

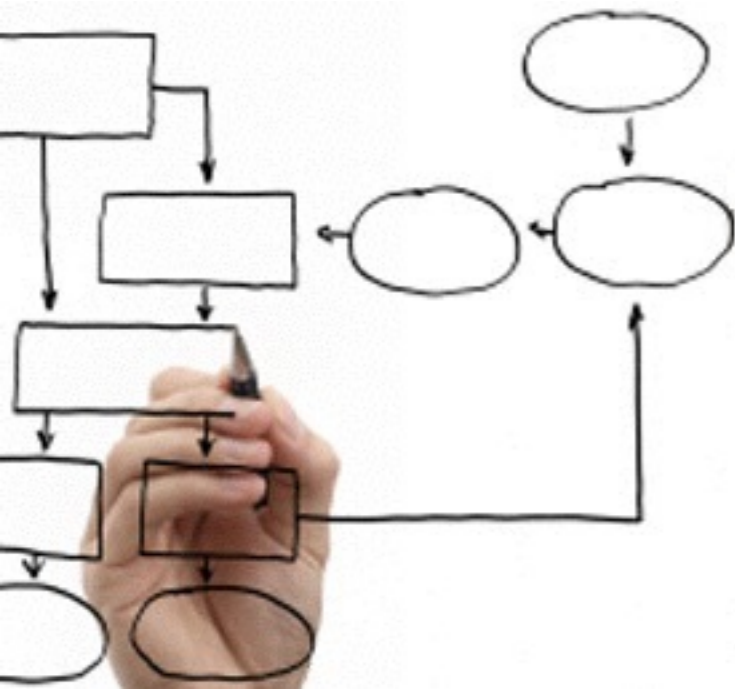
Methods for the specification and verification of business processes

MPB (6 cfu, 295AA)

Roberto Bruni

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

22 - Workflow patterns



Object

We overview some language-independent patterns that identify comprehensive workflow functionalities

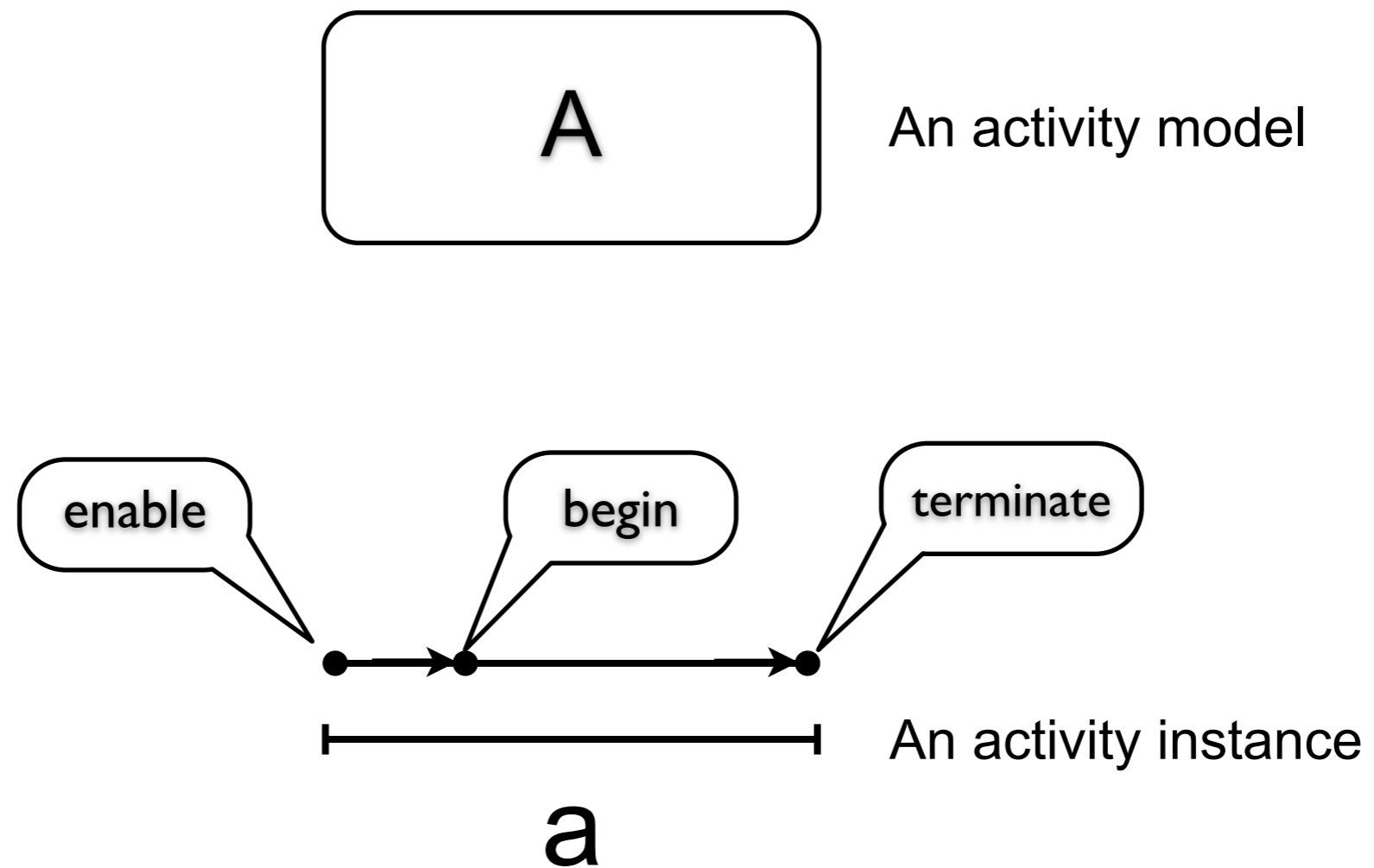
Workflow patterns

Along the years, the so-called workflow patterns have provided a coarse-grain yardstick for expressing, evaluating and comparing process orchestration

They are independent of concrete process languages

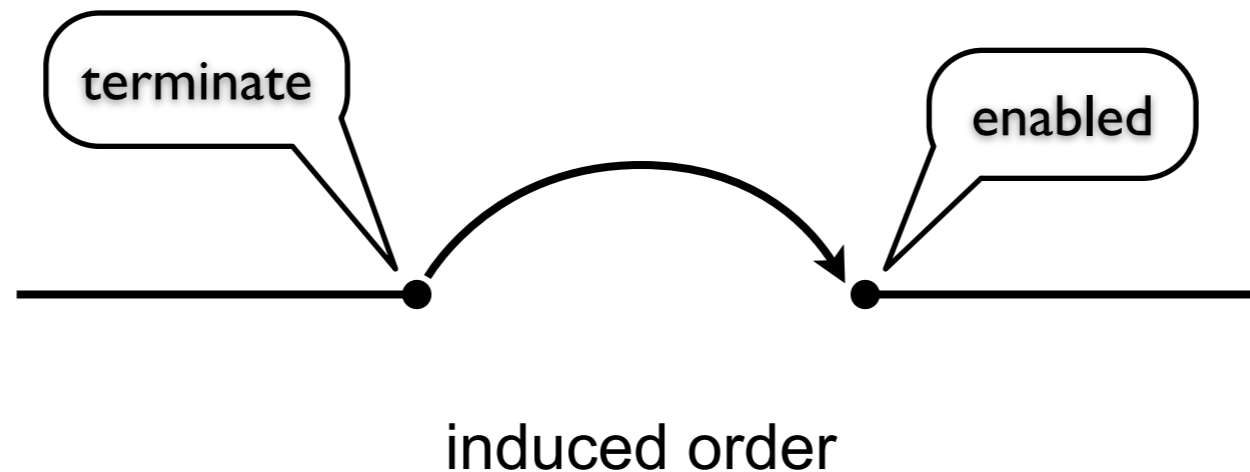
We illustrate them on the basis of a semantic model of events and event orderings

Preliminaries: activities



Preliminaries: control flow

—————→ A control flow construct



Basic Control Flow Patterns

- Pattern 1 (Sequence)
- Pattern 2 (Parallel Split)
- Pattern 3 (Synchronization)
- Pattern 4 (Exclusive Choice)
- Pattern 5 (Simple Merge)

Advanced Branching and Synchronization Patterns

- Pattern 6 (Multi-choice)
- Pattern 7 (Synchronizing Merge)
- Pattern 8 (Multi-merge)
- Pattern 9 (Discriminator)

Structural Patterns

- Pattern 10 (Arbitrary Cycles)
- Pattern 11 (Implicit Termination)

Cancellation Patterns

- Pattern 19 (Cancel Activity)
- Pattern 20 (Cancel Case)

State-based Patterns

- Pattern 16 (Deferred Choice)
- Pattern 17 (Interleaved Parallel Routing)
- Pattern 18 (Milestone)

Patterns involving Multiple Instances

- Pattern 12 (Multiple Instances Without Synchronization)
- Pattern 13 (Multiple Instances With a Priori Design Time Knowledge)
- Pattern 14 (Multiple Instances With a Priori Runtime Knowledge)
- Pattern 15 (Multiple Instances Without a Priori Runtime Knowledge)

WFP #1-5

Most common patterns

Closely match the definitions of elementary aspects of control flow concepts

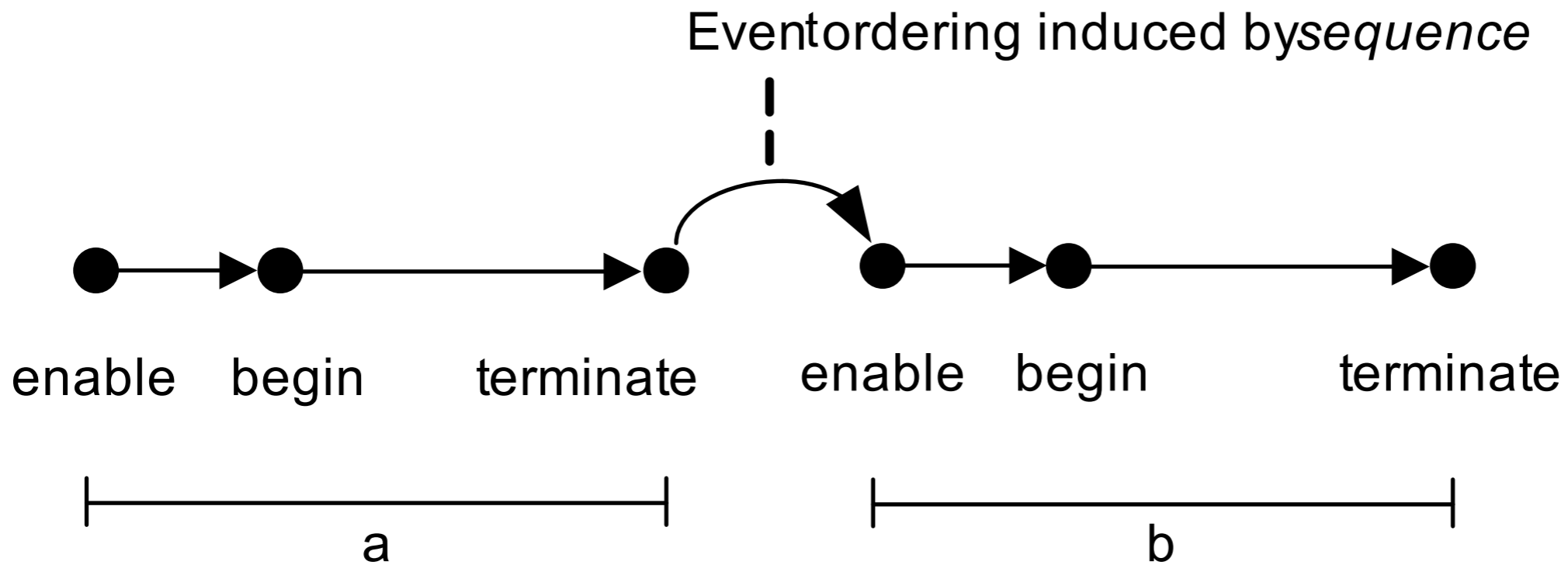
They are present in all workflow engines

#1 Sequence

Description: An activity in a workflow process is enabled after the completion of another activity in the same process

Synonyms: Sequential routing, serial routing

#1 Sequence

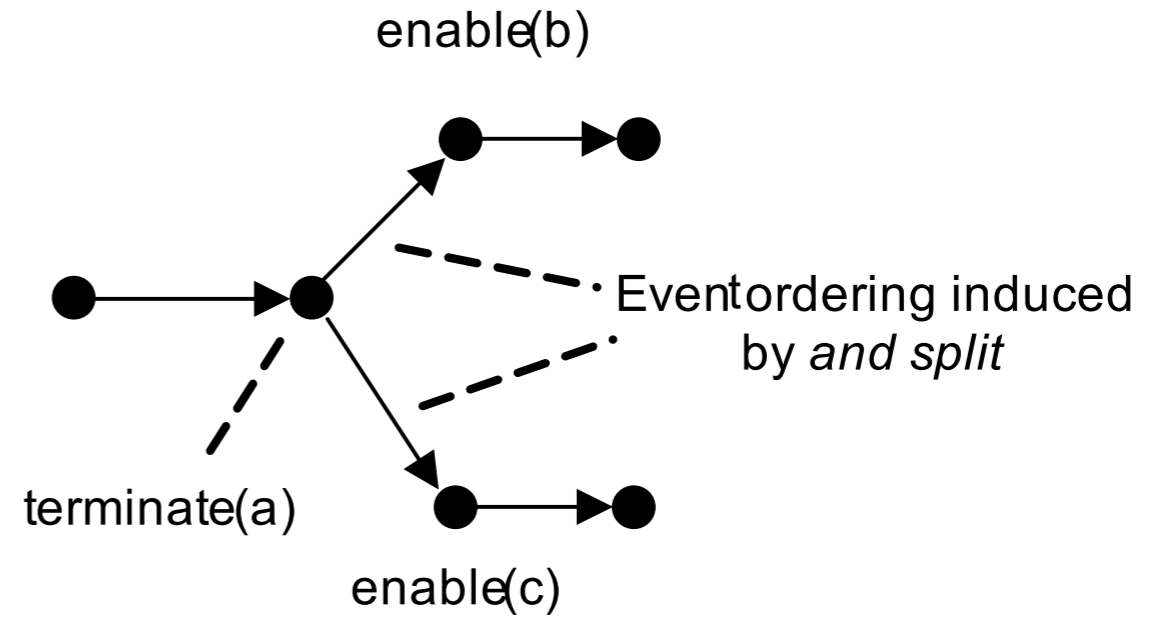
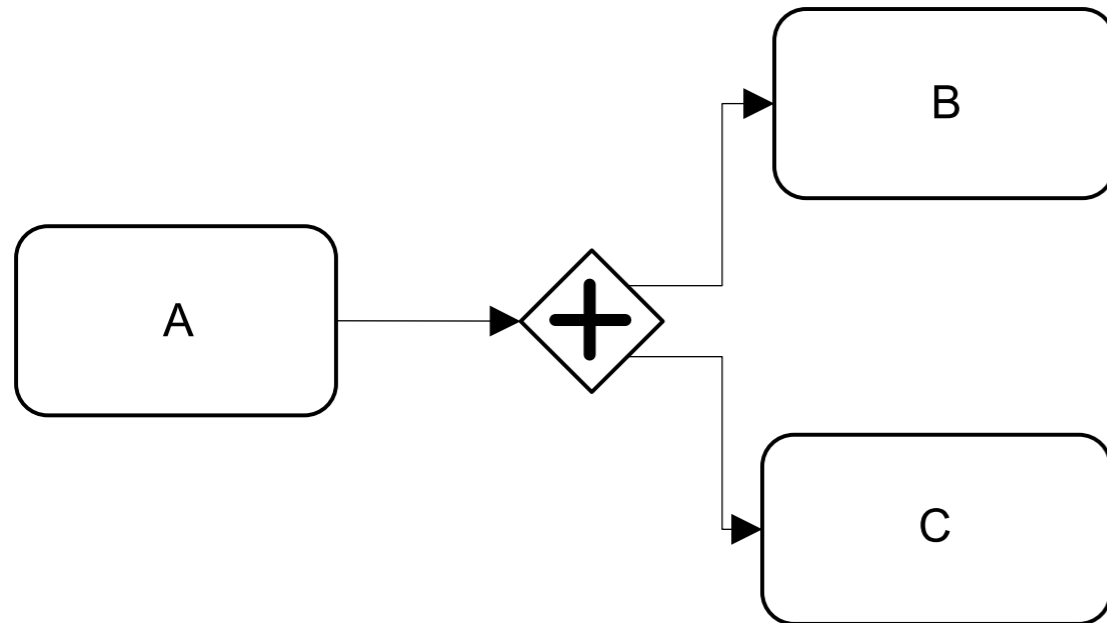


#2 Parallel split

Description: A point in the workflow process where a single thread of control splits into multiple threads of control which can be executed in parallel, thus allowing activities to be executed simultaneously or in any order

Synonyms: AND-split, parallel routing, fork

#2 Parallel split

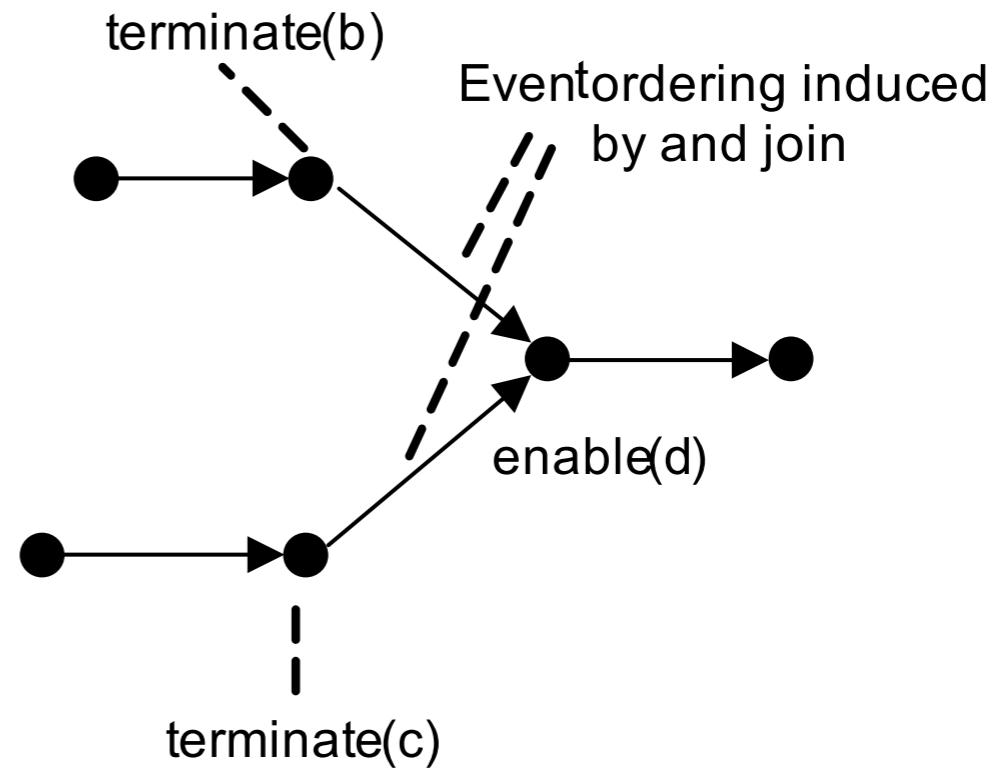
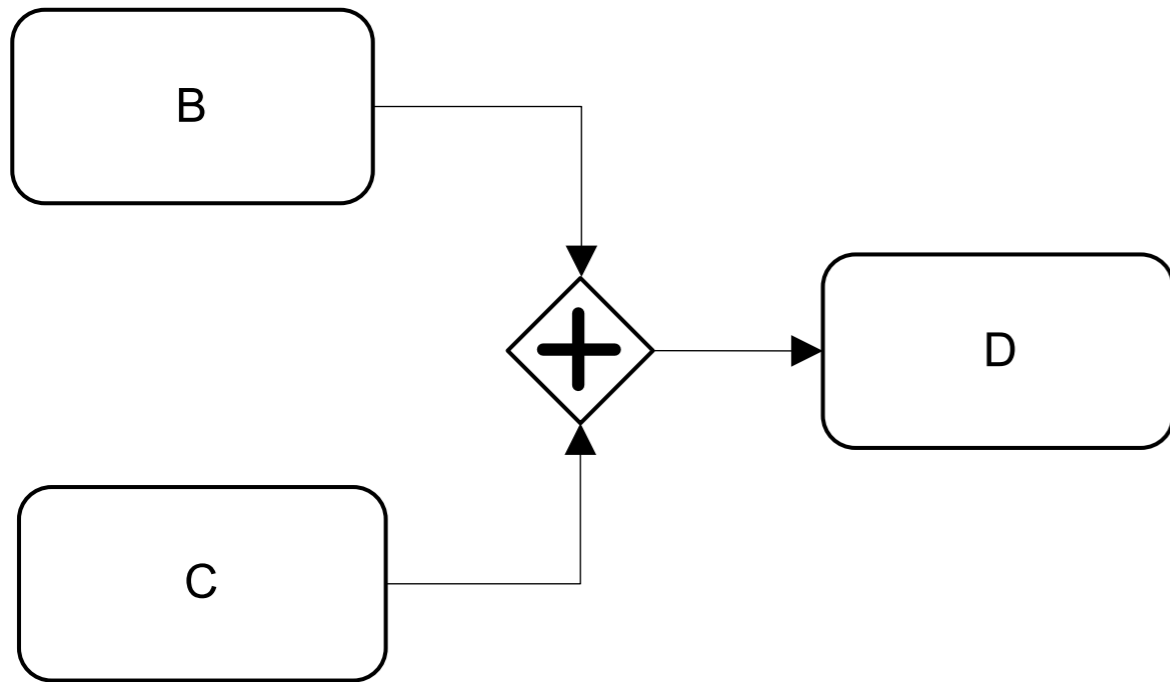


#3 Synchronization

Description: A point in the workflow process where multiple parallel activities converge into one single thread of control, thus synchronizing multiple threads. It is an assumption of this pattern that each incoming branch is executed only once (if this is not the case, see patterns #13--15)

Synonyms: AND-join, rendezvous, synchronizer

#3 Synchronization

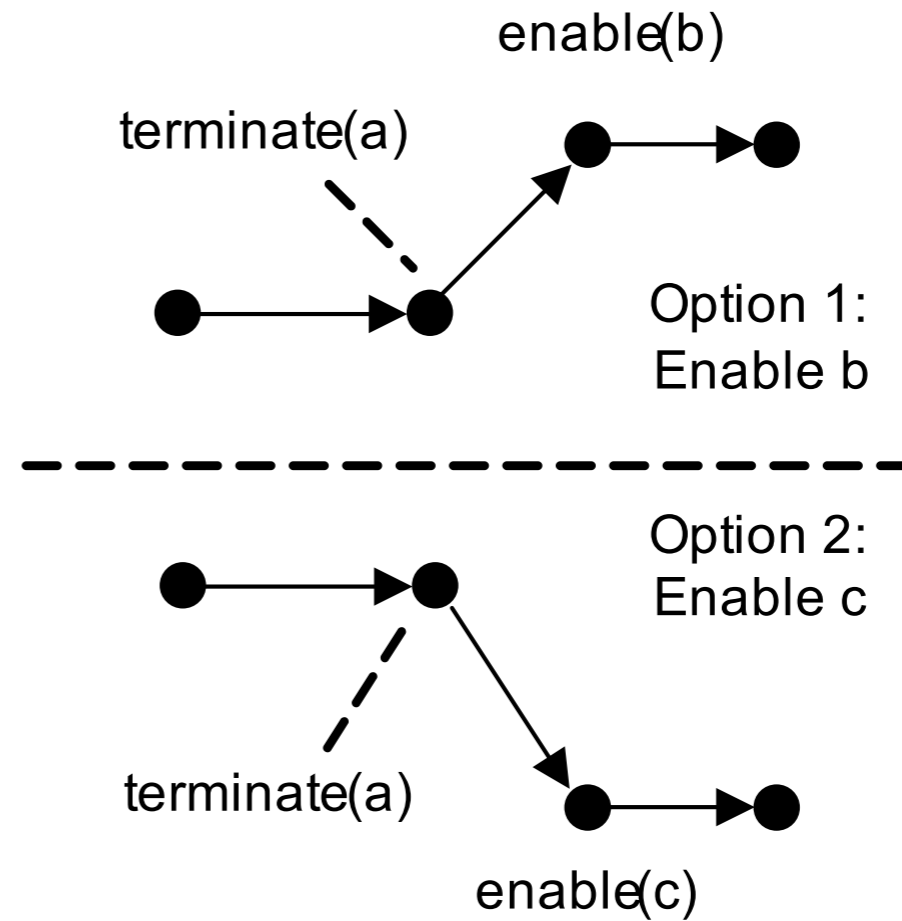
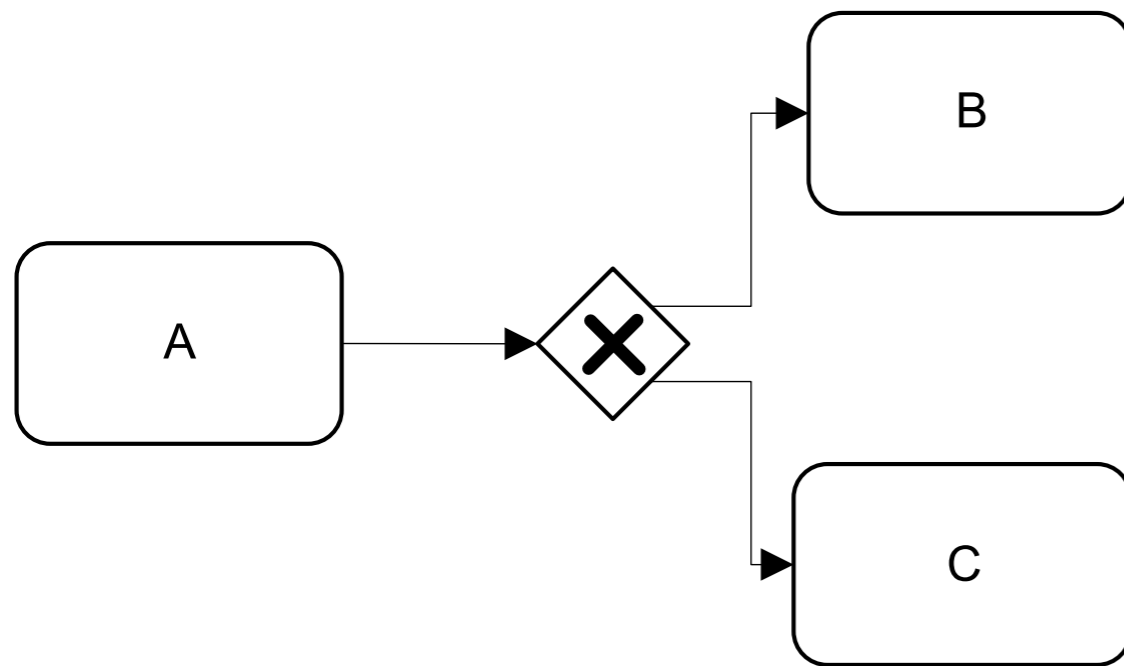


#4 Exclusive choice

Description: A point in the workflow process where, based on a decision or workflow control data, one of several branches is chosen

Synonyms: XOR-split, conditional routing, switch, decision

#4 Exclusive choice



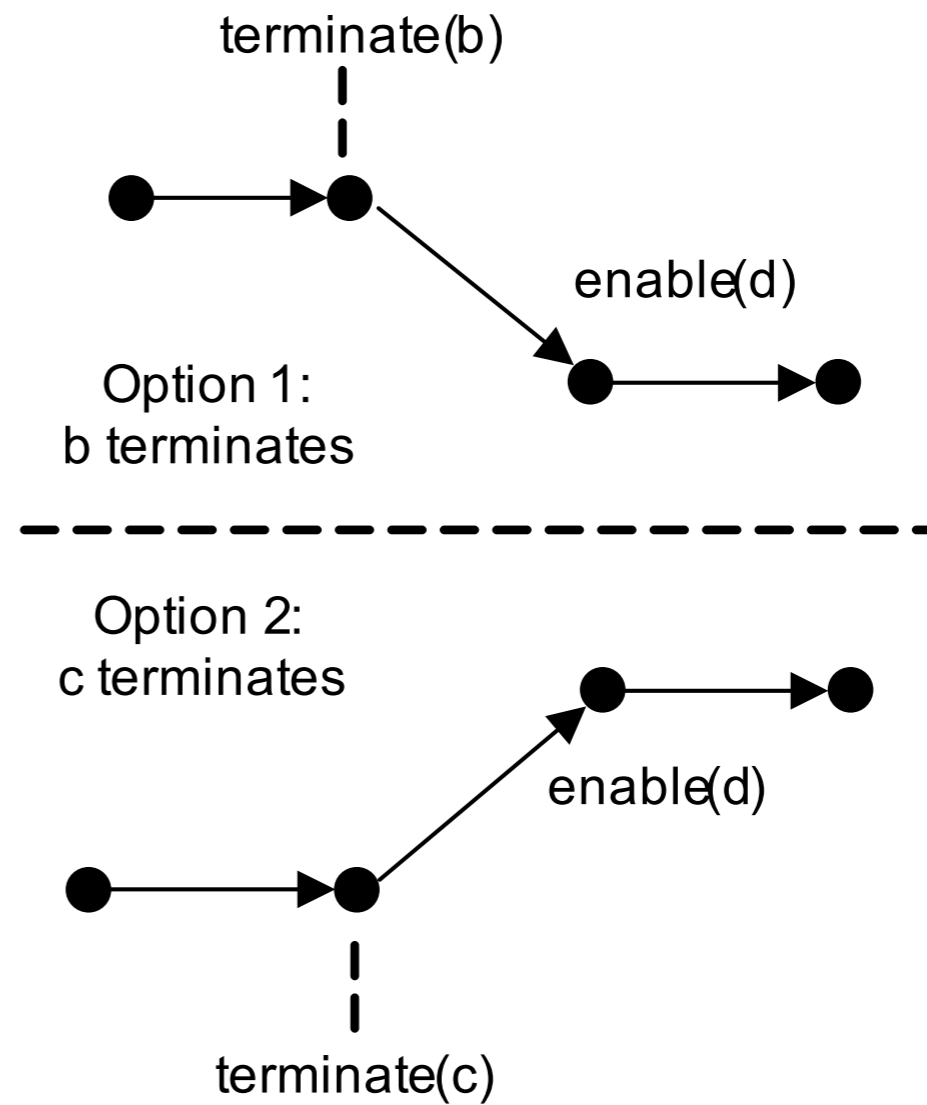
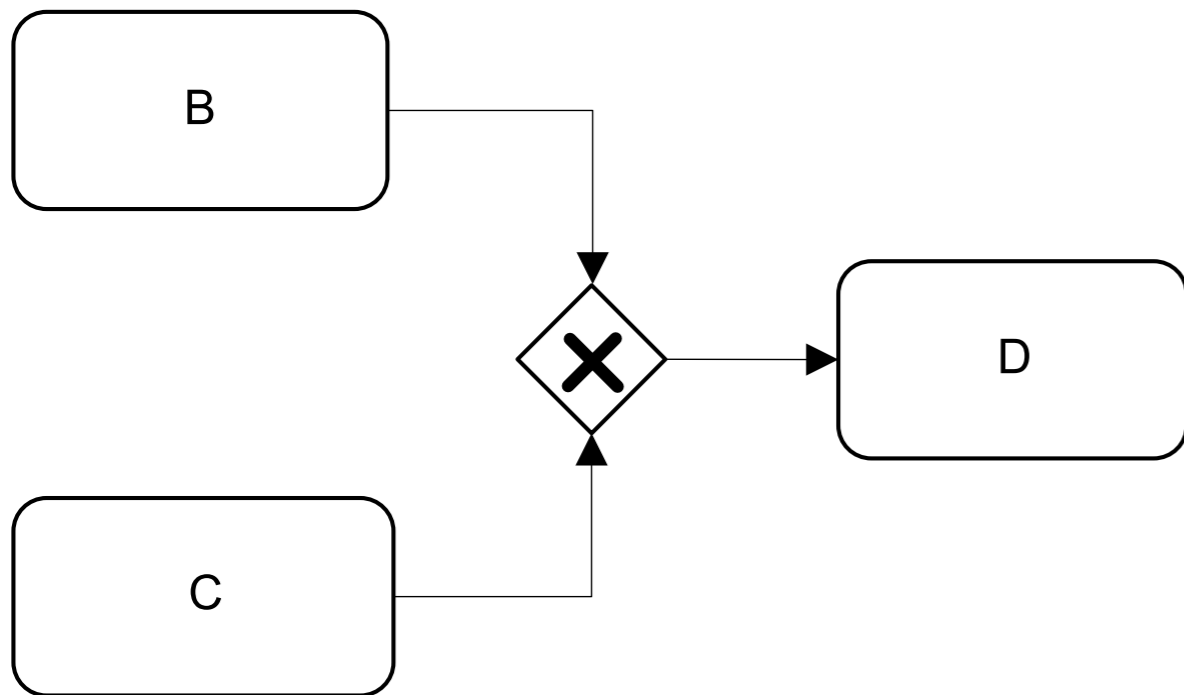
#5 Simple merge

Description: A point in the workflow process where two or more alternative branches come together without synchronization.

It is an assumption of this pattern that none of the alternative branches is ever executed in parallel (if this is not the case, see patterns #8-9)

Synonyms: XOR-join, asynchronous join, merge

#5 Simple merge



Basic Control Flow Patterns

- Pattern 1 (Sequence)
- Pattern 2 (Parallel Split)
- Pattern 3 (Synchronization)
- Pattern 4 (Exclusive Choice)
- Pattern 5 (Simple Merge)

Advanced Branching and Synchronization Patterns

- Pattern 6 (Multi-choice)
- Pattern 7 (Synchronizing Merge)
- Pattern 8 (Multi-merge)
- Pattern 9 (Discriminator)

Structural Patterns

- Pattern 10 (Arbitrary Cycles)
- Pattern 11 (Implicit Termination)

Cancellation Patterns

- Pattern 19 (Cancel Activity)
- Pattern 20 (Cancel Case)

State-based Patterns

- Pattern 16 (Deferred Choice)
- Pattern 17 (Interleaved Parallel Routing)
- Pattern 18 (Milestone)

Patterns involving Multiple Instances

- Pattern 12 (Multiple Instances Without Synchronization)
- Pattern 13 (Multiple Instances With a Prior Design Time Knowledge)
- Pattern 14 (Multiple Instances With a Prior Runtime Knowledge)
- Pattern 15 (Multiple Instances Without a Prior Runtime Knowledge)

WFP #6-9

More advanced patterns for branching and synchronization

Quite common in real-life business scenarios

Do not find straightforward support in workflow engines

#6 Multi-choice

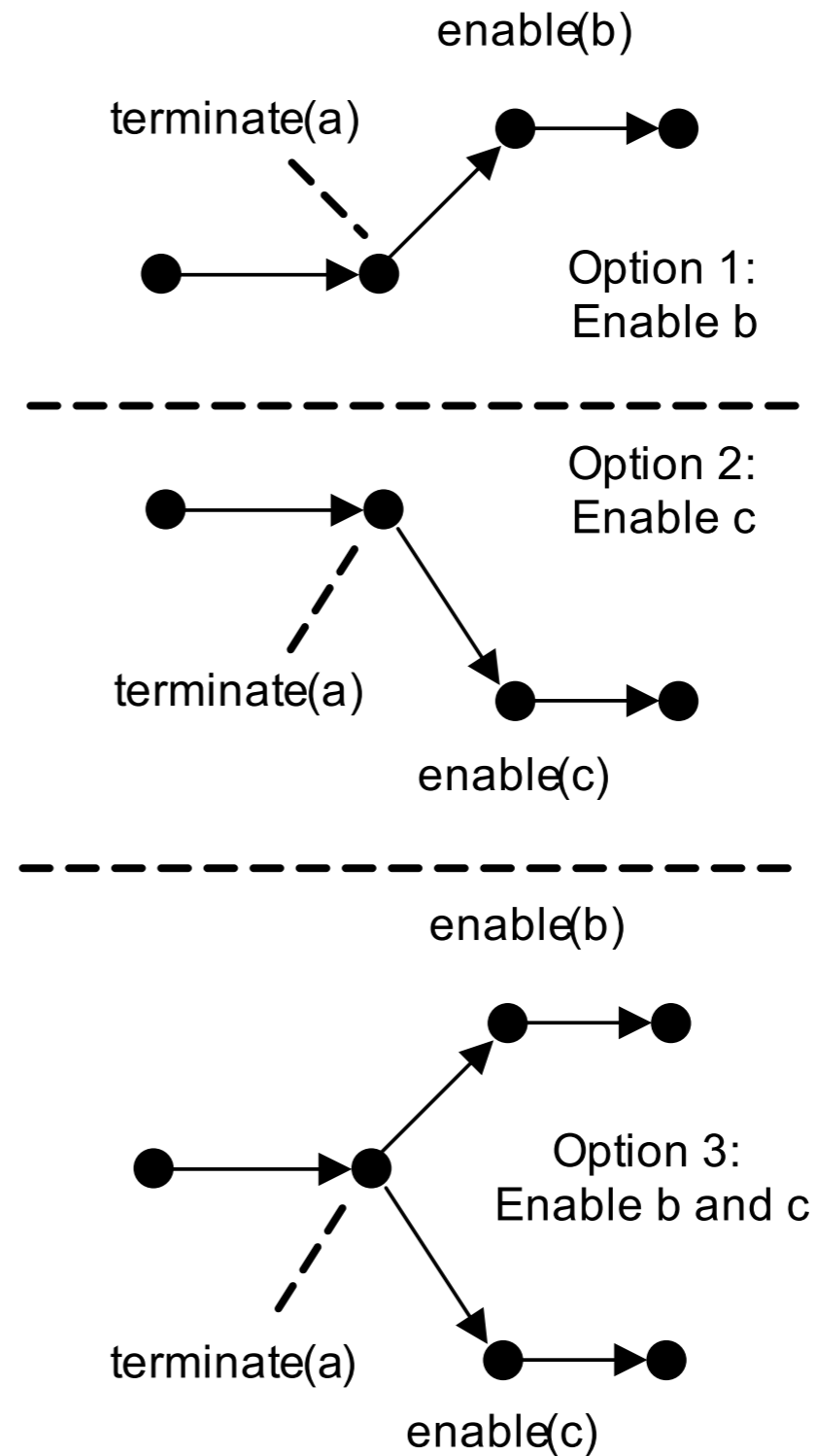
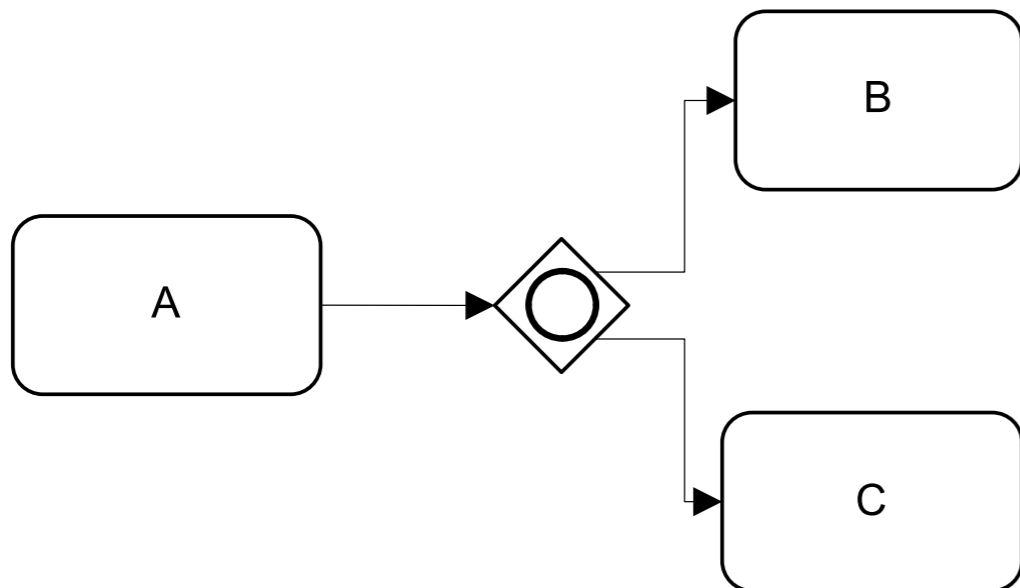
Description: A point in the workflow process where, based on a decision or workflow control data, a number of branches are chosen (sometimes required that at least one is chosen)

Synonyms: Conditional routing, selection, OR-split

Different from Exclusive choice (#4), where exactly one of the alternatives is selected and executed

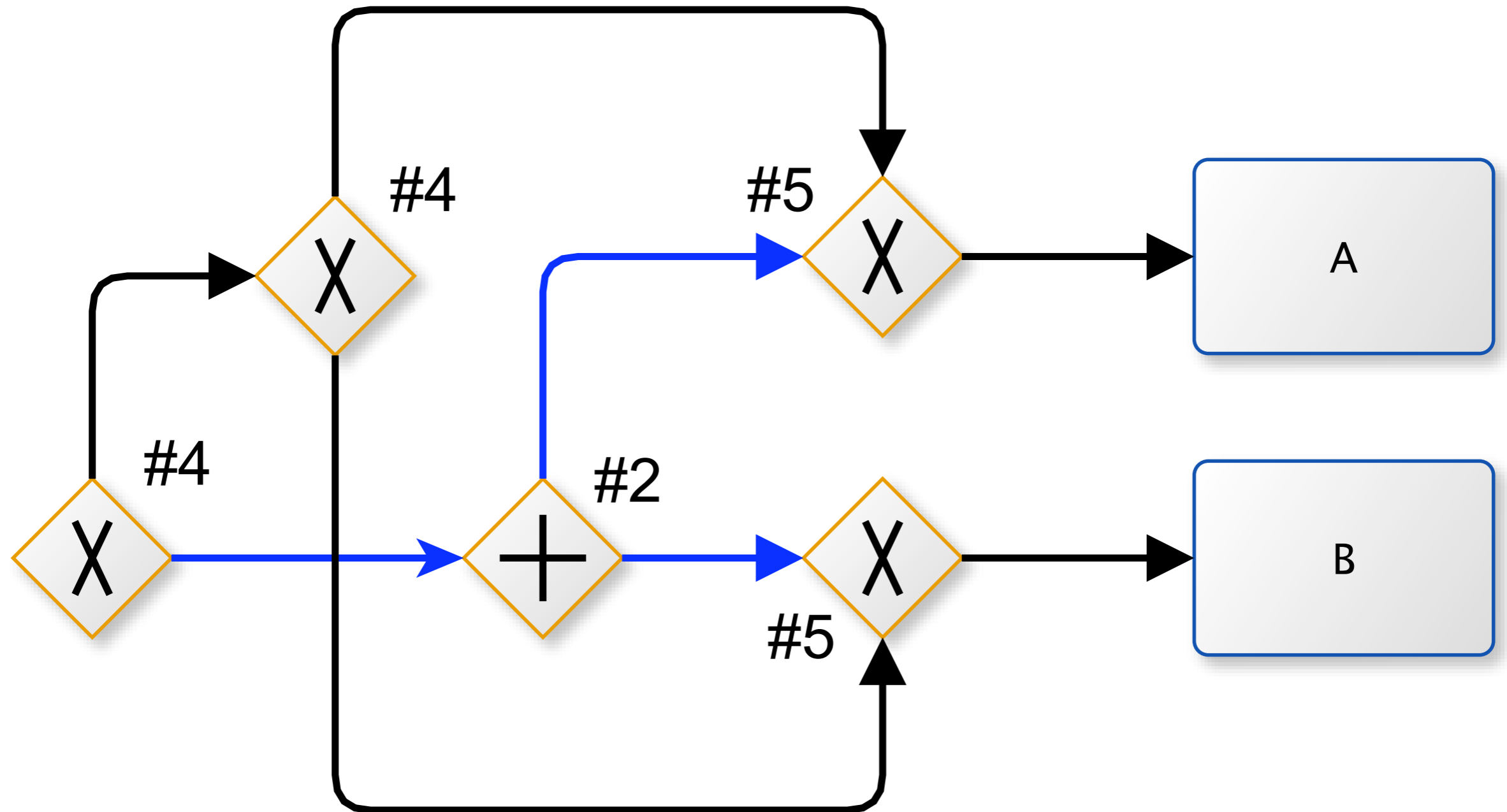
Can be implemented using AND-split and XOR-split

#6 Multi-choice



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

#6 Multi-choice: encoded



#6 Question time

If there are **N** outgoing edges from the gateway,
how many options do we have to consider?

$$2^N - 1$$

#7 Synchronizing merge

Description: A point in the workflow process where multiple paths converge into one single thread.

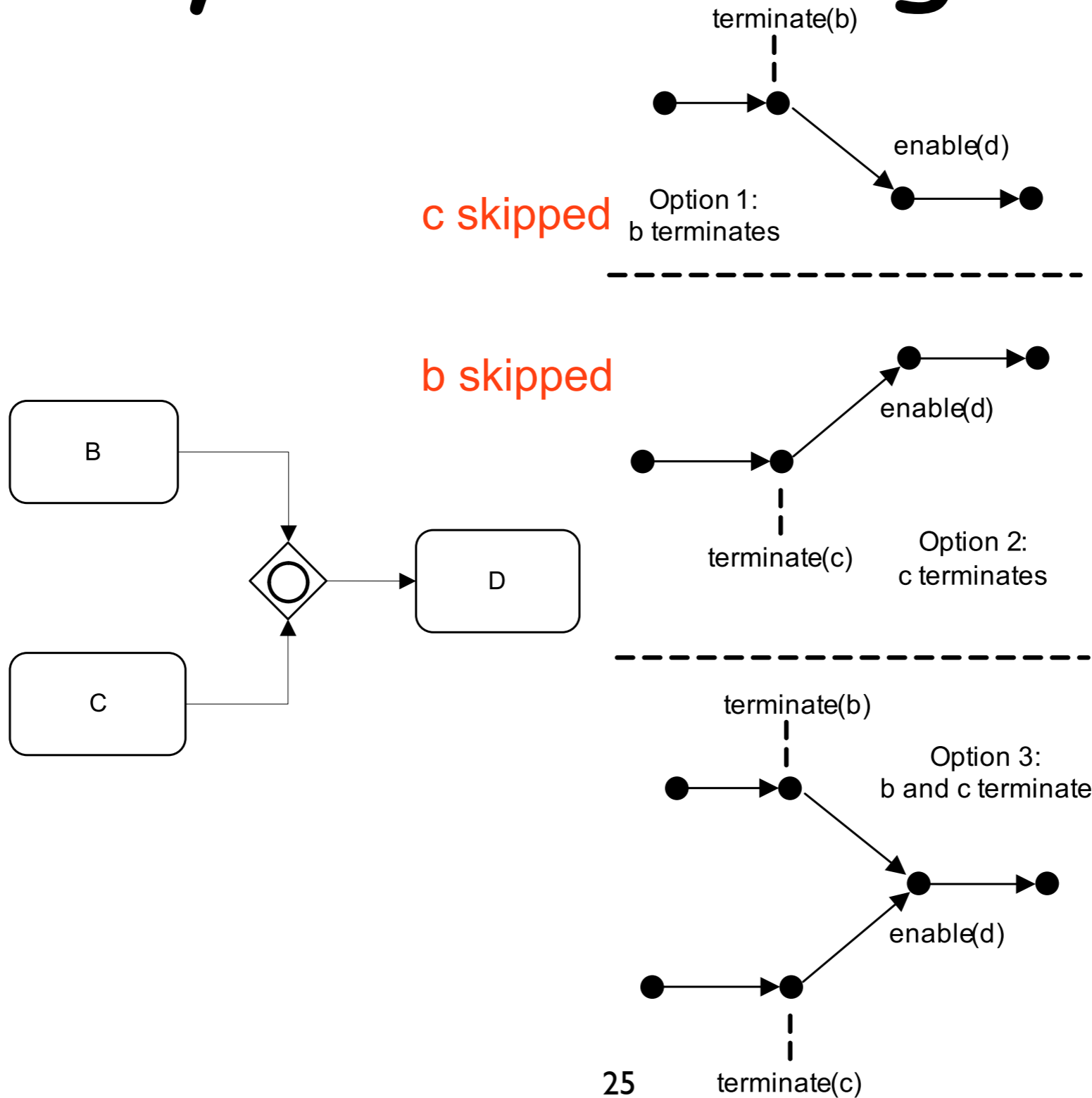
If more than one path is taken, synchronization of the active threads needs to take place.

If only one path is taken, the alternative branches should reconverge without synchronization.

Synonyms: Synchronizing join, OR-join wait-for-all

It is an assumption of this pattern that a branch that has already been activated cannot be activated again while the merge is still waiting for other branches to complete

#7 Synchronizing merge



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

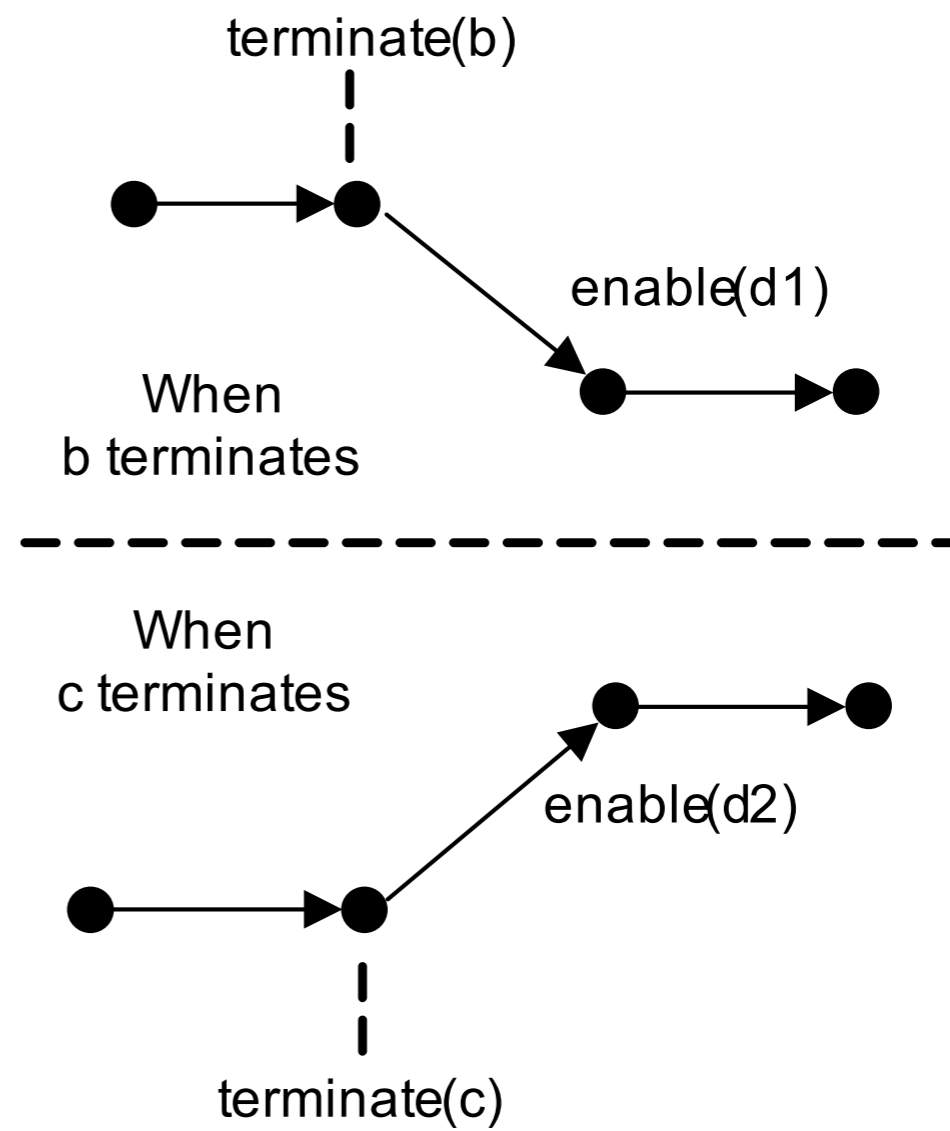
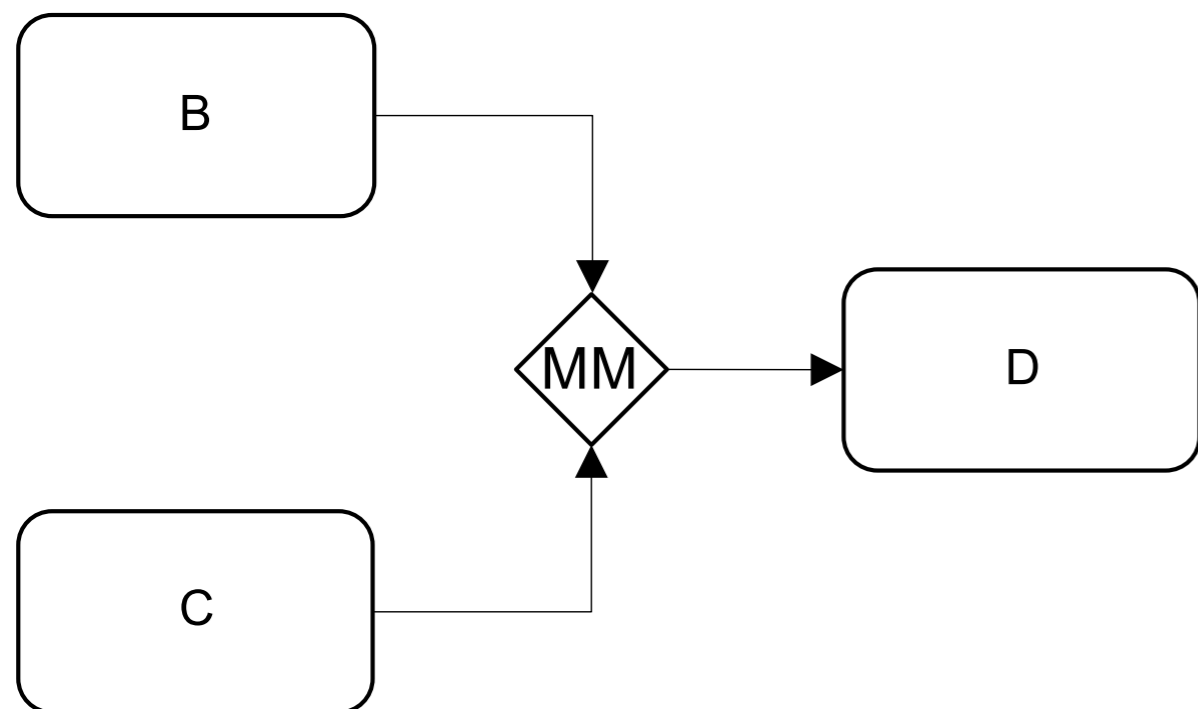
#8 Multi-merge

Description: A point in the workflow process where two or more branches reconverge without synchronization.

If more than one branch gets activated, possibly concurrently, the activity following the merge is started for every activation of every incoming branch

Synonyms: OR-join every-time

#8 Multi-merge



#9 Discriminator

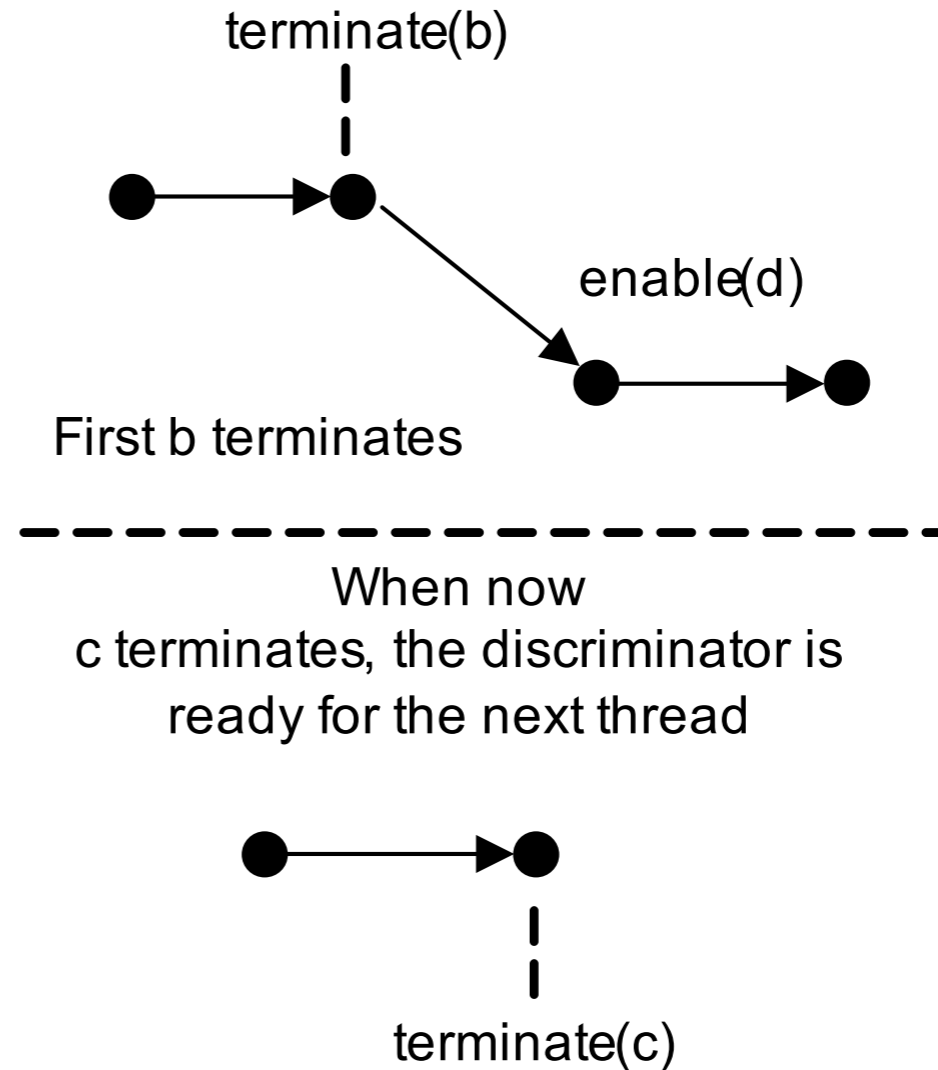
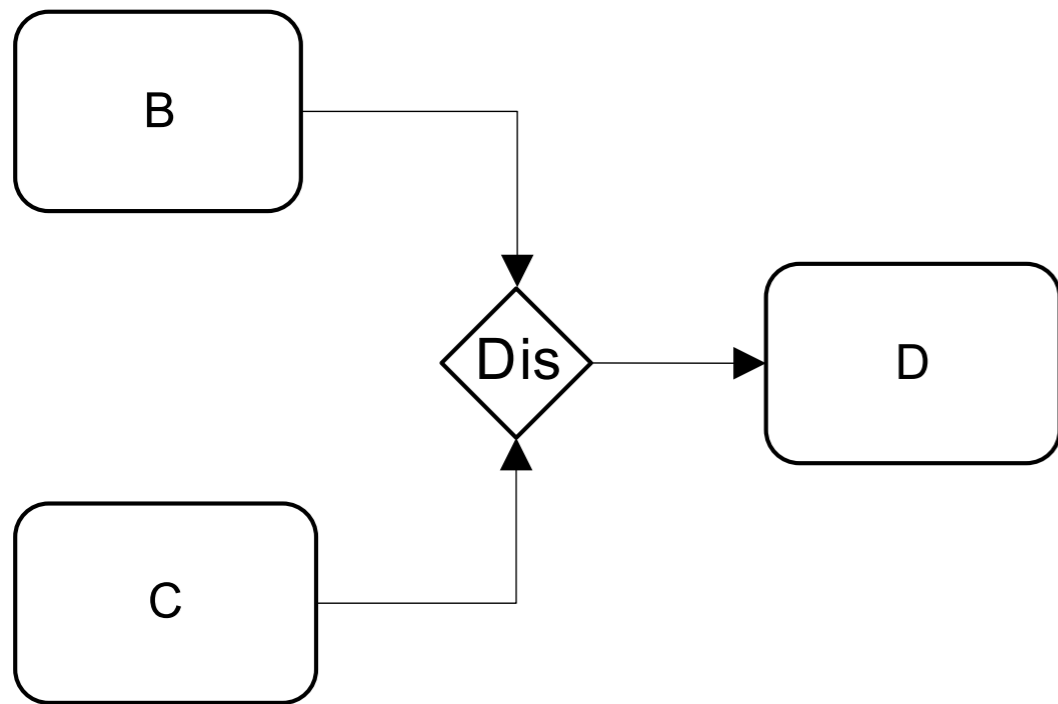
Description: A point in a workflow process that waits for one of the incoming branches to complete before activating the subsequent activity.

Then, it waits for all remaining branches to complete and ignores them.

Once all incoming branches have been triggered, it resets so that it can be triggered again (which makes it suitable for being used inside a loop)

Synonyms: OR-join first-come

#9 Discriminator



(#9bis)

N-out-of-M join

Description: It is a point in a workflow process where M parallel paths converge into one.

The subsequent activity is initiated after $N \leq M$ paths have completed.

Completion of all remaining paths is ignored.

Once all incoming branches have been triggered, it resets so that it can be triggered again

(which is important when put inside a loop)

The N-out-of-M join is a generalization of the discriminator pattern (that coincides with 1-out-of-M join)

Basic Control Flow Patterns

- Pattern 1 (Sequence)
- Pattern 2 (Parallel Split)
- Pattern 3 (Synchronization)
- Pattern 4 (Exclusive Choice)
- Pattern 5 (Simple Merge)

Advanced Branching and Synchronization Patterns

- Pattern 6 (Multi-choice)
- Pattern 7 (Synchronizing Merge)
- Pattern 8 (Multi-merge)
- Pattern 9 (Discriminator)

Structural Patterns

- Pattern 10 (Arbitrary Cycles)
- Pattern 11 (Implicit Termination)

Cancellation Patterns

- Pattern 19 (Cancel Activity)
- Pattern 20 (Cancel Case)

State-based Patterns

- Pattern 16 (Deferred Choice)
- Pattern 17 (Interleaved Parallel Routing)
- Pattern 18 (Milestone)

Patterns involving Multiple Instances

- Pattern 12 (Multiple Instances Without Synchronization)
- Pattern 13 (Multiple Instances With a Priori Design Time Knowledge)
- Pattern 14 (Multiple Instances With a Priori Runtime Knowledge)
- Pattern 15 (Multiple Instances Without a Priori Runtime Knowledge)

WFP #10-11

Some restrictions are often imposed on the workflow structure (loops are not allowed, single exit point,...)

Not always natural (if not annoying)
from the modelling point of view

For example, structured cycles can have only one entry point and only one exit point and cannot be interleaved (like the WHILE vs GOTO controversy)

But the removal of arbitrary cycles can lead to workflows that are much harder to interpret

#10 Arbitrary cycles

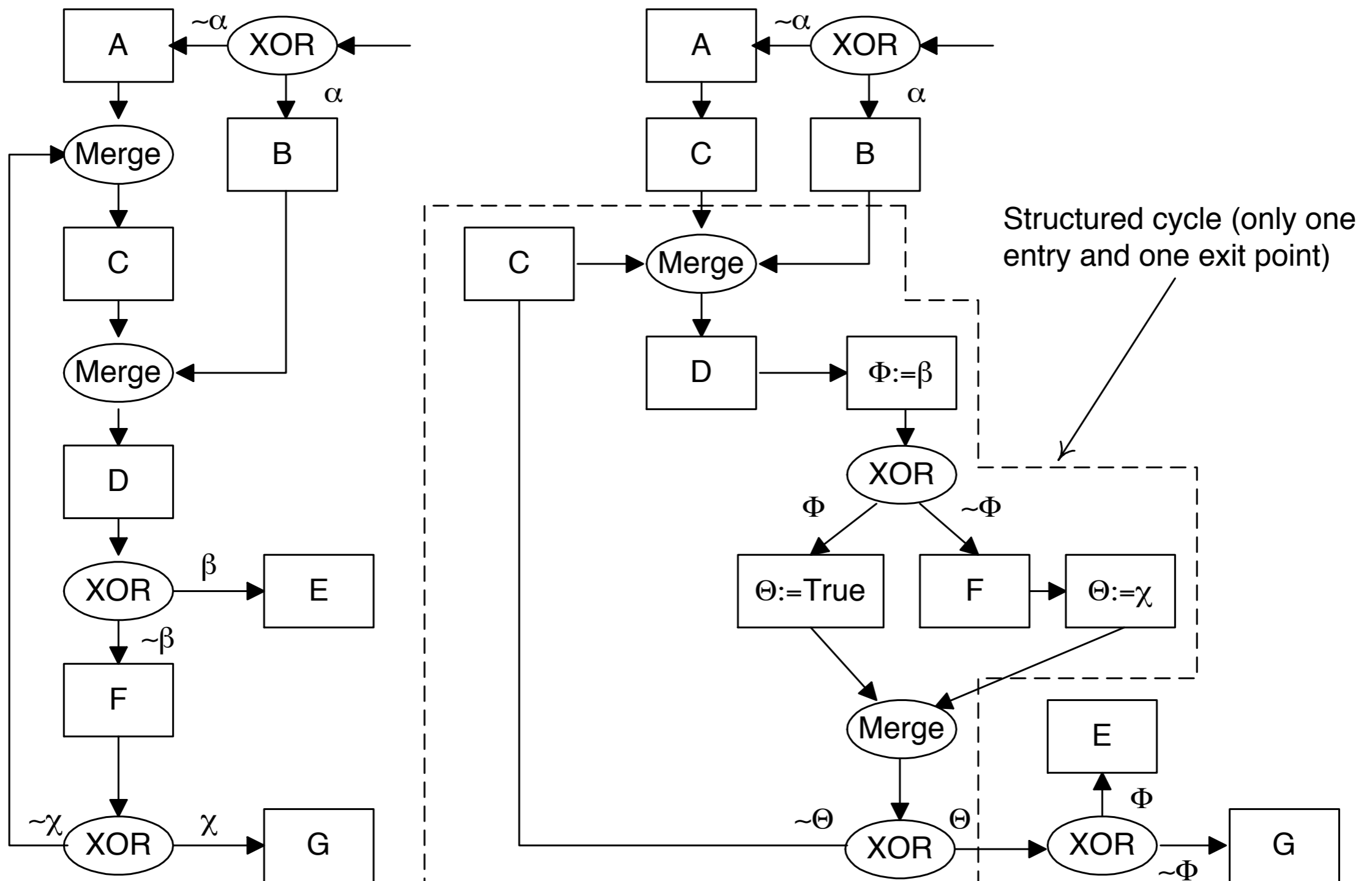
Description: It is a point in a workflow process where one or more activities can be done repeatedly

Synonyms: Loop, iteration, cycle

Often they are just expressed in terms of XOR-split and XOR-join (no dedicated element for cycles)

Arbitrary cycles can often be converted to structured cycles (single entry / exit) unless they contain advanced patterns

#10 Arbitrary cycles



#11 Implicit termination

Description: A given (sub)process should be terminated when there is nothing else to be done.

Most workflow engines terminate the process when an explicit final node is reached, possibly aborting any ongoing activity.

If there are many states in which the process can terminate, then termination is implicit.

Conversion to workflow with only one terminating node is not always possible

Basic Control Flow Patterns

- Pattern 1 (Sequence)
- Pattern 2 (Parallel Split)
- Pattern 3 (Synchronization)
- Pattern 4 (Exclusive Choice)
- Pattern 5 (Simple Merge)

Advanced Branching and Synchronization Patterns

- Pattern 6 (Multi-choice)
- Pattern 7 (Synchronizing Merge)
- Pattern 8 (Multi-merge)
- Pattern 9 (Discriminator)

Structural Patterns

- Pattern 10 (Arbitrary Cycles)
- Pattern 11 (Implicit Termination)

Cancellation Patterns

- Pattern 19 (Cancel Activity)
- Pattern 20 (Cancel Case)

State-based Patterns

- Pattern 16 (Deferred Choice)
- Pattern 17 (Interleaved Parallel Routing)
- Pattern 18 (Milestone)

Patterns involving Multiple Instances

- Pattern 12 (Multiple Instances Without Synchronization)
- Pattern 13 (Multiple Instances With a Priori Design Time Knowledge)
- Pattern 14 (Multiple Instances With a Priori Runtime Knowledge)
- Pattern 15 (Multiple Instances Without a Priori Runtime Knowledge)

WFP #12-15

The phenomenon of multiple instances corresponds to multiple threads that refer to a shared activity definition

An activity in a workflow process can have more than one running active instance at the same time

#12 Multiple instances without synchronization

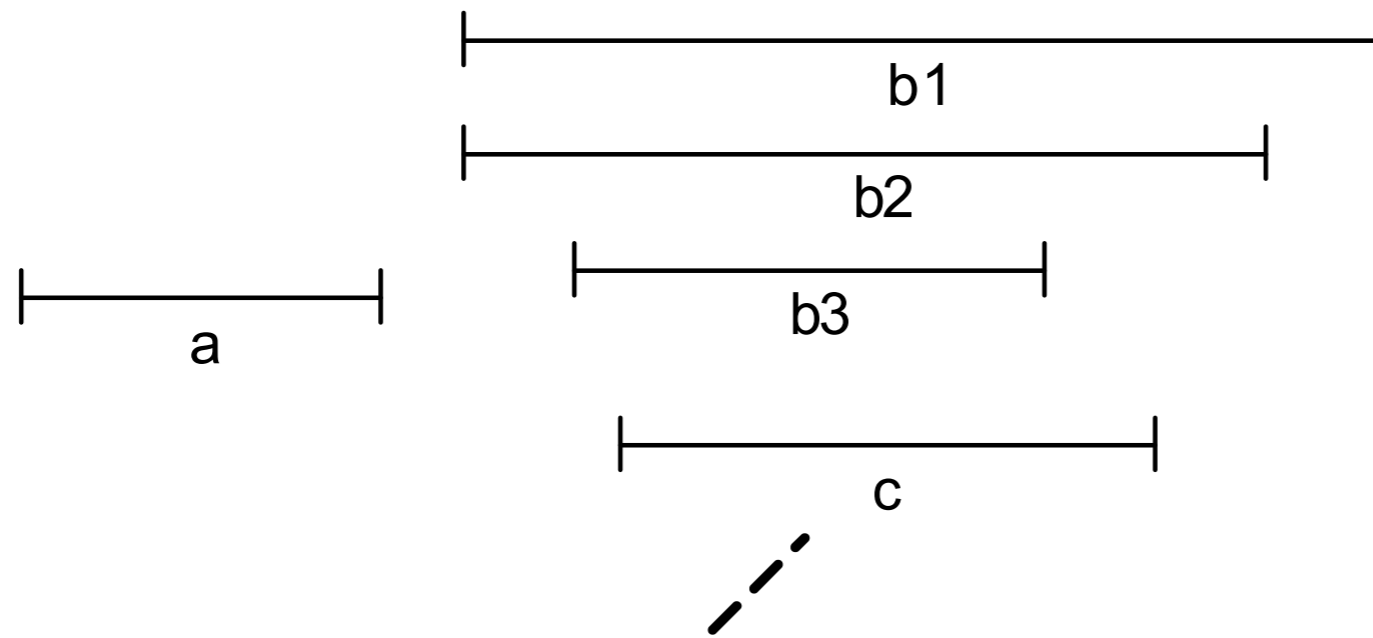
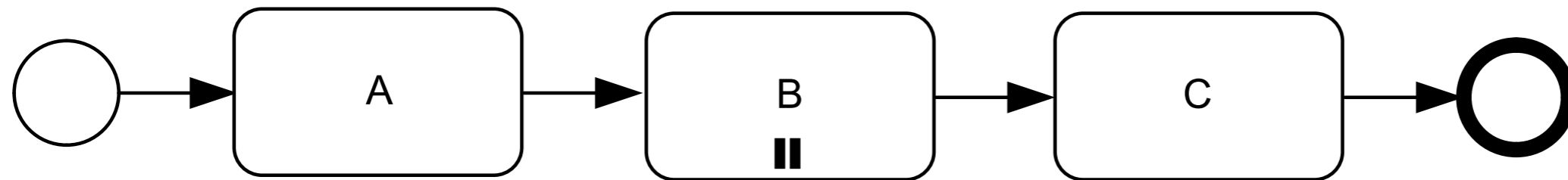
Description: Multiple, independent instances of an activity can be created within the context of a single case.

Synonyms: Multi-threading without synchronization,
spawn off facility

Easy to support
(the problem is not to generate instances,
but to coordinate them)

Can cause many problems from the point of view of
sequencing, termination, soundness...

#12 Multiple instances without synchronization



c enabled immediately after the last b has started (no synchronization)

#13 Multiple instances with a priori design knowledge

Description: An activity is enabled multiple times for handling one case.

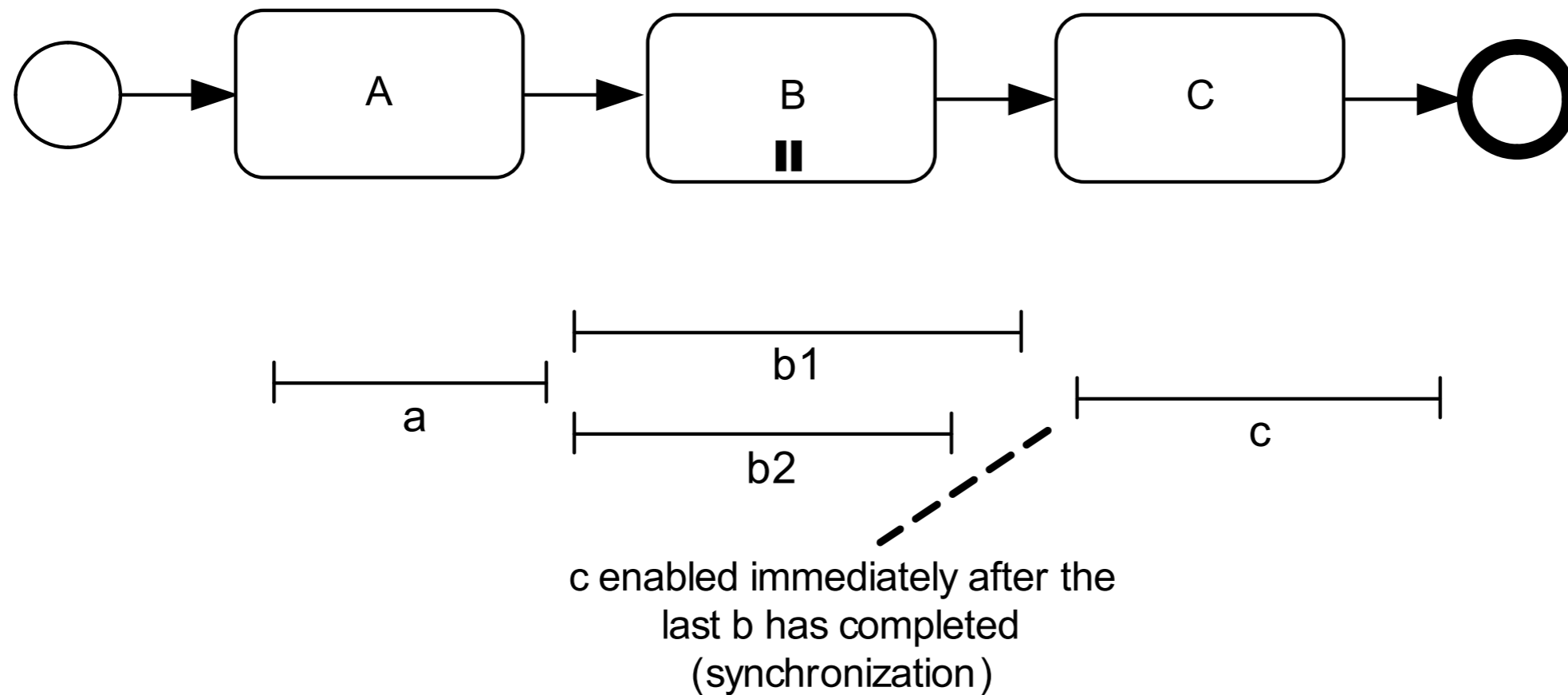
The number of instances for that case is known at design time.

Once all instances have been completed, some other activity needs to be started.

Synonyms: -

Easily implementable
by replicating the task in the process model

#13 Multiple instances with a priori design knowledge



#14 Multiple instances with a priori runtime knowledge

Description: An activity is enabled multiple times for handling one case.

The number of instances for a given case may depend on the availability of resources and will be known only at runtime, before any instance is created. Once all instances have been completed, some other activity needs to be started.

Scarcely supported in workflow engines.
We cannot simply replicate the activity

#15 Mult. instances without a priori runtime knowledge

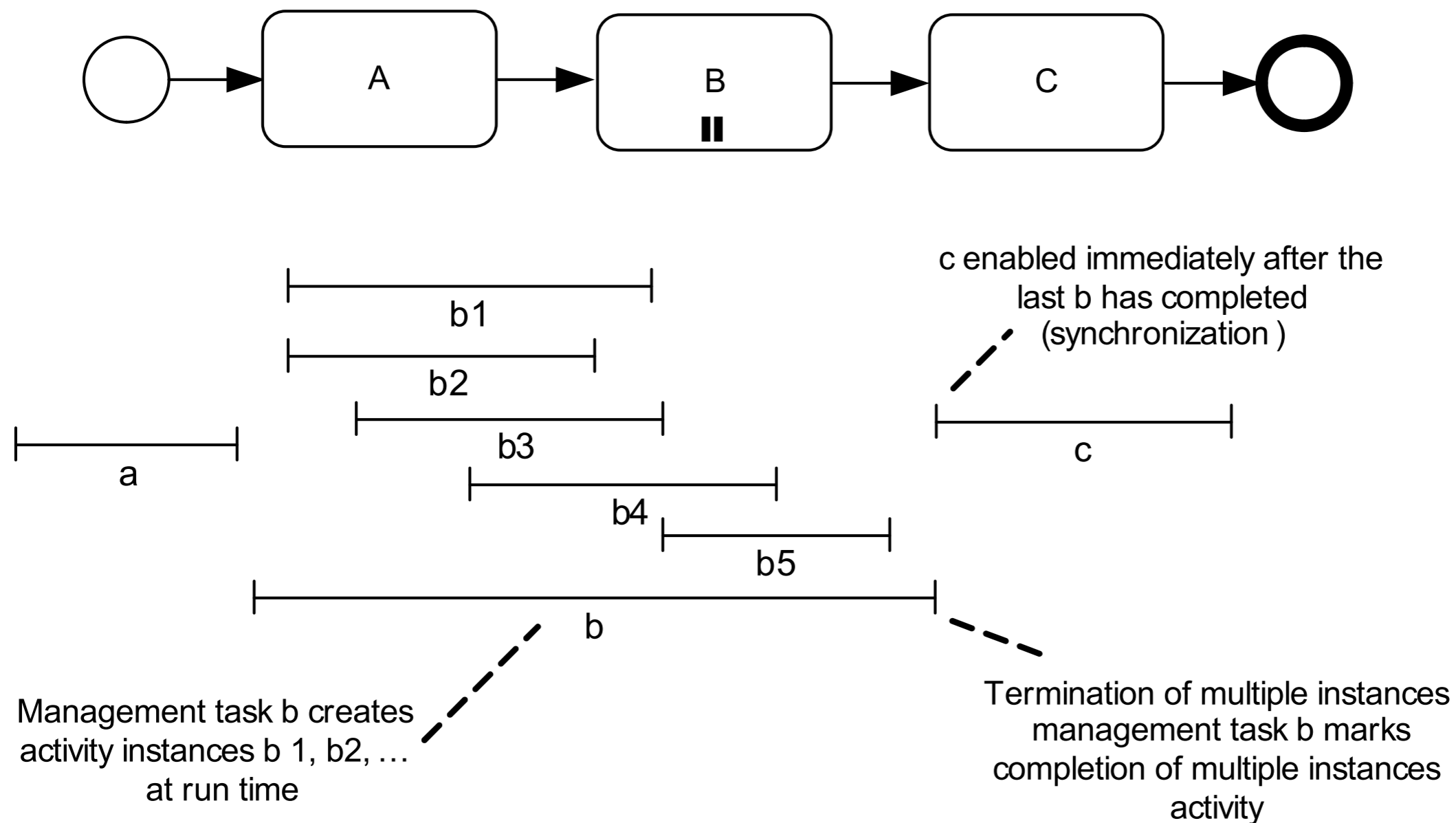
Description: An activity is enabled multiple times for handling one case.

The number of instances for a given case is known neither at design time, nor at runtime before any instance is created.

Once all instances have been completed, some other activity needs to be started.

The difference with #14 is that even while some of the instances are executed or completed, new ones can be created

#15 Mult. instances without a priori runtime knowledge



Basic Control Flow Patterns

- Pattern 1 (Sequence)
- Pattern 2 (Parallel Split)
- Pattern 3 (Synchronization)
- Pattern 4 (Exclusive Choice)
- Pattern 5 (Simple Merge)

Advanced Branching and Synchronization Patterns

- Pattern 6 (Multi-choice)
- Pattern 7 (Synchronizing Merge)
- Pattern 8 (Multi-merge)
- Pattern 9 (Discriminator)

Structural Patterns

- Pattern 10 (Arbitrary Cycles)
- Pattern 11 (Implicit Termination)

Cancellation Patterns

- Pattern 19 (Cancel Activity)
- Pattern 20 (Cancel Case)

State-based Patterns

- Pattern 16 (Deferred Choice)
- Pattern 17 (Interleaved Parallel Routing)
- Pattern 18 (Milestone)

Patterns involving Multiple Instances

- Pattern 12 (Multiple Instances Without Synchronization)
- Pattern 13 (Multiple Instances With a Priori Design Time Knowledge)
- Pattern 14 (Multiple Instances With a Priori Runtime Knowledge)
- Pattern 15 (Multiple Instances Without a Priori Runtime Knowledge)

WFP #16-18

In real workflows, most process instances are in a state awaiting processing rather than being processed

A different notion of “state” is considered with respect to ordinary programming: moments of choice can depend on data or decisions

The state between activities can be explicitly/implicitly represented or not

#16 Deferred choice

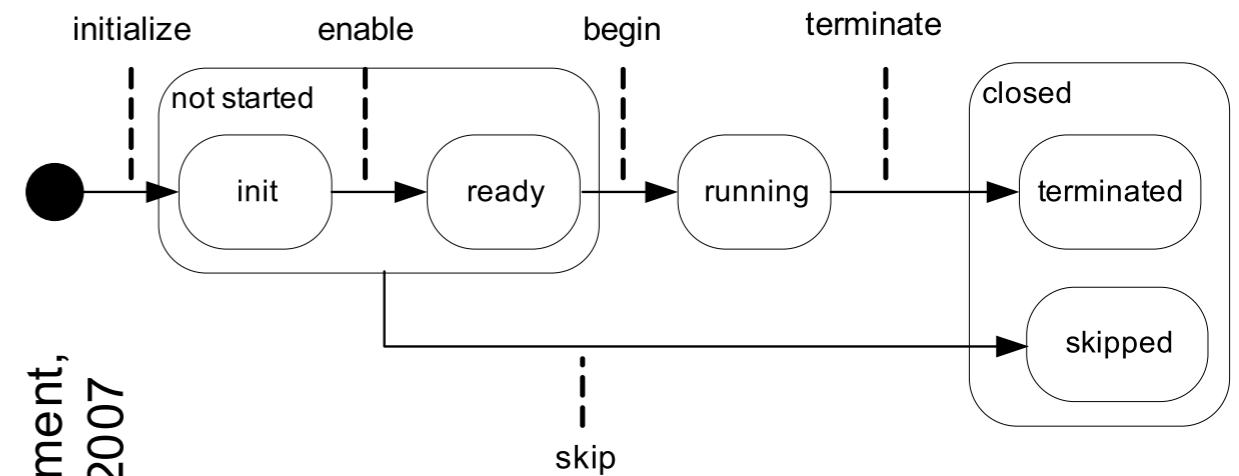
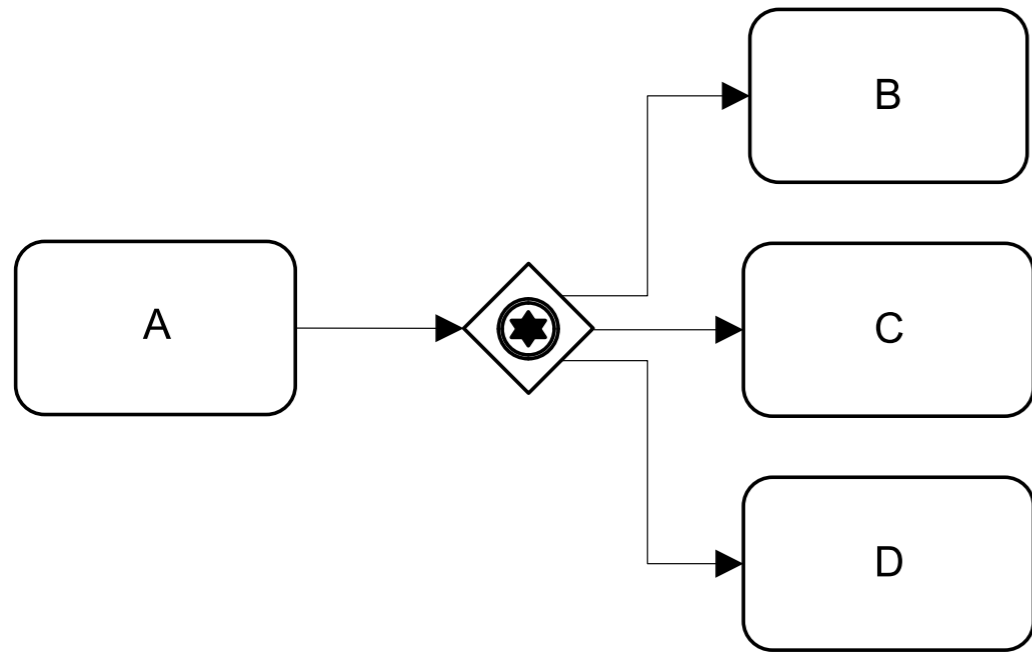
Description: A point in the workflow process where one of several branches is chosen.

Synonyms: External choice, deferred XOR-split

In contrast with XOR-split, the choice is not made explicit, but several alternatives are offered to the environment

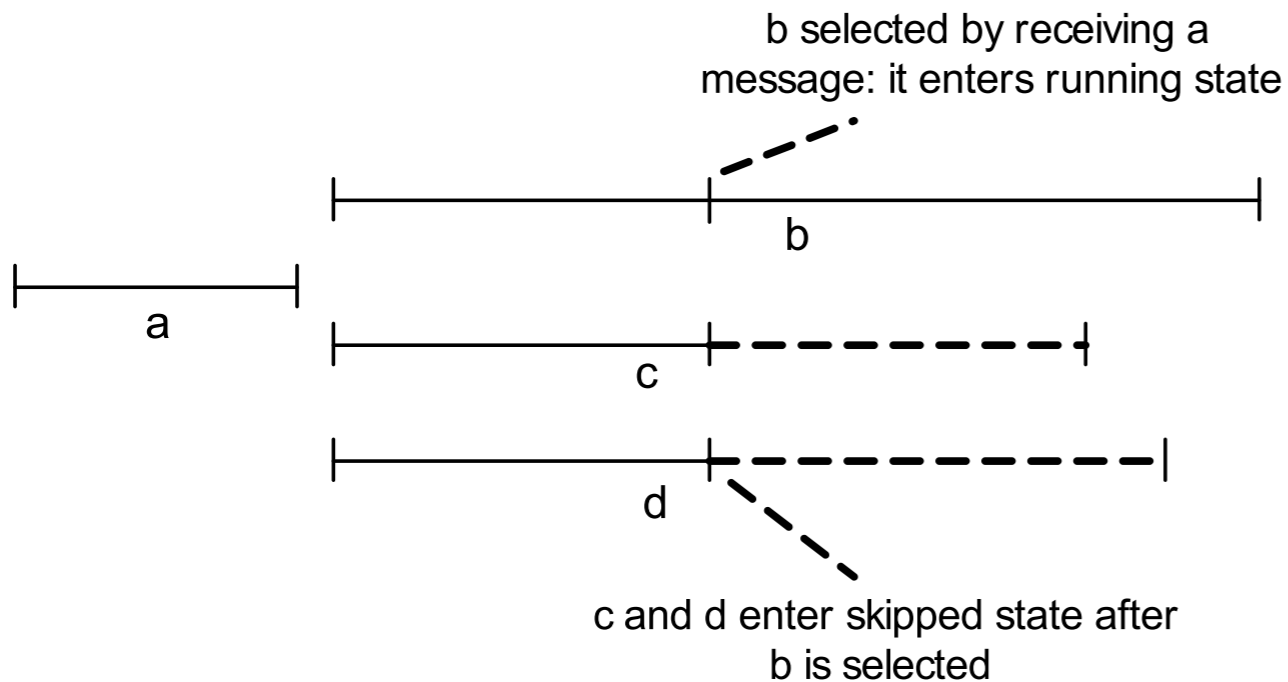
In contrast to the AND-split, only one of the alternatives is selected (and the others are withdrawn)

#16 Deferred choice



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

A closer look at the phases of an activity

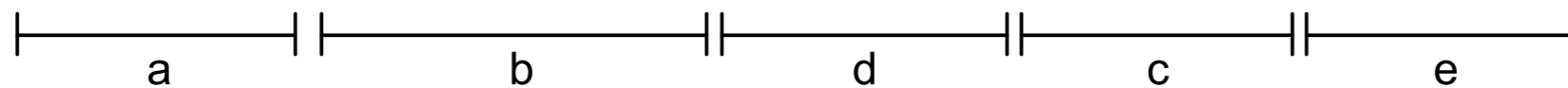
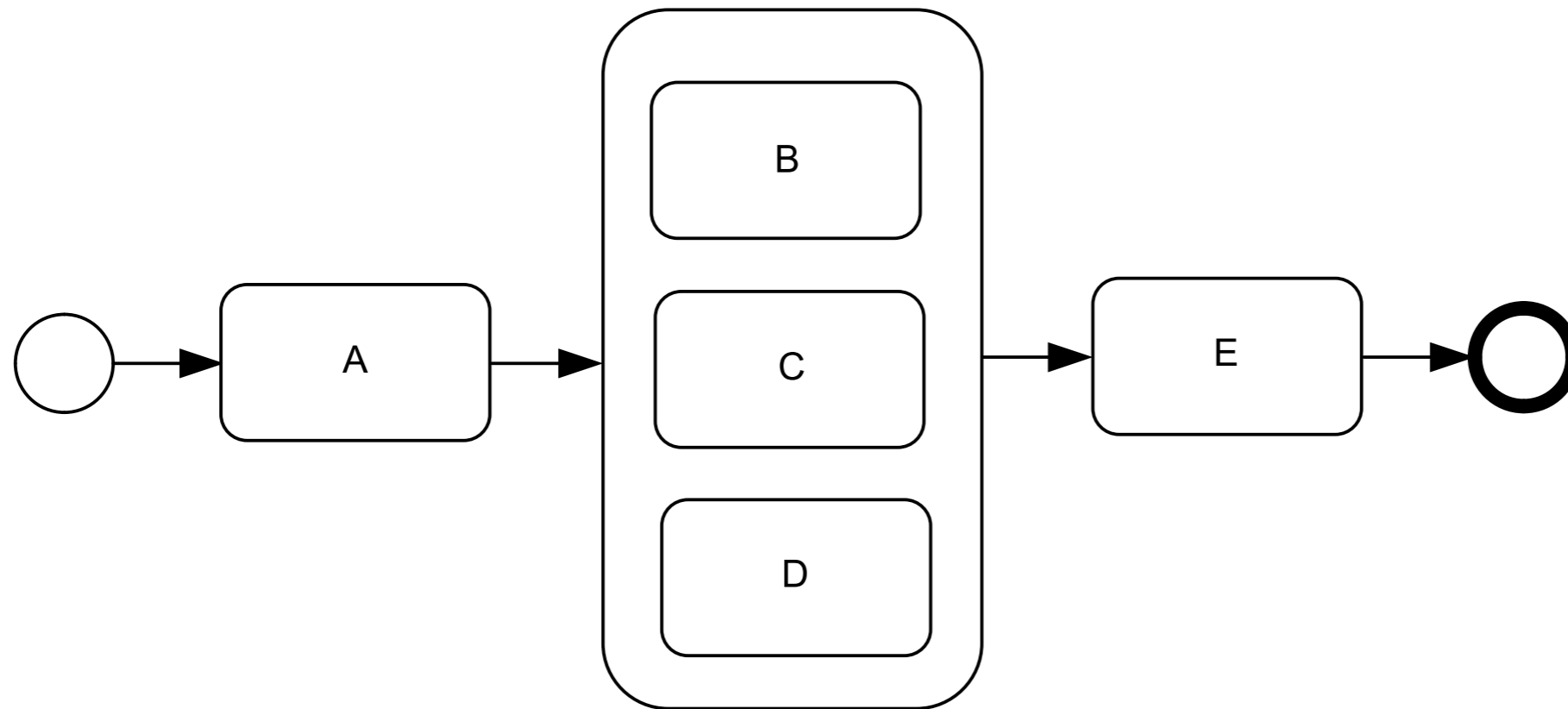


#17 Interleaved parallel routing

Description: A set of activities is executed in an arbitrary order. Each activity in the set is executed, but the order is decided at runtime and no two activities are executed at the same moment.

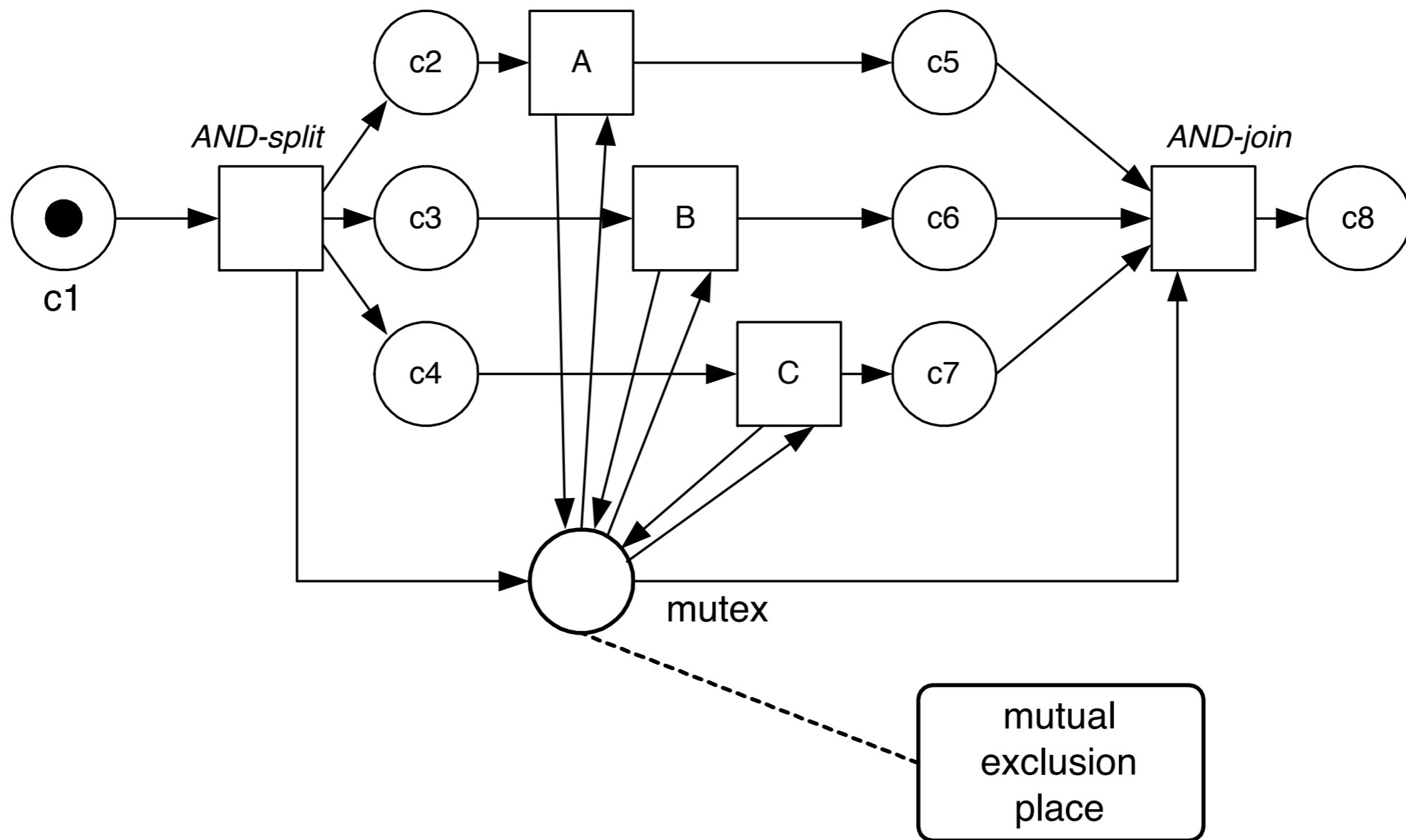
Synonyms: Unordered sequence, ad hoc activity, sequential execution without a priori design time knowledge

#17 Interleaved parallel routing

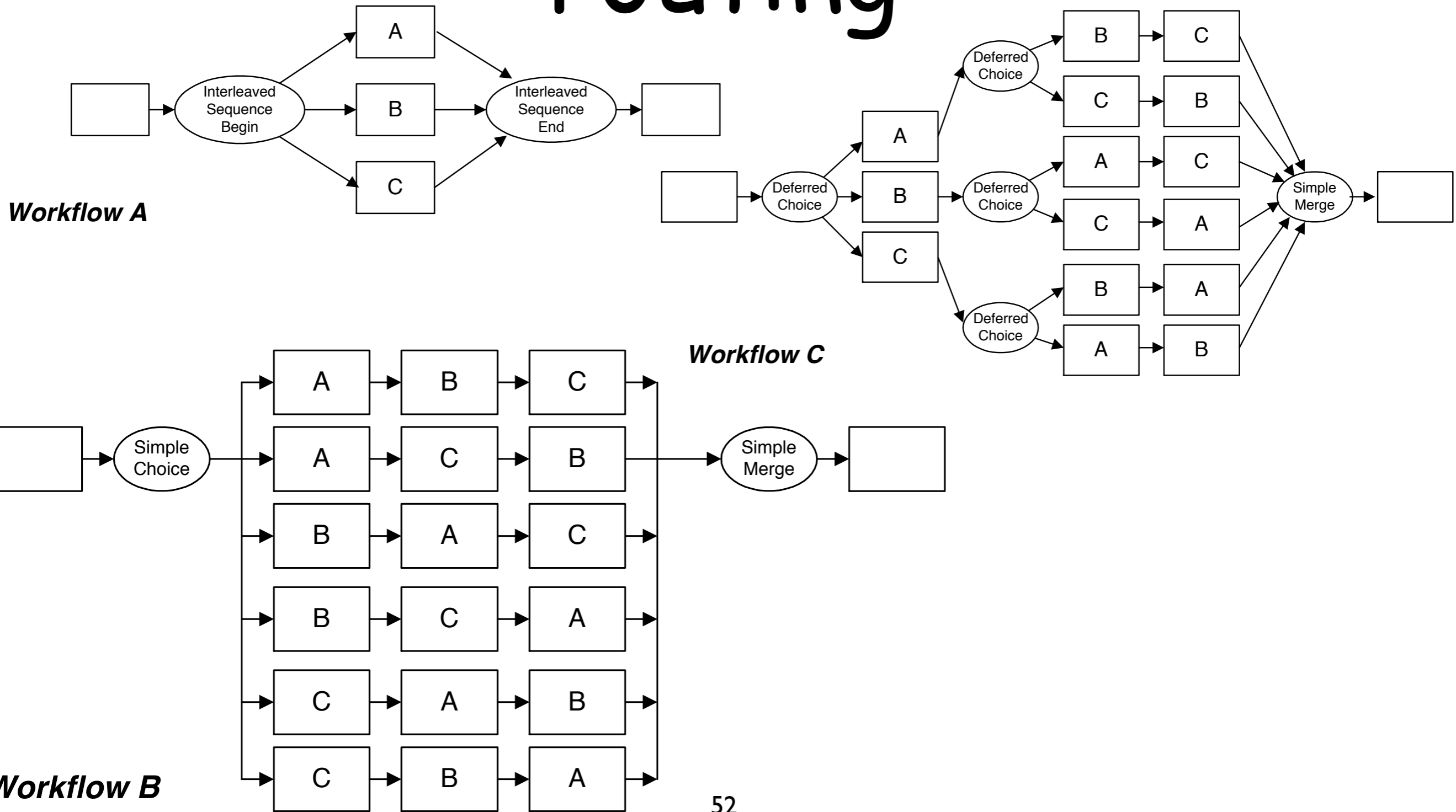


Any sequential ordering
of b,c,d is valid

#17 Interleaved parallel routing



#17 Interleaved parallel routing

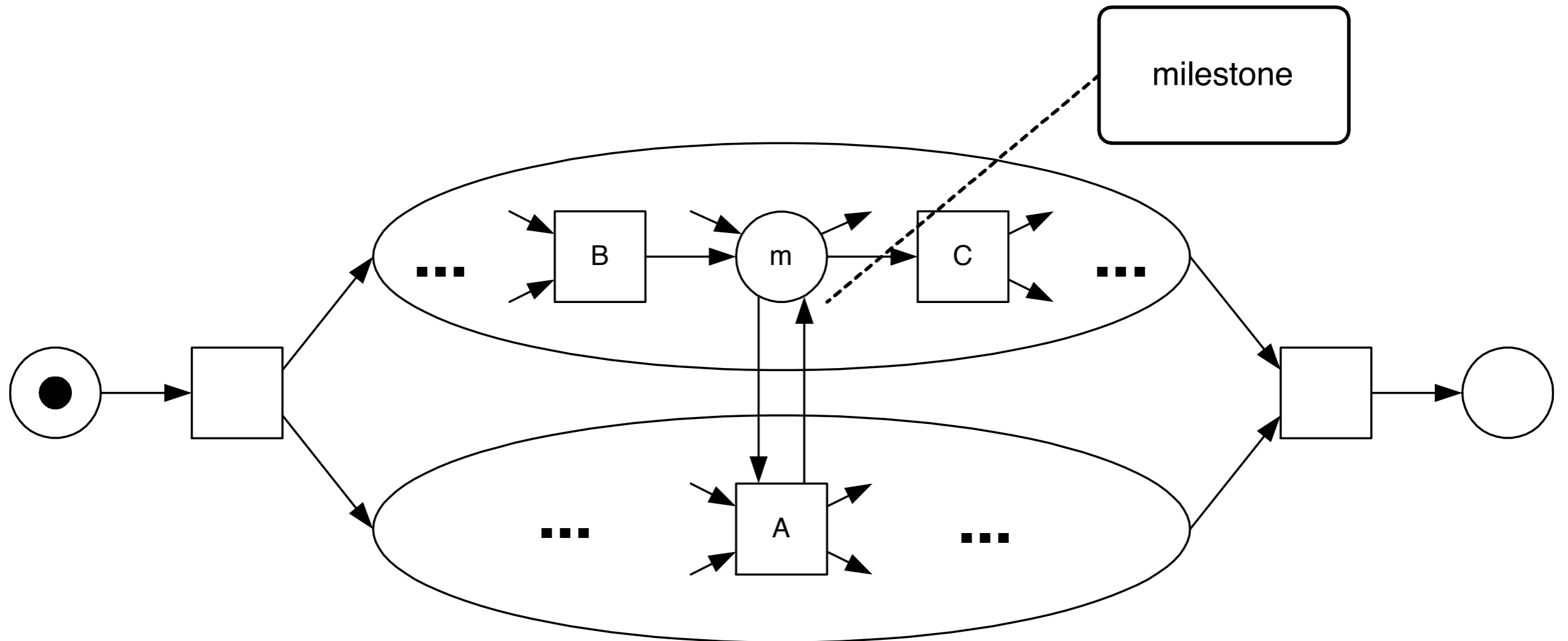


#18 Milestone

Description: The enabling of an activity depends on the case being in a specified state (i.e. a milestone has been reached)

Synonyms: Test arc, deadline, state condition, withdraw message

#18 Milestone



Basic Control Flow Patterns

- Pattern 1 (Sequence)
- Pattern 2 (Parallel Split)
- Pattern 3 (Synchronization)
- Pattern 4 (Exclusive Choice)
- Pattern 5 (Simple Merge)

Advanced Branching and Synchronization Patterns

- Pattern 6 (Multi-choice)
- Pattern 7 (Synchronizing Merge)
- Pattern 8 (Multi-merge)
- Pattern 9 (Discriminator)

Structural Patterns

- Pattern 10 (Arbitrary Cycles)
- Pattern 11 (Implicit Termination)

Cancellation Patterns

- Pattern 19 (Cancel Activity)
- Pattern 20 (Cancel Case)

State-based Patterns

- Pattern 16 (Deferred Choice)
- Pattern 17 (Interleaved Parallel Routing)
- Pattern 18 (Milestone)

Patterns involving Multiple Instances

- Pattern 12 (Multiple Instances Without Synchronization)
- Pattern 13 (Multiple Instances With a Priori Design Time Knowledge)
- Pattern 14 (Multiple Instances With a Priori Runtime Knowledge)
- Pattern 15 (Multiple Instances Without a Priori Runtime Knowledge)

WFP #19-20

Cancellation patterns are very powerful mechanisms to clean up the pending activities

They offer ways to implement other patterns we have seen

They are also known as runtime patterns

#19 Cancel activity

Description: An enabled activity is disabled

Synonyms: Withdraw activity

The semantics of this pattern can become ill-defined if it is used in combination with multiple instances.

We assume that the cancellation of an activity refers to the cancellation of an instance of that activity.

#20 Cancel case

Description: A workflow instance is removed completely

Synonyms: Withdraw case

The cancellation of a case requires
all its activities are cancelled
so that the process comes to halt