

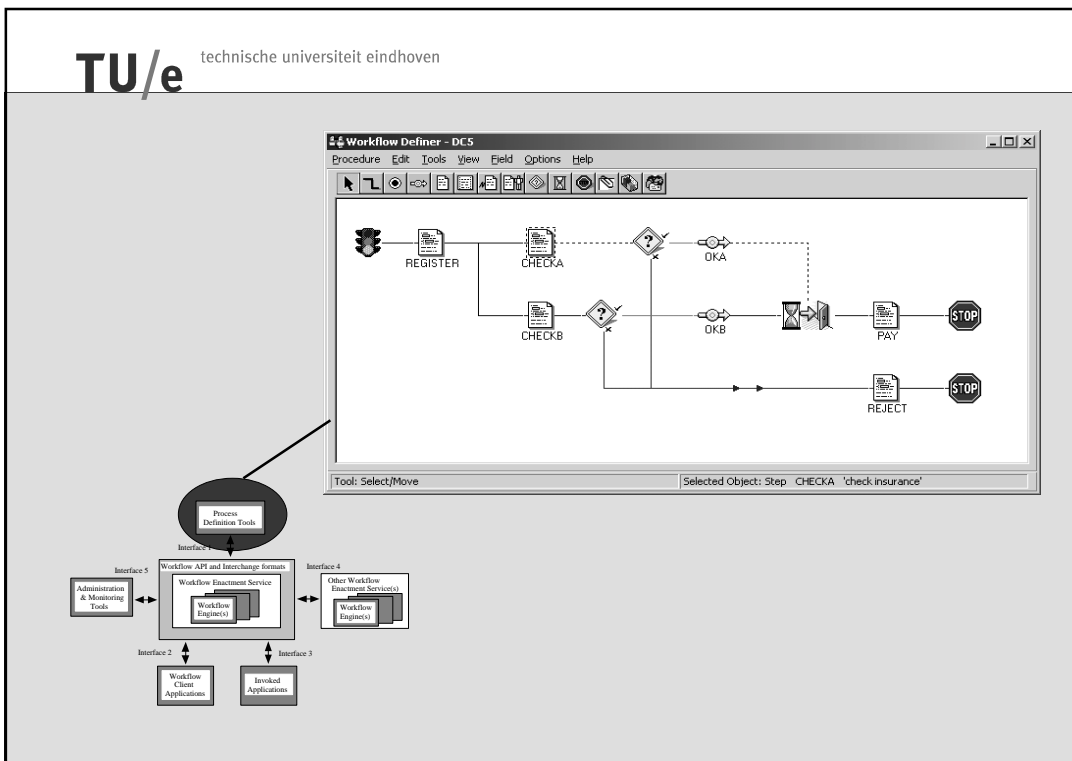
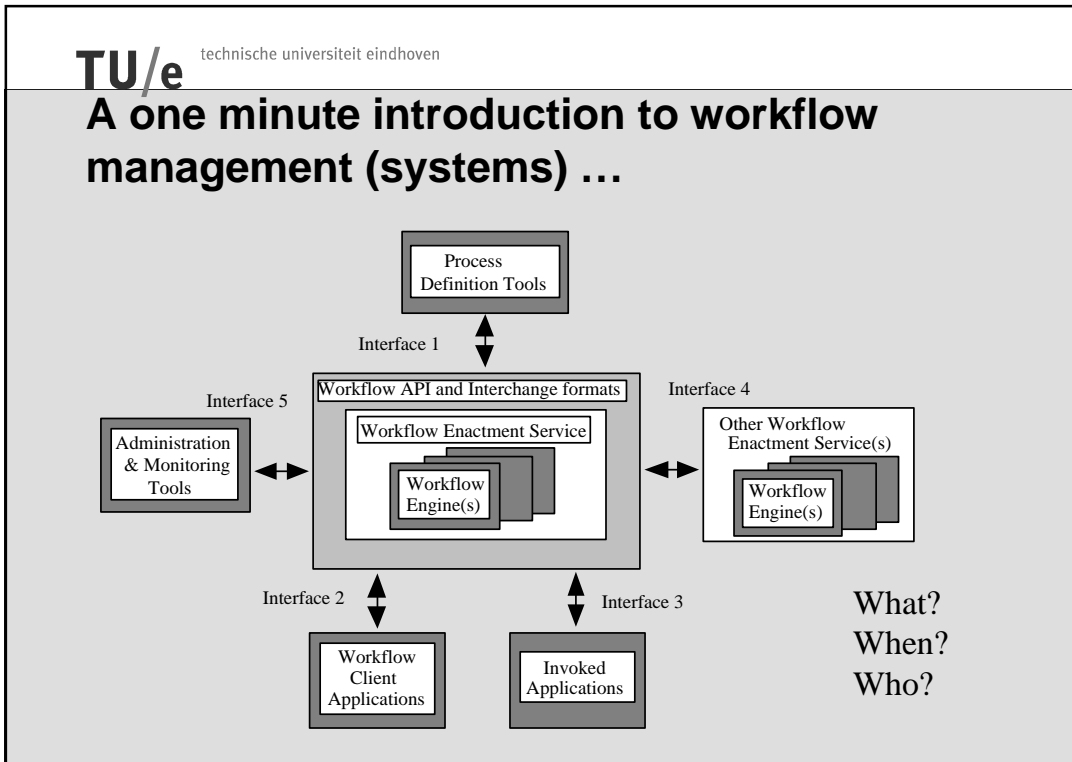
Towards a Workflow Language based on XML, Petri Nets, and Workflow Patterns

Wil van der Aalst

Eindhoven University of Technology, Faculty of Technology Management
Department of Information and Technology, P.O.Box 513, NL-5600 MB,
Eindhoven, The Netherlands.

Outline

- Introduction
 - Workflow management
 - Limitations of existing systems
- Part I: Workflow patterns
 - Examples
 - Evaluation of systems
- Part II: XRL
 - XML based Language
 - Petri net Semantics
 - Architecture
- Conclusion



TU/e technische universiteit eindhoven

Staffware 2000 System Administrator: DC_register - Work Items

Status	Case Ref.	Case Description	Form Description
	81-5	DC6 - Case Jansen	register
	81-6	DC6 - case Pieteresen	register
	81-7	DC6 - Case Van Balen	register

Form: register

Register claim
This is the first step of the Double Check Process

Enter name:

Close Form

Close the Form Window and

the Form

Queue: DC_register 3

Architecture Diagram:

- Process Definition Tools
- Interface 1
- Interface 5
- Administration & Monitoring Tools
- Workflow API and Interchange Services
- Workflow Enactment Services
- Workflow Enactment Services
- Other Workflow Enactment Services
- Workflow Enactment Services
- Interface 4
- Interface 3
- Workflow Client Applications
- Invoked Applications

TU/e technische universiteit eindhoven

Staffware Administration Managers: staffw_edlbp

User Manager Backup Manager Table Manager Case Manager

List Manager Network Manager Move SysInfo Exit Admin

System Administrator (swadmin) 11:48 18/04/2003

Staffware User Manager

Users Groups Users/Groups Roles Attributes

User Name Search: wvdaalst

Users	Description
edlbp	EDL-BP
swadmin	System Administrator
swanne	Demo user Anne
swchris	Demo user Chris
swmartin	Demo user Martin
swvera	Demo user Vera
tmitgast	Gastaccount IT
wvdaalst	Wil van der Aalst

Attributes For: wvdaalst

Current Value: Wil van der Aalst

Attribute Name	Type	Length	Dec	Value
DESCRIPTION	Text	24	0	Wil van der Aalst
LANGUAGE	Text	24	0	English
MENUNAME	Text	24	0	ADMIN
SORTMAIL	Text	24	0	PROCEDURE
USERFLAGS	Text	24	0	Step Forward

Architecture Diagram:

- Process Definition Tools
- Interface 1
- Interface 4
- Administration & Monitoring Tools
- Workflow API and Interchange Services
- Workflow Enactment Services
- Workflow Enactment Services
- Other Workflow Enactment Services
- Workflow Enactment Services
- Interface 2
- Interface 3
- Workflow Client Applications
- Invoked Applications

Contemporary WFM systems

- Features:
 - generic support for operational processes
 - adaptable processes, no programming, graphical, etc.
- Limitations:
 - difficulties supporting complex processes (lack of expressive power) } *Part I: Patterns*
 - limited run-time flexibility } *Part II: XRL*
 - limited support for interorganizational workflows
 - limited support for analysis

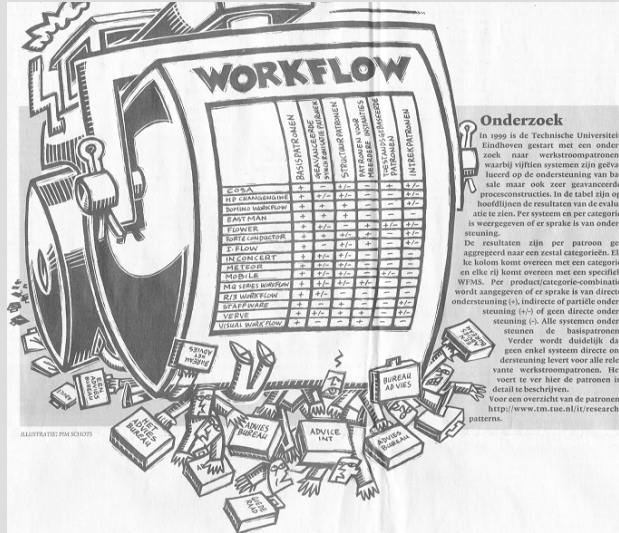
Part I: workflow patterns

Joint work with Arthur ter Hofstede (QUT), Bartek Kiepuszewski (QUT), Alistair Barros (UQ), Oscar Ommert (EUT), Ton Pijpers (ATOS), et al.

<http://www.tm.tue.nl/it/research/patterns/>

Workflow patterns

- The academic response
- A quest for the basic requirements
- 20 basic patterns
- 16 systems
- Joint work with QUT, ATOS, etc.



Categories of patterns

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)

Patterns involving Multiple Instances

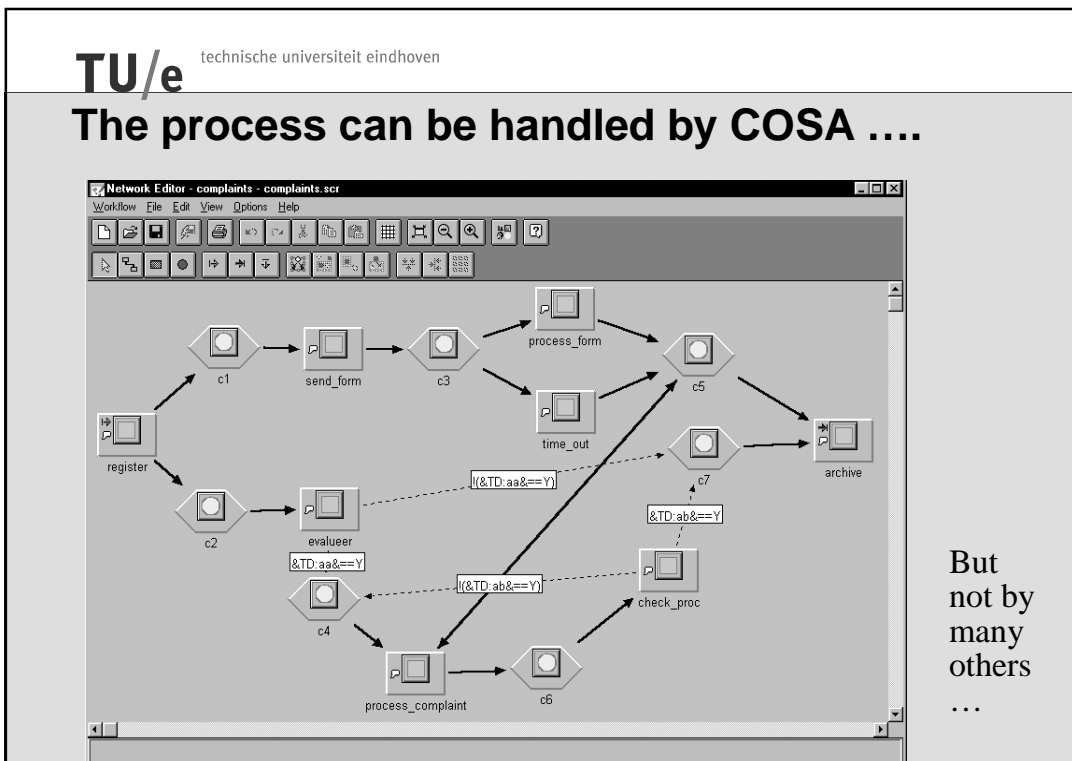
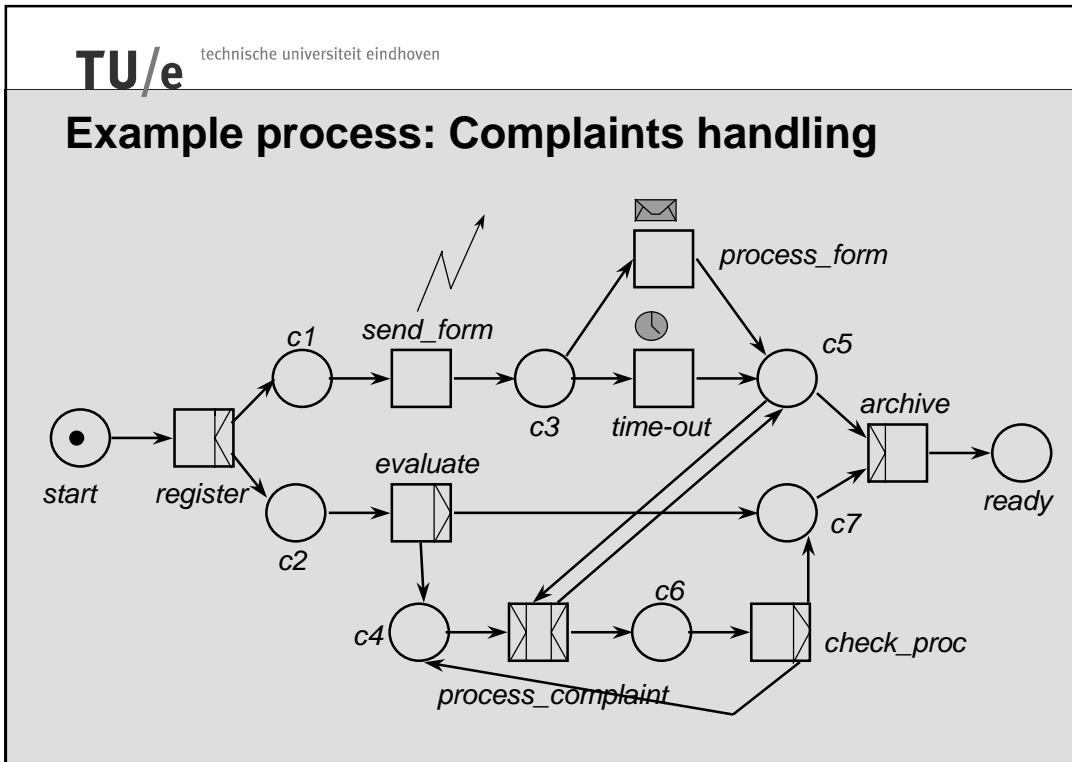
- Pattern 12 (Multiple Instances Without Synchronization)
- Pattern 13 (Multiple Instances With a Prior Design Time Knowledge)
- Pattern 14 (Multiple Instances With a Prior Runtime Knowledge)
- Pattern 15 (Multiple Instances Without a Prior Runtime Knowledge)

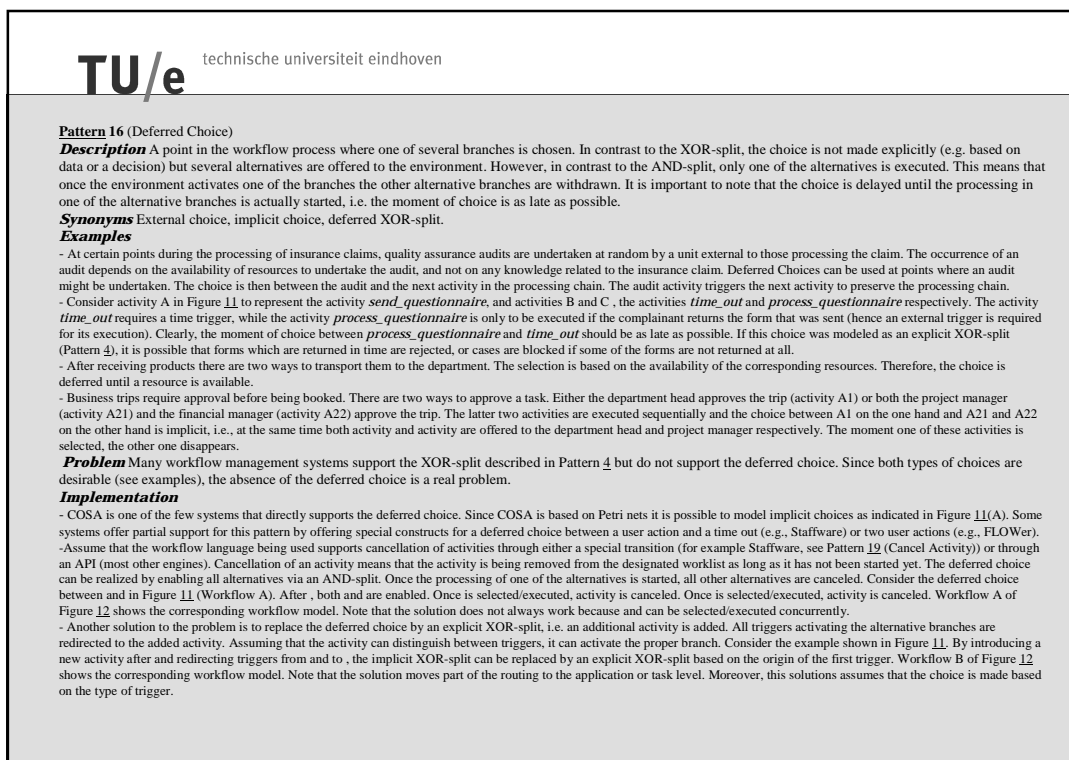
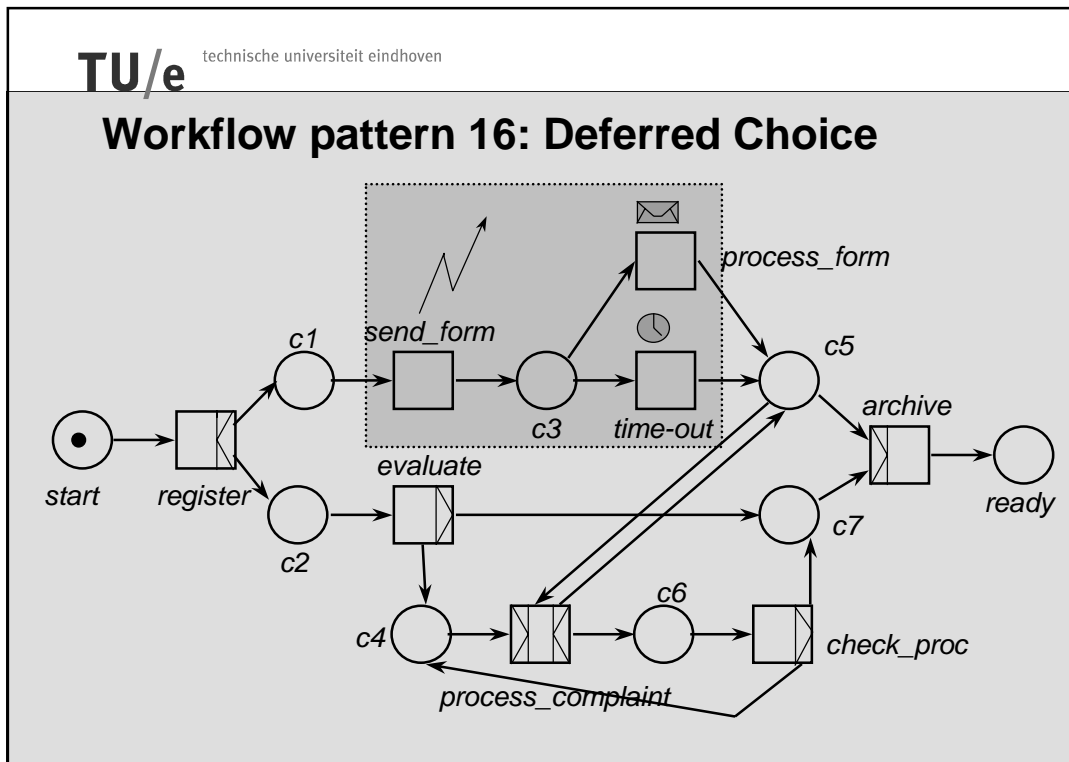
State-based Patterns

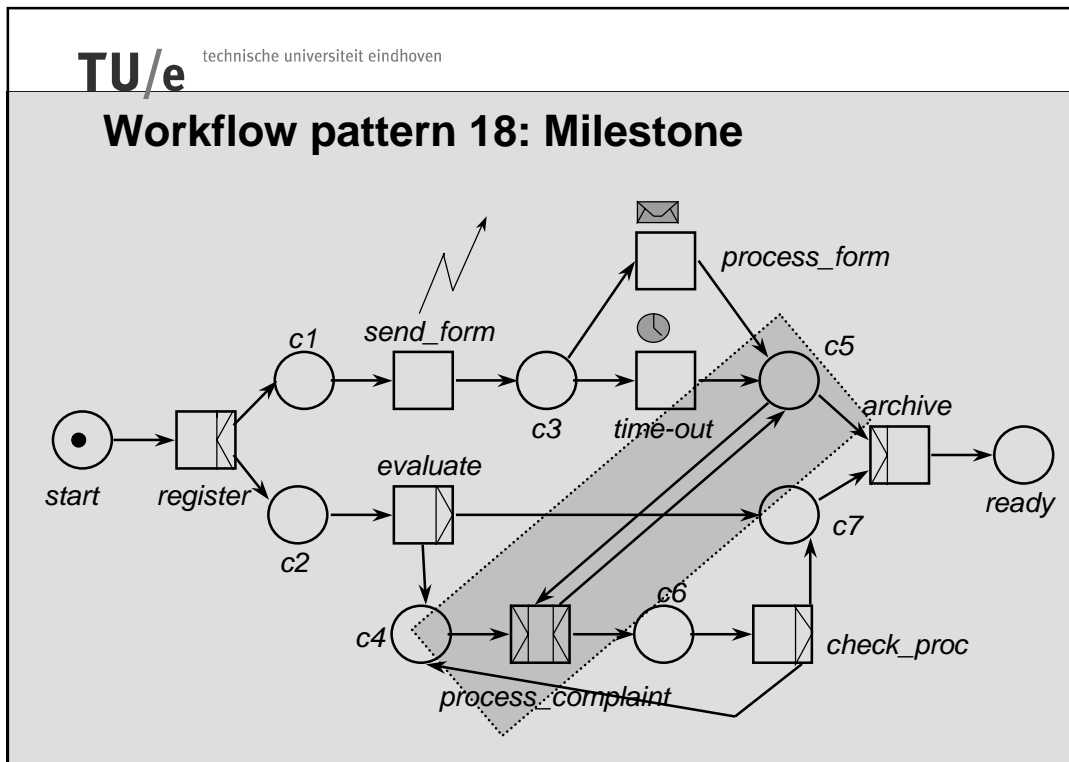
- Pattern 16 (Deferred Choice)
- Pattern 17 (Interleaved Parallel Routing)
- Pattern 18 (Milestone)

Cancellation Patterns

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







TU/e technische universiteit eindhoven

pattern	product							
	Staffware	COSA	InConcert	Eastman	FLOWer	Domino	Meteor	Mobile
1 (seq)	+	+	+	+	+	+	+	+
2 (par-spl)	+	+	+	+	+	+	+	+
3 (synch)	+	+	+	+	+	+	+	+
4 (ex-ch)	+	+	+/-	+	+	+	+	+
5 (simple-m)	+	+	+/-	+	+	+	+	+
6 (m-choice)	-	+	+/-	+/-	-	+	+	+
7 (sync-m)	-	+/-	+	+	-	+	-	-
8 (multi-m)	-	-	-	+	+/-	+/-	+	-
9 (disc)	-	-	-	+	+/-	-	+/-	+
10 (arb-c)	+	+	-	+	-	+	+	-
11 (impl-t)	+	-	+	+	-	+	-	-
12 (mi-no-s)	-	+/-	-	+	+	+/-	+	-
13 (mi-dt)	+	+	+	+	+	+	+	+
14 (mi-rt)	-	-	-	-	+	-	-	-
15 (mi-no)	-	-	-	-	+	-	-	-
16 (def-c)	-	+	-	-	+/-	-	-	-
17 (int-par)	-	+	-	-	+/-	-	-	+
18 (milest)	-	+	-	-	+/-	-	-	-
19 (can-a)	+	+	-	-	+/-	-	-	-
20 (can-c)	-	-	-	-	+/-	+	-	-

basic: 1-5
 adv. synch.: 6-9
 struct.: 10-12
 mult. inst.: 13-15
 state: 16-18
 cancel: 19-20

TU/e technische universiteit eindhoven

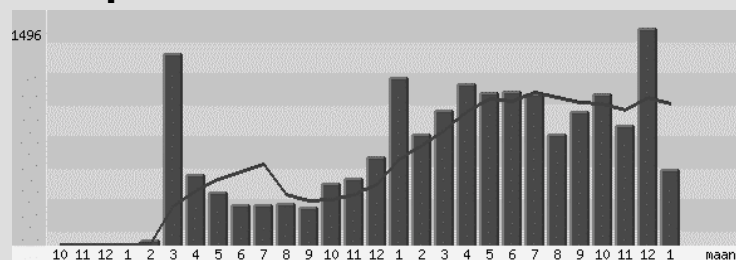
pattern	product						
	MQSeries	Forté	Verve	Vis. WF	Changeng.	I-Flow	SAP/R3
1 (seq)	+	+	+	+	+	+	+
2 (par-spl)	+	+	+	+	+	+	+
3 (synch)	+	+	+	+	+	+	+
4 (ex-ch)	+	+	+	+	+	+	+
5 (simple-m)	+	+	+	+	+	+	+
6 (m-choice)	+	+	+	+	+	+	+
7 (sync-m)	+	-	-	-	-	-	-
8 (multi-m)	-	+	+	-	-	-	-
9 (disc)	-	+	+	-	+	-	+
10 (arb-c)	-	+	+	+/-	+	+	-
11 (impl-t)	+	-	-	-	-	-	-
12 (mi-no-s)	-	+	+	+	-	+	-
13 (mi-dt)	+	+	+	+	+	+	+
14 (mi-rt)	+/-	-	-	-	-	-	+/-
15 (mi-no)	-	-	-	-	-	-	-
16 (def-c)	-	-	-	-	-	-	-
17 (int-par)	-	-	-	-	-	-	-
18 (milest)	-	-	-	-	-	-	-
19 (can-a)	-	-	-	-	-	-	+
20 (can-c)	-	+	+	-	+	-	+

basic: 1-5
 adv. synch.: 6-9
 struct.: 10-12
 mult. inst.: 13-15
 state: 16-18
 cancel: 19-20

Scientific results

- Mapping onto WF-nets for analysis etc.
- Classification of mechanisms:
 - single thread (safe) / multiple threads (non-safe)
 - state machine / marked graph / free-choice / well-structured / arbitrary
 - graph structured / block structured
 - normal token / true-false token / maximal input
- Observation: New systems/standards are more expressive!

Practical impact



- <http://www.tm.tue.nl/it/research/patterns>
- +/- 80 pageviews per working day (>17.000 in total)
- patterns are used in selection processes
- role of vendors has been opportunistic

Part II: XRL

Joint work with Eric Verbeek (EUT), Akhil Kumar (CU/BL),
and Alexander Hirnschall (EUT).

<http://www.tm.tue.nl/it/staff/wvdaalst/workflow/xrl>

XRL: Motivation

- The eXchangeable Routing Language (XRL) has been developed to address limitations of existing systems, cf.
 - difficulties supporting complex processes (lack of expressive power)
 - limited run-time flexibility
 - limited support for interorganizational workflows
 - limited support for analysis
- language is inspired by patterns research
- goal: testbed/play yard for scientific results

Features of XRL

- **Syntax** is **XML** based: Allows for the application of XML technology (XSLT, etc.).
- **Semantics** is based on **Petri nets**: Allows for analysis and enactment.
- **Extendible** with new routing primitives (exploits XML/Petri net base).
- Processes described at **instance** level (allows for run-time flexibility and additional patterns).

Language: XRL Routing Elements

- Task
- Sequence
- Any_sequence
- Choice
- Condition
- Parallel_sync
- Parallel_no_sync
- Parallel_part_sync
- Parallel_part_sync_cancel
- Wait_all
- Wait_any
- While_do
- Terminate

XRL – DTD (1)

```

<!ENTITY % routing_element
    "task|sequence|any_sequence|choice|condition|parallel_sync|parallel_no_sync|parallel_part_sync|parallel_part_sync_cancel|wait_all|wait_any|while_do|terminate">
<!ELEMENT route ((%routing_element), event*)>
<!ATTLIST route
    name ID #REQUIRED
    created_by CDATA #IMPLIED
    date CDATA #IMPLIED>
<!ELEMENT task (event*)>
<!ATTLIST task
    name ID #REQUIRED
    address CDATA #REQUIRED
    role CDATA #IMPLIED
    doc_read NMTOKENS #IMPLIED
    doc_update NMTOKENS #IMPLIED
    doc_create NMTOKENS #IMPLIED
    result CDATA #IMPLIED
    status (ready|running|enabled|disabled|aborted|null) #IMPLIED
    start_time NMTOKENS #IMPLIED
    end_time NMTOKENS #IMPLIED
    notify CDATA #IMPLIED>

```

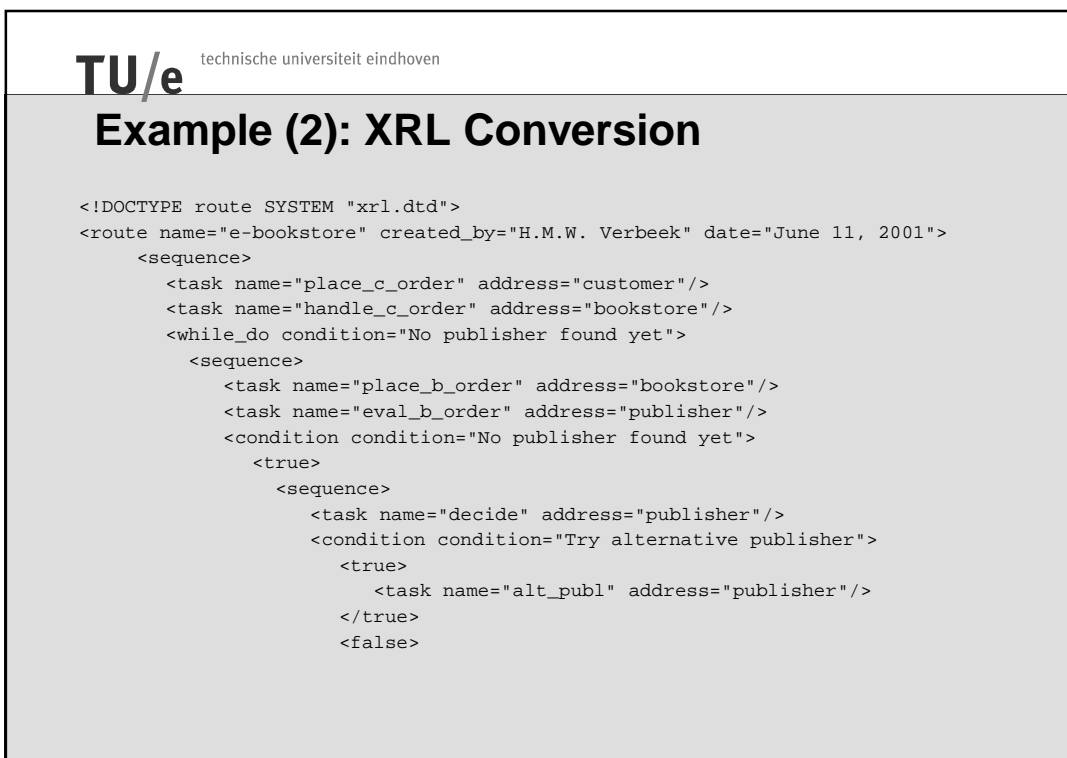
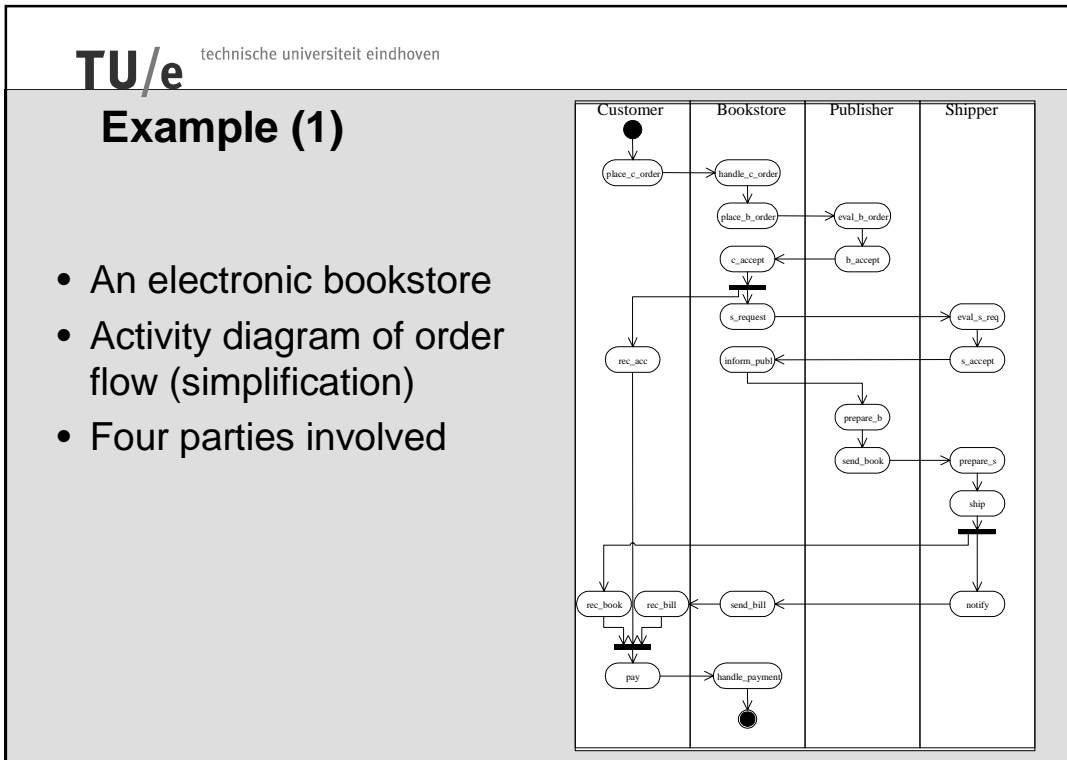
XML (eXtensible Markup Language)
 DTD (Document Type Definition)
 XSLT (eXtensible Stylesheet Language Transformations)

XRL – DTD (2)

```

<!ELEMENT event EMPTY>
<!ATTLIST event
    name ID #REQUIRED>
<!ELEMENT sequence ((%routing_element;|state)+)>
<!ELEMENT any_sequence ((%routing_element;)+)>
<!ELEMENT choice ((%routing_element;)+)>
<!ELEMENT condition ((true|false)*)>
<!ATTLIST condition
    condition CDATA #REQUIRED>
<!ELEMENT true (%routing_element;)>
<!ELEMENT false (%routing_element;)>
<!ELEMENT parallel_sync ((%routing_element;)+)>
<!ELEMENT parallel_no_sync ((%routing_element;)+)>
<!ELEMENT parallel_part_sync ((%routing_element;)+)>
<!ATTLIST parallel_part_sync
    number NMTOKEN #REQUIRED>
<!ELEMENT parallel_part_sync_cancel ((%routing_element;)+)>
<!ATTLIST parallel_part_sync_cancel
    number NMTOKEN #REQUIRED>
<!ELEMENT wait_all ((event_ref|timeout)+)>
<!ELEMENT wait_any ((event_ref|timeout)+)>
<!ELEMENT event_ref EMPTY>
<!ATTLIST event_ref
    name IDREF #REQUIRED>
<!ELEMENT timeout ((%routing_element;)?)>
<!ATTLIST timeout
    time CDATA #REQUIRED
    type (relative|s_relative|absolute) "absolute">
<!ELEMENT while_do (%routing_element;)>
<!ATTLIST while_do
    condition CDATA #REQUIRED>
<!ELEMENT terminate EMPTY>
<!ELEMENT state EMPTY>

```



Example (3): XRL Conversion

```

    <sequence>
      <task name="b_reject" address="publisher"/>
      <task name="c_reject" address="bookstore"/>
      <task name="rec_decl" address="customer"/>
    </sequence>
  </false>
</condition>
</sequence>
</true>
<false>
  <sequence>
    <task name="b_accept" address="publisher"/>
    <task name="c_accept" address="bookstore"/>
    <parallel_sync>
      <task name="rec_acc" address="customer">
        <event name="accept"/>
      </task>
    </parallel_sync>
    <sequence>
      <while_do condition="No shipper found yet">

```

Example (4): XRL Conversion

```

    <sequence>
      <task name="s_request" address="bookstore"/>
      <task name="eval_s_req" address="shipper"/>
    </sequence>
  </while_do>
  <condition condition="Shipper found">
    <true>
      <sequence>
        <task name="s_accept" address="shipper"/>
        <task name="inform_publ" address="bookstore"/>
        <task name="prepare_b" address="publisher"/>
        <task name="send_book" address="publisher"/>
        <task name="prepare_s" address="shipper"/>
        <task name="ship" address="shipper"/>
        <parallel_sync>
          <sequence>
            <task name="notify" address="shipper"/>
            <task name="send_bill" address="bookstore"/>
          </sequence>
        </parallel_sync>
      </sequence>
    </true>
  </condition>

```

Example (5): XRL Conversion

```

    <wait_all>
      <event_ref name="accept" />
    </wait_all>
    <task name="rec_bill" address="customer" />
  </sequence>
  <sequence>
    <wait_all>
      <event_ref name="accept" />
    </wait_all>
    <task name="rec_book" address="customer" />
  </sequence>
</parallel_sync>
<task name="pay" address="customer" />
<task name="handle_payment" address="bookstore" />
</sequence>
</true>
<false>
  <task name="s_reject" address="shipper" />
</false>

```

Example (6): XRL Conversion

```

    </condition>
  </sequence>
</parallel_sync>
</sequence>
</false>
</condition>
</sequence>
</while_do>
</sequence>
</route>

```

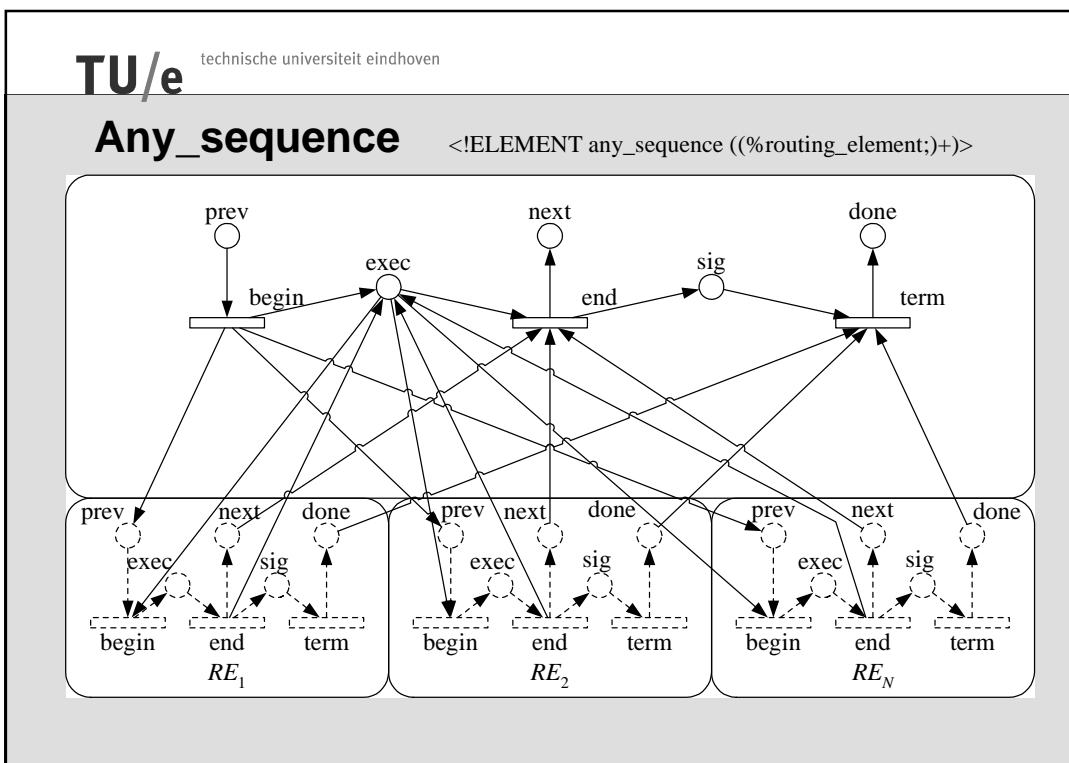
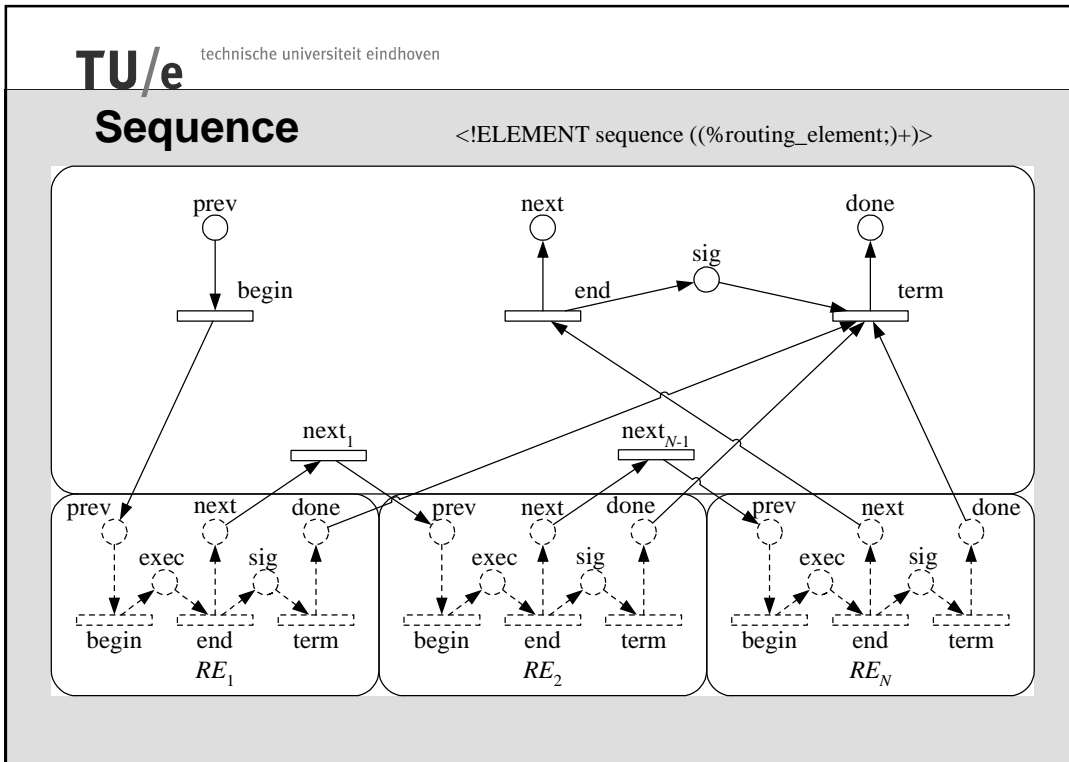
Also a DTD for organizational matters

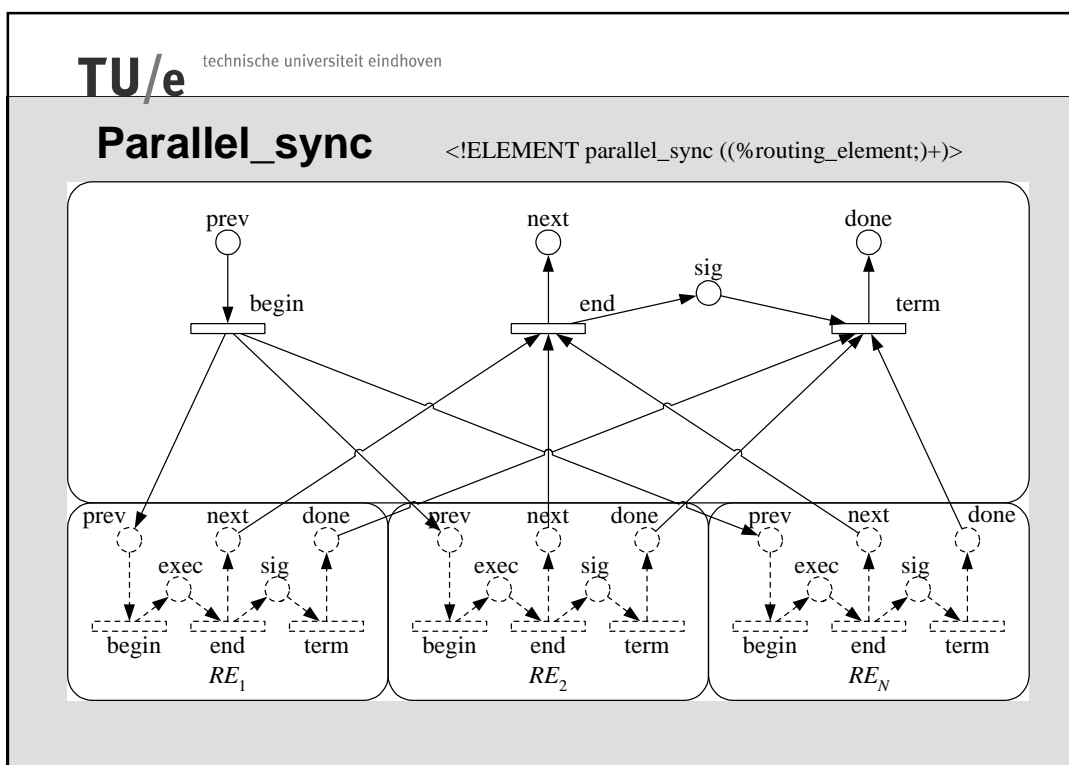
```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT organization (resources?,
  resource_types?, collections?,
  relations?)>
<!ELEMENT resources (user*, machine*,
  space*)>
<!ELEMENT user EMPTY>
<!ATTLIST user
  user_id IDREF #REQUIRED
  first_name CDATA #IMPLIED
  last_name CDATA #IMPLIED
  department IDREF #IMPLIED
  e-mail CDATA #IMPLIED
  login_id CDATA #IMPLIED
  address CDATA #IMPLIED
  phone CDATA #IMPLIED
  skills CDATA #IMPLIED
>
<!ELEMENT machine EMPTY>
<!ATTLIST machine
  machine_id IDREF #REQUIRED
  machine_name CDATA #IMPLIED
  description CDATA #IMPLIED
  number CDATA #IMPLIED
...

<!ELEMENT resource_types (role*,
  machine_type*, space_type*)>
<!ELEMENT role EMPTY>
<!ATTLIST role
  role_id IDREF #REQUIRED
  name CDATA #IMPLIED
  description CDATA #IMPLIED
>
<!ELEMENT can_inherit (role_ref,
  role_ref)>
<!ATTLIST can_inherit
  transitive_flag CDATA #IMPLIED
  restrictions CDATA #IMPLIED
>
<!ELEMENT can_delegate ((role_ref,
  role_ref) | (user_ref, user_ref) |
  (user_ref, role_ref) | (role_ref,
  user_ref))>
<!ATTLIST can_delegate
  transitive_flag CDATA #IMPLIED
  restrictions CDATA #IMPLIED
>
<!ELEMENT availability (user_ref*,
  machine_ref*, space_ref*)>
...
```

Petri-net Semantics

- DTD describes syntax but does not specify semantics
- Transformation to Petri net allows for analysis and enactment
- XRL document forms a tree
 - route element as root
 - child routing elements interface with parent elements
- Three examples: sequence, any_sequence, and parallel_sync

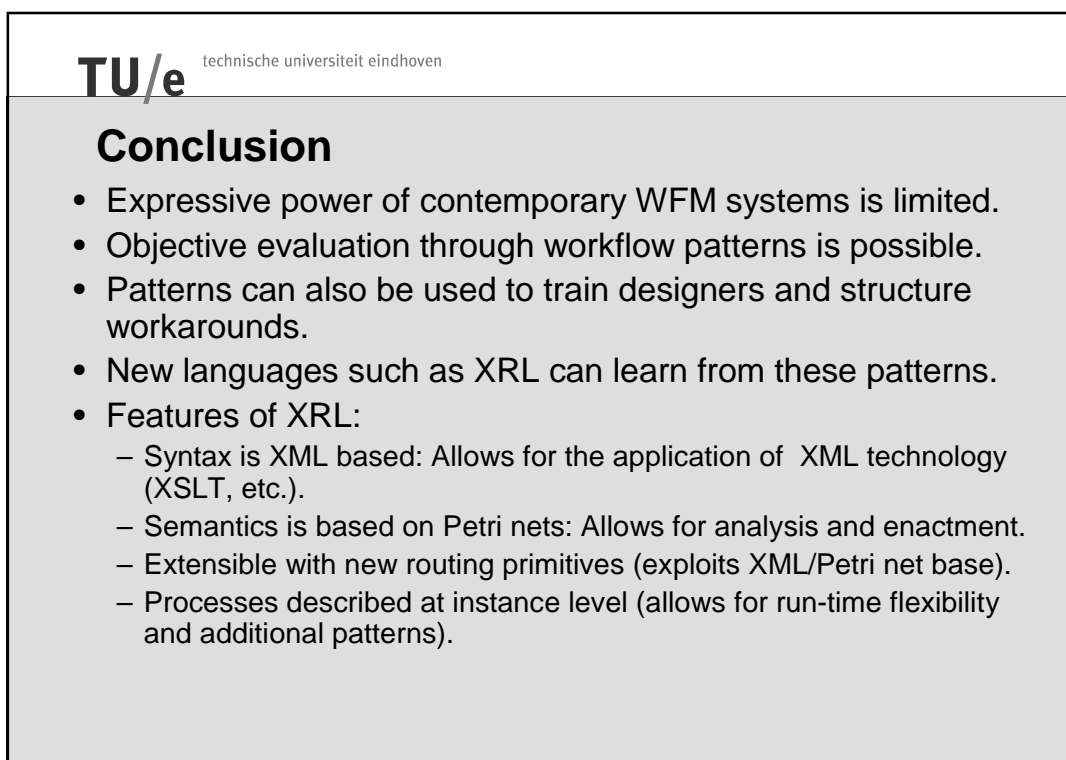
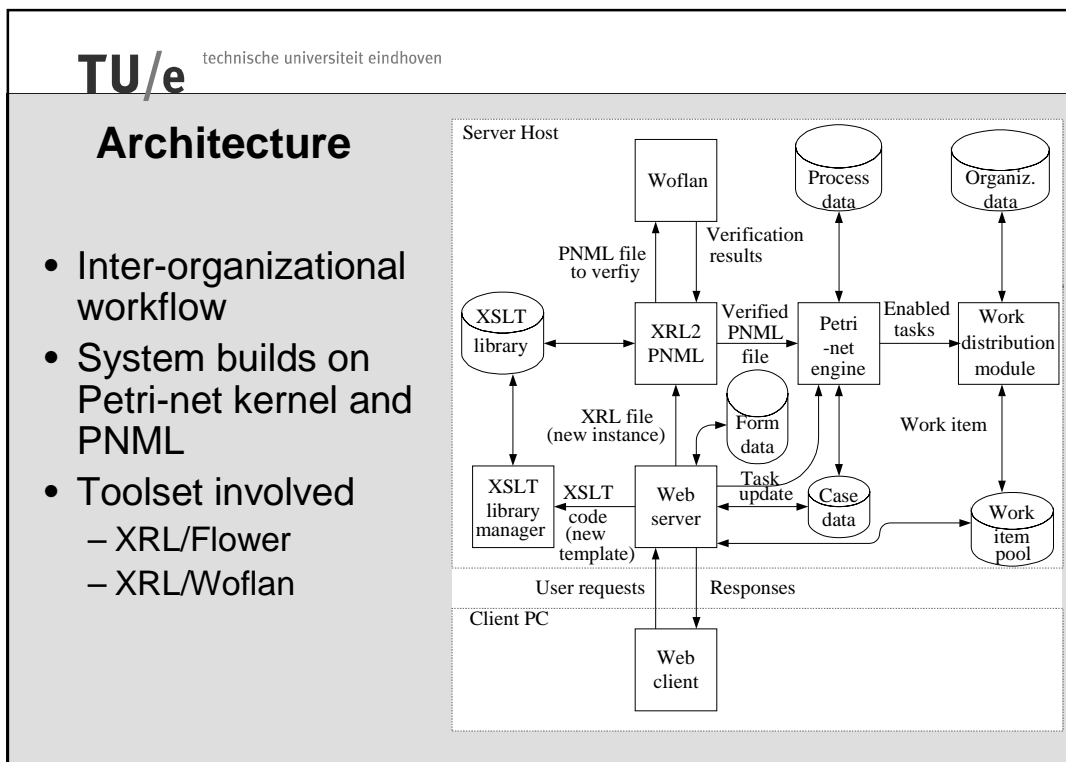




TU/e technische universiteit eindhoven

Benefits of Petri-net Semantics

- Analysis of correctness is possible
 - e.g. Woflan
- Efficient implementation of workflow engine
- Extendibility
 - DTD definition extension of XRL
 - XSLT supported translation to PNML
 - No changes of Petri-net engine required



How about the standards in this domain?

<i>pattern</i>	<i>standard</i>						
	XPDL	UML	BPEL	XLANG	WSFL	BPML	WSCI
Sequence	+	+	+	+	+	+	+
Parallel Split	+	+	+	+	+	+	+
Synchronization	+	+	+	+	+	+	+
Exclusive Choice	+	+	+	+	+	+	+
Simple Merge	+	+	+	+	+	+	+
Multi Choice	+	-	+	-	+	-	-
Synchronizing Merge	-	-	+	-	+	-	-
Multi Merge	-	-	-	-	-	+/-	+/-
Discriminator	-	-	-	-	-	-	-
Arbitrary Cycles	+	-	-	-	-	-	-
Implicit Termination	+	-	+	-	+	+	+
MI without Synchronization	-	-	+	+	+	+	+
MI with a Priori Design Time Knowledge	+	+	+	+	+	+	+
MI with a Priori Runtime Knowledge	-	+	-	-	-	-	-
MI without a Priori Runtime Knowledge	-	-	-	-	-	-	-
Deferred Choice	-	+	+	+	-	+	+
Interleaved Parallel Routing	-	-	+/-	-	-	-	-
Milestone	-	-	-	-	-	-	-
Cancel Activity	-	+	+	+	+	+	+
Cancel Case	-	+	+	+	+	+	+