Modélisation des procédés et des workflow.
Une approche basée sur les réseaux de Pétri.

Warning !
Course under construction

Acknowledgement to W.v.d. Aalst, F. Charoy, D. Grigori,
O. Perrin and H. Skaf
This part mainly from W.V.D. Aalst lecture.

Références
• Livres :
  – W. v.d. Aalst, Workflow Management: Models, Methods and
    Systems, MIT Press
  – Frank Leymann and Dieter Roller, Production Workflow:
    Concepts and Technics

• Workflow Management Coalition http://www.wfmc.org/
• Workflow (Cours) www.workflowcourse.com
• Workflow patterns www.workflowpatterns.com
• Découverte de procédés www.processmining.org
Overview of process modeling techniques

- Flowcharts
- Dataflow diagrams (DFD, ISAC, SADT, IDEF)
- Unified Modeling Technique
- Transition systems, state transition diagrams (extensions: e.g. state charts)
- Queueing networks and Markov chains
- Process algebra's (ACP, CCS, CSP)
- (High level) Petri nets
- Vendor specific diagramming techniques used in WFMS's, simulation tools and CASE tools.

UML: Unified Modeling Language

- Use cases
- Class Diagram
- Interaction Diagrams
- State Diagrams
- Activity Diagrams
Petri Nets

Petri nets

- The classical Petri net was invented by Carl Adam Petri in 1962.
- A lot of research has been conducted (>10,000 publications).
- Until 1985 it was mainly used by theoreticians.
- Since the 80's the practical use is increasing because of the introduction of high level Petri nets and the availability of many tools.

The classical Petri net model

A Petri net is a network composed of places (○) and transitions (□).

Connections are directed and between a place and a transition.
Tokens (*) are the dynamic objects.
The state of a Petri net is determined by the distribution of tokens over the places.
Transition t1 has three input places (p1, p2 and p3) and two output places (p3 and p4).
Place p3 is both an input and an output place of t1.

Enabling condition

Transitions are the active components and places and tokens are passive.
A transition is enabled if each of the input places contains tokens.

Transition t1 is not enabled, transition t2 is enabled.

Firing

An enabled transition may fire.
Firing corresponds to consuming tokens from the input places and producing tokens for the output places.

Firing is atomic.
Example

Non-determinism

Two transitions fight for the same token: conflict.
Even if there are two tokens, there is still a conflict.

Modeling

States of a process are modeled by tokens in places and state transitions leading from one state to another are modeled by transitions.

- Tokens represent objects (humans, goods, machines), information, conditions or states of objects.
- Places represent buffers, channels, geographical locations, conditions or states.
- Transitions represent events, transformations or transportations.
Example: Dutch traffic light

Two traffic lights

Two safe traffic lights
Two safe and fair traffic lights

Ball game

- The number of arcs between two objects specifies the number of tokens to be produced/consumed.
- This can be used to model (dis)assembly processes.

Some definitions

- **current state**
  The configuration of tokens over the places.
- **reachable state**
  A state reachable from the current state by firing a sequence of enabled transitions.
- **dead state**
  A state where no transition is enabled.
High level Petri Nets

High-level Petri nets

In practice the classical Petri net is not very useful:
• The Petri net becomes too large and too complex.
• It takes too much time to model a given situation.
• It is not possible to handle time and data.

Therefore, we use high level Petri nets, i.e. Petri nets extended with:
• color
• time
• hierarchy
To explain the three extensions we use the following example of a hairdresser's saloon.

client waiting  hairdresser ready to begin

waiting  busy  ready

Note how easy it is to model the situation with multiple hairdressers.

The extension with color

A token often represents an object having all kinds of attributes. Therefore, each token has a value (color) with refers to specific features of the object modeled by the token.

waiting  busy  ready

Each transition has an (in)formal specification which specifies:
• the number of tokens to be produced,
• the values of these tokens,
• and (optionally) a precondition.

The complexity is divided over the network and the values of tokens.

This results in a compact, manageable and natural process description.
The extension with time
For performance analysis we need to model durations, delays, etc. Therefore, each token has a timestamp and transitions determine the delay of a produced token.

The extension with hierarchy
- A mechanism to structure complex Petri nets comparable to DFD's.
- A subnet is a net composed out of places, transitions and subnets.

Exercise: remove hierarchy
Mapping workflow concepts onto Petri nets

Workflow management concepts

A workflow definition is composed out of three parts:

- **process definition:** a description of the process itself
- **resource classification:** a classification of the resources to be used
- **resource management rules:** how to map work onto resources

Process definition

A **process definition** specifies which steps are required and in what order they should be executed.

- (routings definition, procedure, workflow script)
- (purchase order, tax declarations, insurance claims process)

A process definition consists of:

- **Tasks** (step, activity, process element)
  - A task is atomic: commit or rollback.

- **Conditions** (state, phase, requirement)
  - A condition is used to determine the enabling of a task.

- **Subprocesses**
Case

A case is the 'thing' which needs to be processed by following the process definition.

(process instance, job, project)
(insurance claim, purchase order, complaint, loan application)

The state of a case is determined by:

• **case variables** (case parameters)
  The logistic attributes of a case which are used to route the case.

• **conditions**
  The requirements which are satisfied.

• **(Application data)**
  Beyond the scope of the WFMS.

Mapping a process definition onto Petri nets

Routing of cases

• Sequential
  "first A then B"

• Parallel
  "A en B at the same time or in any order"
  – AND split
  – AND join

• Choice
  "A or B"
  – OR split
  – OR join

• Iteration
  "multiple A's"
Sequential routing

"First A then B"

Parallel routing

"A and B at the same time or in any order"

Choice (1)

"A or B"

Overkill ?!
Choice (2)

Implicit choice: it depends on the "eagerness" of A and B!

Choice (3)

We use high level Petri nets:
- tokens have values: case variables
- transitions determine the number of tokens produced: explicit OR-split

Choice is explicit and may be based on logistic attributes!

Syntactic sugaring

AND-split

Explicit OR-split

OR-join

AND-join
B may be executed several times.

Exercise
- A travel agency organizes trips. To organize a trip the following tasks are executed.
  - First the customer request is registered, then an employee searches for opportunities which are communicated to the customer. Then the customer is contacted to find out whether (s)he is still interested and whether more alternatives are desired.
  - If the customer selects a trip, then the trip is booked. In parallel (if desired) one or two types of insurance are prepared.
  - Two weeks before the start date the documents are sent to the customer.
  - It is possible that the customer cancels the trip at any time before the start date.

Triggers
- The workflow system cannot force things to happen in reality:
  - The arrival of an electronic message (EDM) which is needed to execute a task.
  - A resource which starts to work on a case.
  - The arrival of a paper document.
  - A phone call to confirm a purchase order.
- A workflow system is a reactive system, i.e. it is triggered by the environment.
- Some tasks require a trigger.
We identify four kinds of tasks:

- **Automatic**
  - No trigger is required.

- **User**
  - A resource takes the initiative.

- **External**
  - An external event (message, phone call) is required.

- **Time**
  - The task requires a time trigger.

The triggering concept can be modeled in terms of PN:

```
  trigger token
```

```
  o  A  o
      ↙   ↘
```

However, we will omit the extra place.

Mapping …

- Organizational aspects are weakly considered
Analysis

Techniques to analyze workflows

Analysis techniques are used to answer two types of questions:

- **qualitative questions**
  - Is there a deadlock possible?
  - Is it possible to successfully handle a specific case?
  - Will all cases terminate eventually?
  - Is it possible to execute two tasks in any order?

- **quantitative questions**
  - How many cases can be handled in one hour?
  - What is the average throughput time?
  - How many resources are required?
  - How many cases are handled within 2 days?

Analysis techniques can be used to avoid logical errors.

Is this a correct workflow? If not, how to correct it?
Error 1: dangling tasks

begin task1 task2 task3 end

start task4

Error 2: deadlock (task2)

begin task1 task2 end

start

Error 3: unbounded and never ending

begin task1 task2 task3 end

start
Error 4: deadlock before or after termination

Petri-nets: a solver-independent medium

Good structuring: well balanced operators
Reachability analysis

Reachability graph
- Each node corresponds to a reachable state.
- Done by a computer.
- A computer can cope with reachability graphs with millions of nodes.

\[
\begin{align*}
(0,0,1,0,0,0) & \quad \rightarrow \quad (1,0,0,1,0,0) \\
(0,1,0,1,0,0) & \quad \rightarrow \quad (1,0,0,0,1,0) \\
\end{align*}
\]

The traffic lights are safe!

Structural analysis

Many techniques are available:
- **place invariants**
- transition invariants
- traps and siphons
- reduction rules
- decomposition techniques
- S-covers/T-covers
- special techniques for subclasses:
  - state machines
  - marked graphs
  - free-choice nets
  - asymmetric free-choice nets
Place invariants

- A place invariant assigns a **weight** to each place such that the **weighted token sum** remains constant.

- $1\text{man}+1\text{woman}=2\text{couple} \Rightarrow \text{constant}$
  - $(\text{man}+\text{woman}=2\text{couple}=?)$
  - **woman + couple**
  - **man + couple**
  - **man - woman**

---

Example

- $\text{red1} + \text{yellow1} + \text{green1} = 1$
- $\text{red2} + \text{yellow2} + \text{green2} = 1$
- $\text{safe} + \text{green1} + \text{green2} + \text{yellow1} + \text{yellow2} = 1$
- $\text{red1} + \text{red2} - \text{safe} = 1$

---

Soundness property

- The soundness property corresponds to two standard Petri-net properties (liveness and boundedness).
- Standard Petri-net-based tools can be used.
- For (almost) free-choice nets this can be checked in polynomial time!
Performance analysis

Questions:
- throughput, waiting and service times
- service levels
- occupation rates

Techniques:
- simulation
- queuing theory
- Markovian analysis

Example: sequential (1)

- average throughput time: 22.2 minutes
- service time: 8.0 minutes
- waiting time: 14.2 minutes

Parallel (2)

- average throughput time: 15 minutes
- service time: 4 minutes
- waiting time: 11 minutes
Compose (3)

- average throughput time: 9.5 minutes
- service time: 7.0 minutes
- waiting time: 2.5 minutes

Flexible resources (4)

- average throughput time: 14.0 minutes
- service time: 8.0 minutes
- waiting time: 6.0 minutes

Triage (5)

- average throughput time: 31.1 minutes
- service time: 8.0 minutes
- waiting time: 23.1 minutes
Priority (6)

- average throughput time: 14 minutes
- service time: 8 minutes
- waiting time: 6 minutes

Results

<table>
<thead>
<tr>
<th>situation</th>
<th>description</th>
<th>average throughput</th>
<th>average service time</th>
<th>average waiting time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Situation 1</td>
<td>sequential</td>
<td>22.2</td>
<td>8.0</td>
<td>14.2</td>
</tr>
<tr>
<td>Situation 2</td>
<td>parallel</td>
<td>15</td>
<td>4(8)</td>
<td>11(7)</td>
</tr>
<tr>
<td>Situation 3</td>
<td>triage</td>
<td>9.5</td>
<td>7.0</td>
<td>2.5</td>
</tr>
<tr>
<td>Situation 4</td>
<td>triage</td>
<td>14.0</td>
<td>8.0</td>
<td>6.0</td>
</tr>
<tr>
<td>Situation 5</td>
<td>triage</td>
<td>31.1</td>
<td>8.0</td>
<td>23.1</td>
</tr>
<tr>
<td>Situation 6</td>
<td>triage</td>
<td>14.0</td>
<td>8.0</td>
<td>6.0</td>
</tr>
</tbody>
</table>

Queuing models

arrivals

| x |

waiting

| s |

service

| μ |

Basic characteristics:
- average number of arrivals per time unit: λ (mean arrival rate)
- average number that can be handled by one server per time unit: μ (mean service rate)
- number of servers: x
Simulation

- Random walk through the reachability graph
- Computer experiment
  - pseudo random numbers
  - random generator
- Validation
- Statistical aspects
  - start run
  - subruns
- Animation
- Flexible
- No proof!

Workflow Patterns

http://www.workflowpatterns.com/

Workflow Patterns

- In the last decade more than 200 workflow management systems have
  become available.
- Despite the efforts of the Workflow Management Coalition (WfMC) a good
  standard for workflow design is missing.
- Available systems are really different with respect of basic workflow
  functionality.
- Selection processes typically do not address these differences.
- In this research we (joint work with Arthur ter Hofstede and Bartek
  Kiepuszewski) provide the academic answer to the poor evaluations made by
  prestigious consulting firms like Gartner, Moret, Ernst & Young, etc.
- Thus far more than 30 patterns have been collected.
- Moreover, the patterns give insight into the basic constructs in operational
  processes and are a means to characterize processes and to train designers.
Categories of patterns

- Basic Control Flow Patterns
- Advanced Branching and Synchronization Patterns
- Structural Patterns
- Patterns involving Multiple Instances
- State-based Patterns
- Cancellation Patterns

Basic Control Flow Patterns

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

Pattern 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.
**Pattern 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.

![Diagram of Pattern 2](image)

**Pattern 3** (Synchronization)
*Description* A point in the workflow process where multiple parallel subprocesses/activities converge into one single thread of control, thus synchronizing multiple threads. It is an assumption of this pattern that each incoming branch of a synchronizer is executed only once.
*Synonyms* AND join, rendezvous, synchronizer.

![Diagram of Pattern 3](image)

**Pattern 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.

![Diagram of Pattern 4](image)
**Pattern 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.

**Synonyms** XOR, asynchronous join, merge.

![Diagram](BD_CD)

---

**Advanced Branching and Synchronization Patterns**

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

---

**Pattern 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.

**Synonyms** Conditional routing, selection, OR, split.

![Diagram](A_B C)
Pattern 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. 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.
Synonyms: Synchronizing join.
ABD; ACD; ABCD; ACBD

Pattern 8 (Multi-merge)
Description: A point in a 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.
ABD, ACD, ABCDD, ACBDD
(cf., non-safe Petri nets.)
**Pattern 9** (Discriminator)

**Description** The discriminator is a point in a workflow process that waits for one of the incoming branches to complete before activating the subsequent activity. From that moment on it waits for all remaining branches to complete and “ignores” them. Once all incoming branches have been triggered, it resets itself so that it can be triggered again (which is important otherwise it could not really be used in the context of a loop).

---

**N out of M**

- **N out of M** is a point in a workflow process where M parallel paths converge into one. The subsequent activity should be activated once N paths have completed. Completion of all remaining paths should be ignored. Similarly to the discriminator, once all incoming branches have “fired”, the join resets itself so that it can fire again.
Structural Patterns

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

Pattern 10 (Arbitrary Cycles)
Description: A point in a workflow process where one or more activities can be done repeatedly.
Synonyms: Loop, iteration, cycle.

Pattern 11 (Implicit Termination)
Description: A given subprocess should be terminated when there is nothing else to be done. In other words, there are no active activities in the workflow and no other activity can be made active (and at the same time the workflow is not in deadlock).
State-based Patterns

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

**Pattern 16 (Deferred Choice)**

*Description* A point in the workflow process where one of several branches is chosen. In contrast to the XOR split, the choice is not made explicitly (e.g., based on data or a decision) but several alternatives are offered to the environment. However, in contrast to the AND split, only one of the alternatives is executed. This means that once the environment activates one of the branches, the other alternative branches are withdrawn. It is important to note that the choice is delayed until the processing in one of the alternative branches is actually started, i.e., the moment of choice is as late as possible.

*Synonyms* External choice, implicit choice, deferred XOR split.
**Pattern 17** (Interleaved Parallel Routing)

**Description** A set of activities is executed in an arbitrary order: each activity in the set is executed, the order is decided at run time, and no two activities are executed at the same moment (i.e. no two activities are active for the same workflow instance at the same time).

**Synonyms** Unordered sequence

**Pattern 18** (Milestone)

**Description** The enabling of an activity depends on the case being in a specified state, i.e. the activity is only enabled if a certain milestone has been reached which did not expire yet.

**Synonyms** Test arc, deadline, state condition, withdraw message.
Cancellation Patterns

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

**Pattern 19** (Cancel Activity)
*Description* An enabled activity is disabled, i.e. a thread waiting for the execution of an activity is removed.
*Synonyms* Withdraw activity.

**Pattern 20** (Cancel Case)
*Description* A case, i.e. workflow instance, is removed completely (i.e., even if parts of the process are instantiated multiple times, all descendants are removed).
<table>
<thead>
<tr>
<th>pattern</th>
<th>basic</th>
<th>adv.</th>
<th>synth.</th>
<th>struct.</th>
<th>mult.</th>
<th>inst.</th>
<th>state</th>
<th>cancel</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (frq)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 (ms)</td>
<td>+</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>