Business Processes Modelling

MPB (6 cfu, 295AA)

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

13 - Workflow nets
We study some special kind of Petri nets, that are suitable models of workflows.
There are many, many variants of Petri nets
Example: Coloured nets (also called High-Level)

A coloured net is a Petri net whose tokens can carry data and whose transitions can check data (see exact definition in Weske’s book)

[Paula, 15000], [Mary, 22000], [Peter, 23000]

Credit Request

If a > 20000 (c,a)

AssessRisk

If a <= 20000 (c,a)

SimpleRisk Assessment

AdvancedRisk Assessment

Inform Customer I-Z

[BeginsWith (c, I-Z)]

Inform Customer A-H

[BeginsWith (c, A-H)]
Workflow nets
Workflow nets features

Tailored to the representation of business processes

  Formal (unambiguous) semantics

  Structural restrictions

  Decorated graphical representation
Workflow net: idea
Workflow net

Definition:
A Petri net \((P, T, F)\) is called workflow net if:

1. there is a distinguished initial place \(i \in P\) with \(\bullet i = \emptyset\)
2. there is a distinguished final place \(o \in P\) with \(o \bullet = \emptyset\)
3. every other place and transition belongs to a path from \(i\) to \(o\)
Workflow net: Rationale

1. a token in $i$ represents a process instance not yet started
2. a token in $o$ represents a finished case
3. each place and each transition can participate in a case

Definition:
A Petri net $(P, T, F)$ is called workflow net if:
1. there is a distinguished initial place $i \in P$ with $\bullet i = \emptyset$
2. there is a distinguished final place $o \in P$ with $o\bullet = \emptyset$
3. every other place and transition belongs to a path from $i$ to $o$
WF net: Example
Basic properties

Lemma: In a workflow net there is a unique node with no incoming arc.

Lemma: In a workflow net there is a unique node with no outgoing arc.

Exercise: Guess which nodes are those.

Exercise: Prove the above lemmas (hint: suppose the nodes are not unique, reach a contradiction).
Question time: WF net?

![Workflow diagram]

- **l** -> **t1** -> **p2** -> **t3** -> **o**
Question time: WF net?

Yes
Question time: WF net?
Question time: WF net?

No
(no initial place)
Question time: WF net?
Question time: WF net?

No
(no path from i to o via t2)
Question time: WF net?
Question time: WF net?

Yes
Question time: WF net?
Question time: WF net?
Question time: WF net?
Question time: WF net?

Yes
Syntax sugar: split

AND-split stands for

XOR-split stands for
Syntax sugar: join

stands for

AND-join

stands for

XOR-join
Syntax sugar: any combination is also possible

stands for
Syntax sugar: any combination is also possible

stands for
Syntax sugar: a personal note

- **AND-join**
  - Chosen decorations are too similar!
  - Different meanings if differently placed!

- **XOR-join**
  - Unnecessary for AND (redundant)!
Syntax sugar:
a personal note

Why there?  Because of gateways
Syntax sugar: a personal note

Let us avoid any source of confusion!
Hierarchical structuring

Uniqueness of entry / exit point facilitate the hierarchical structuring of WF nets

A transition can be realized by another workflow net
WoPeD
Workflow Petri Net Designer
Download WoPeD at sourceforge!
Language of a workflow net

The language of a workflow net is the set of firing sequences that go from $i$ to $o$

$$L(N) = \{ \sigma \mid i \xrightarrow{\sigma} o \}$$

$L(N)$ defines the admissible traces of the workflow
Typical control flow aspects

Sequencing

Parallelism (AND-split + AND-join)

Selection (XOR-split + XOR-join)

Iteration (XOR-join + XOR-split)

Capacity constraints:
  Feedback loop
  Mutual exclusion
  Alternating
Sequencing

B is executed after A
Parallelism

(AND-split + AND-join)

A and B are both executed in no particular order
Explicit choice
(XOR-split + XOR-join)

Either A or B is executed (choice is explicit)
Explicit choice ("sugared" version)

Decorated version
Deferred choice

Either A or B is executed (choice is implicit)
Remember

Explicit choice ≠ Implicit choice

The decision is made at different points in time
Iteration
(one or more times)

A is executed 1 or more times
One-or-more iteration ("sugared" version)

Decorated version
Iteration
(zero or more times)

A is executed 0 or more times
Zero-or-more iteration ("sugared" version)

Decorated version
One serve per time

Multiple activations are handled one by one
Mutual exclusion

A and B cannot execute concurrently
Alternation

A and B execute one time each (A first)
Consider the workflow net below

How many times can A be executed?
How many times can B be executed?
Can a firing sequence contain two As in a row?
Can a firing sequence contain two Bs in a row?
Can a firing sequence contain more Bs than As?
Consider the workflow net below

How many times can A be executed? 1 or more
How many times can B be executed? 0 or more
Can a firing sequence contain two As in a row? yes
Can a firing sequence contain two Bs in a row? no
Can a firing sequence contain more Bs than As? no
Exercises

- Which "patterns" can be found in the workflow net below?
- Draw the corresponding Reachability Graph
- What is its language?
Exercises

• Which "patterns" can be found in the workflow net below?
• "Sugarize" the net (where it makes sense)
• Name all places and draw the Reachability Graph
• What is its language?
Exercises

• "Desugarize" the workflow net below
• Name all nodes and draw the Reachability Graph
• What is its language?
Exercises

• "Desugarize" the workflow nets below
• Name all nodes and draw the Reachability Graphs
• What are their languages?

![Workflow Nets Diagram]

Execution constraints can depend on the environment in which processes are enacted.

In workflow nets, transitions can be decorated with the information on who (or what) is responsible for the "firing" of that task.

Such annotations are called triggers.
Triggers

Triggers can be:

- a human interaction
- the receipt of a message
- the expiration of a time-out

Transitions with no trigger can fire automatically
Symbols for triggers

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
Triggers: example

send request -> collect response -> archive file

send reminder
Triggers: example

A starts with the process of charging the credit card. There is a choice point where the credit card can either succeed or fail. If it succeeds, the process moves to packing the completed order. If it fails, the process goes to spamming the customer. If the customer is spammed, there is an option to cancel the order. The final step is updating the billing information.
Explicit vs Implicit XOR-split

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

(b) *Implicit xor split* enables A and B concurrently
Motivation for the analysis

L(N) shows the correct ways to run the process if it is empty there is clearly some problem.

Are we guaranteed that nothing can go wrong? Are we guaranteed that once a case is started it will reach an end?

BPs are large, with increasing complexity flawed situations are frequent.
Is this WF net ok?

What does it mean "to be ok"?