Bridging the Heterogeneity of Orchestrations

– A Petri Net-based Integration of BPEL and Windows Workflow

Stefan Kolb, Jörg Lenhard and Guido Wirtz
Distributed Systems Group
University of Bamberg, Germany

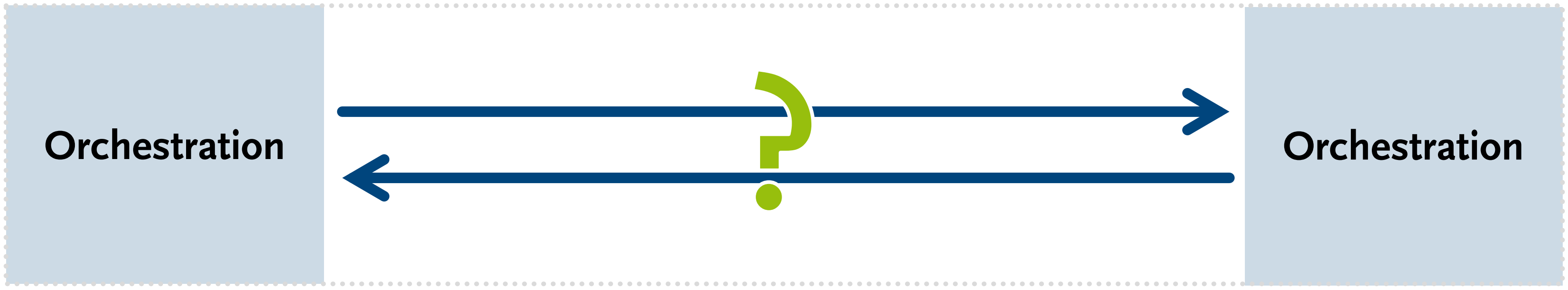
Why?

Orchestrations in today‘s process-aware IS have to deal with:

- **Incompatibility:** Services are often incompatible
  - Complex processes ⇨ manual integration is error prone & inefficient
  - Existing processes ⇨ bottom-up integration should be possible
- **Heterogeneity:** Variety of orchestration languages
  - Heterogeneity of communication
  - Heterogeneity of control flow description
- **Informatility:** (Mostly) No analyzable formal foundations
  - Difficult to prove properties like correctness of interaction

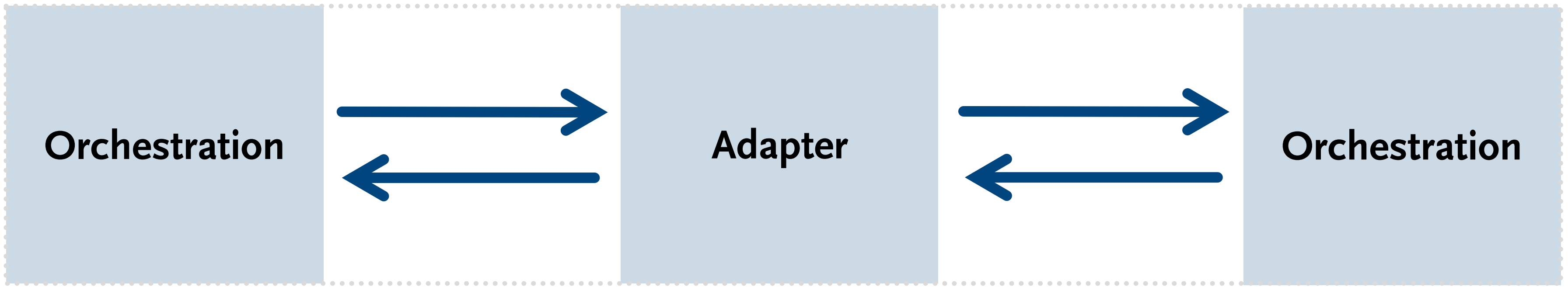
How can we tackle this?

⇨ Bridge **Incompatibility**, **Heterogeneity**, **Informality**



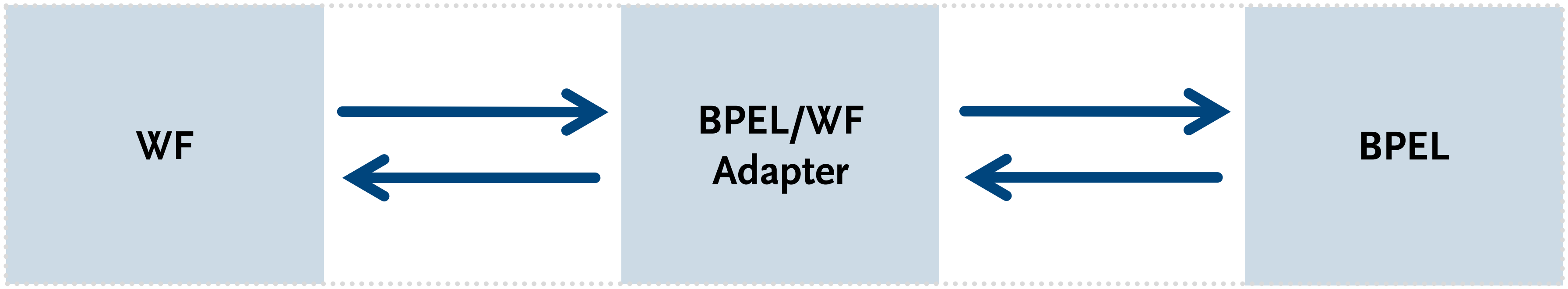
How can we tackle this?

⇨ Bridge **Incompatibility**, **Heterogeneity**, **Informality**



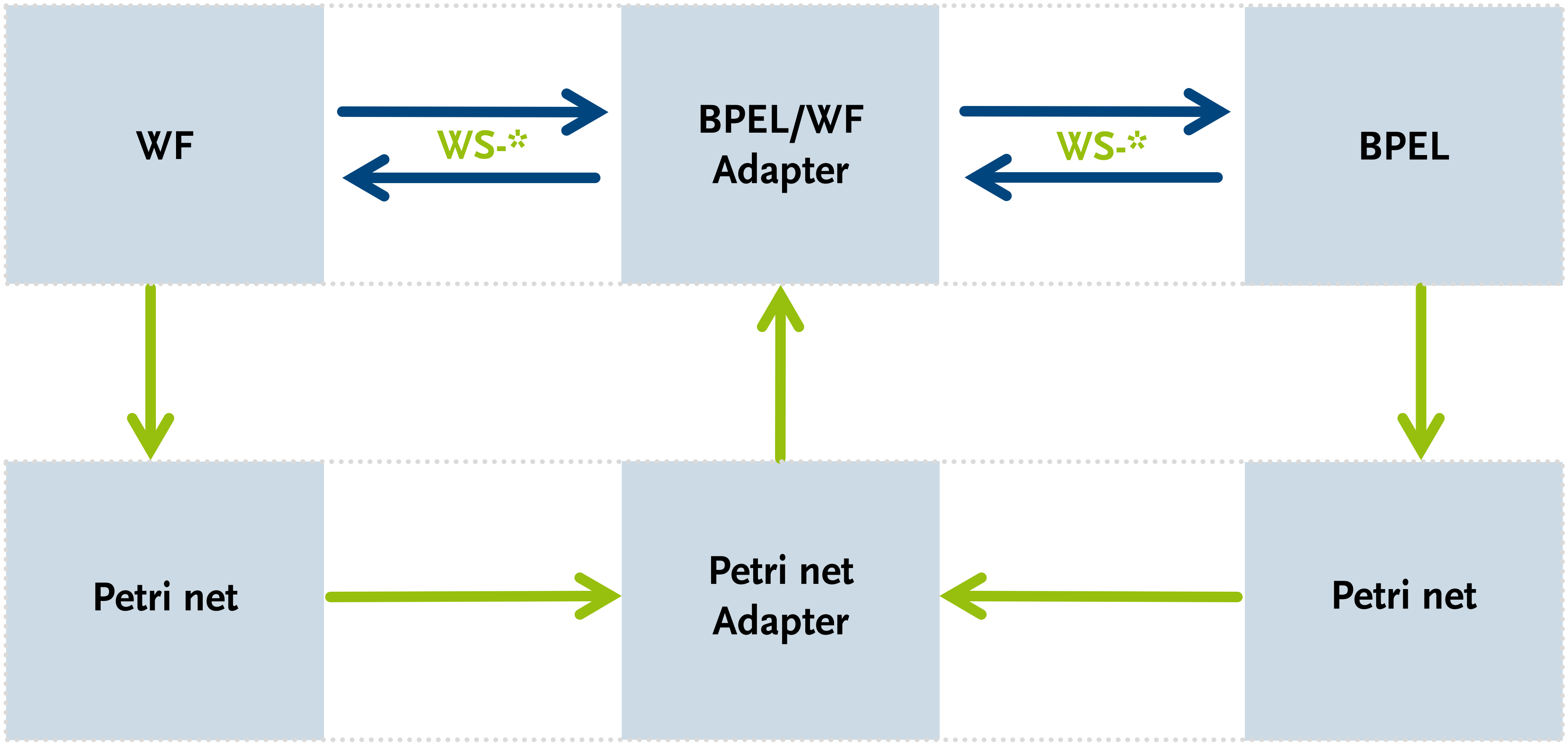
How can we tackle this?

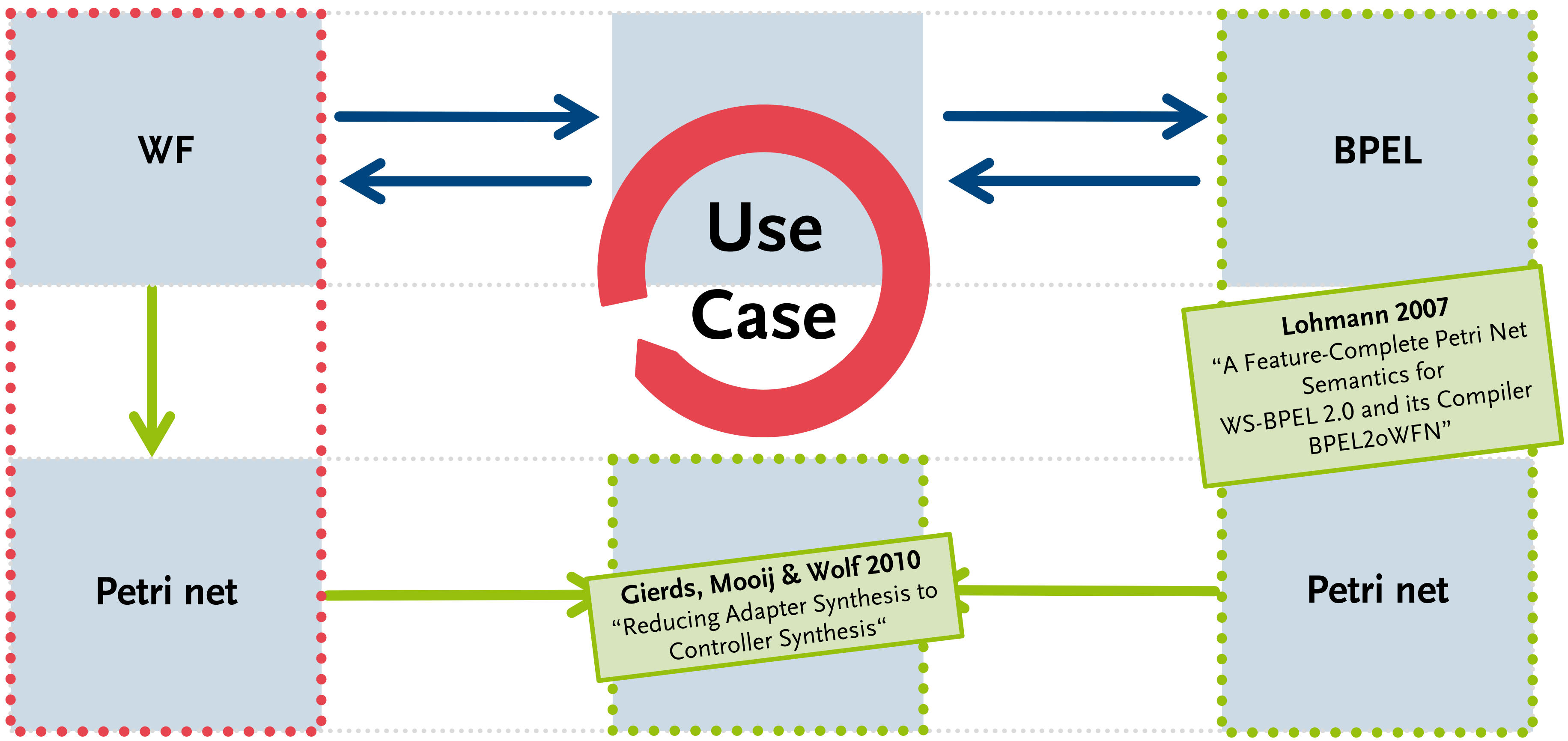
⇨ Bridge **Incompatibility**, **Heterogeneity**, **Informality**



How can we tackle this?

⇨ Bridge **Incompatibility**, **Heterogeneity**, **Informality**



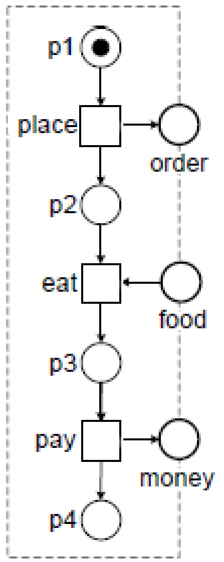

WF
Use Case
BPEL
Lohmann 2007
“A Feature-Complete Petri Net Semantics for WS-BPEL 2.0 and its Compiler BPEL2oWFN”
Petri net
Gierds, Mooij & Wolf 2010
“Reducing Adapter Synthesis to Controller Synthesis“
Petri net

Windows Workflow

- .NET API for process-based applications
- All-in-one solution (designer, runtime, ...)
- WS-communication with *WCF* makes it a natural candidate for orchestrations
- Block-structured, **graph-based** and **state machine** modeling styles
- One of the well known and widely spread languages

Open Workflow Nets

- PNs proved to be good for workflow modeling
- Several formal analysis methods available
- *Open Workflow Nets* (oWFNs) ⇨ PN services



GS_1

Input places

I = {food}

Output places

O = {order, money}

Final markings

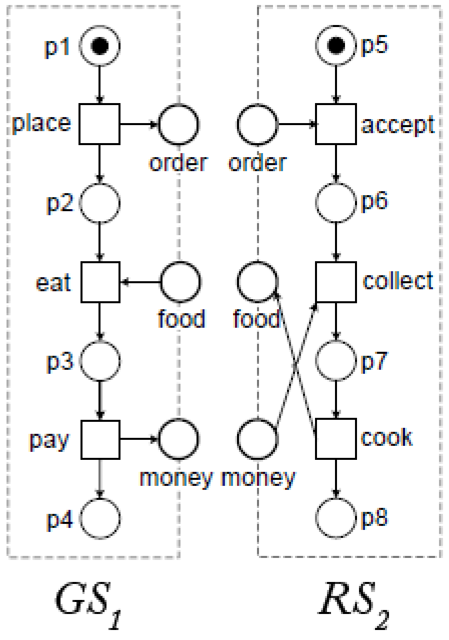
Ω = {[p4]}

[ASMW09]

Open Workflow Nets

- PNs proved to be good for workflow modeling
- Several formal analysis methods available
- *Open Workflow Nets* (oWFNs) ⇨ PN services

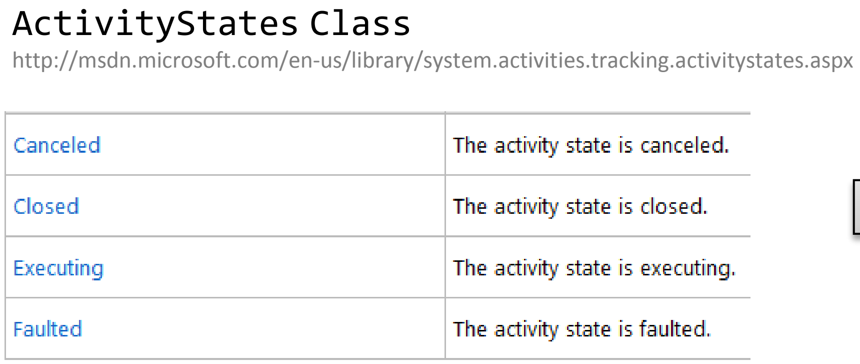
Composition
⇨ Place Fusion



[ASMW09]

Petri Net Semantics

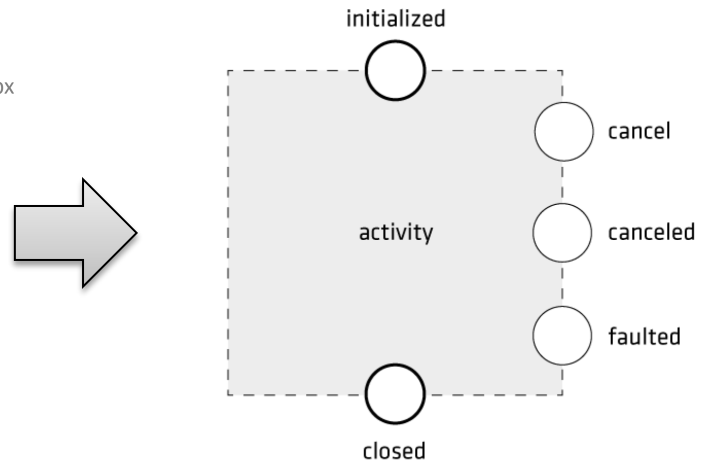
- *Hierarchical approach*: The different activities of a process are separately transformed into a petri net representation (*pattern*) and subsequently merged to a full process
- Common interface is derived of the activity lifecycle



ActivityStates Class

http://msdn.microsoft.com/en-us/library/system.activities.tracking.activitystates.aspx

Canceled	The activity state is canceled.
Closed	The activity state is closed.
Executing	The activity state is executing.
Faulted	The activity state is faulted.



⇨ **Currently only faultless termination and abstraction of data and time!**

Petri Net Semantics

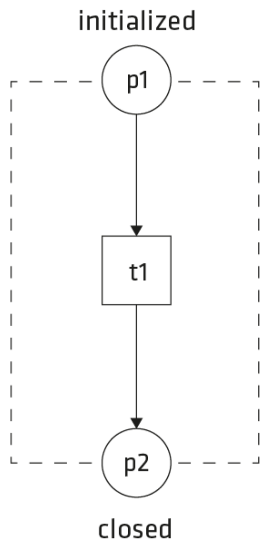
In Numbers

- 16 activities
- 89 % of control-flow activities
- 43 % of total activities

Activity	Supported	Not supported
Assign	X	
WriteLine	X	
Delay	X	
Receive	X	
Send	X	
ReceiveAndSendReply	X	
SendAndReceiveReply	X	
Sequence	X	
If	X	
Switch<T>	X	
Pick	X	
While	X	
DoWhile	X	
Parallel	X	
ForEach<T>		X
ParallelForEach<T>		X
Flowchart	X	
StateMachine	X	

Petri Net Semantics

Primitives

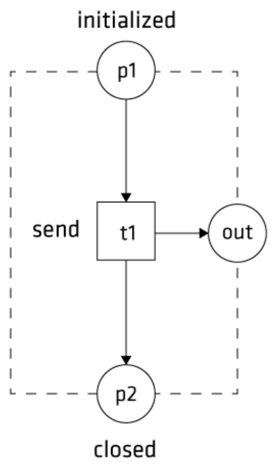


Assign, WriteLine, Delay

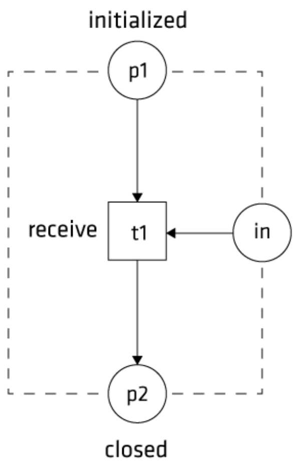
Activity	Supported	Not supported
Assign	X	
WriteLine	X	
Delay	X	
Receive	X	
Send	X	
ReceiveAndSendReply	X	
SendAndReceiveReply	X	
Sequence	X	
If	X	
Switch<T>	X	
Pick	X	
While	X	
DoWhile	X	
Parallel	X	
ForEach<T>		X
ParallelForEach<T>		X
Flowchart	X	
StateMachine	X	

Petri Net Semantics

Messaging



Send

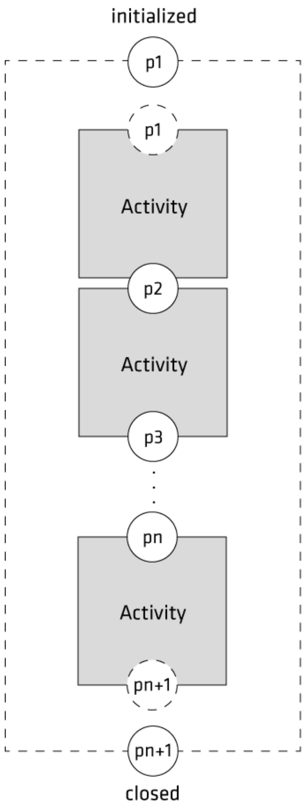


Receive

Activity	Supported	Not supported
Assign	X	
WriteLine	X	
Delay	X	
Receive	X	
Send	X	
ReceiveAndSendReply	X	
SendAndReceiveReply	X	
Sequence	X	
If	X	
Switch<T>	X	
Pick	X	
While	X	
DoWhile	X	
Parallel	X	
ForEach<T>		X
ParallelForEach<T>		X
Flowchart	X	
StateMachine	X	

Petri Net Semantics

Sequential/Block-structured

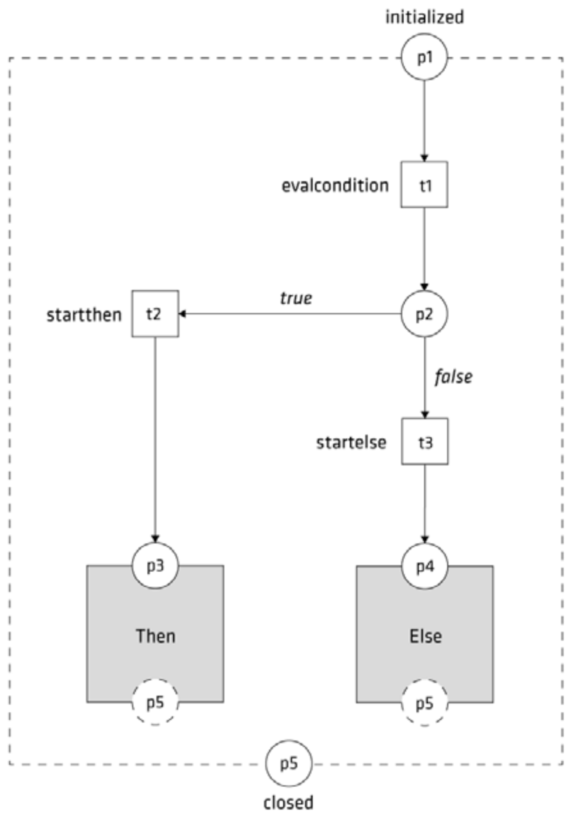


Sequence

Activity	Supported	Not supported
Assign	X	
WriteLine	X	
Delay	X	
Receive	X	
Send	X	
ReceiveAndSendReply	X	
SendAndReceiveReply	X	
Sequence	X	
If	X	
Switch<T>	X	
Pick	X	
While	X	
DoWhile	X	
Parallel	X	
ForEach<T>		X
ParallelForEach<T>		X
Flowchart	X	
StateMachine	X	

Petri Net Semantics

Sequential/Block-structured

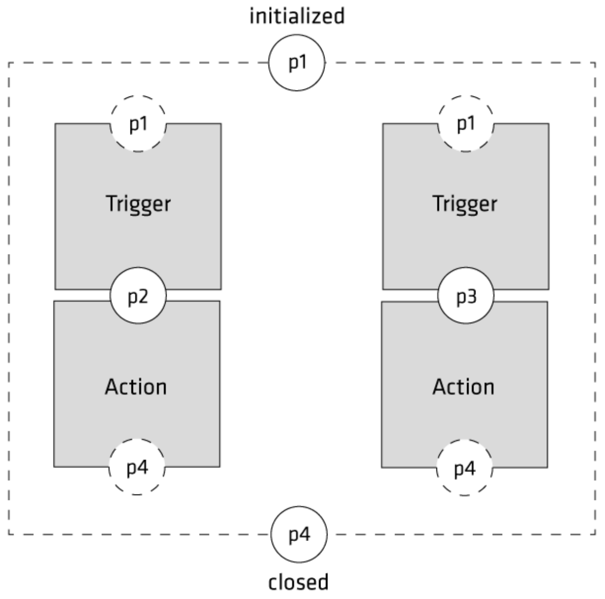


If

Activity	Supported	Not supported
Assign	X	
WriteLine	X	
Delay	X	
Receive	X	
Send	X	
ReceiveAndSendReply	X	
SendAndReceiveReply	X	
Sequence	X	
If	X	
Switch<T>	X	
Pick	X	
While	X	
DoWhile	X	
Parallel	X	
ForEach<T>		X
ParallelForEach<T>		X
Flowchart	X	
StateMachine	X	

Petri Net Semantics

Sequential/Block-structured



Pick

Activity	Supported	Not supported
Assign	X	
WriteLine	X	
Delay	X	
Receive	X	
Send	X	
ReceiveAndSendReply	X	
SendAndReceiveReply	X	
Sequence	X	
If	X	
Switch<T>	X	
Pick	X	
While	X	
DoWhile	X	
Parallel	X	
ForEach<T>		X
ParallelForEach<T>		X
Flowchart	X	
StateMachine	X	

You said automization!

- *Proof-of-Concept compiler* prototype *WF2oWFN*
- Implements all supported patterns
- Compatible with many tools from *service-technology.org* (Model checking, partner synthesis, …)
- Plugin-based structure for simple addition of `CustomActivities`
- Validated with 137 tests from two process libraries

[Len11, Mic10]

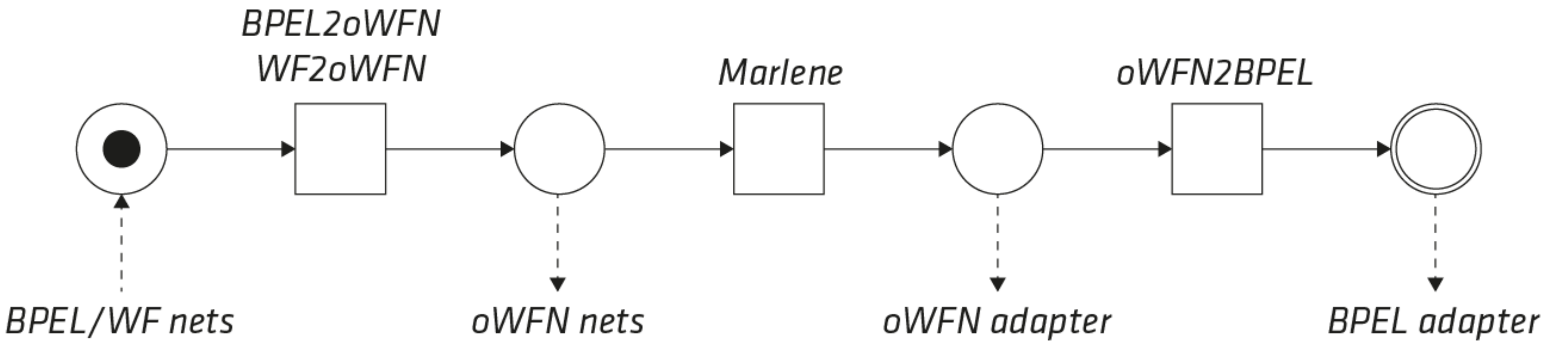
Try it!

`https://github.com/uniba-dsg/wf2owfn`

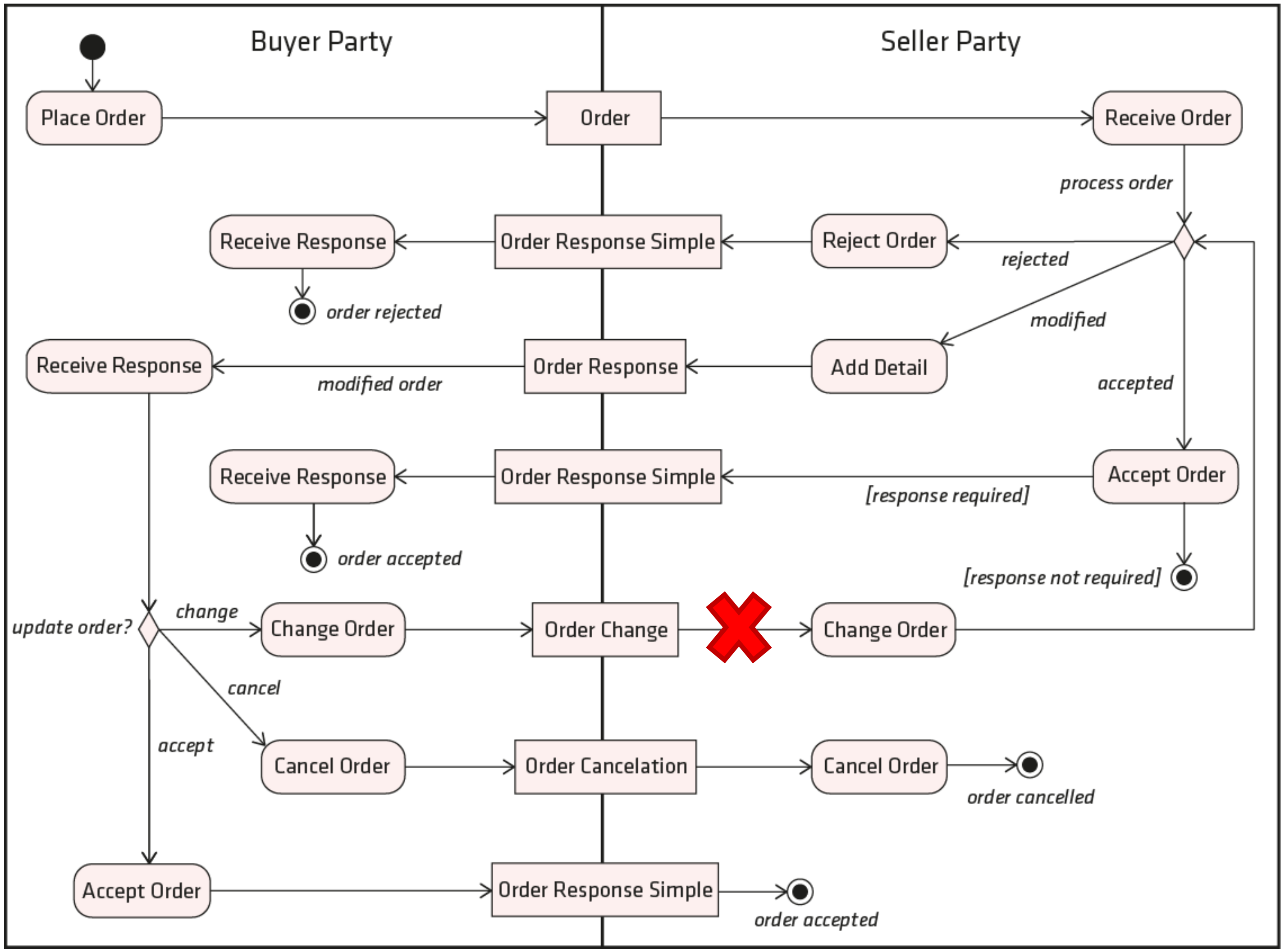
...and in practice?

- Standards-based *realistic* use case with continuous toolchain from WF/BPEL processes to an executable adapter process
- Taken from the *Universal Business Language* (UBL)

[OAS11]

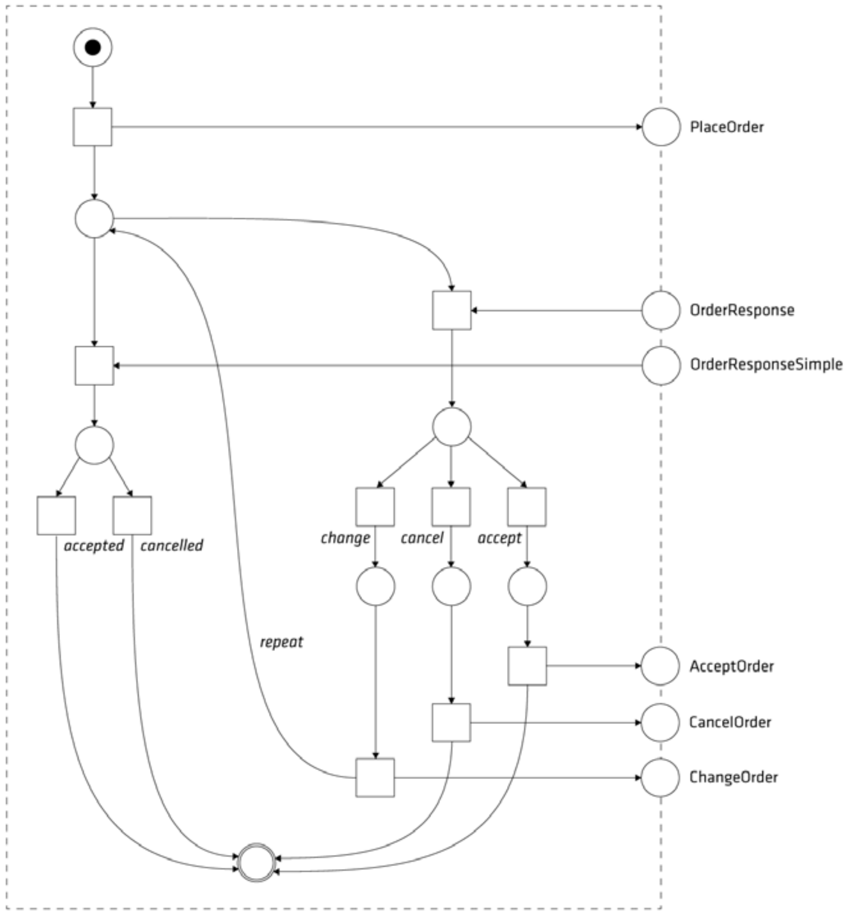


UBL Ordering Process [OAS11]

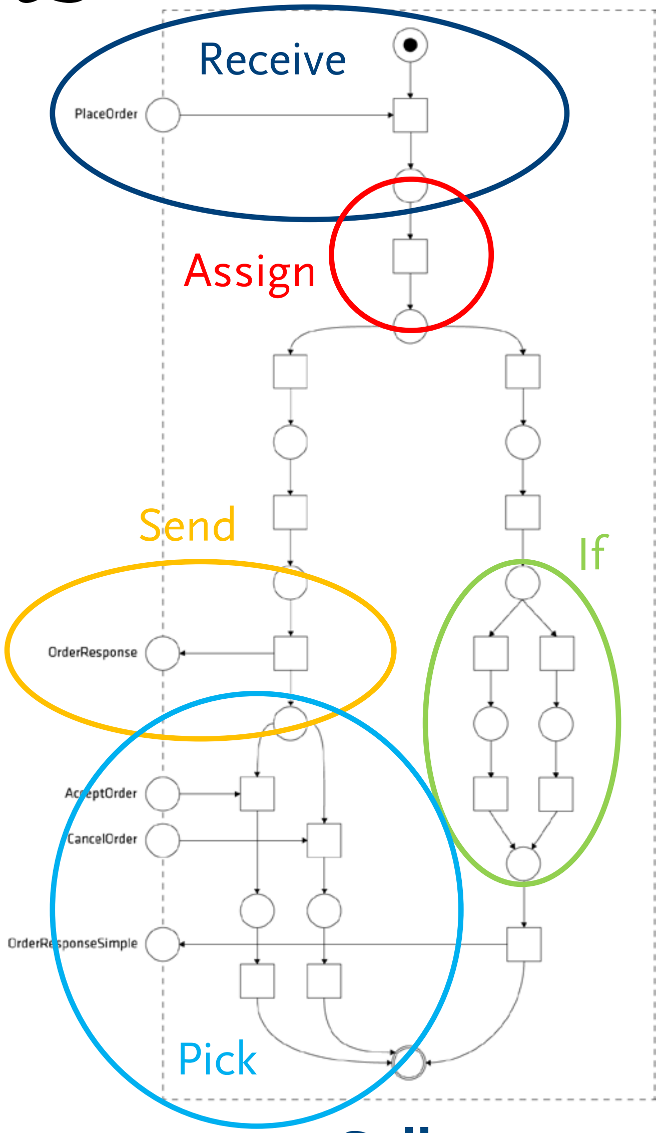


BPEL **WF**

UBL Petri Nets



Buyer



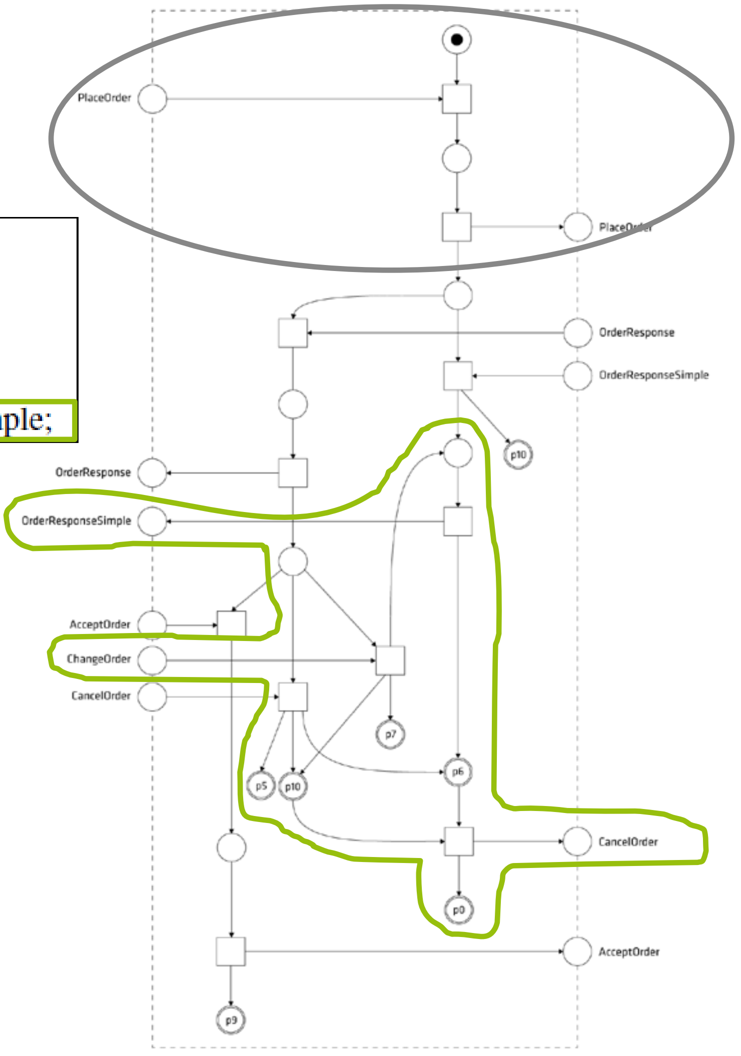
Seller

Generate an adapter!

Specification of Elementary Activities (SEA)

1	s.OrderResponseSimple	$\longmapsto$	b.OrderResponseSimple;
2	s.OrderResponse	$\longmapsto$	b.OrderResponse;
3	b.PlaceOrder	$\longmapsto$	s.PlaceOrder;
4	b.AcceptOrder	$\longmapsto$	s.AcceptOrder;
5	b.CancelOrder	$\longmapsto$	s.CancelOrder;
6	b.ChangeOrder	$\longmapsto$	s.CancelOrder, b.OrderResponseSimple;

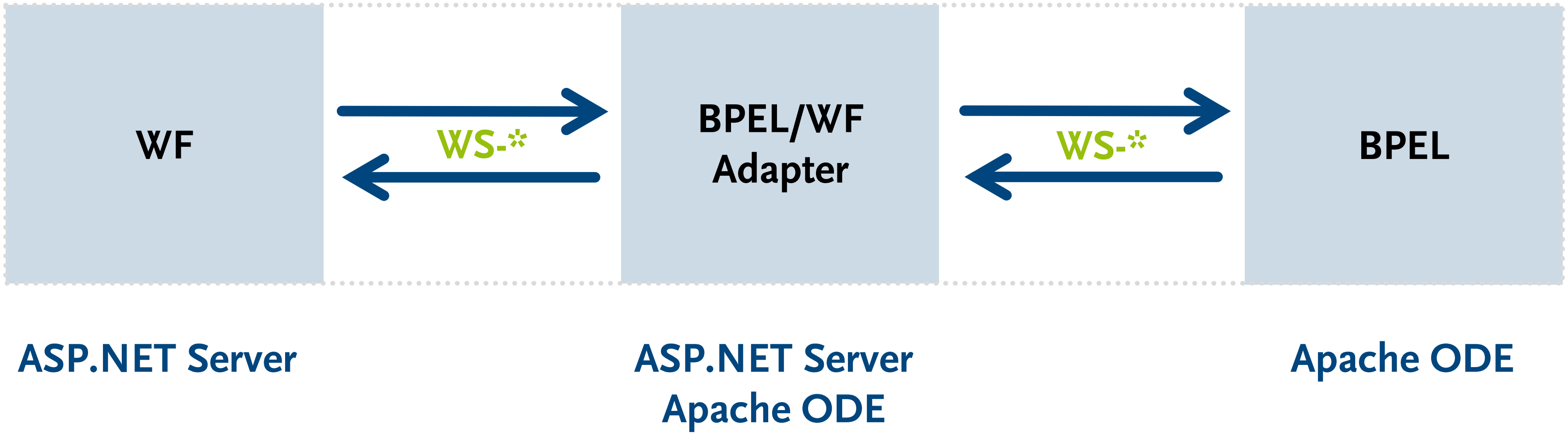
- Routing of syntactial equivalent message types (*rules 1-5*)
- Seller supports no *ChangeOrder*
  - ➪ Reject Buyer‘s request by an *OrderResponseSimple*
  - ➪ Reject Seller‘s modified order by a *CancelOrder* (*rule 6*)



$\Omega = \{ (p7, p0), (p0, p5), (p6, p10), (p9) \}$

Evaluation in Production

⇨ *Adapter*: Transformation to abstract BPEL with *oWFN2BPEL* and manual transformation to WF



What‘s still missing?

- Completion of standard activity library
- Extension with fault-, cancellation and compensation handling

- Abstraction of data and time aspects feasible for scientific work ⇨ problems in real world applications
- Currently we can only guarantee a **weak preservation of controllability** between the BP and the PN

Thank you!

Questions?!

References

[AMSW09]	W. van der Aalst, A. Mooij, C. Stahl und K. Wolf: *Service Interaction: Patterns, Formalization, and Analysis*, Formal Methods for Web Services, pp. 42-88, 2009.
[Mic10]	Microsoft: *Windows Communication Foundation (WCF) and Windows Workflow Foundation (WF) Samples for .NET Framework 4*, 2010, Available online at http://www.microsoft.com/en-us/download/details.aspx?id=21459
[Mic12]	Microsoft: *Xaml Object Mapping Specification 2009*, April 2012. Available online at http://download.microsoft.com/download/0/A/6/0A6F7755-9AF5-448B-907D-13985ACCF53E/[MS-XAML-2012].pdf
[Len11]	Lenhard, J.: *A Pattern-based Analysis of WS-BPEL and Windows Workflow*. Technischer Bericht 88, Fakultät Wirtschaftsinformatik und Angewandte Informatik, 2011.
[GMW10]	Gierds, C., A. J. Mooij und K. Wolf: *Reducing Adapter Synthesis to Controller Synthesis*. IEEE Transactions on Services Computing, 99:72-85,2010.
[OAS11]	OASIS: *Universal Business Language Version 2.1 - Committee Specification Draft 02 / Public Review Draft 02*, Mai 2011.

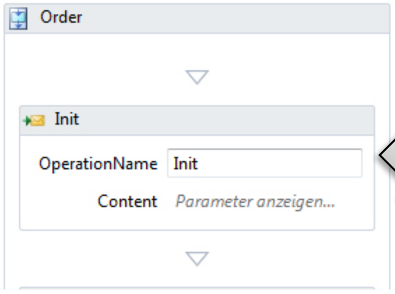
Backup

The “Specification“

- WF is mapped to a vocabulary of the *Extensible Application Markup Language* (XAML) [Mic12]
- XAML is one of Microsoft‘s Open Specifications

XAML Specification
.NET API
Unit Tests

WF XAML Specification



Order

Init

OperationName Init

Content *Parameter anzeigen...*

```
<Activity>
   <Sequence>
      <WriteLine Text="Hello Workflow" />
   </Sequence>
</Activity>
```

```
<Activity x:Class="NCNAME" namespace+>
   activity
</Activity>
```

```
<Sequence DisplayName="NCNAME"? standard-attributes>
   variables ?
   activity *
</Sequence>
```

```
<WriteLine
   Text="vb-expr"?
   TextWriter="vb-expr"?
   DisplayName="NCNAME"?
   standard-attributes />
```