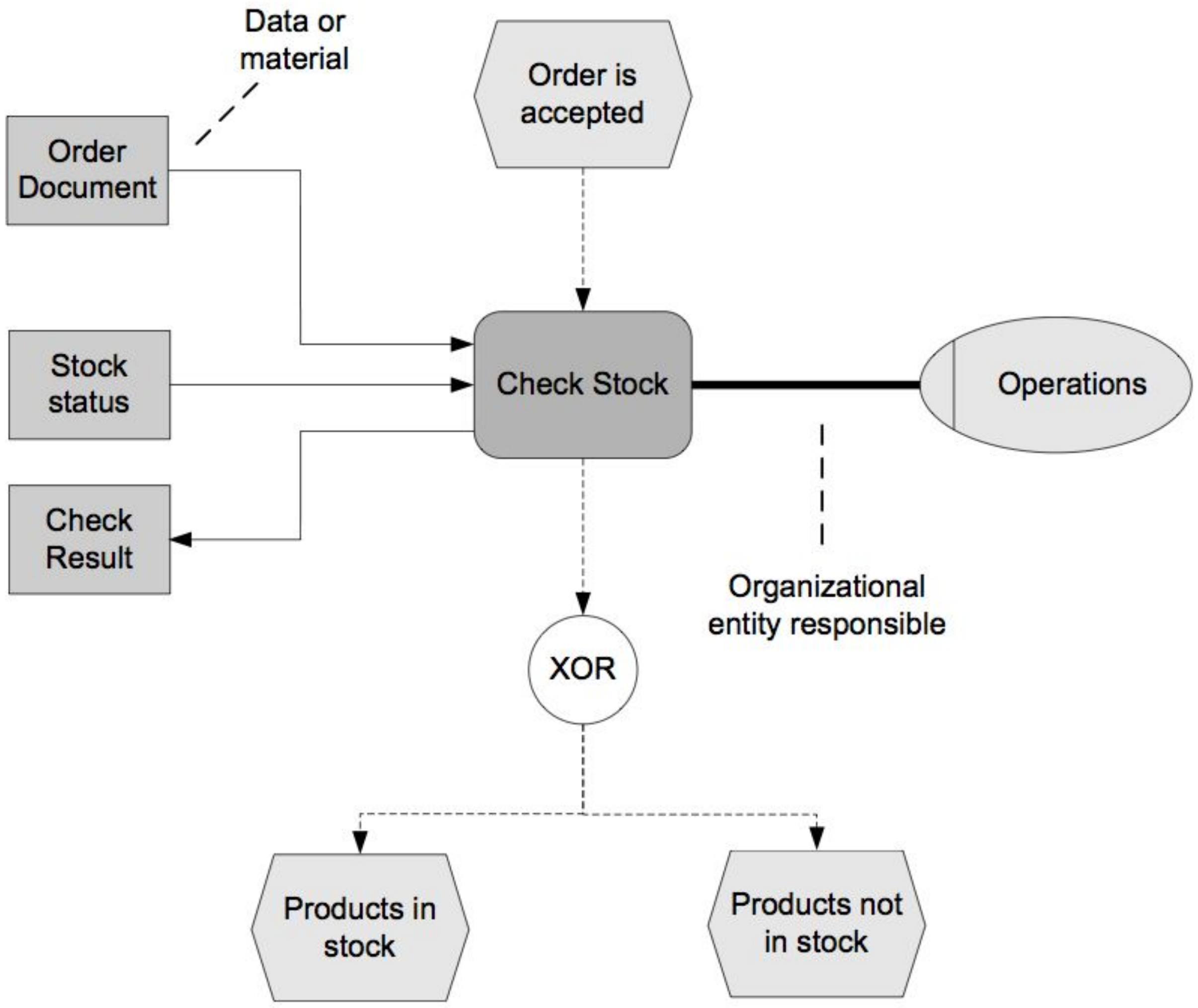


Fig. 4.39. Example of extended event-driven process chain

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

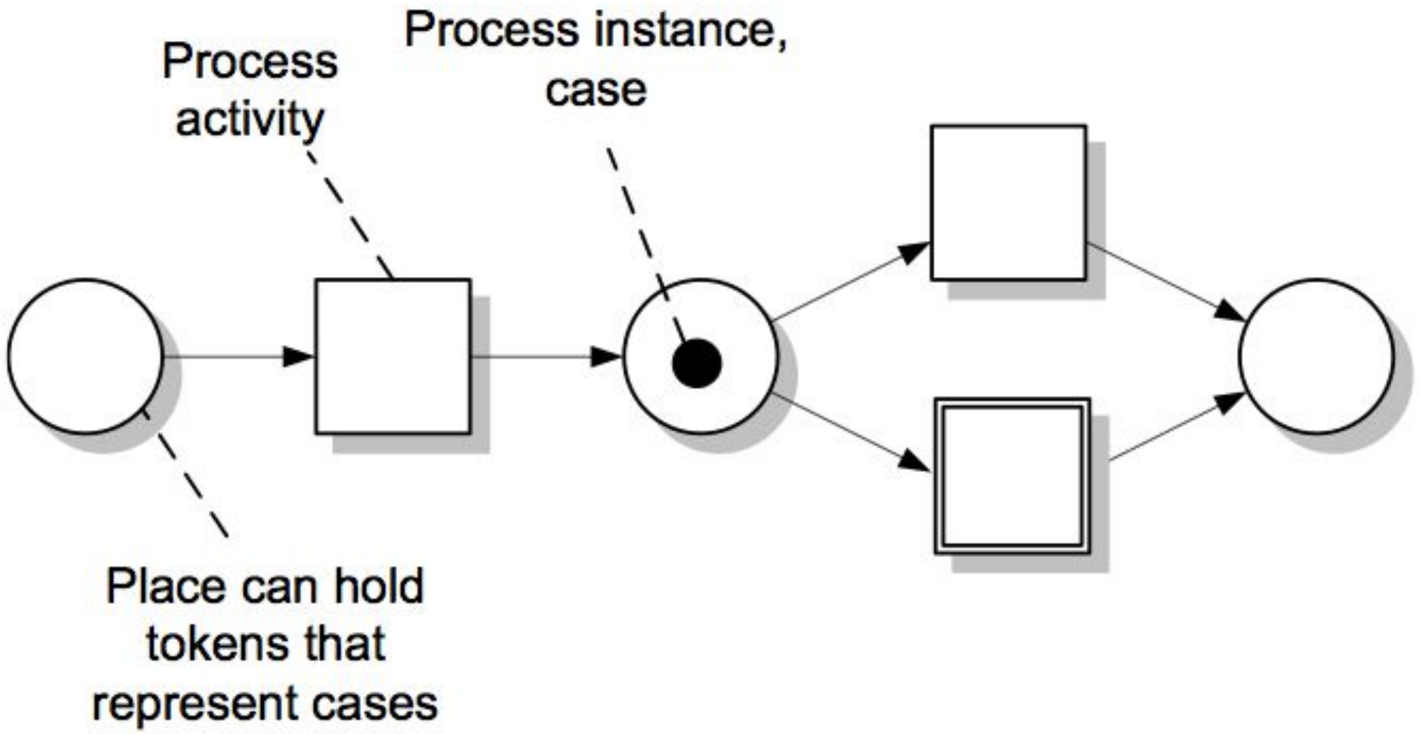


Fig. 4.40. Sample workflow net

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

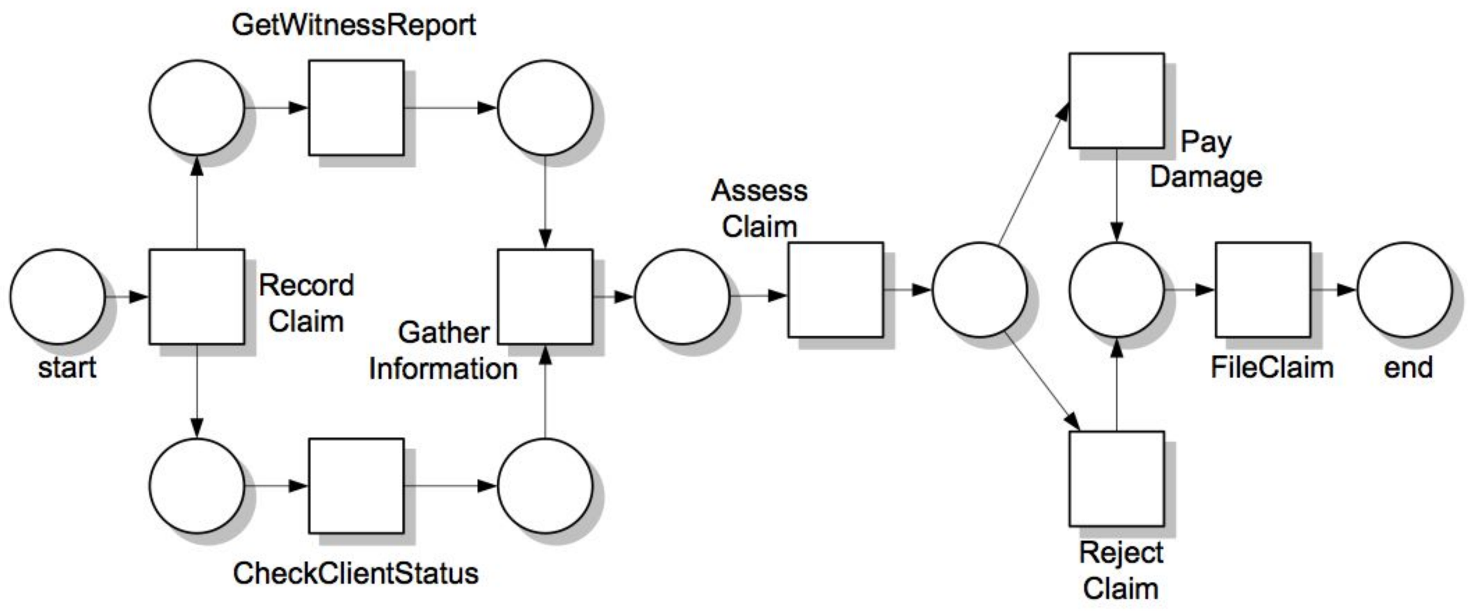


Fig. 4.41. Sample workflow net

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

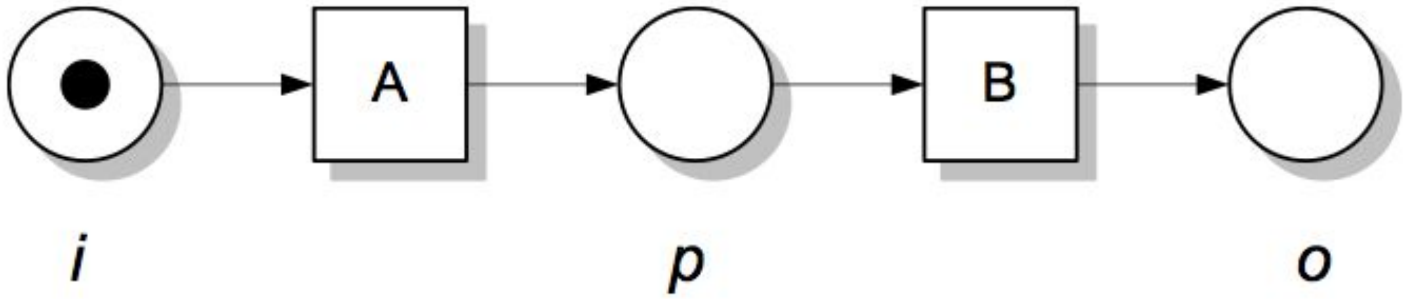


Fig. 4.42. Sequence pattern in workflow net

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

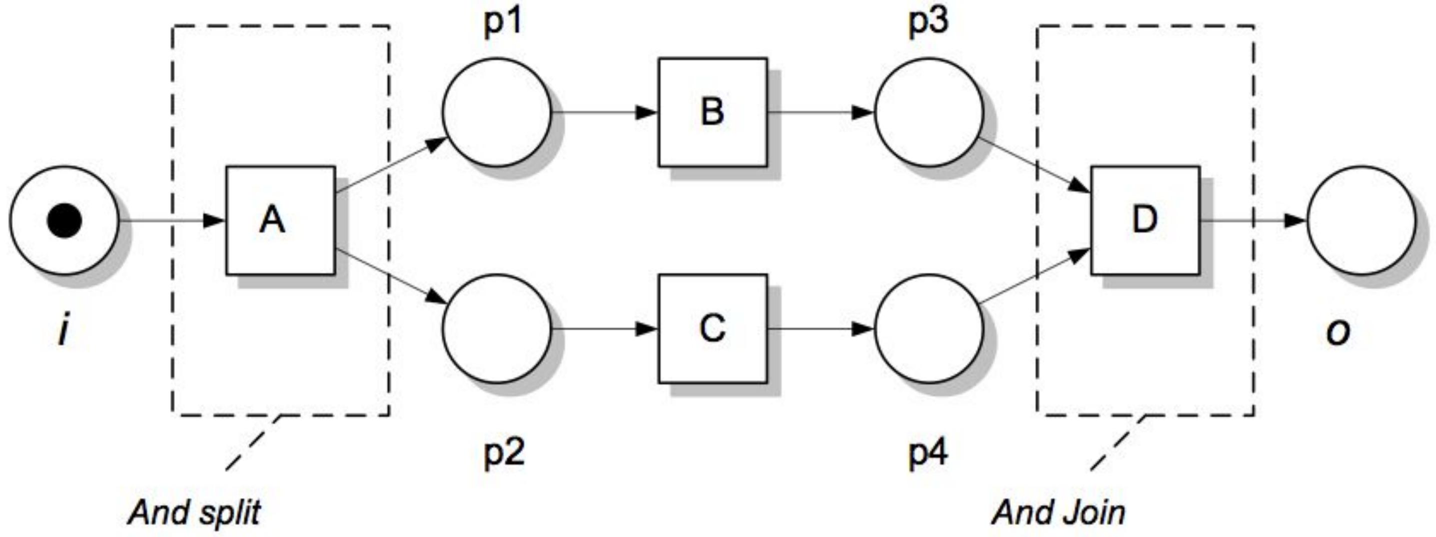


Fig. 4.43. *And split* and *and join* patterns in workflow nets

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

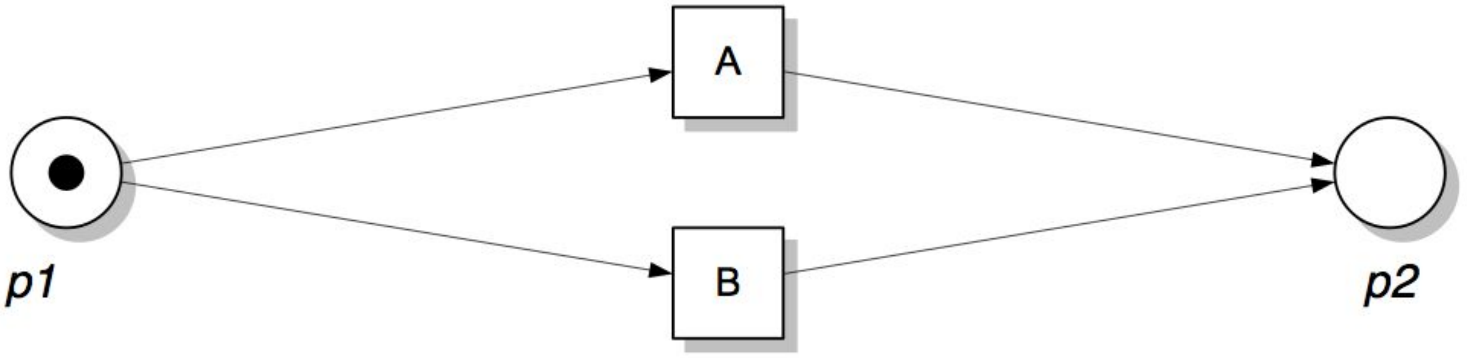


Fig. 4.44. *Implicit exclusive or split* also known as *deferred choice*

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

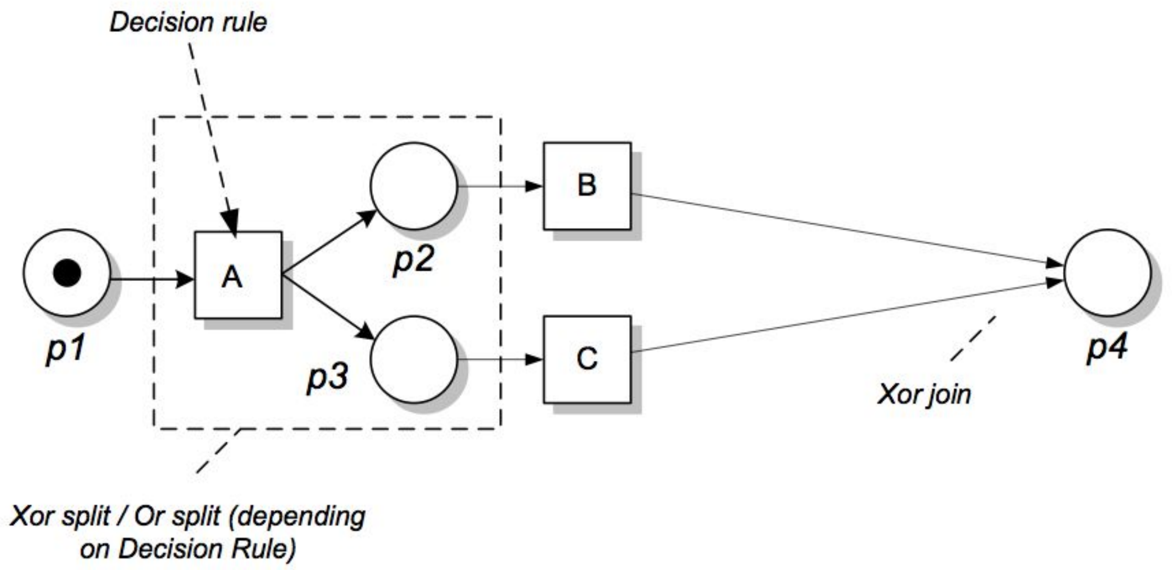


Fig. 4.45. Decision rule based split, can realize *or split*, *exclusive or split*, and *and split*

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

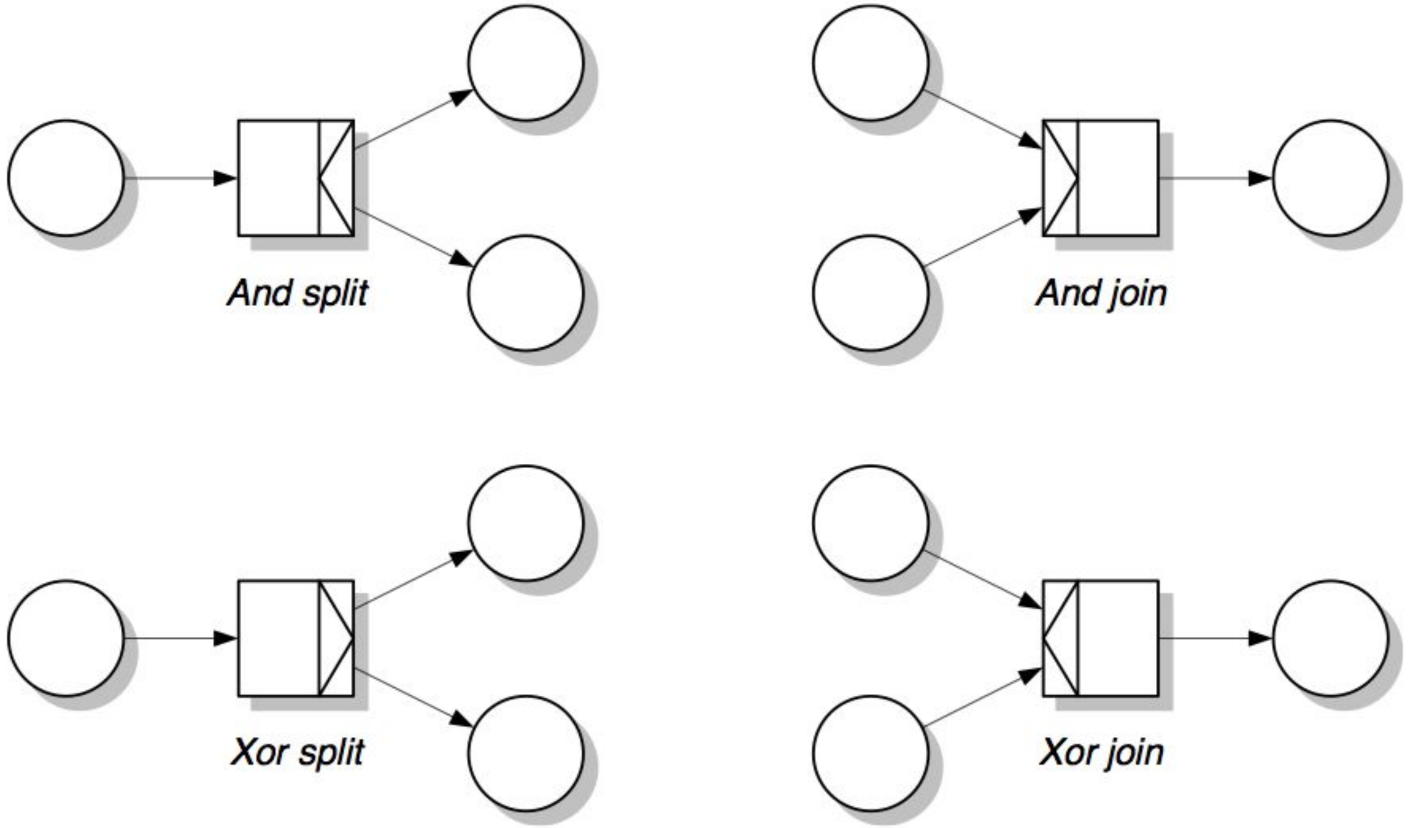
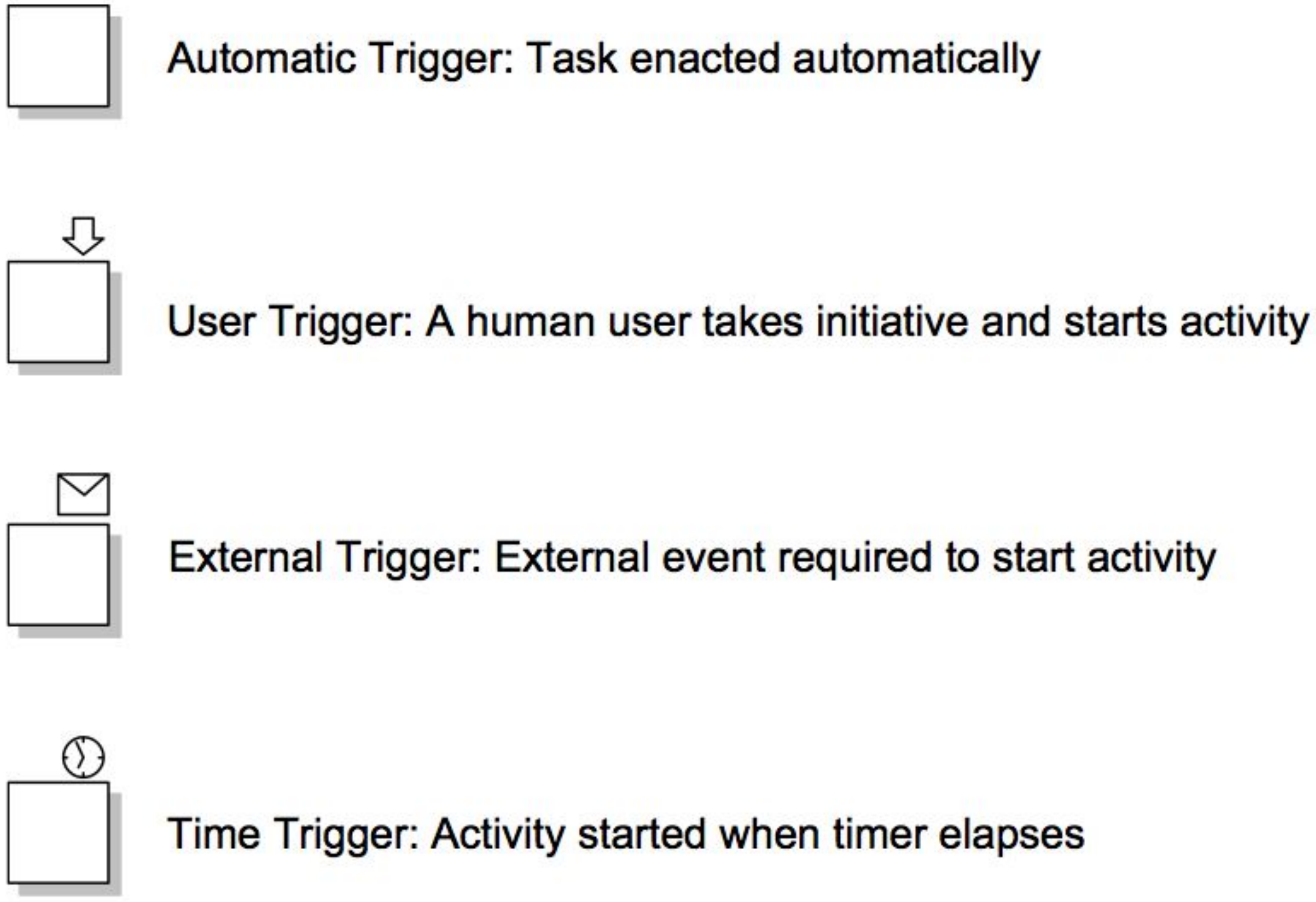


Fig. 4.46. Syntactic sugaring of transitions in workflow nets

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007



Automatic Trigger: Task enacted automatically

User Trigger: A human user takes initiative and starts activity

External Trigger: External event required to start activity

Time Trigger: Activity started when timer elapses

Fig. 4.47. Triggers in workflow nets

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

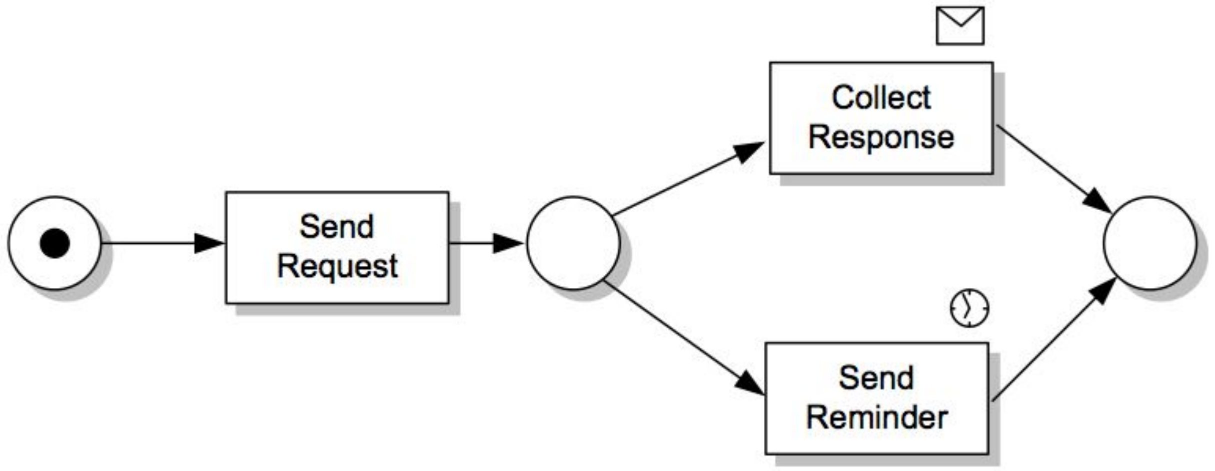
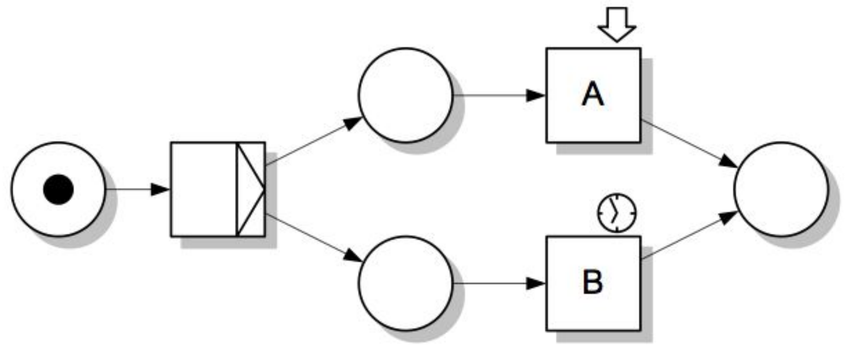


Fig. 4.48. Sample workflow net with external trigger and time trigger

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007


(a) *Explicit xor split* does not enable A and B concurrently

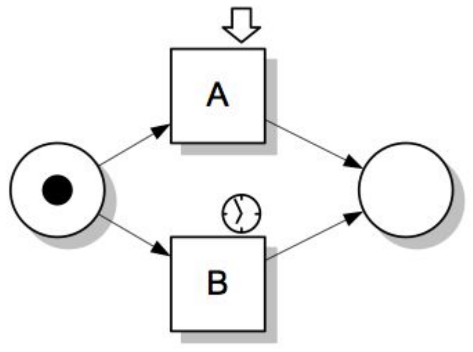
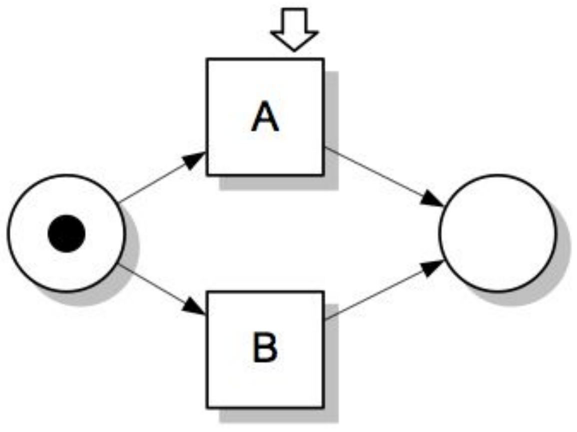
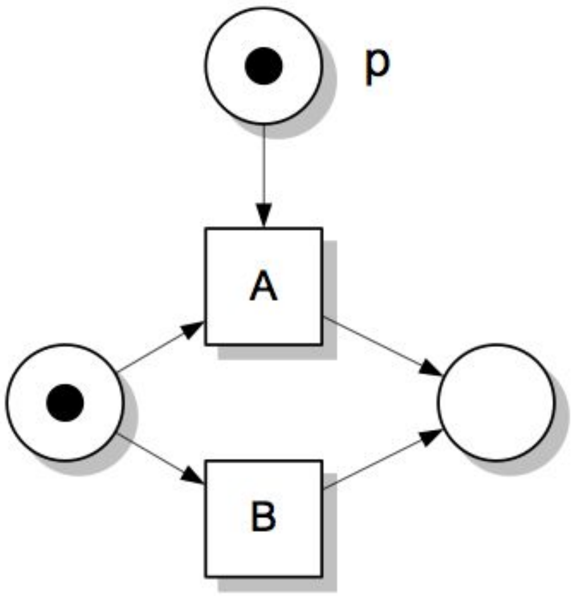

(b) *Implicit xor split* enables A and B concurrently

Fig. 4.49. Sample workflow nets illustrate the difference between *explicit xor split* (a) and *implicit xor split* (b)

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007



(a) Transition A started by user trigger



(b) Representation of user trigger by additional place and additional arc

Fig. 4.50. Representation of triggers

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

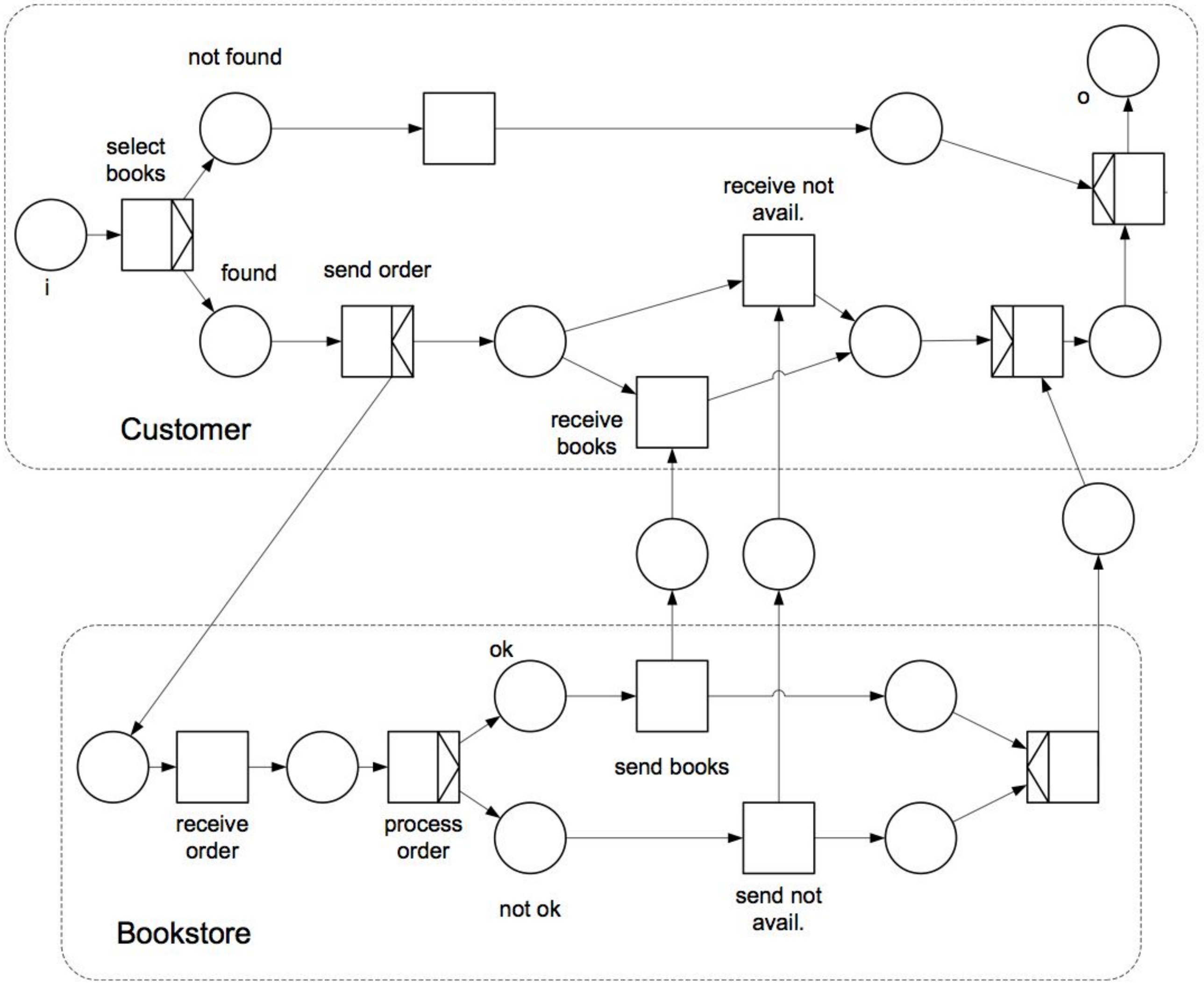


Fig. 4.51. Sample workflow net involving multiple parties

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

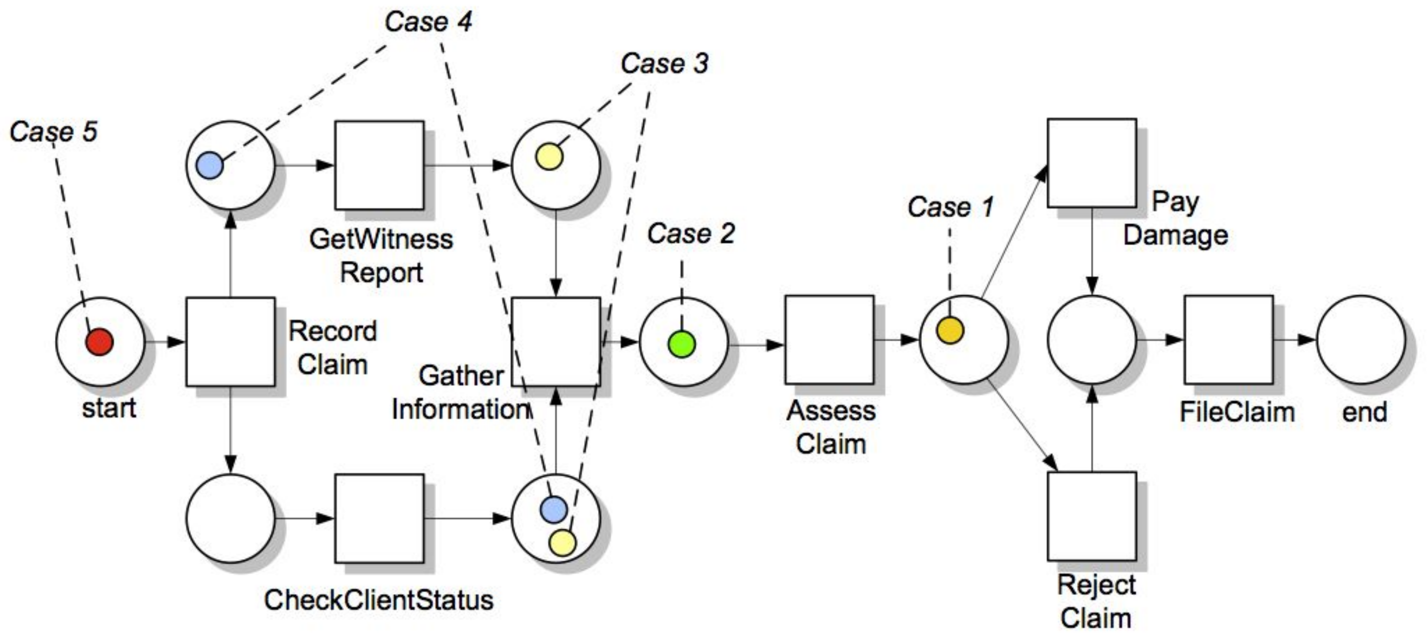


Fig. 4.52. Sample workflow net with coloured tokens representing process instances

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

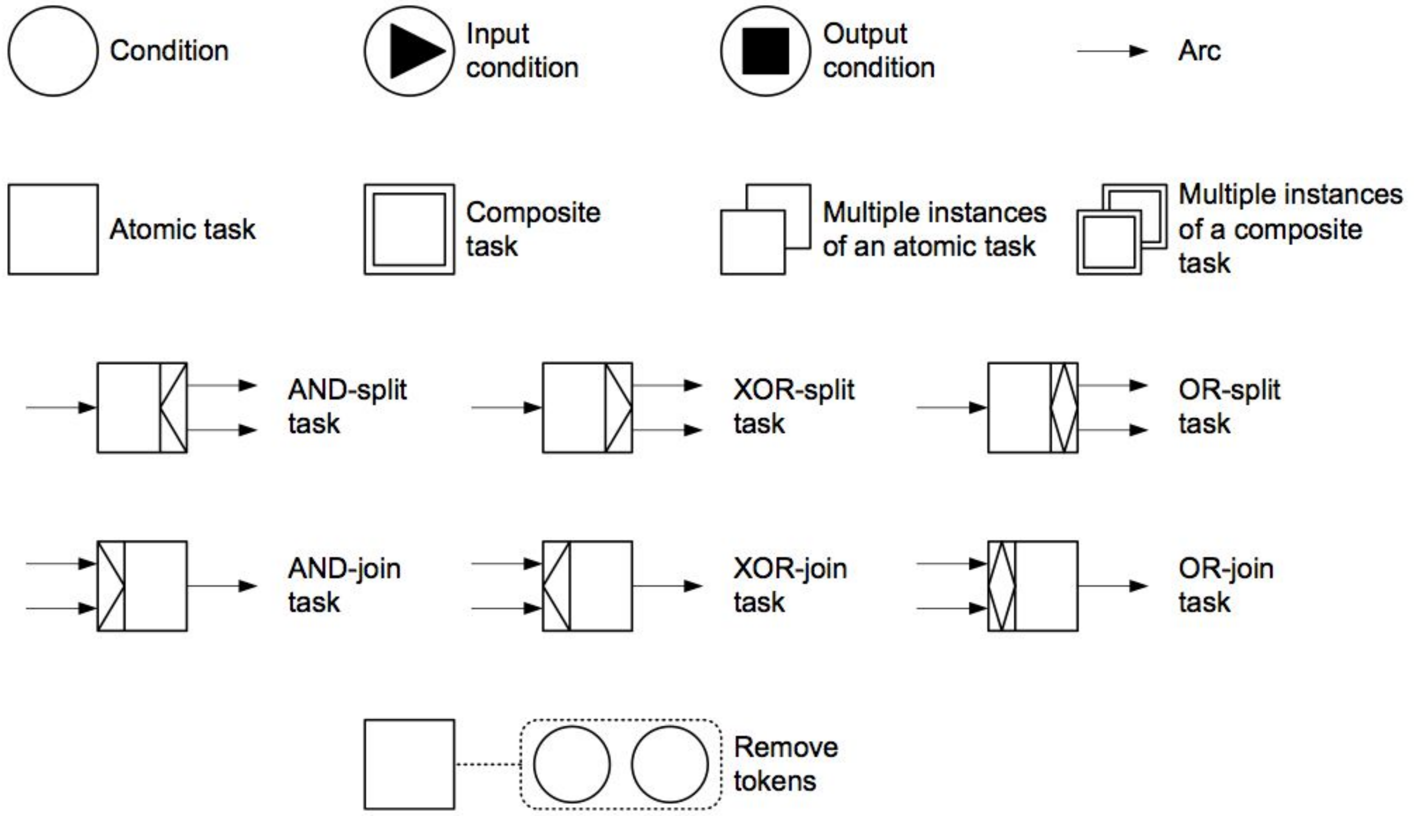


Fig. 4.53. Notational elements of YAWL, van der Aalst and ter Hofstede (2005)

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

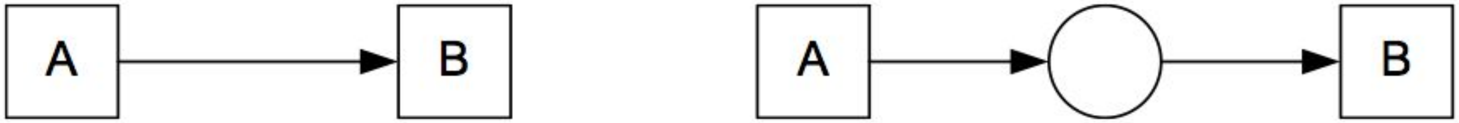


Fig. 4.54. Representations of sequence pattern in YAWL

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

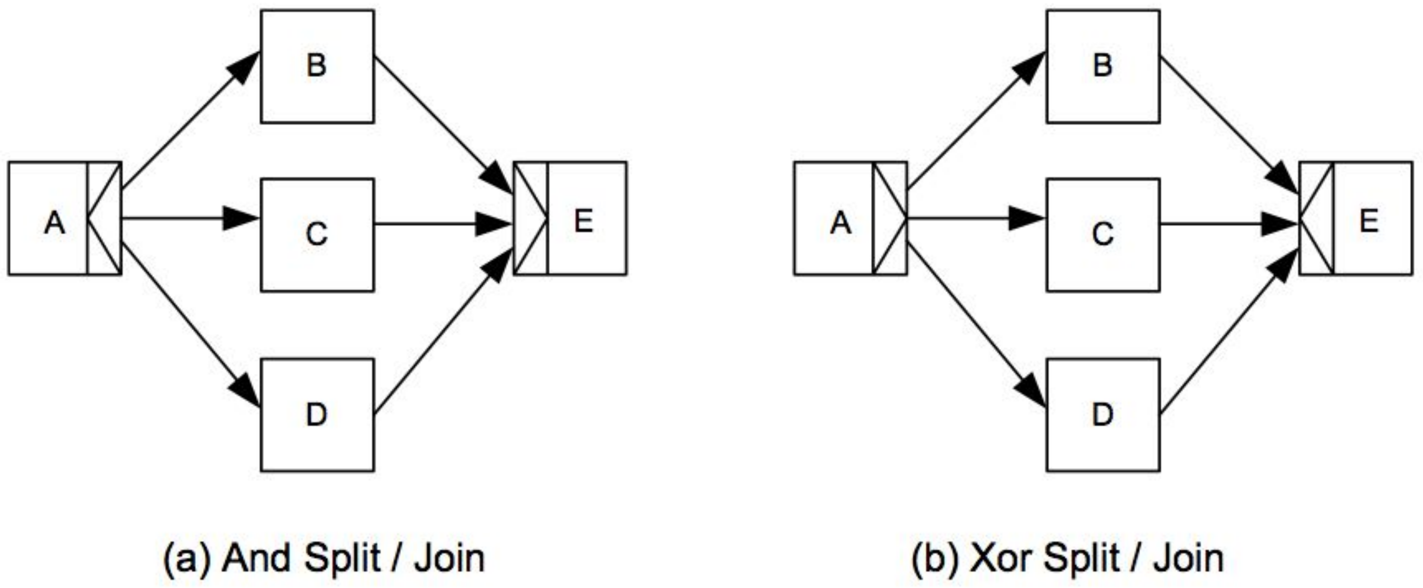


Fig. 4.55. *And split/join* and *xor split/join* patterns

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

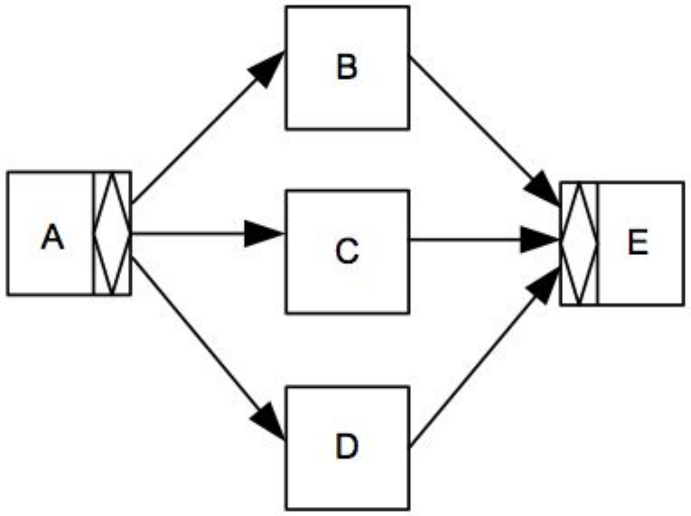


Fig. 4.56. *Inclusive or split* and *inclusive or join* patterns

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

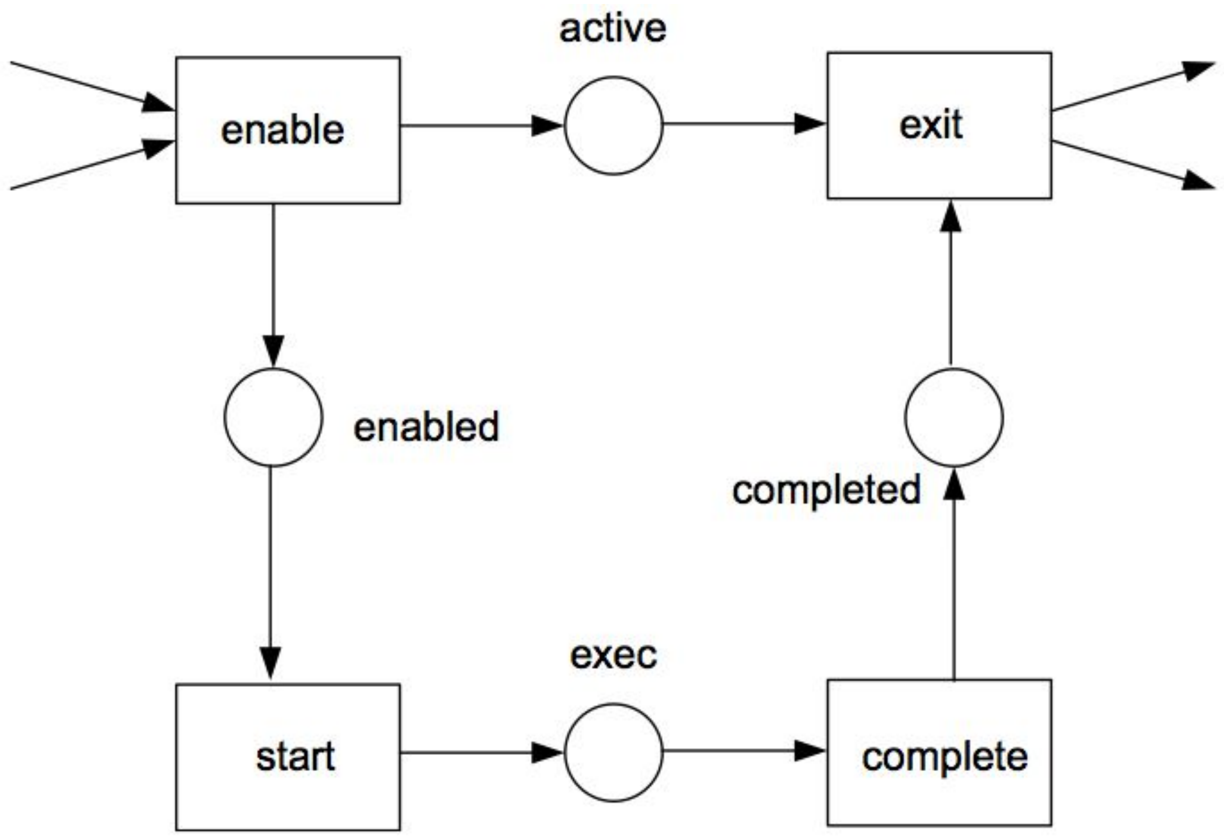


Fig. 4.57. State transition diagrams for single instance tasks

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

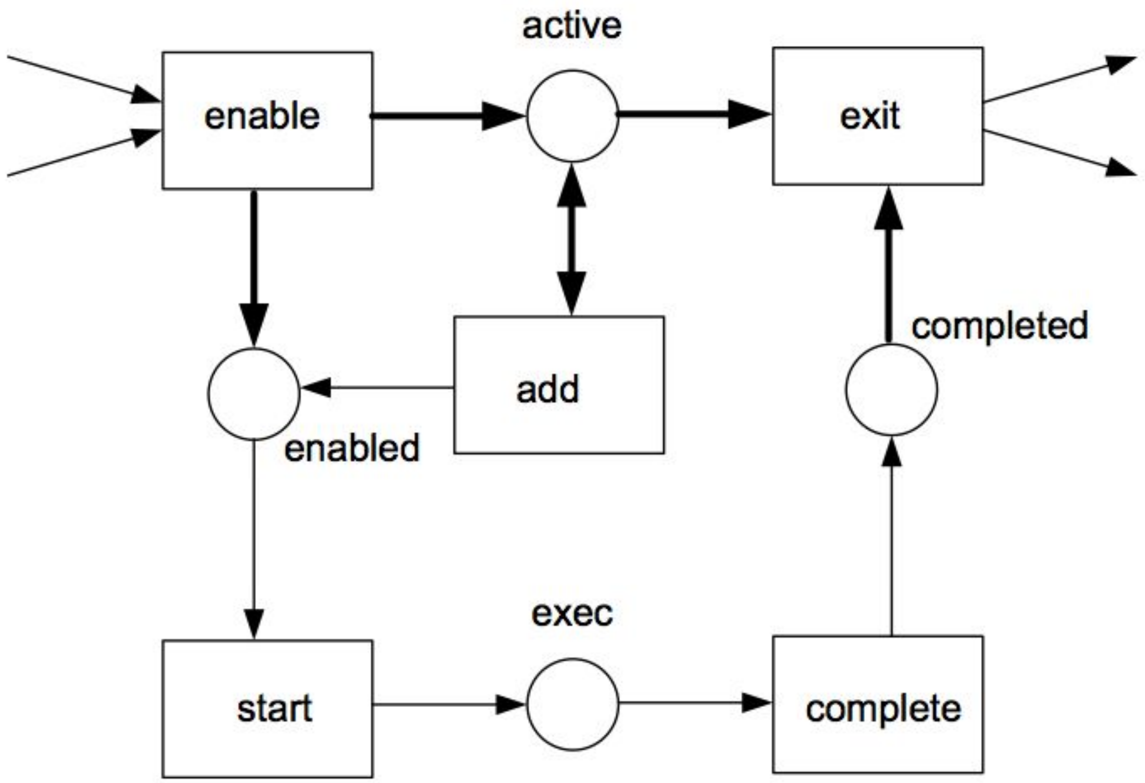


Fig. 4.58. State transition diagram for multiple instances task

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

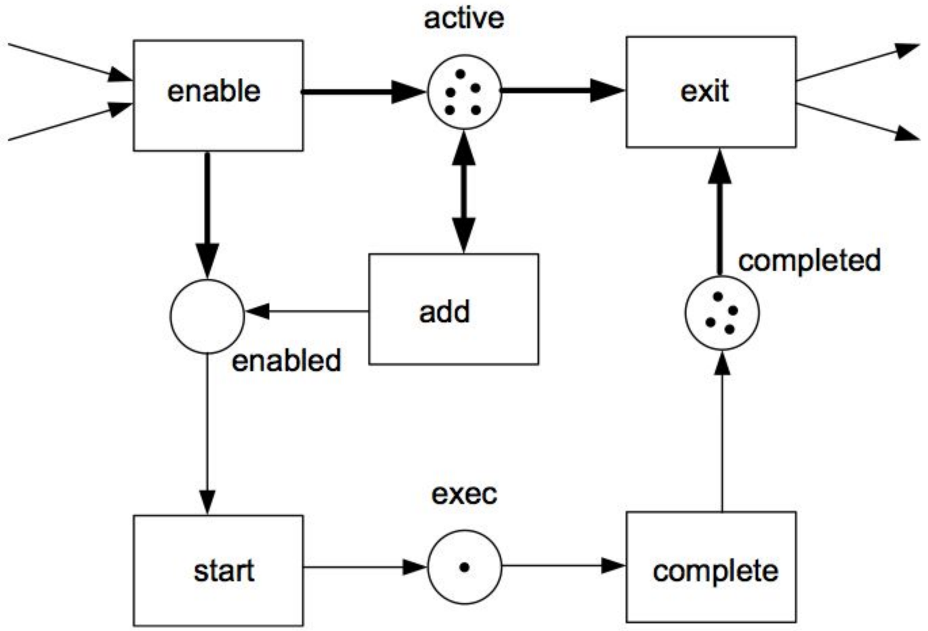


Fig. 4.59. State transition diagram for multiple instances task with five instances, four of which have completed

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

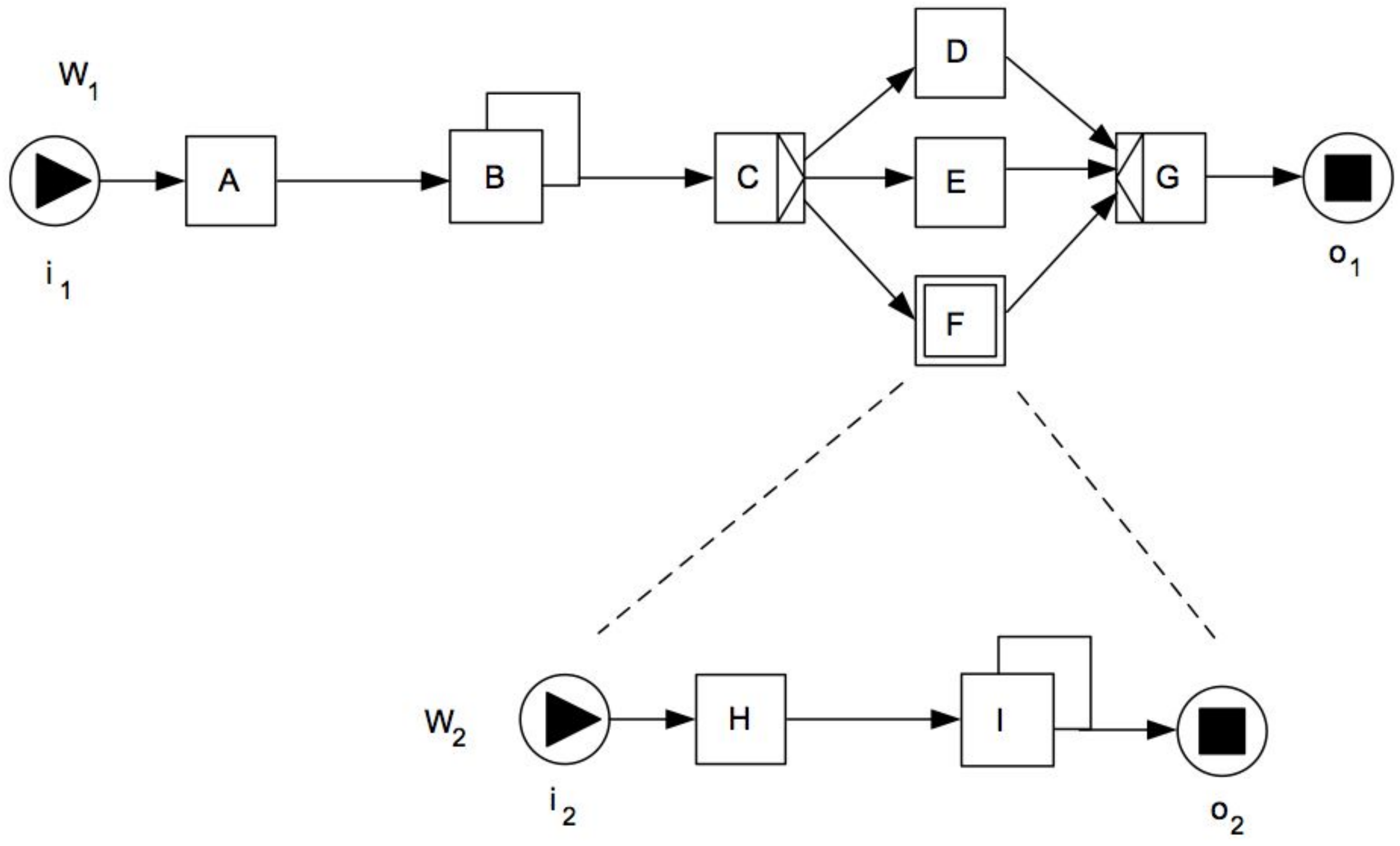


Fig. 4.60. YAWL specification

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

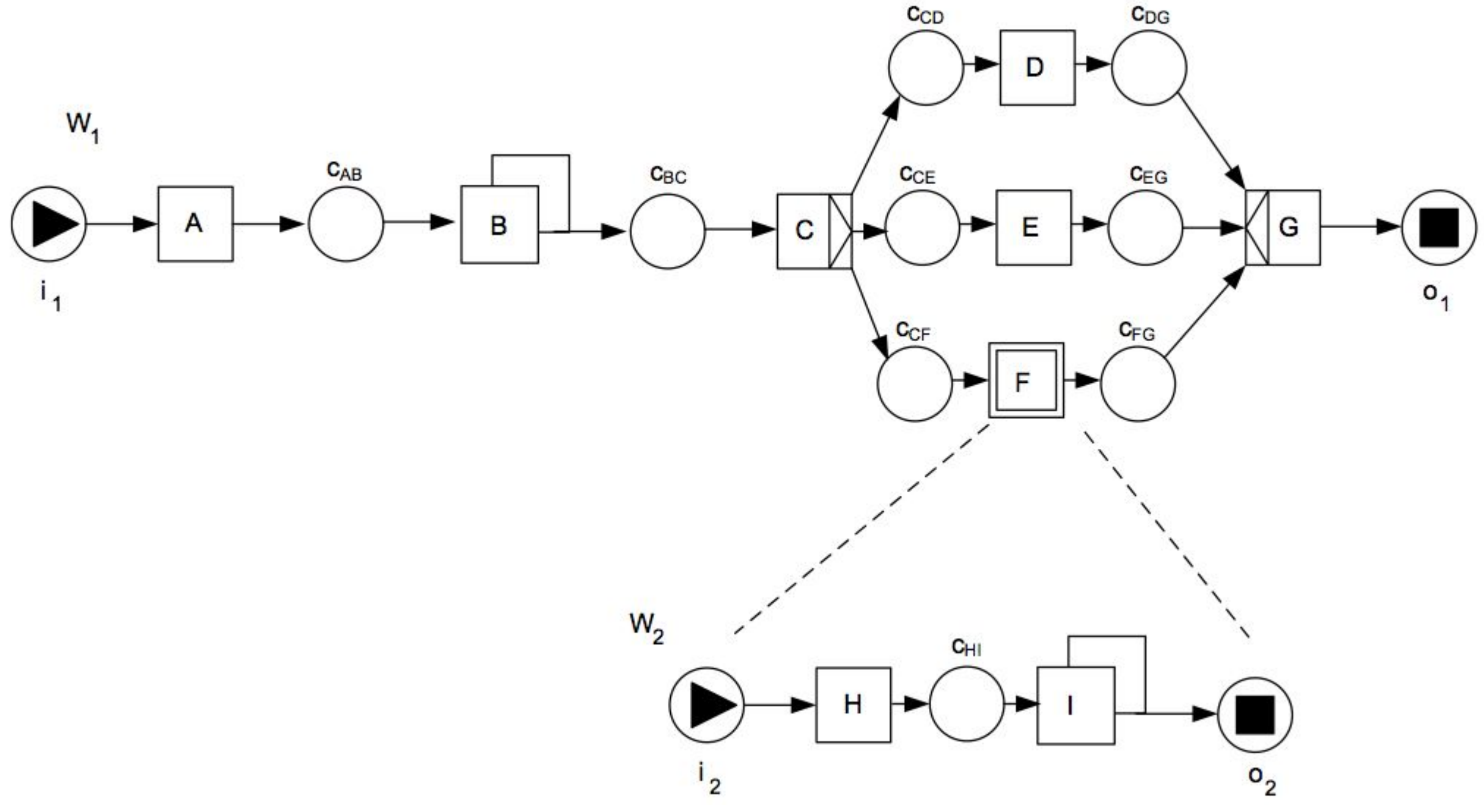


Fig. 4.61. YAWL specification with extended condition set C^{ext}

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

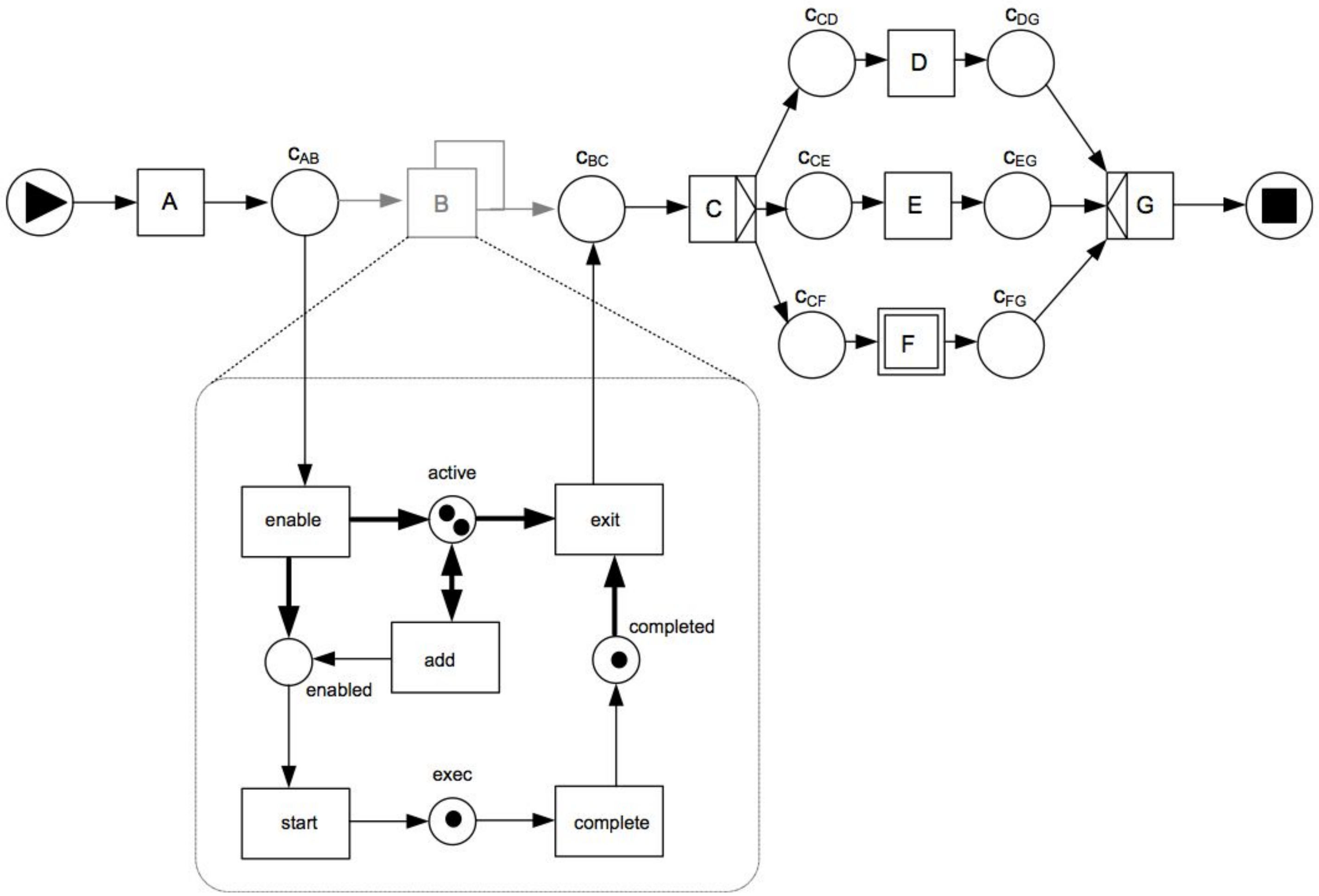


Fig. 4.62. Workflow state with two task instances of *B* active, one is executing and one has completed

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

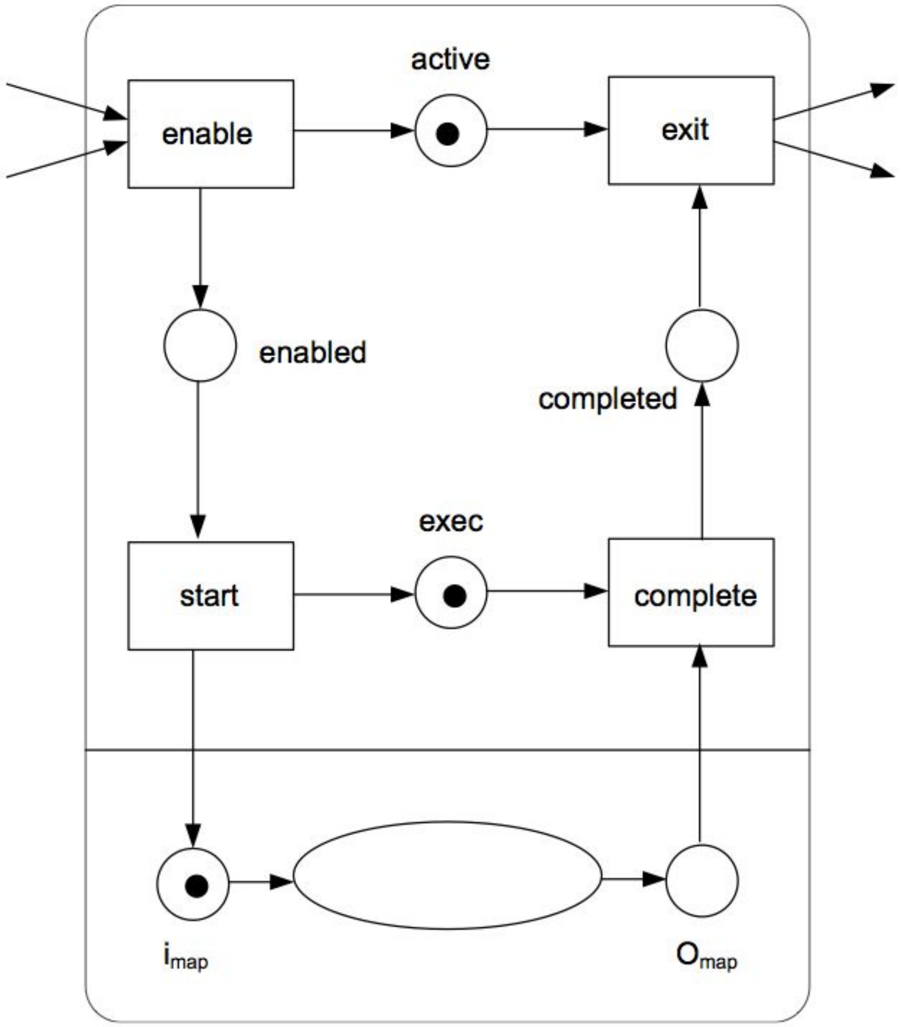


Fig. 4.63. State transition system for composite tasks

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

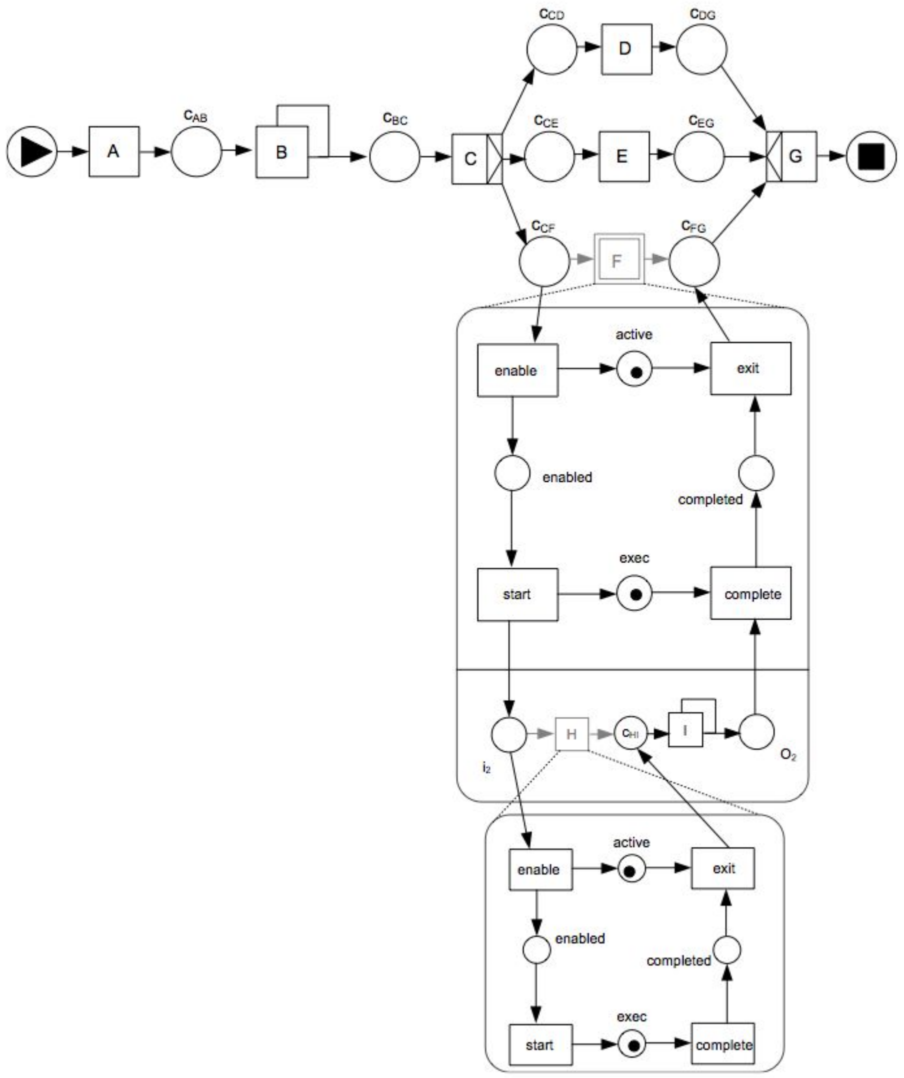


Fig. 4.64. Workflow state, where composite task *F* is currently active

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

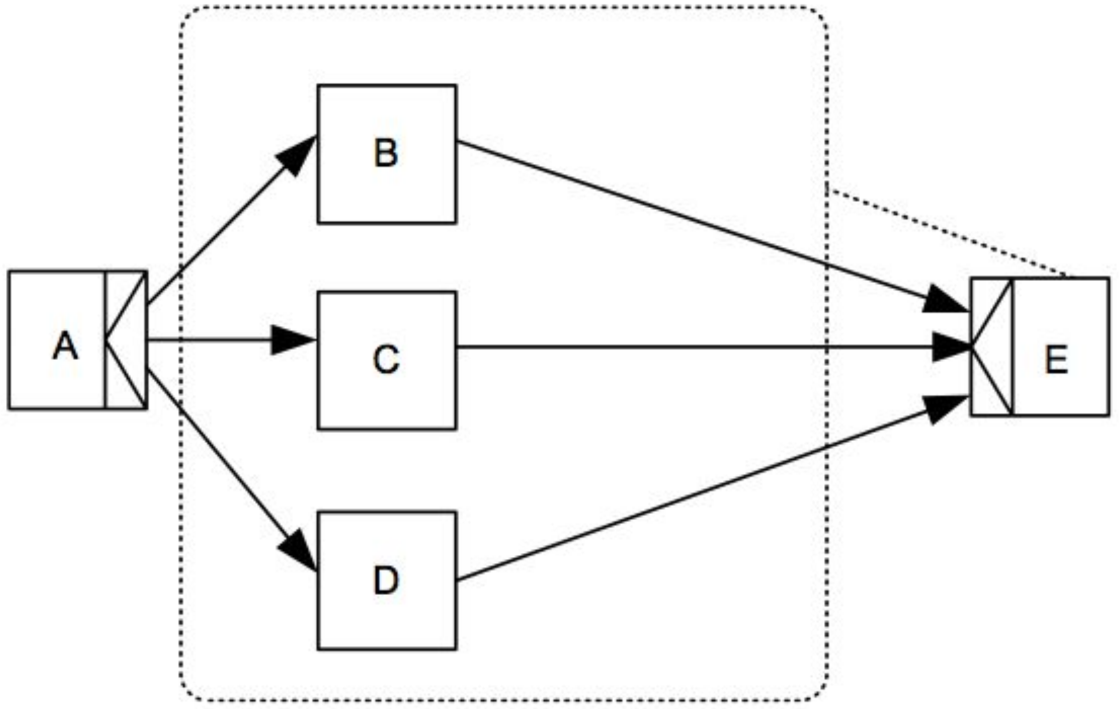


Fig. 4.65. Discriminator in YAWL using cancellation

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

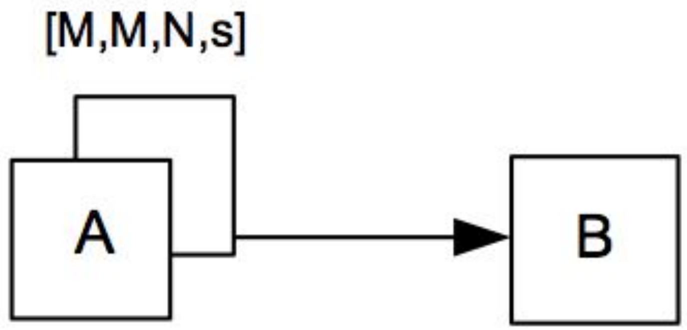


Fig. 4.66. *N-out-of-M join* using multiple instances

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

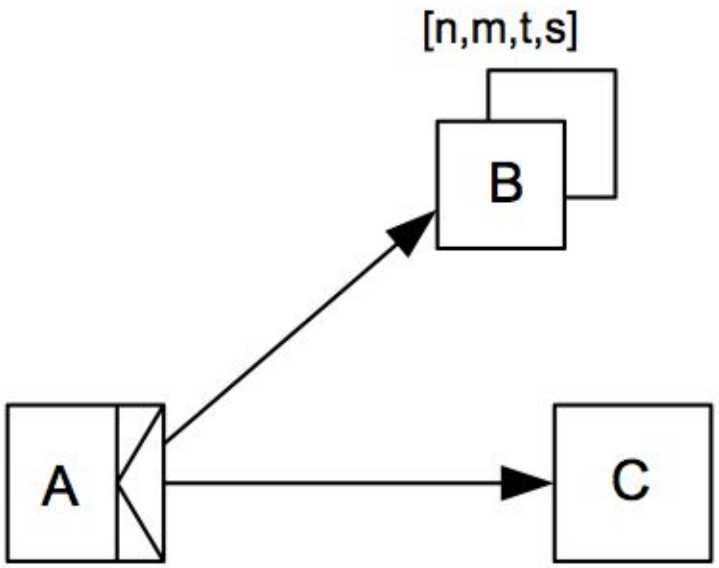


Fig. 4.67. Multiple instances without synchronization

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

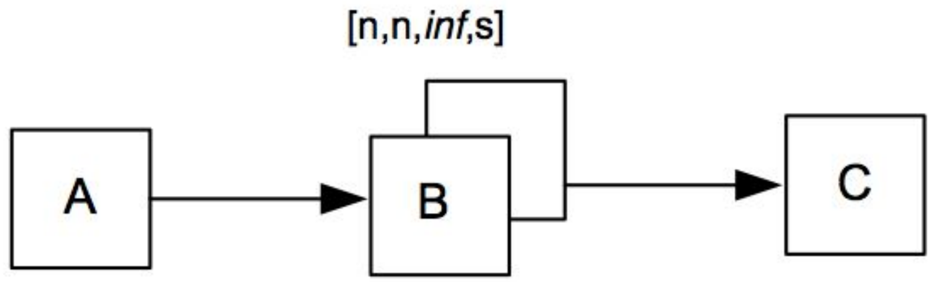
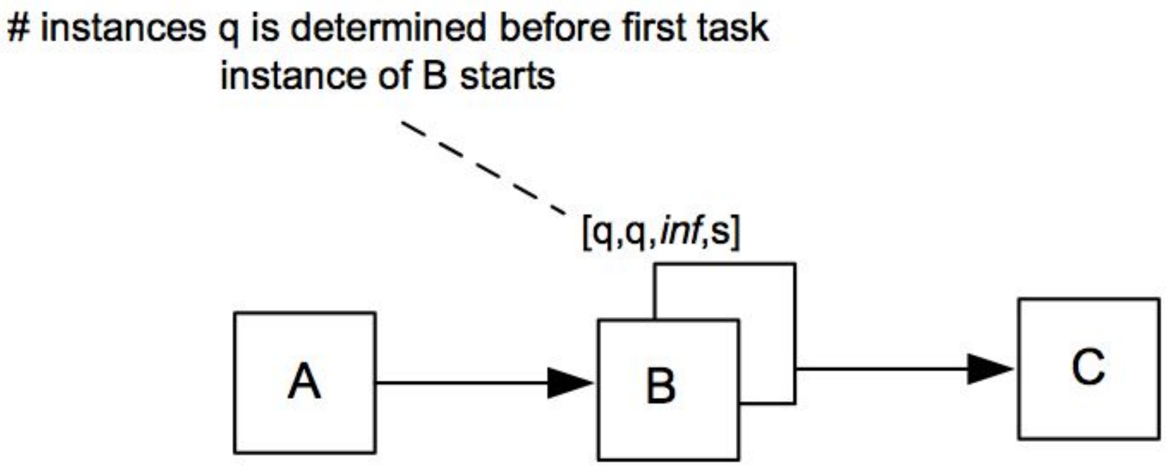


Fig. 4.68. Multiple instances with a priori design time knowledge

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007



# instances q is determined before first task instance of B starts
[q,q,*inf*,s]
A
B
C

Fig. 4.69. Multiple instances with a priori run time knowledge

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

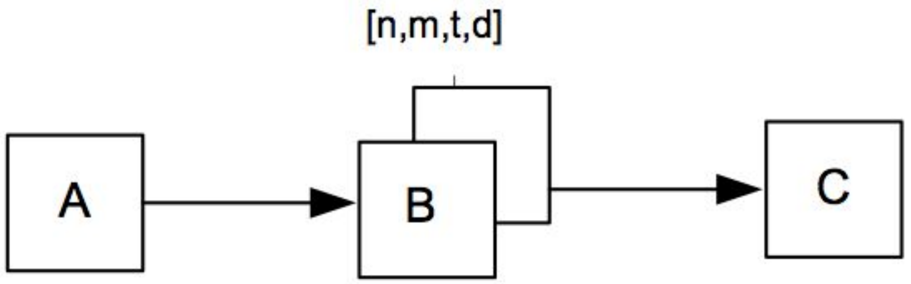


Fig. 4.70. Multiple instances without a priori run time knowledge

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

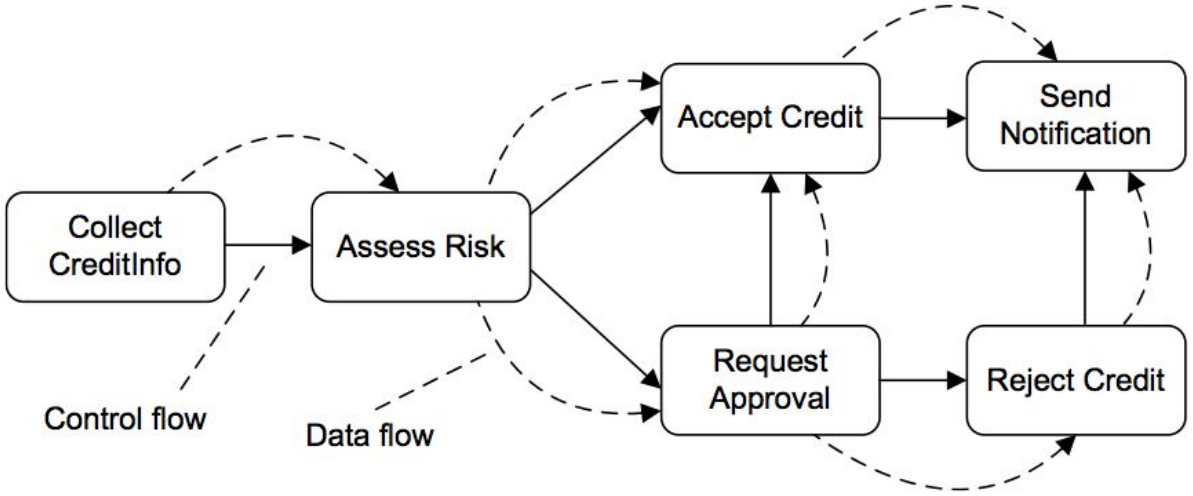


Fig. 4.71. Credit request process model, expressed in graph-based workflow language

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

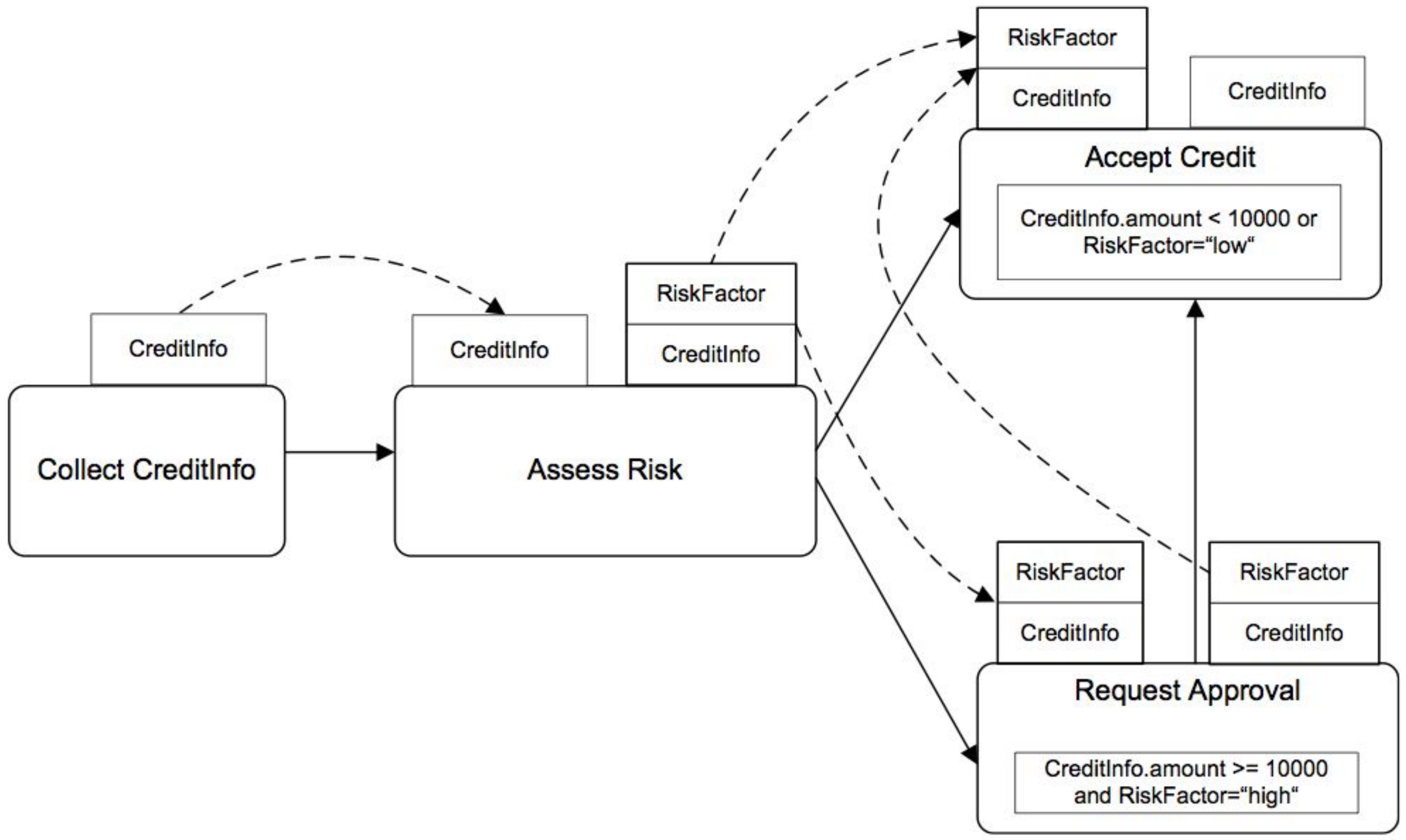


Fig. 4.72. Detailed view on parameters and conditions (partial process)

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

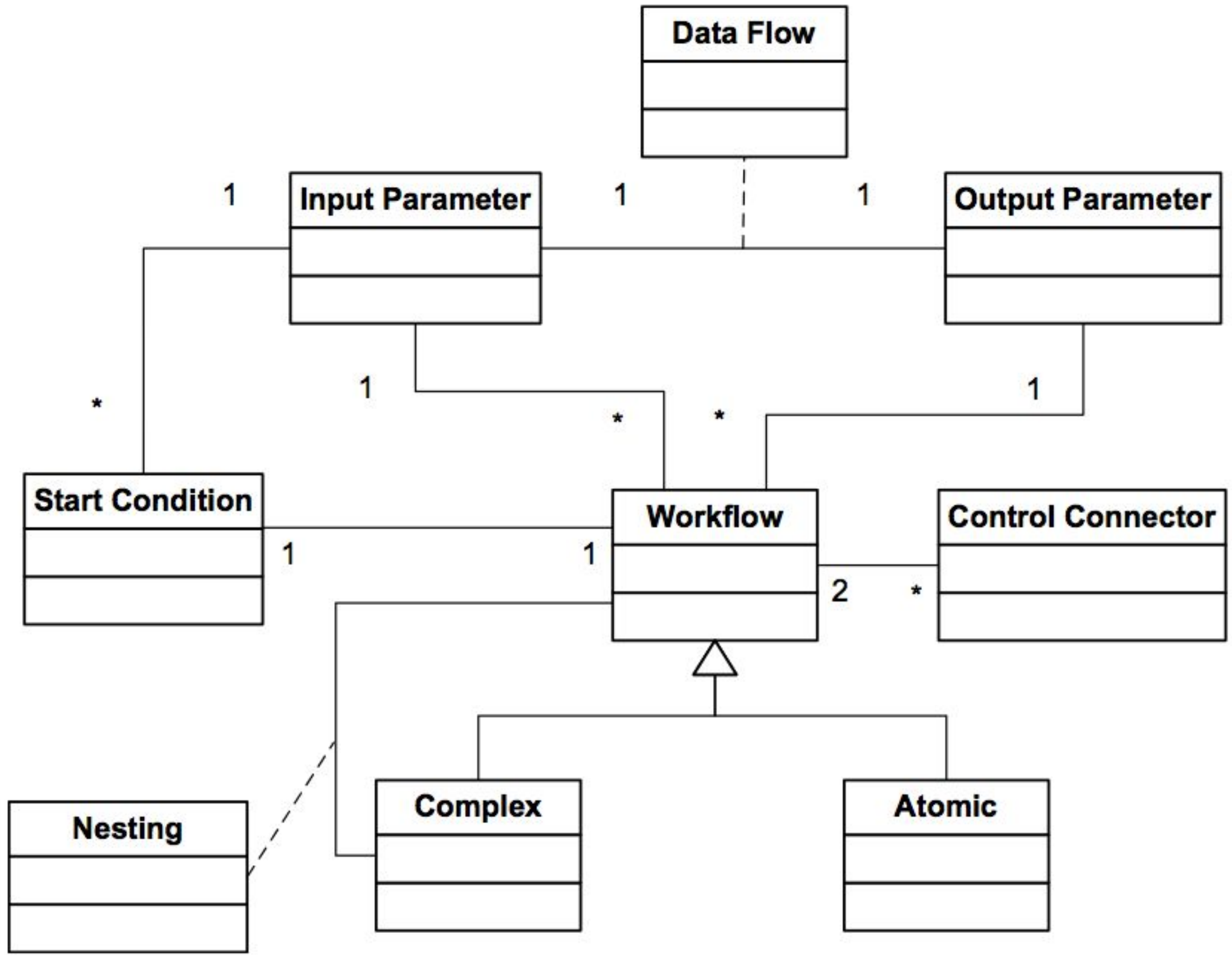


Fig. 4.73. Metamodel of a graph-based workflow language

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

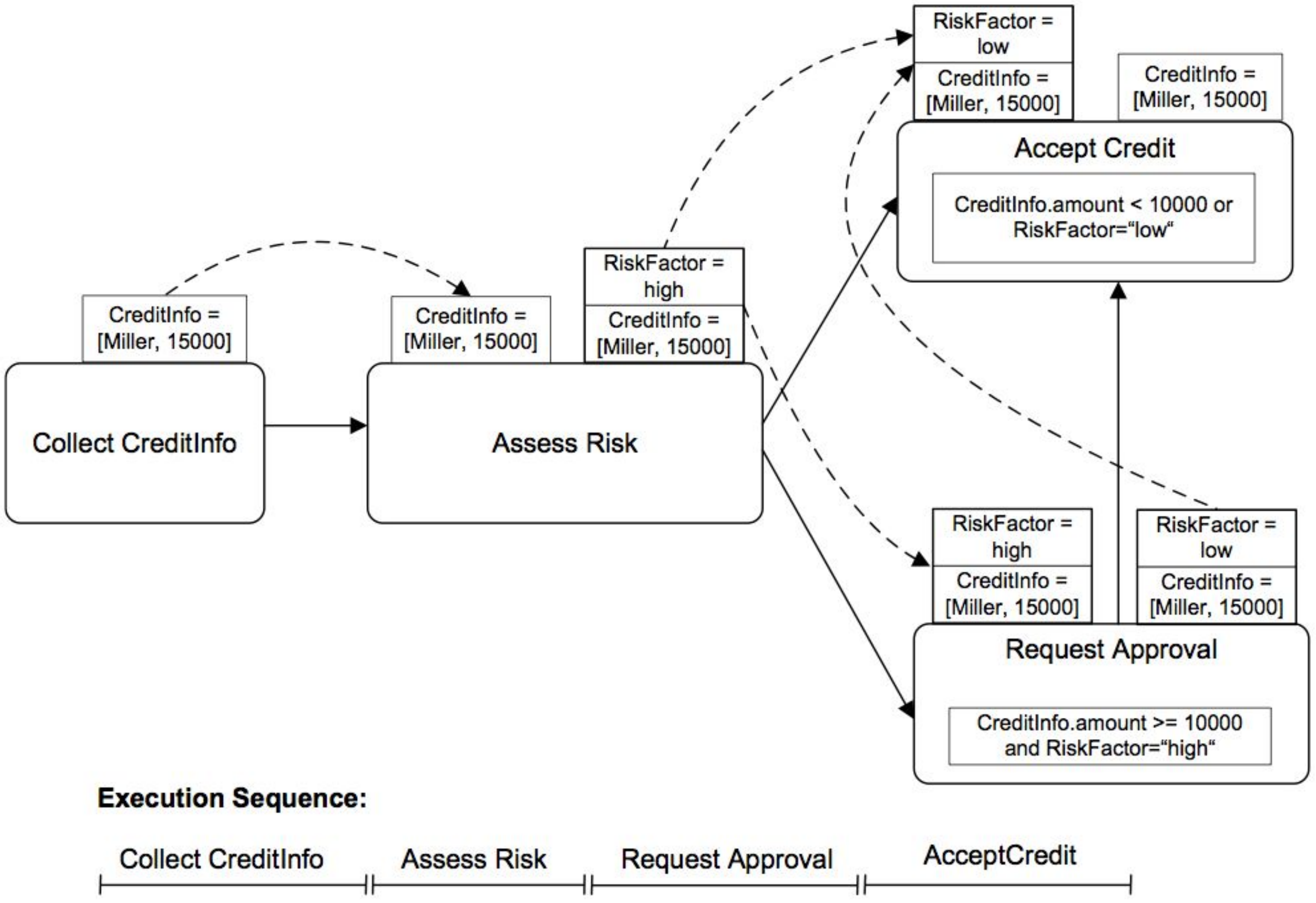

RiskFactor = low
CreditInfo = [Miller, 15000]
CreditInfo = [Miller, 15000]
Accept Credit
CreditInfo.amount < 10000 or RiskFactor="low"
RiskFactor = high
CreditInfo = [Miller, 15000]
CreditInfo = [Miller, 15000]
CreditInfo = [Miller, 15000]
Collect CreditInfo
Assess Risk
RiskFactor = high
CreditInfo = [Miller, 15000]
RiskFactor = low
CreditInfo = [Miller, 15000]
Request Approval
CreditInfo.amount >= 10000 and RiskFactor="high"
Execution Sequence:
Collect CreditInfo
Assess Risk
Request Approval
AcceptCredit

Fig. 4.74. Process instance based on process model shown in Figure 4.72

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

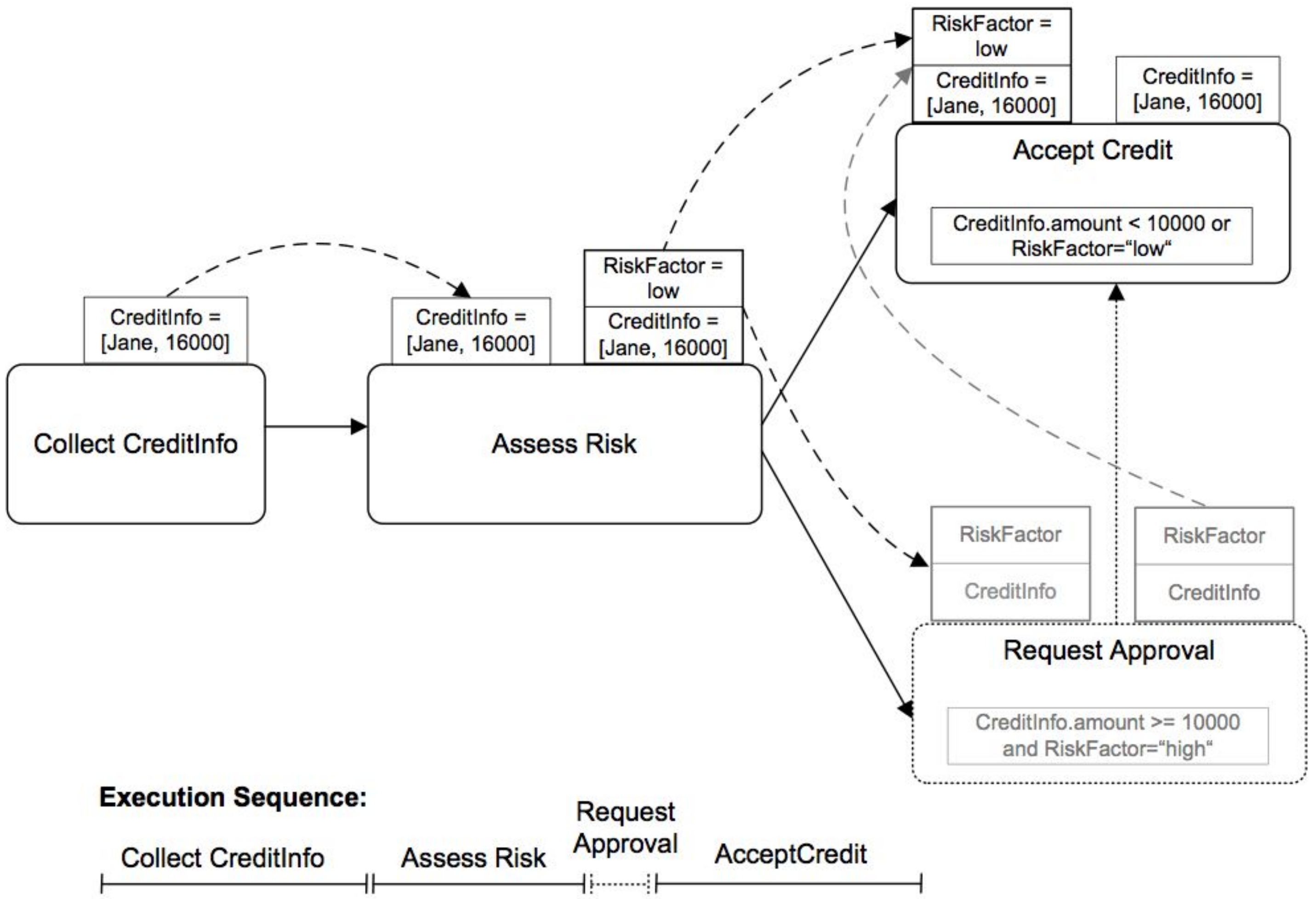

RiskFactor = low
CreditInfo = [Jane, 16000]
CreditInfo = [Jane, 16000]
Accept Credit
CreditInfo.amount < 10000 or RiskFactor="low"
RiskFactor = low
CreditInfo = [Jane, 16000]
CreditInfo = [Jane, 16000]
CreditInfo = [Jane, 16000]
Collect CreditInfo
Assess Risk
RiskFactor
CreditInfo
RiskFactor
CreditInfo
Request Approval
CreditInfo.amount >= 10000 and RiskFactor="high"
Execution Sequence:
Collect CreditInfo
Assess Risk
Request Approval
AcceptCredit

Fig. 4.75. Process instance where request approval activity is not required

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

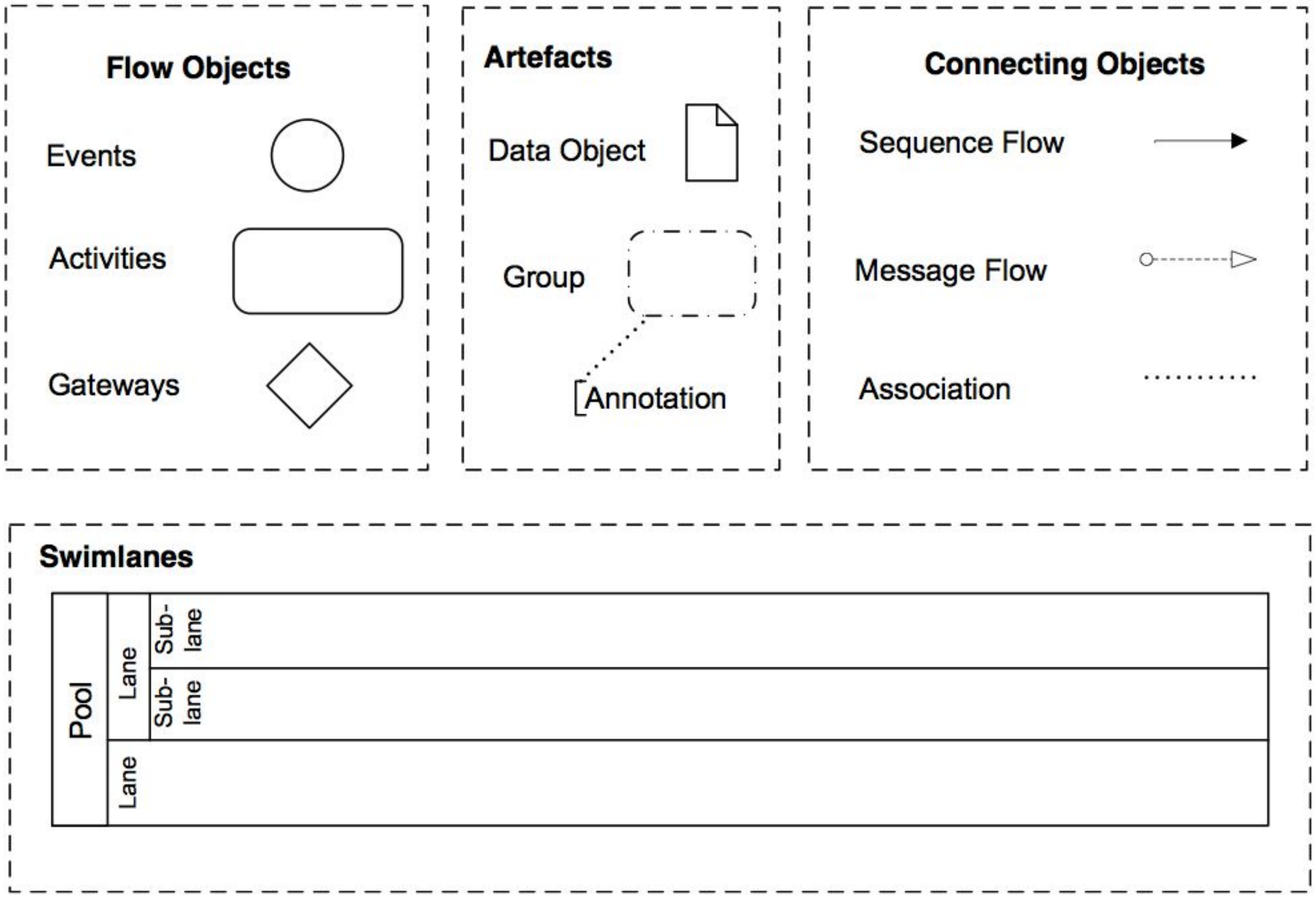


Fig. 4.76. BPMN: categories of elements

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

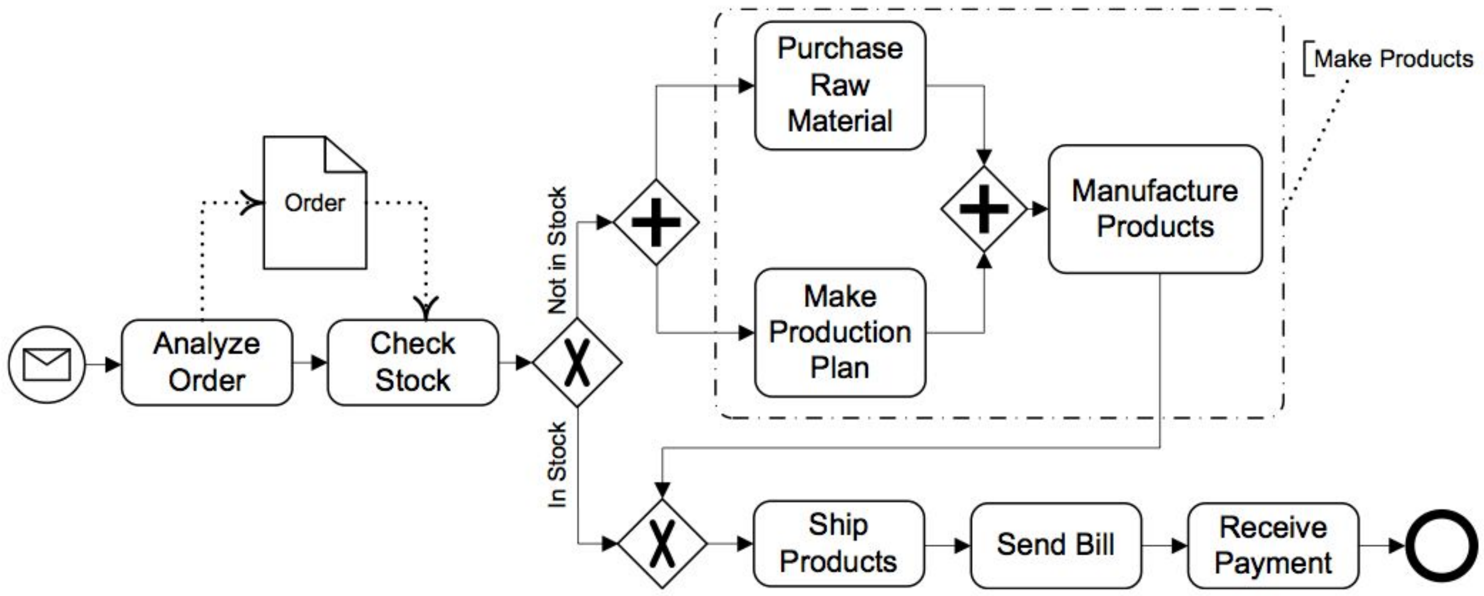


Fig. 4.77. Business process diagram expressed in BPMN

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

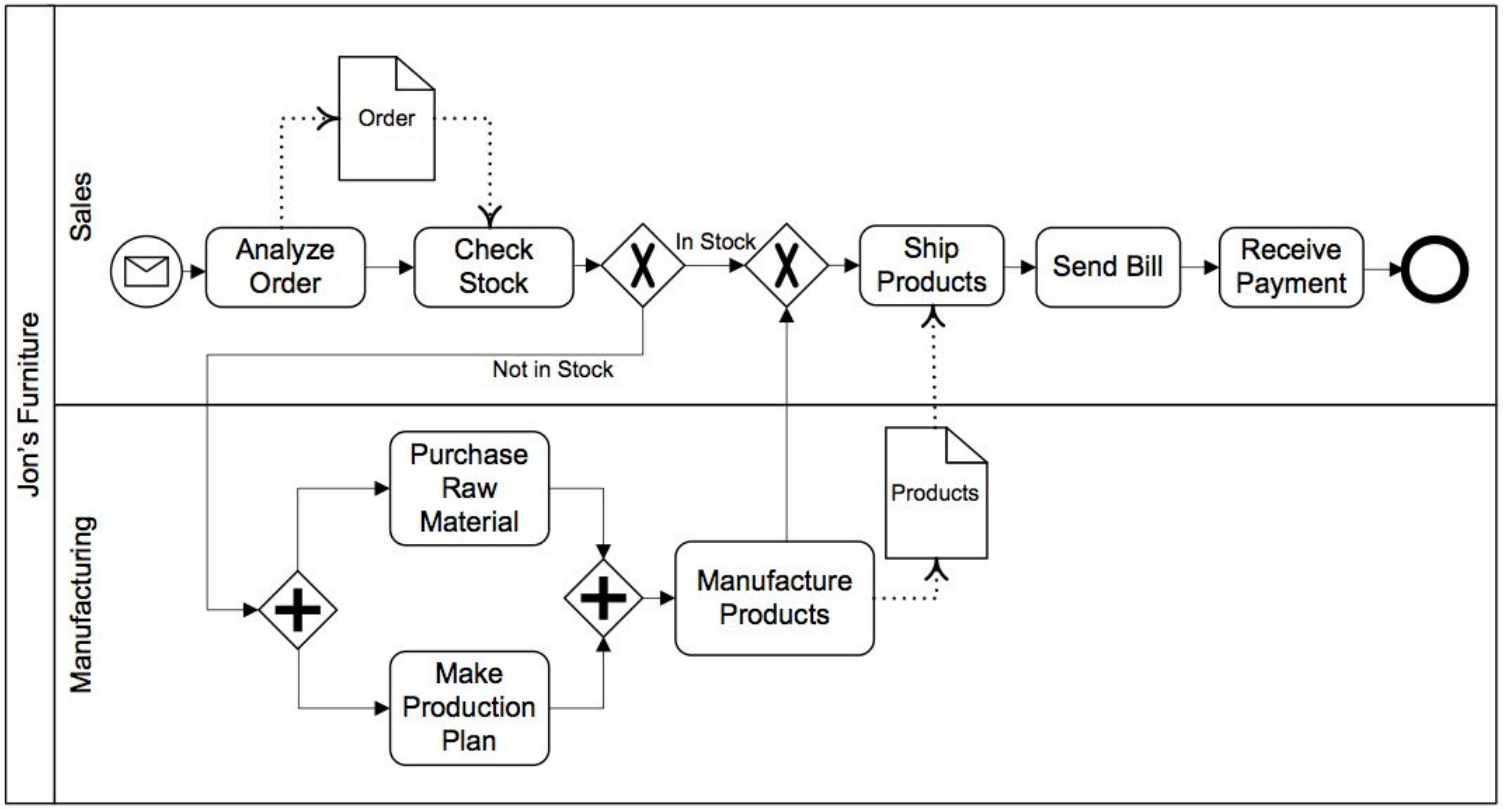


Fig. 4.78. Business process diagram with role information

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

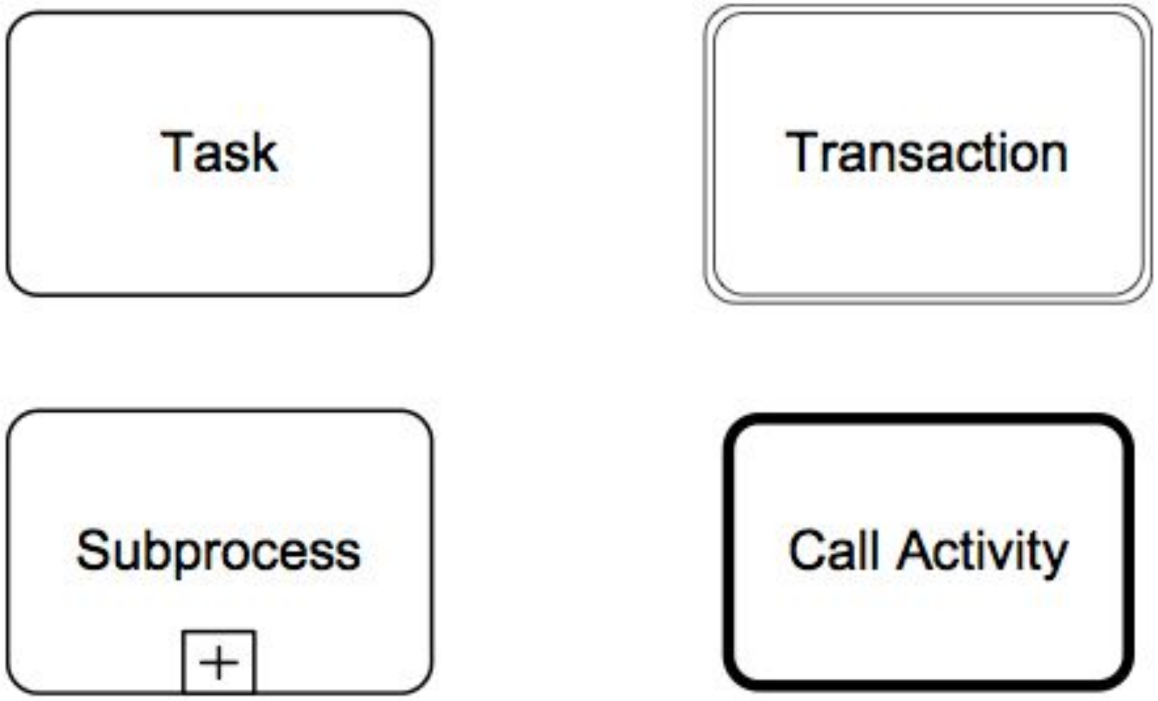


Fig. 4.79. Activity types in the BPMN

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

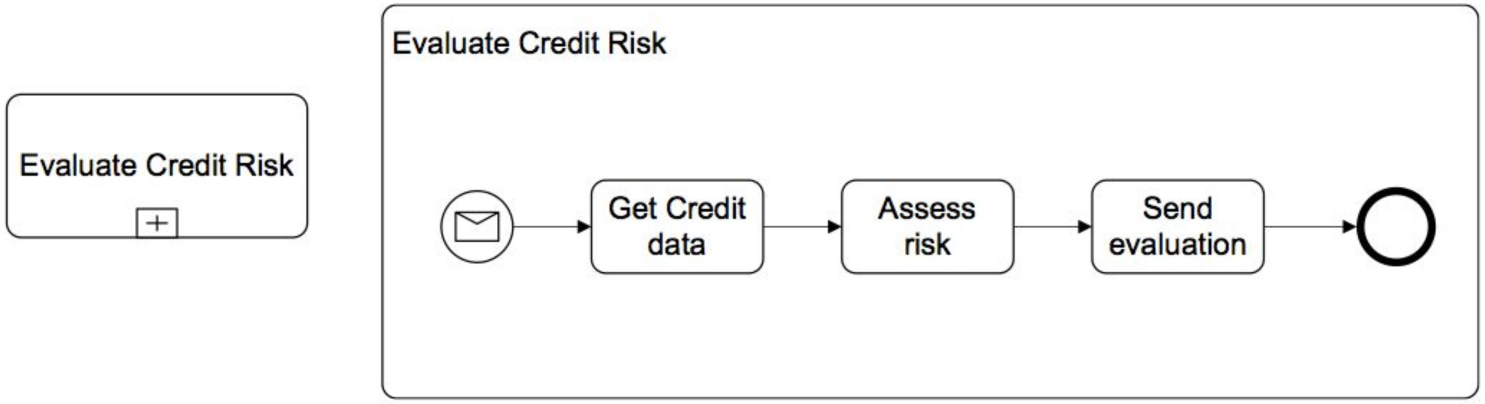


Fig. 4.80. Collapsed and expanded subprocess

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

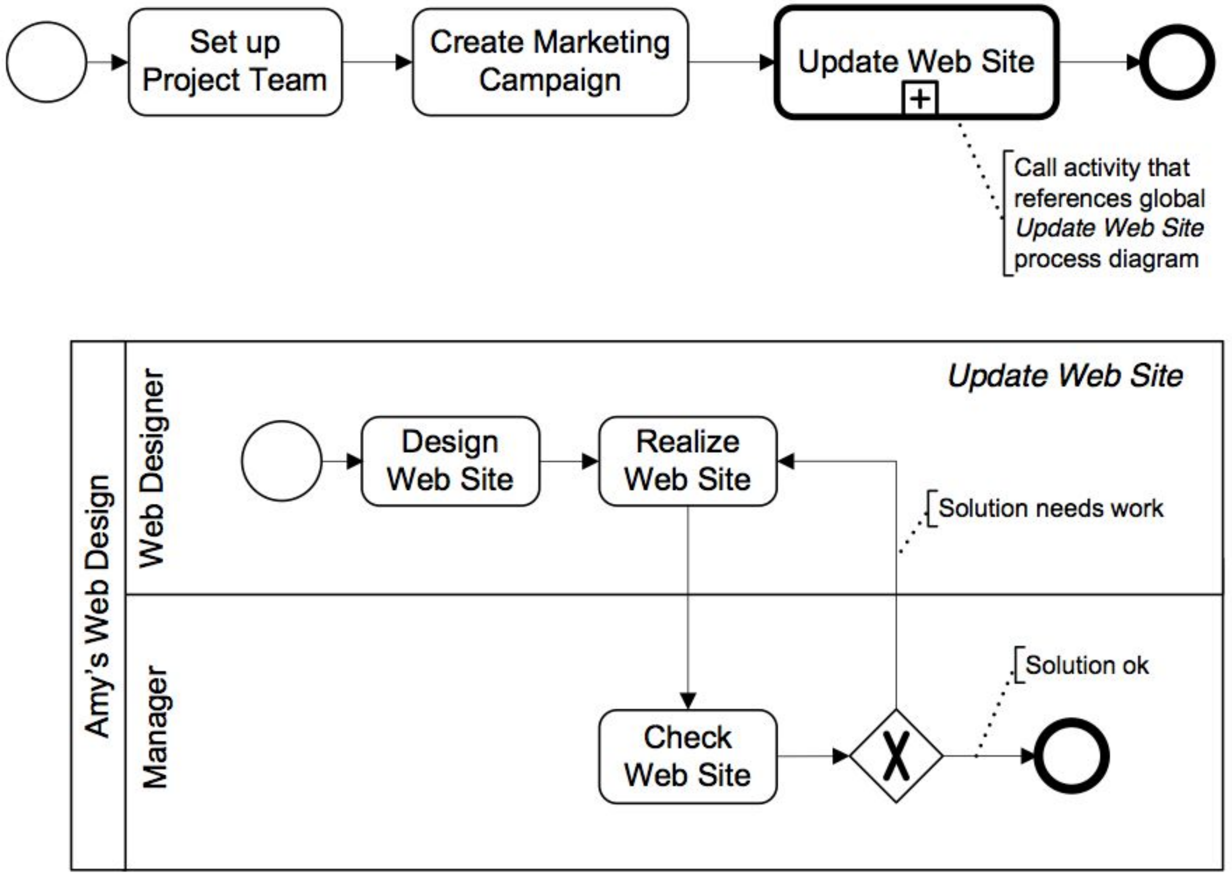


Fig. 4.81. Process diagram with a call activity that references a global process diagram; the reference is maintained in the respective attribute of the call activity

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

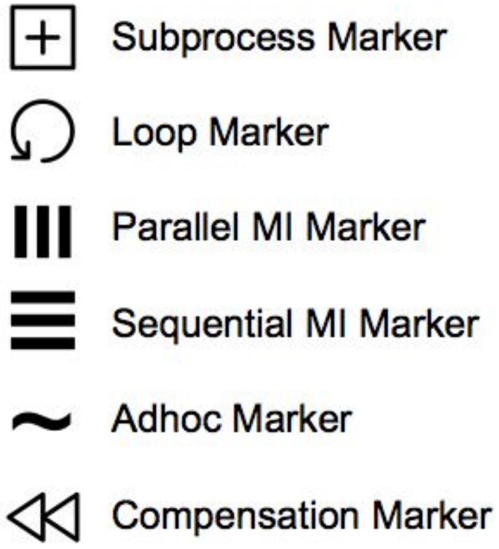


Fig. 4.82. Activity markers refine the behaviour of activities

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

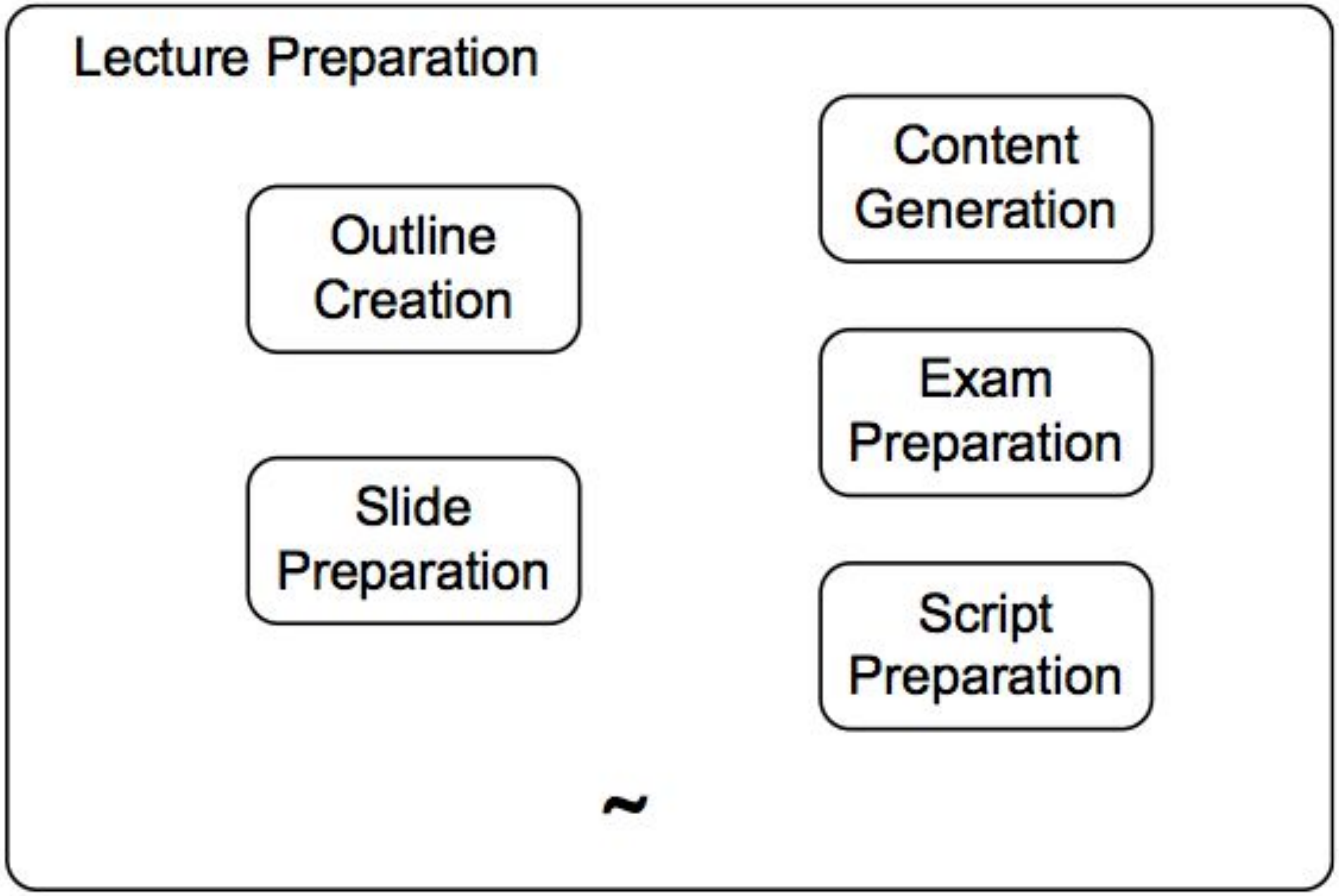


Fig. 4.83. Sample adhoc process

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

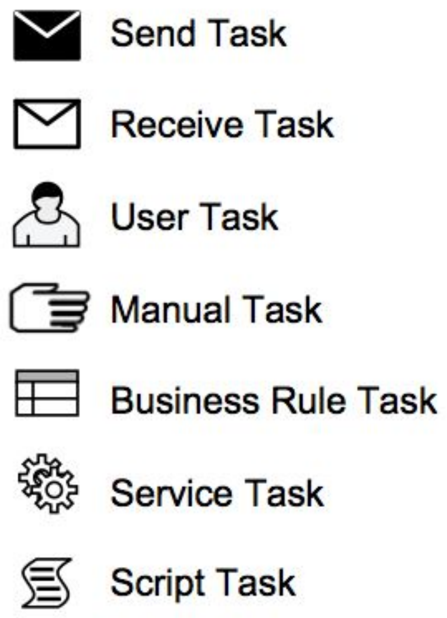
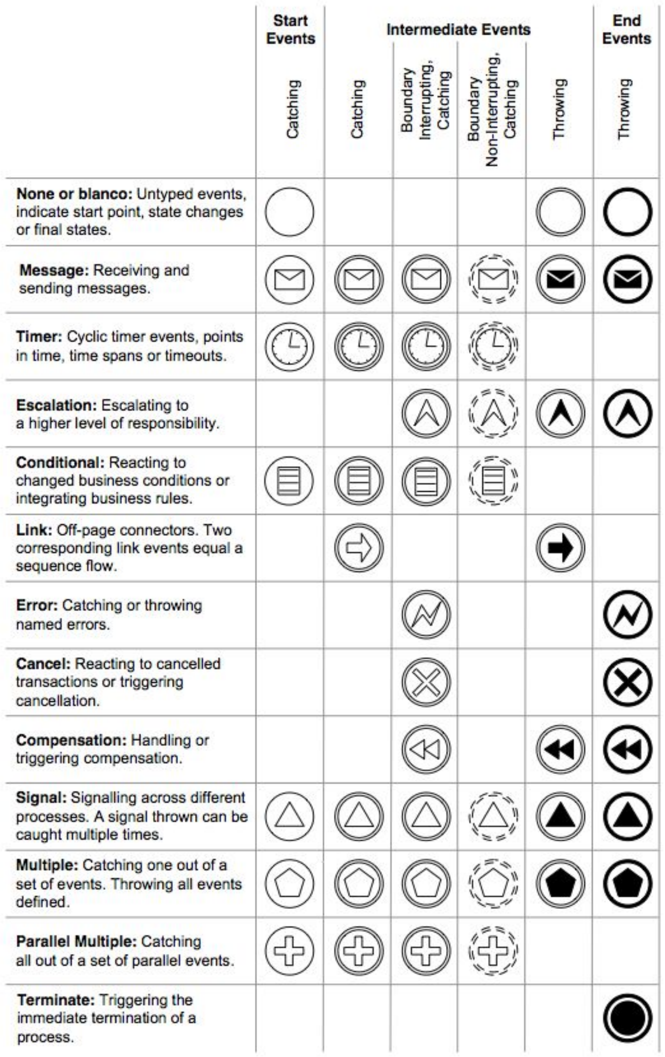


Fig. 4.84. Task types specify the kind of task that is represented

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007



	Start Events	Intermediate Events				End Events
	Catching	Catching	Boundary Interrupting, Catching	Boundary Non-Interrupting, Catching	Throwing	Throwing
None or blanco: Untyped events, indicate start point, state changes or final states.	○				○	○
Message: Receiving and sending messages.	✉	✉	✉	✉	✉	✉
Timer: Cyclic timer events, points in time, time spans or timeouts.	◷	◷	◷	◷		
Escalation: Escalating to a higher level of responsibility.			⮝	⮝	⮝	⮝
Conditional: Reacting to changed business conditions or integrating business rules.	≡	≡	≡	≡		
Link: Off-page connectors. Two corresponding link events equal a sequence flow.		⇨			➡	
Error: Catching or throwing named errors.			ϟ			ϟ
Cancel: Reacting to cancelled transactions or triggering cancellation.			✕			✕
Compensation: Handling or triggering compensation.			⏪		⏪	⏪
Signal: Signalling across different processes. A signal thrown can be caught multiple times.	△	△	△	△	▲	▲
Multiple: Catching one out of a set of events. Throwing all events defined.	⬠	⬠	⬠	⬠	⬟	⬟
Parallel Multiple: Catching all out of a set of parallel events.	✚	✚	✚	✚		
Terminate: Triggering the immediate termination of a process.						●

Fig. 4.85. Common event types in the BPMN, adapted from the BPMN Poster, BPM Offensive Berlin (2011)

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

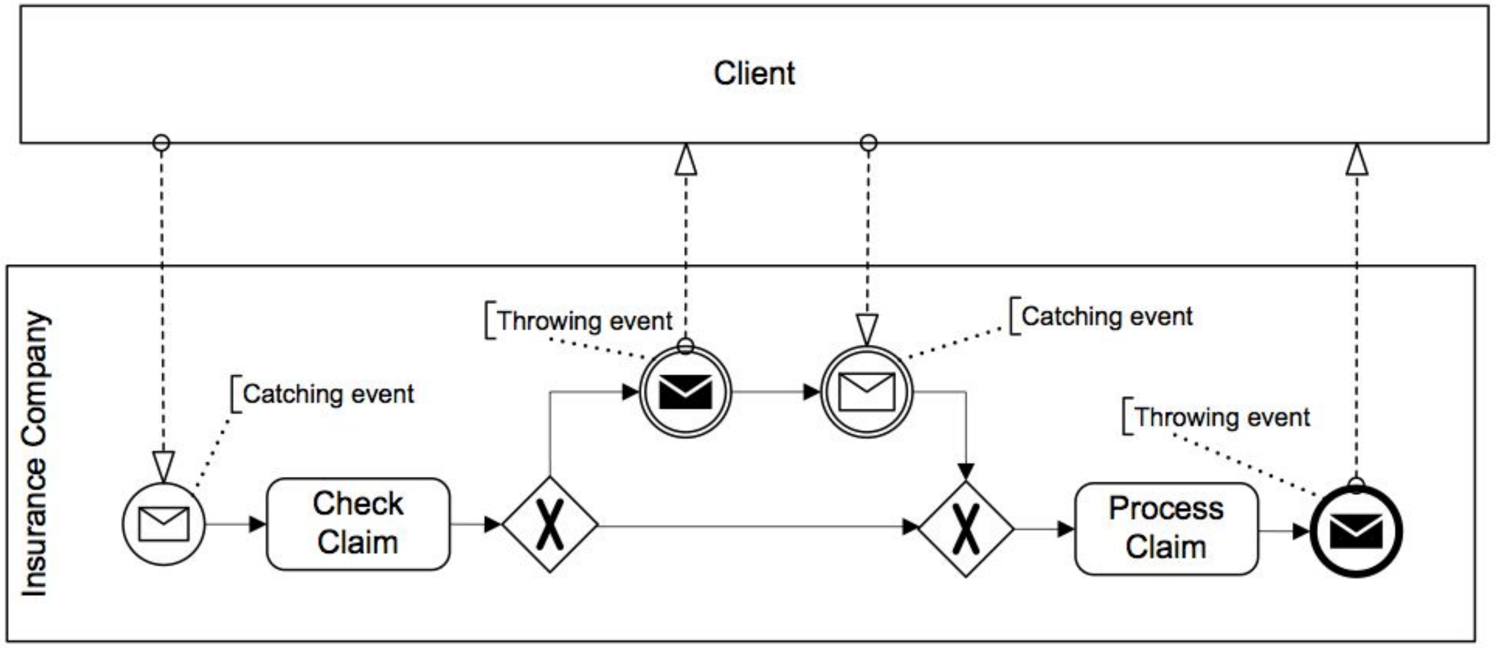


Fig. 4.86. Throwing and catching events

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

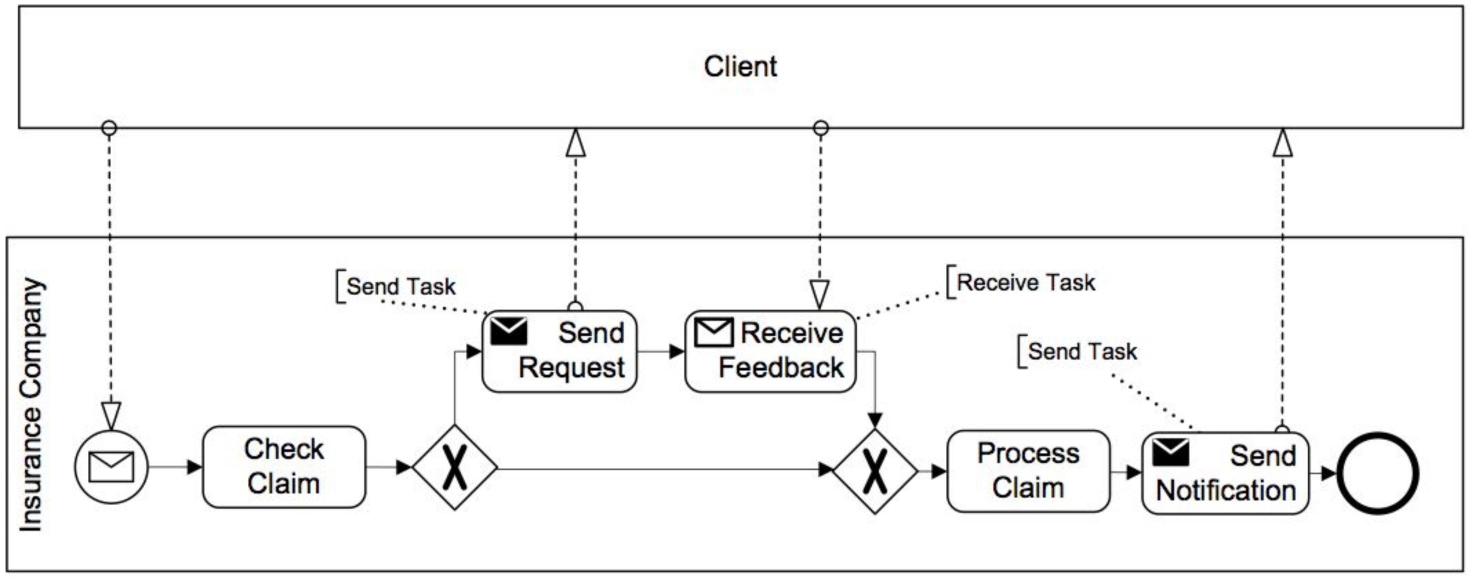


Fig. 4.87. Using markers to identify send tasks and receive tasks

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

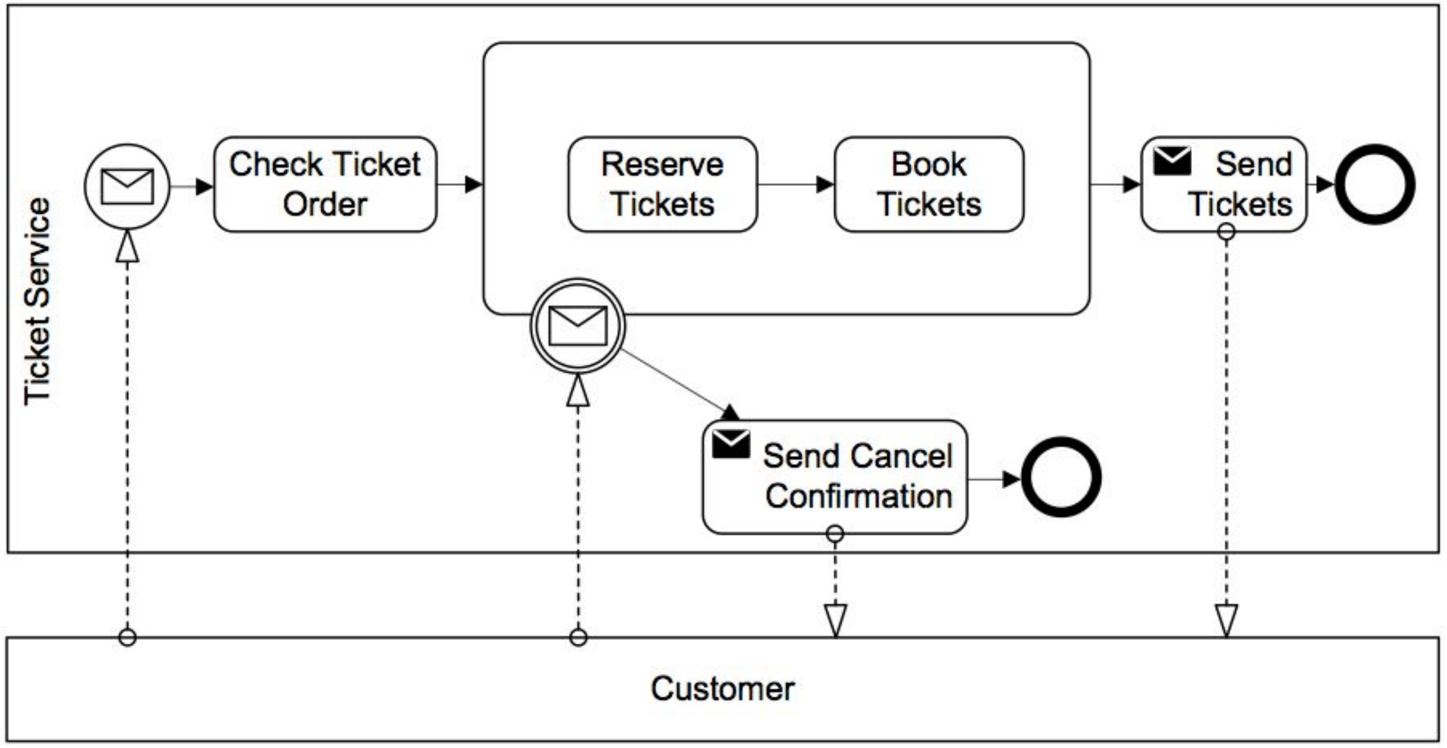


Fig. 4.88. Process diagram with interrupting boundary event

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

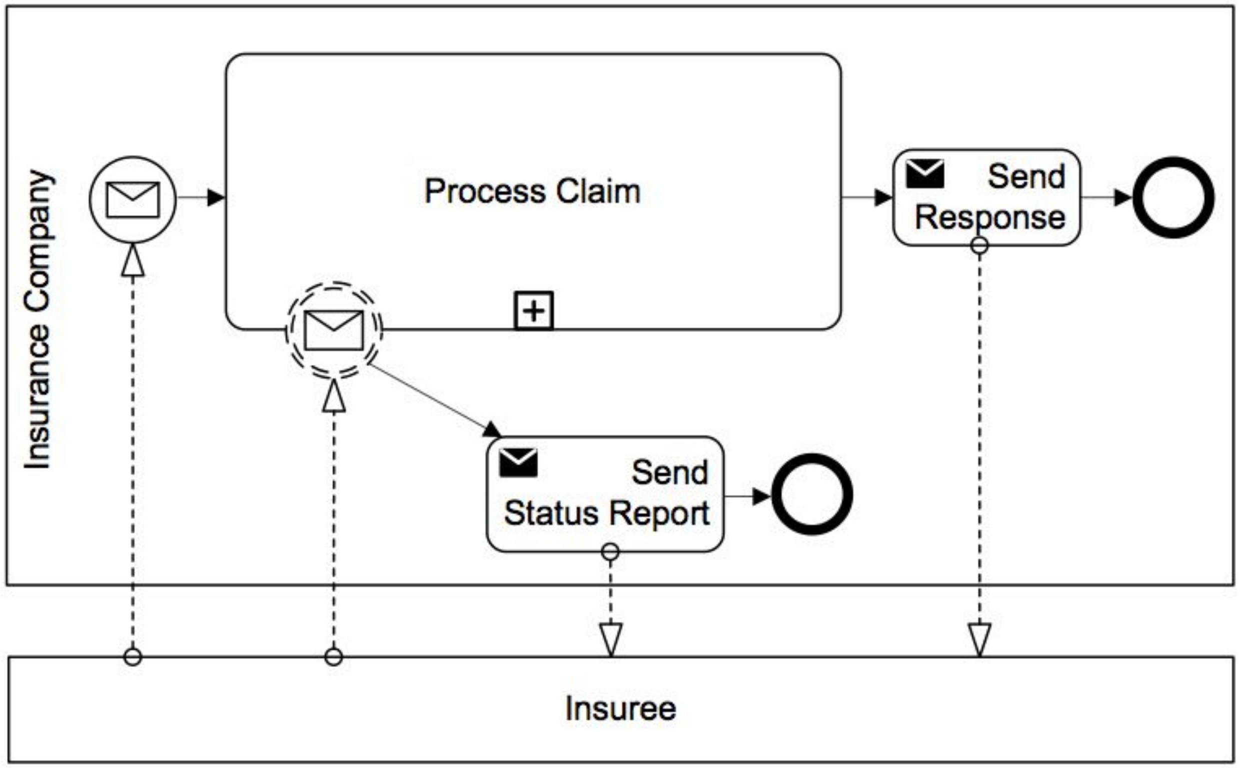


Fig. 4.89. Process diagram with non-interrupting boundary event

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

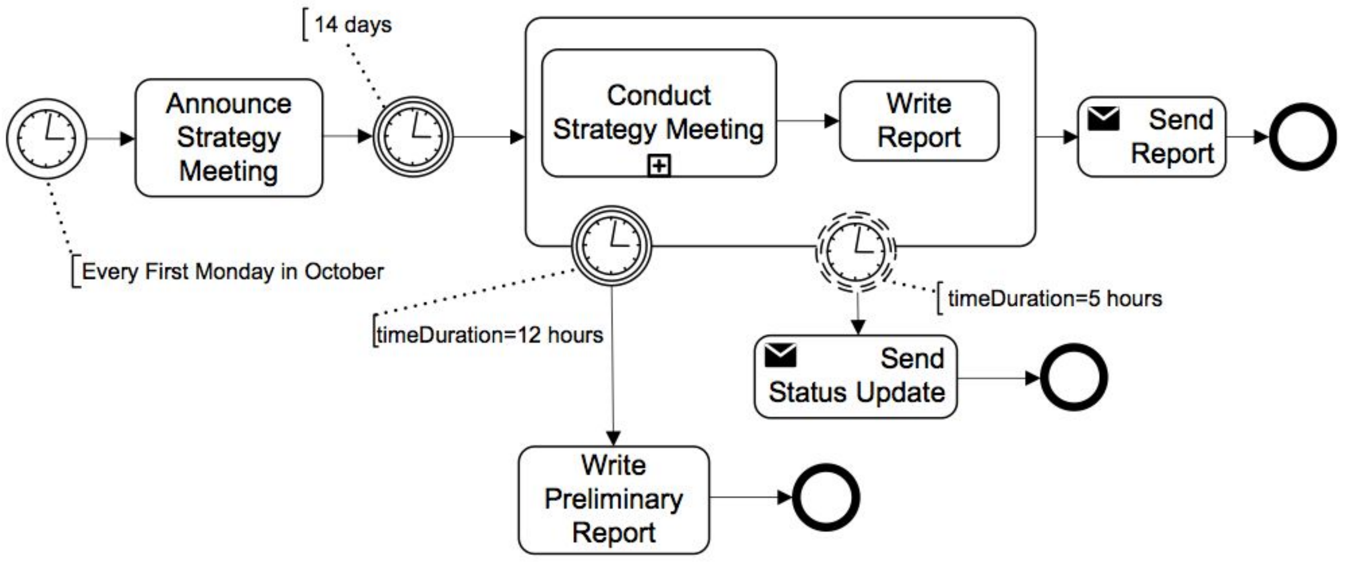


Fig. 4.90. Process diagram with interrupting and non-interrupting boundary timer events

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

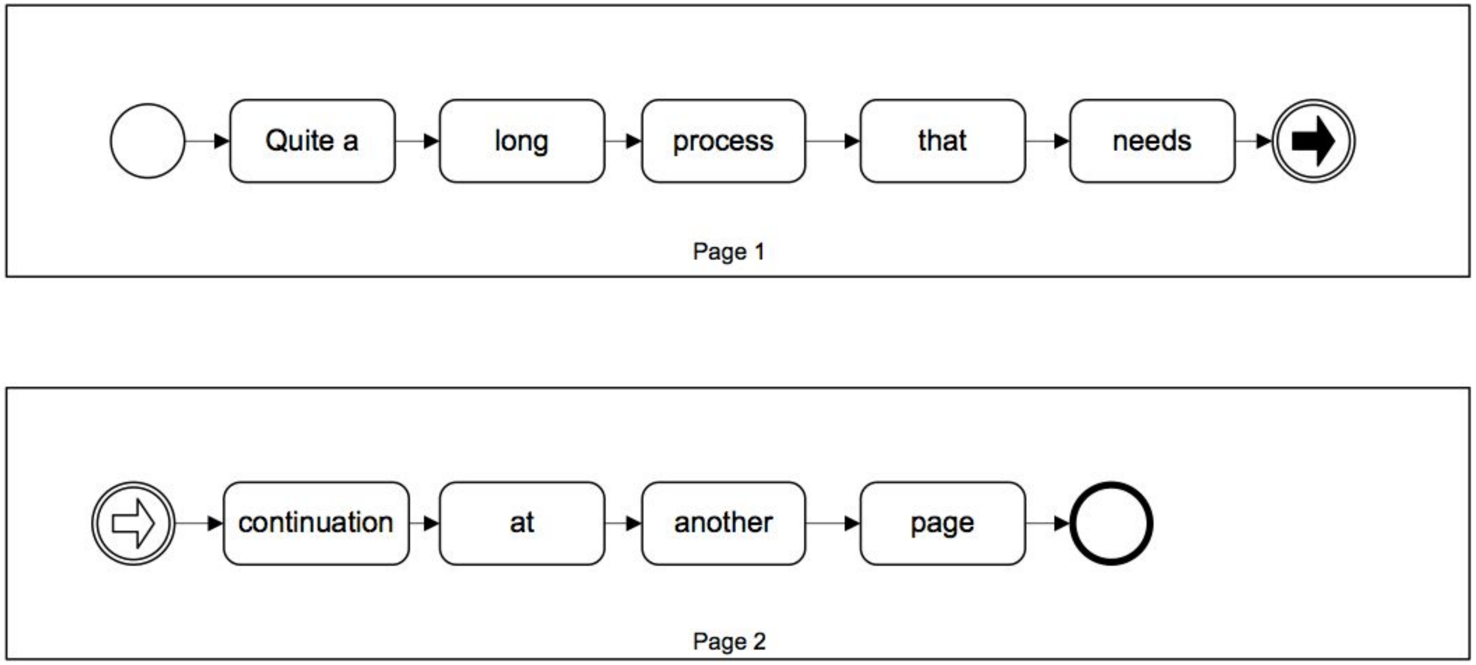


Fig. 4.91. Link events connect different parts of one process

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

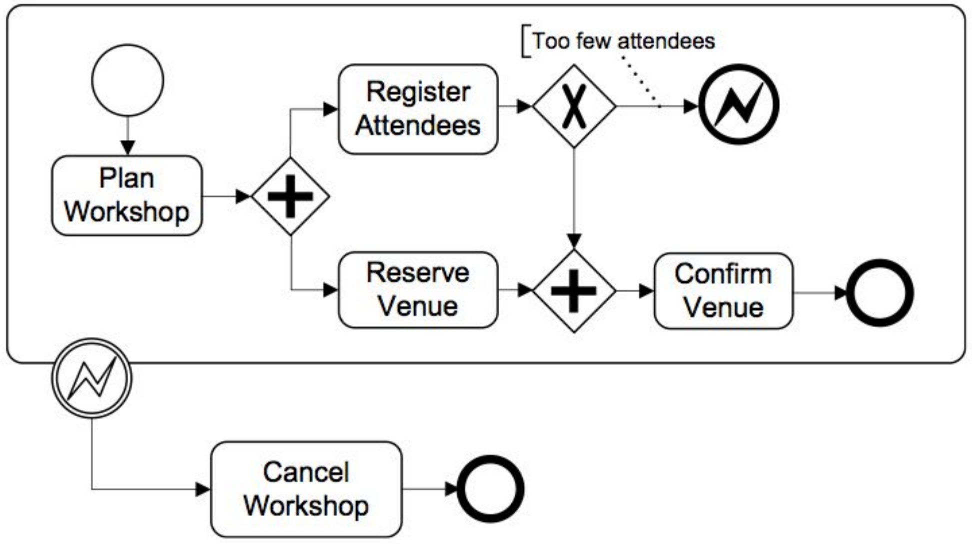


Fig. 4.92. An error is thrown in a subprocess; it is caught by an error boundary event attached to that subprocess

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

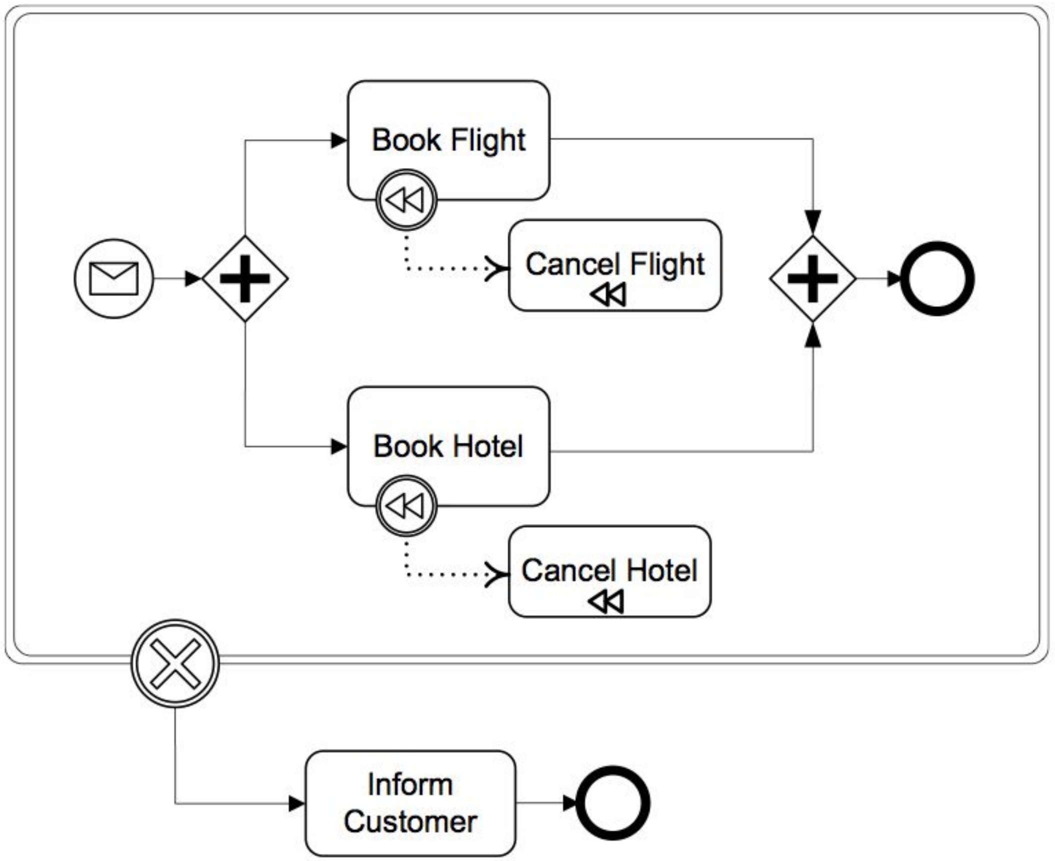


Fig. 4.93. Business process diagram with transaction and compensation elements, adapted from Object Management Group (2011)

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

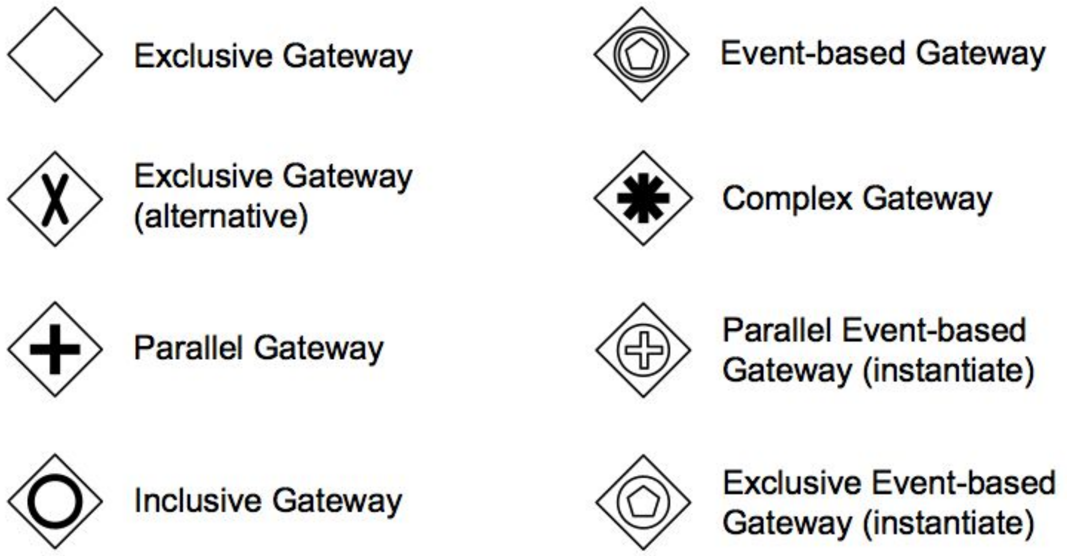

Exclusive Gateway
Exclusive Gateway (alternative)
Parallel Gateway
Inclusive Gateway
Event-based Gateway
Complex Gateway
Parallel Event-based Gateway (instantiate)
Exclusive Event-based Gateway (instantiate)

Fig. 4.94. Gateway types in the BPMN, Object Management Group (2011)

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

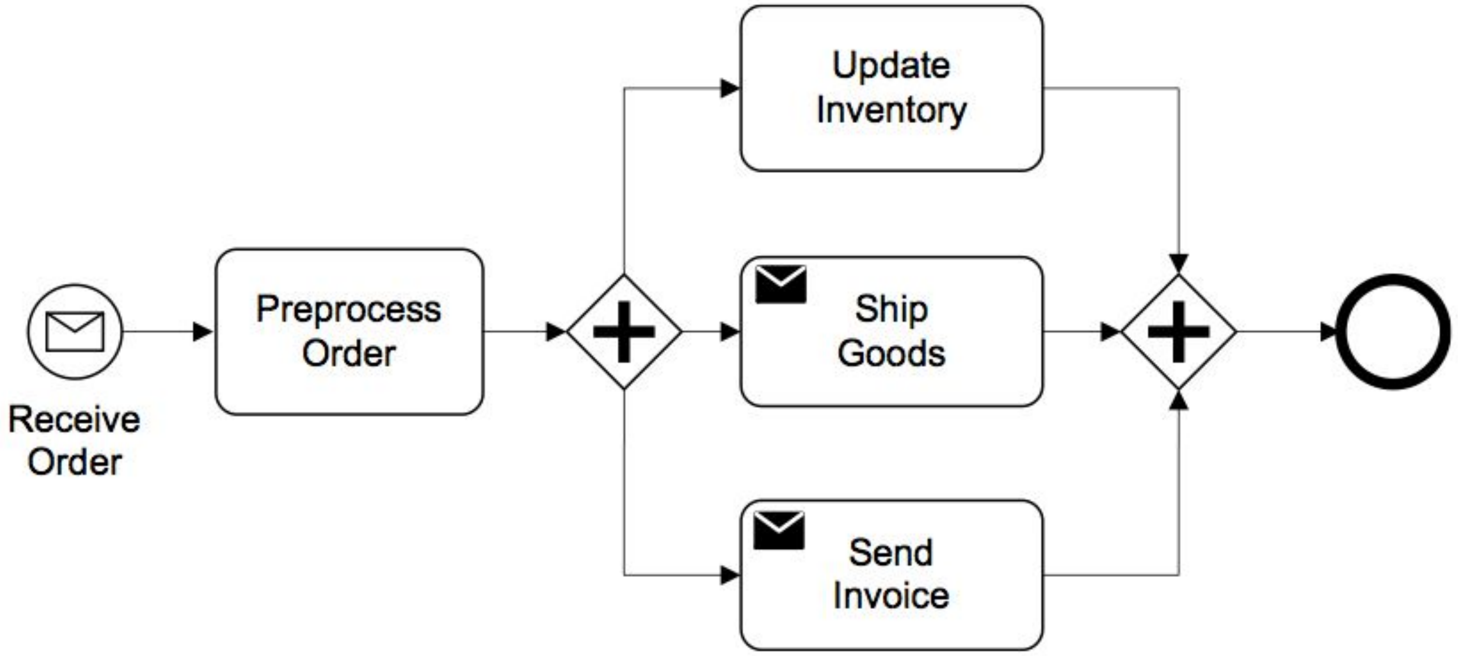


Fig. 4.95. Example involving the parallel gateway

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

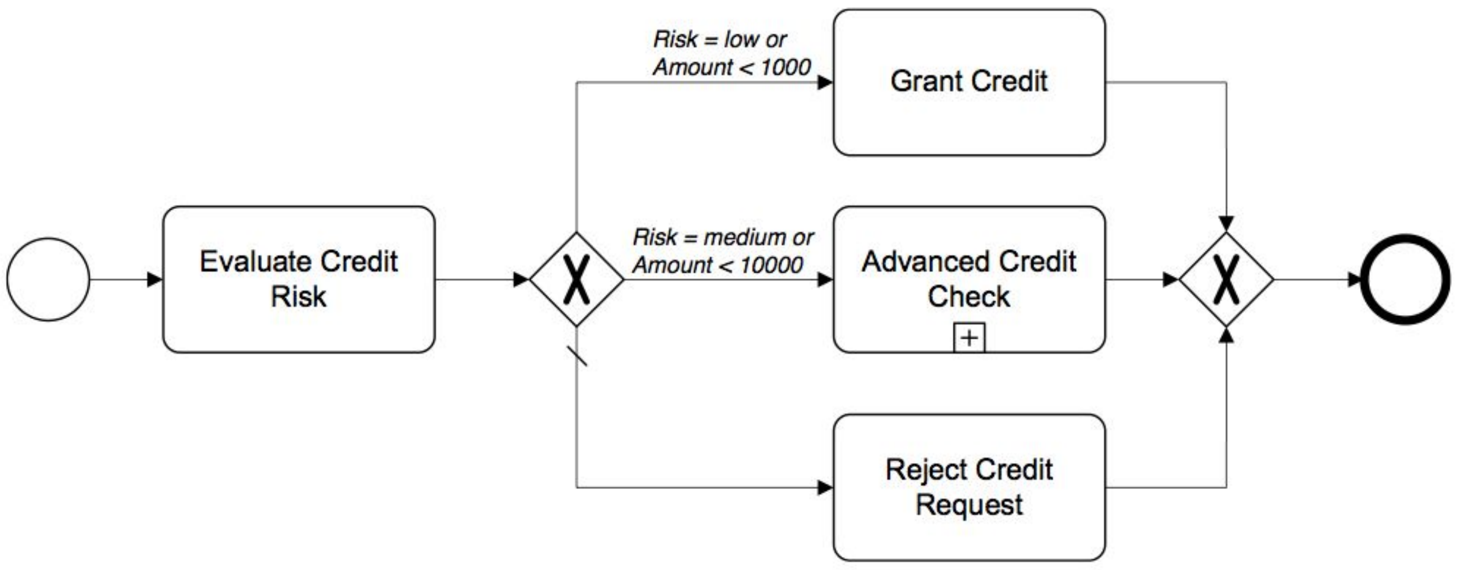


Fig. 4.96. Exclusive gateway with conditions and default flow

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

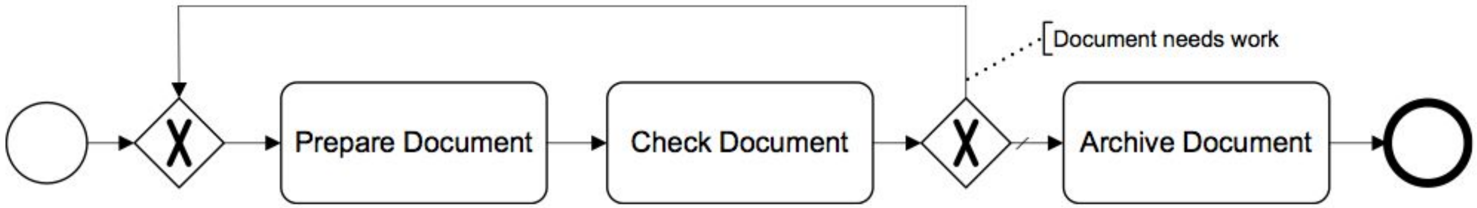


Fig. 4.97. Exclusive gateways realizing a loop

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

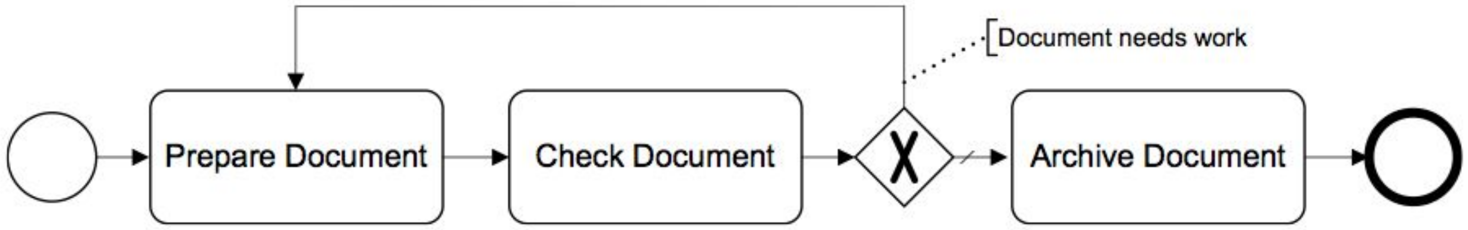


Fig. 4.98. Process diagram with uncontrolled flow

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

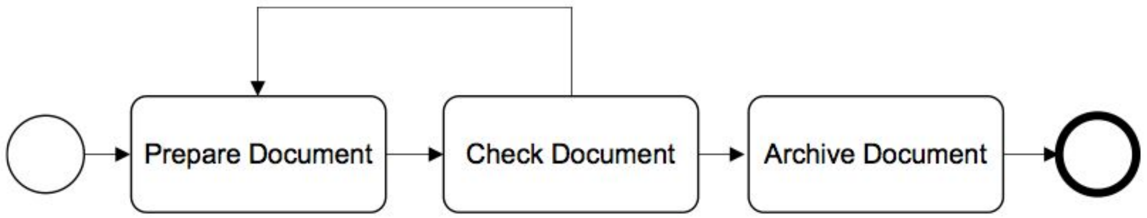


Fig. 4.99. Process diagram with split and join activities, representing a livelock

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

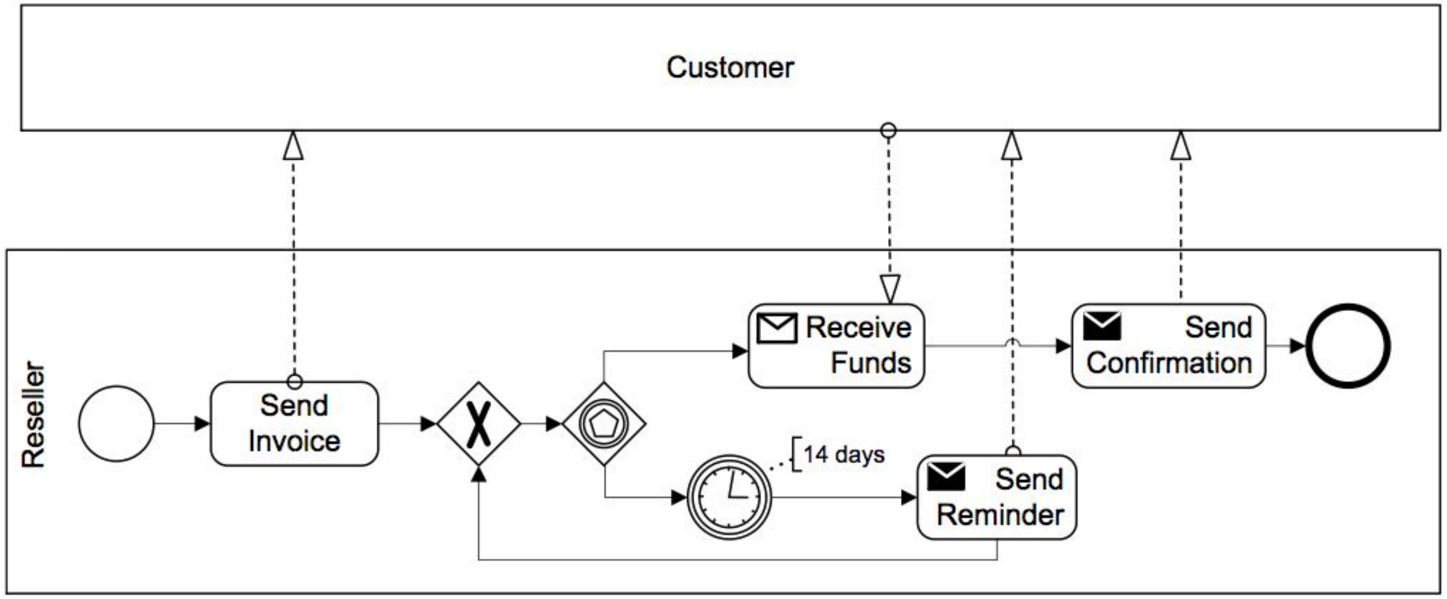


Fig. 4.100. Example of an *event-based* gateway

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

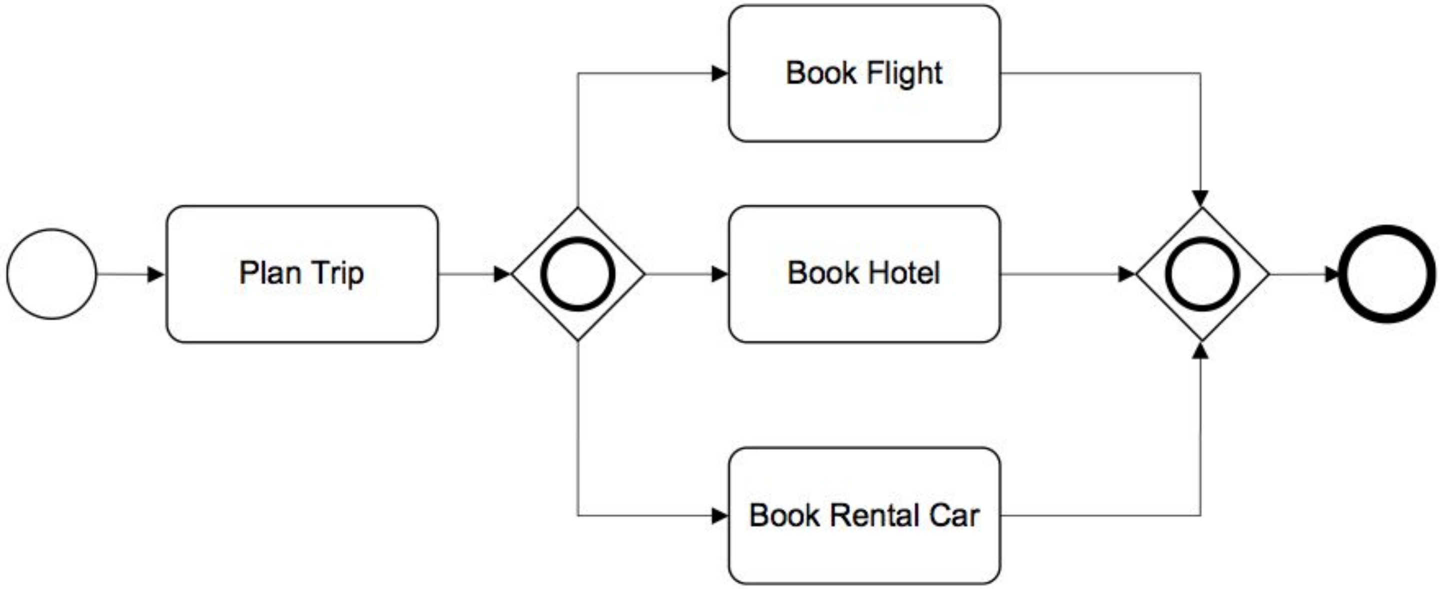


Fig. 4.101. Example of an *inclusive or* gateway

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

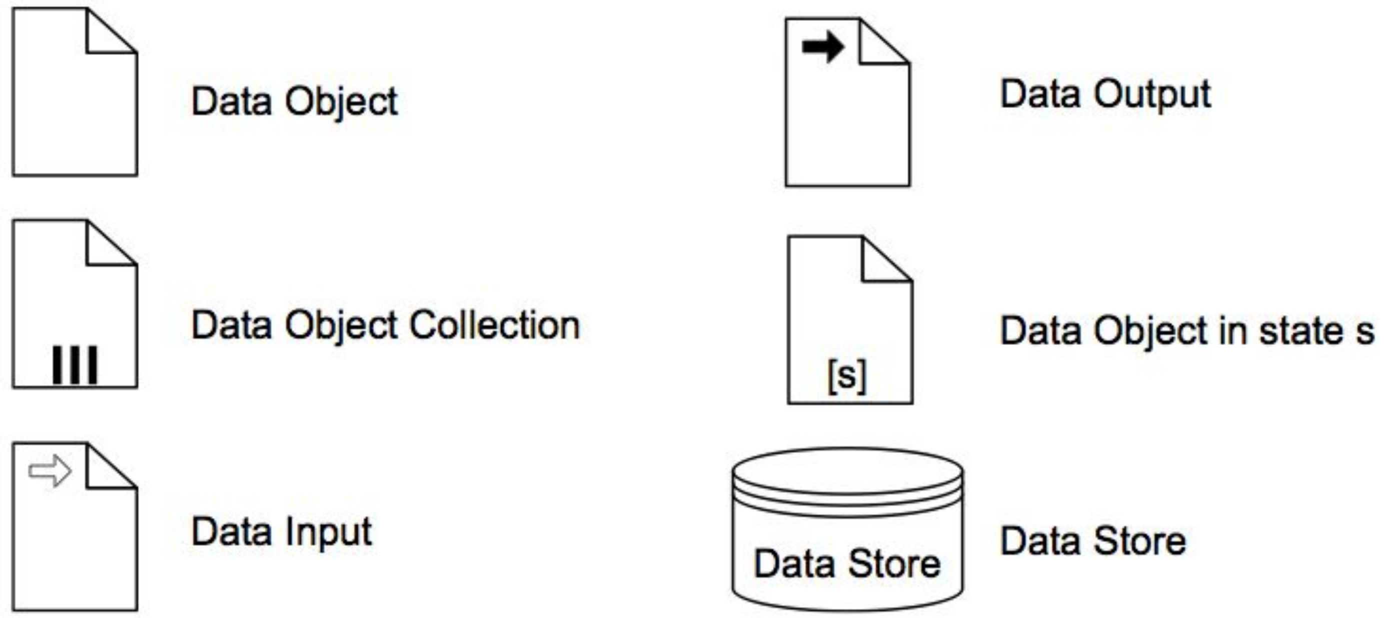


Fig. 4.102. Notational elements regarding data

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

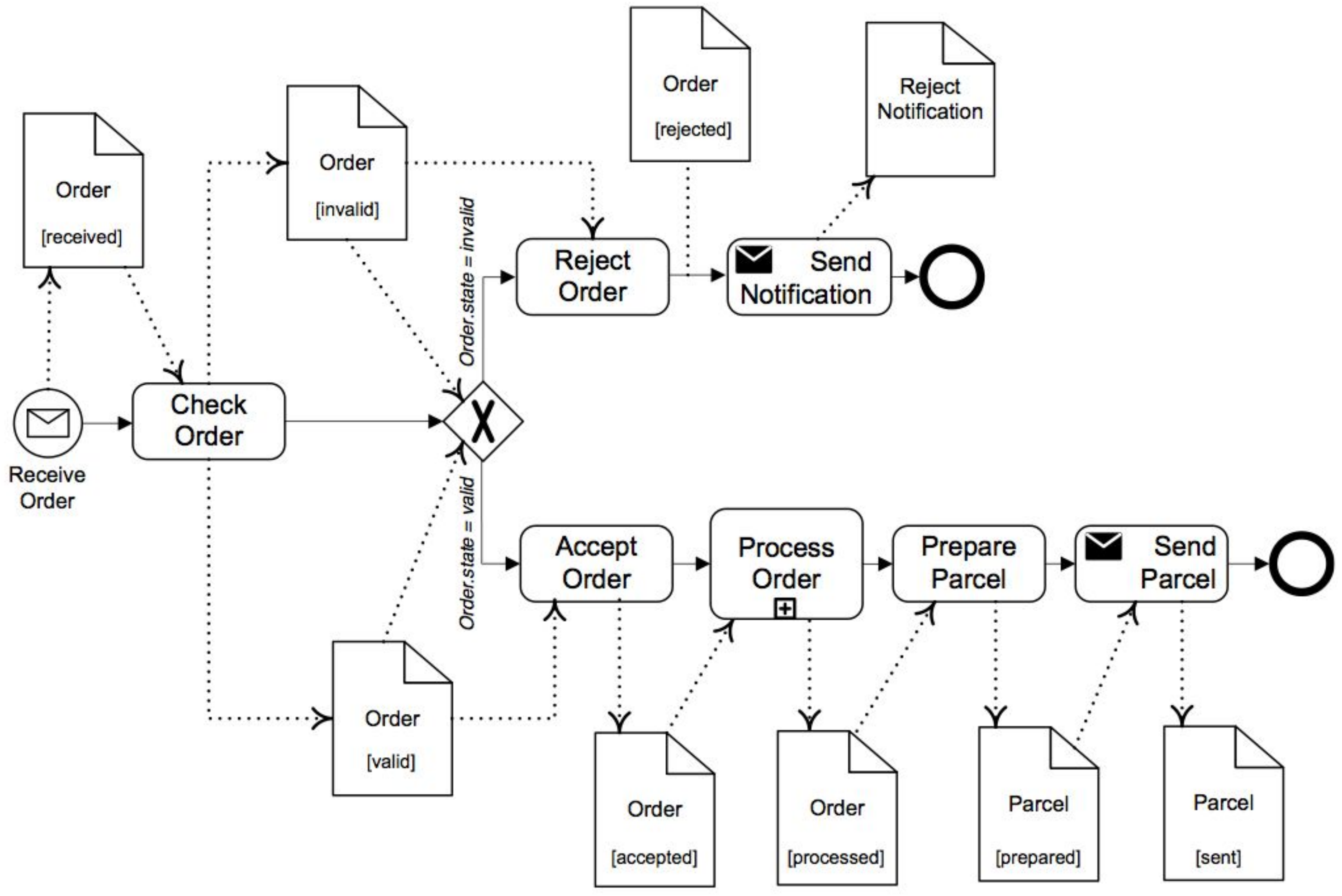


Fig. 4.103. Process diagram involving data objects

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

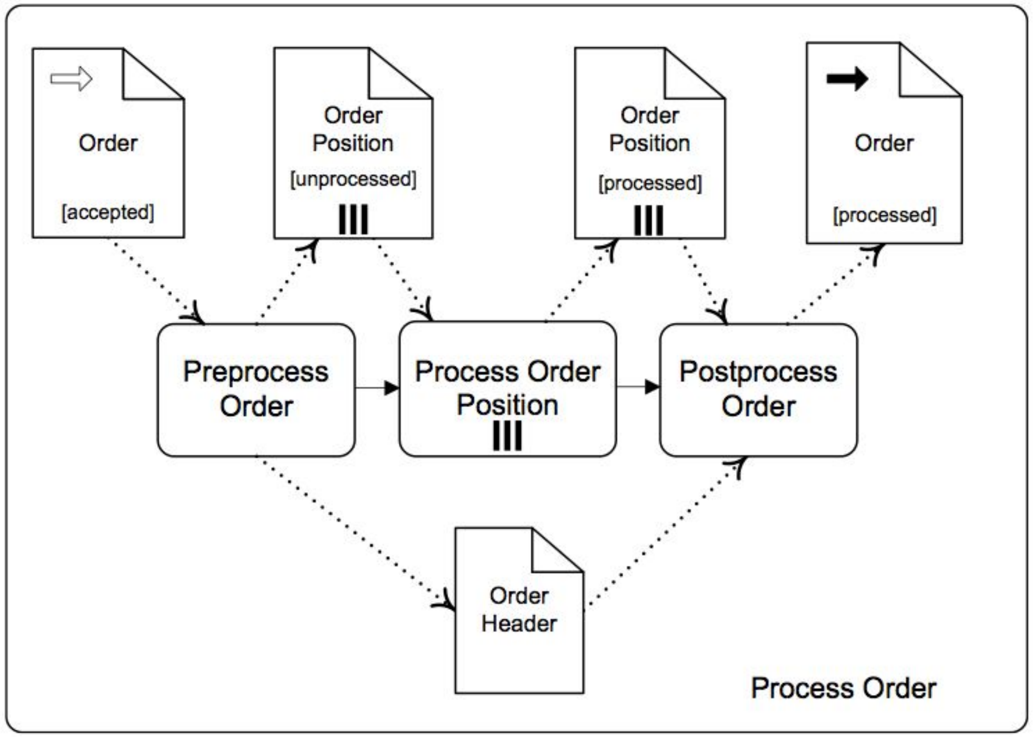


Fig. 4.104. Diagram of the *Process Order* subprocess from Figure 4.103, involving data object collections

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

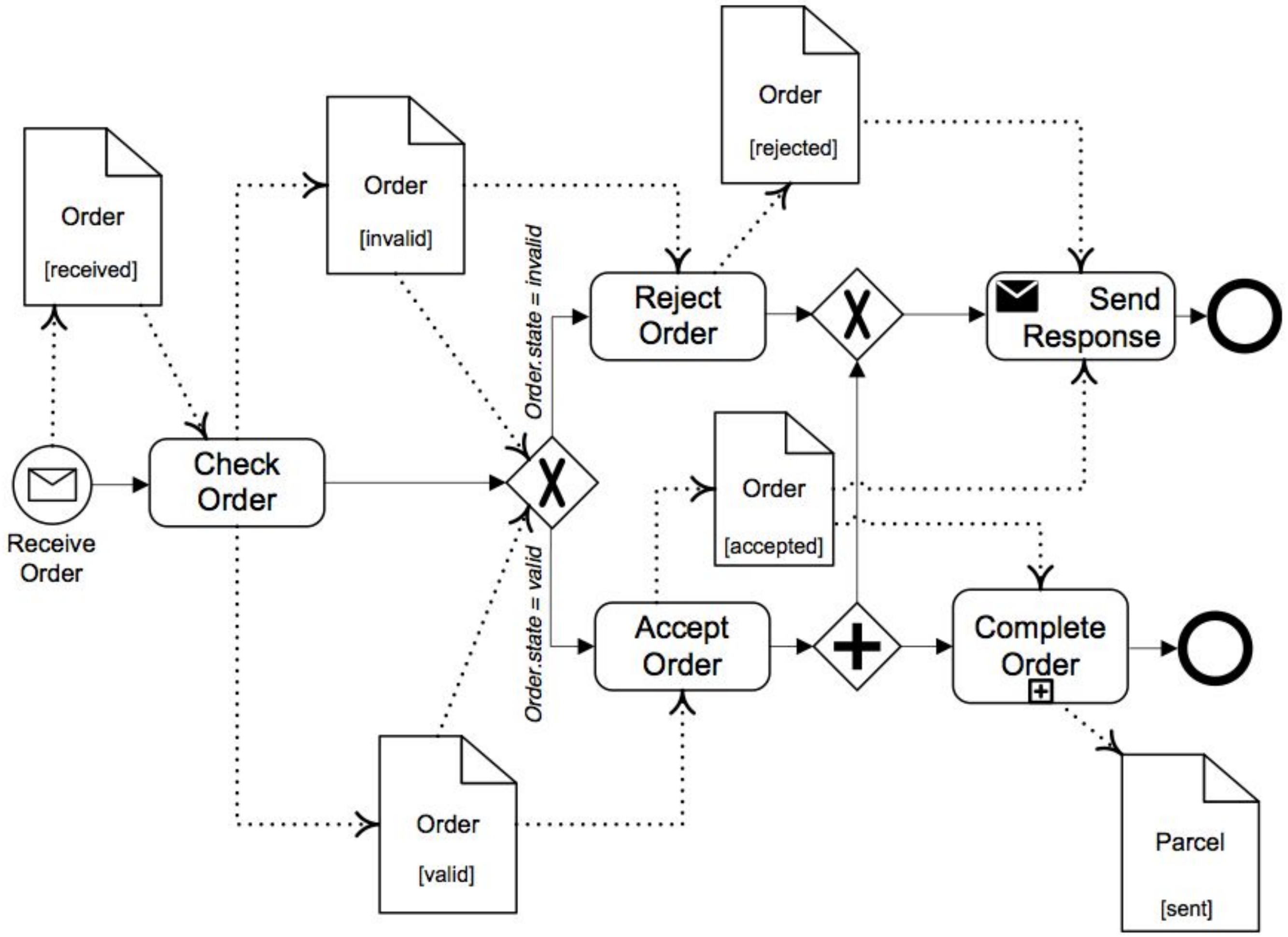
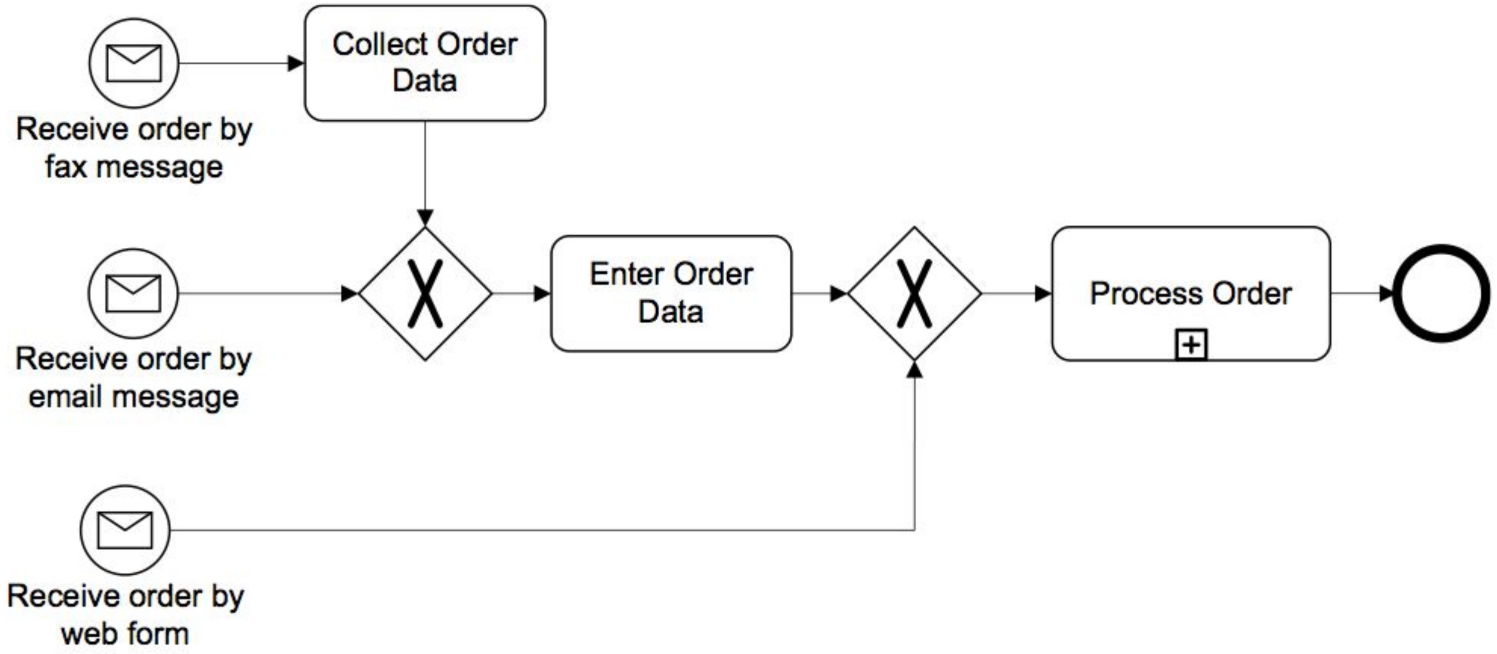


Fig. 4.105. Process diagram involving multiple input sets of an activity

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007



Receive order by fax message
Collect Order Data
Receive order by email message
Enter Order Data
Process Order
Receive order by web form

Fig. 4.106. Process diagram with multiple alternative start events

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

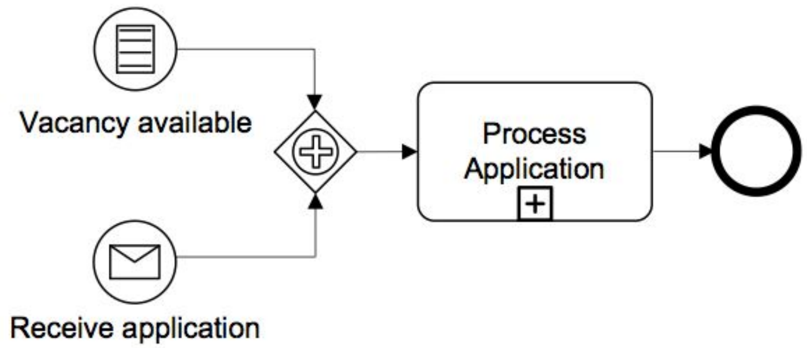


Fig. 4.107. Process diagram with two start events, both of which need to occur to instantiate the process

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

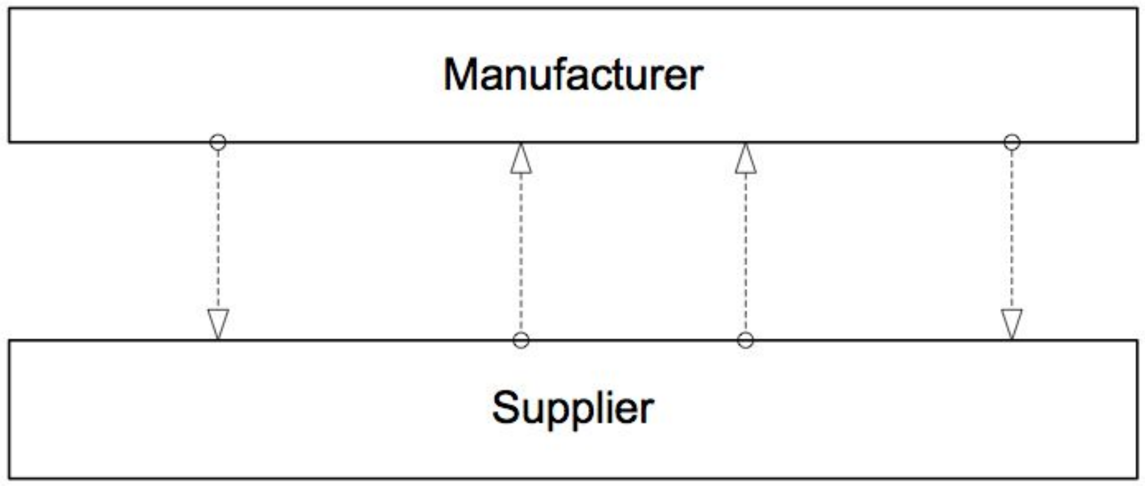


Fig. 4.108. Business processes collaborating through message flow

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

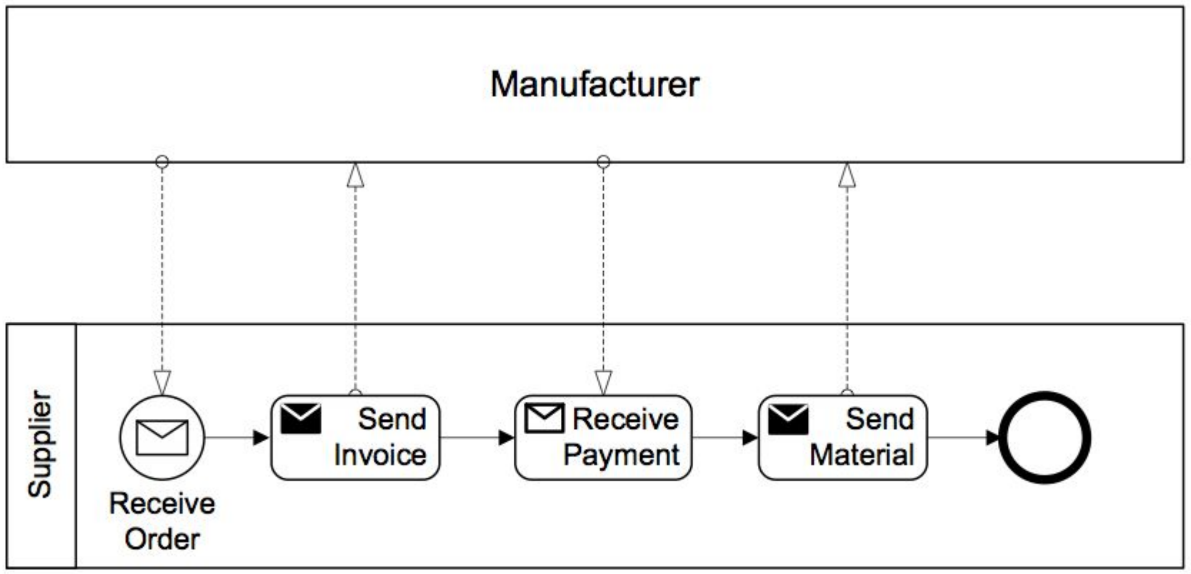
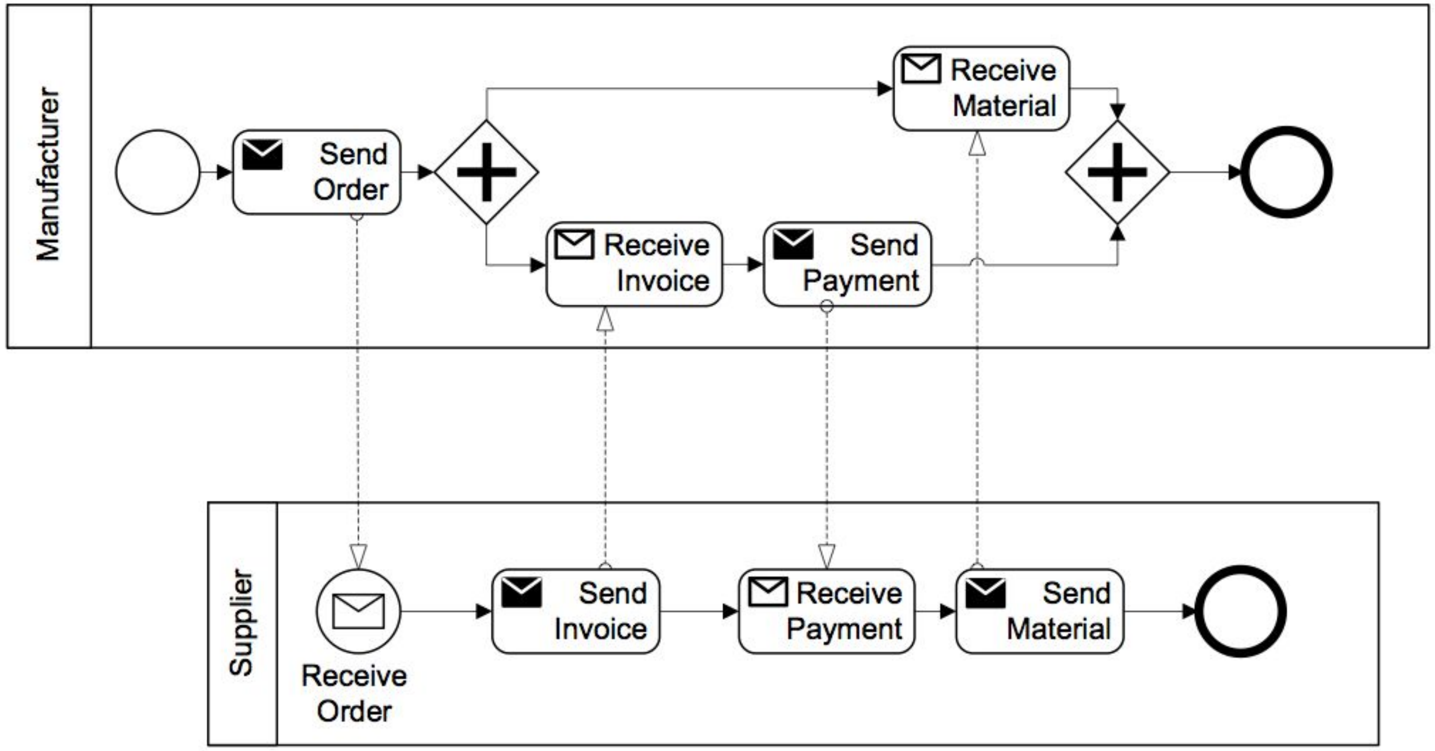


Fig. 4.109. Collaborating business processes with public process of the *Supplier*

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007



M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

Fig. 4.110. Collaborating business processes with public processes of both partners

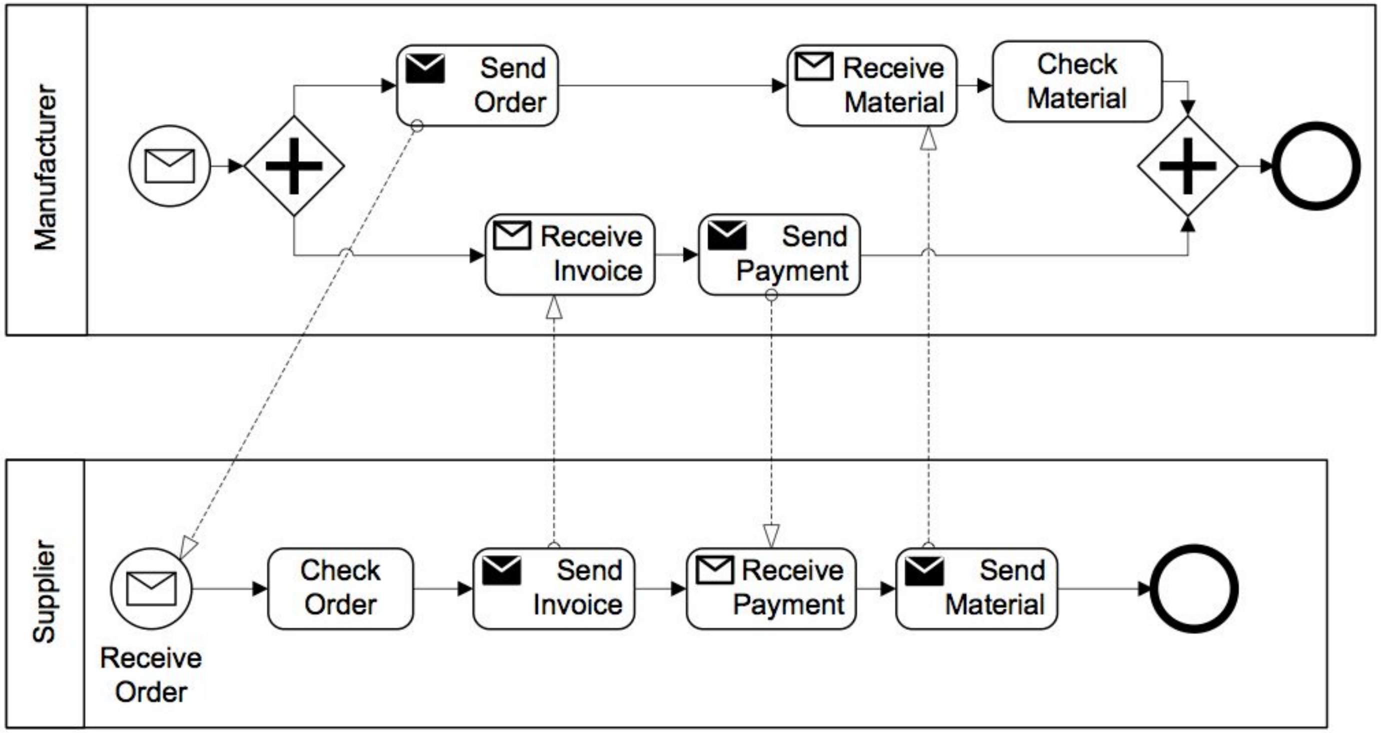


Fig. 4.111. Collaborating business processes with private processes of both partners

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

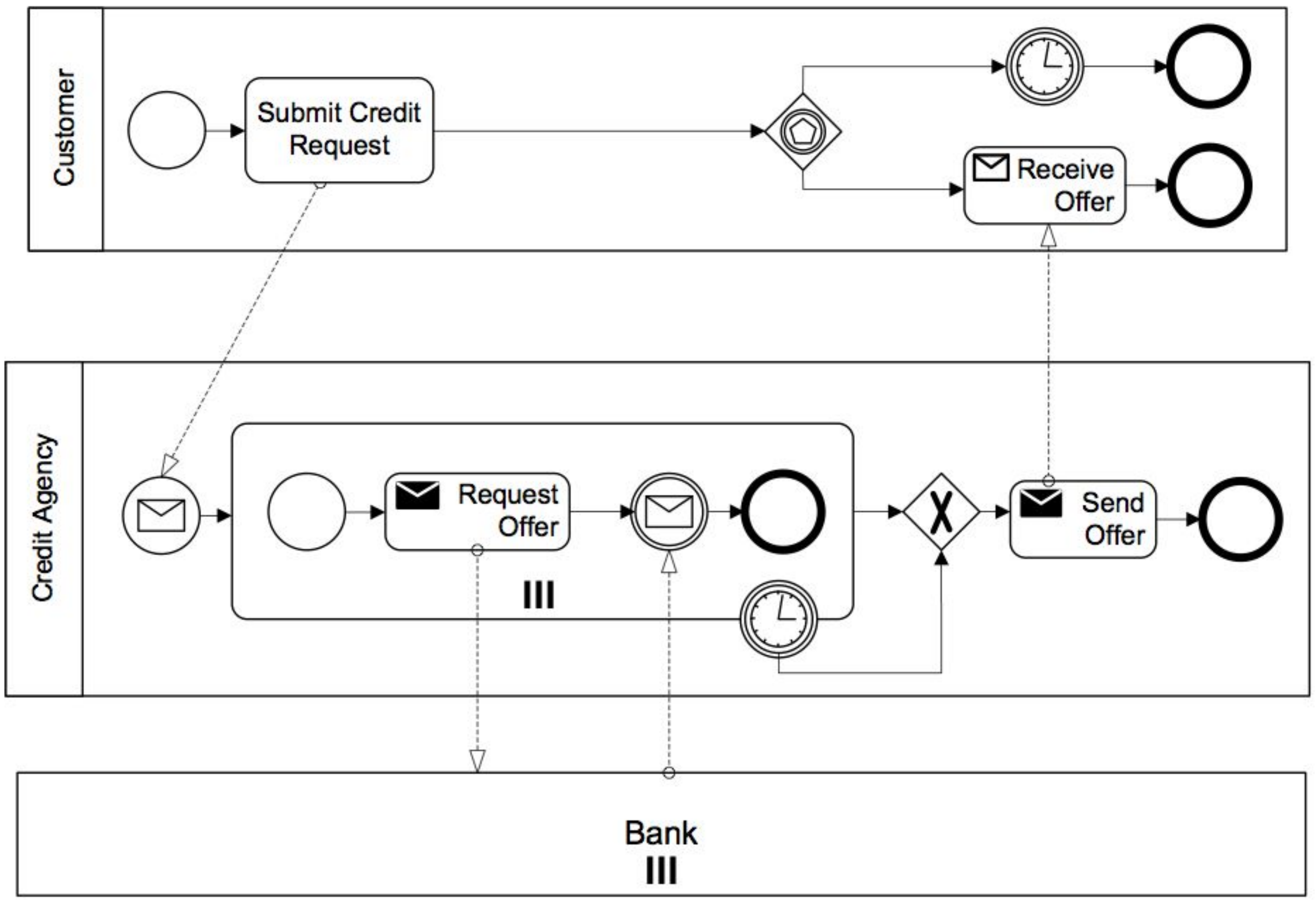


Fig. 4.112. Collaborating processes with a multiple instances pool

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007