Exploring the declarative nature of XProc 3.0

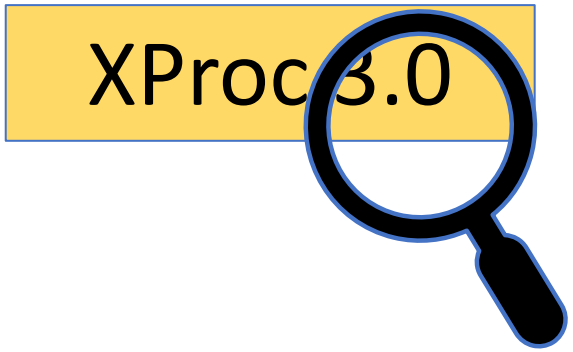
Achim Berndzen
www.xml-project.com
<xml-project />

Geert Bormans
www.c-moria.com
C-MORIA
<XML and Linked Data Solutions />

Agenda:

- What do we mean by “declarative”?
- A quick look at XProc 3.0
- Declarative aspects of XProc 3.0
- Take aways

What do we mean by “declarative”?



Declarative languages

Imperative languages

What do we mean by “declarative”?

XProc 3.0

Declarative languages

Fortran

Basic

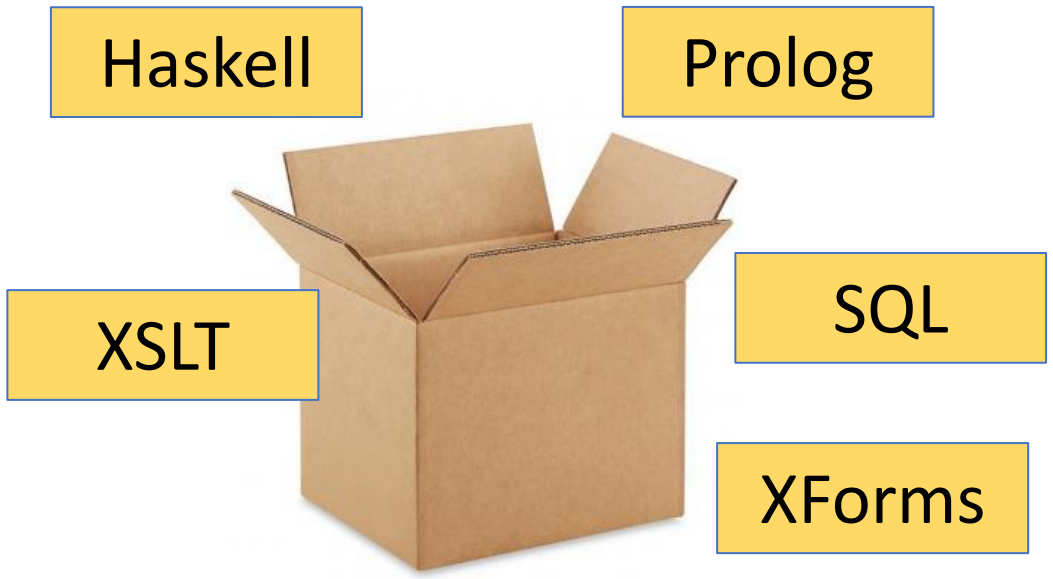
Java

C

Imperative languages

What do we mean by “declarative”?

XProc 3.0



Declarative languages

Imperative languages

What do we mean by “declarative”?

[…] declarative programming is a programming paradigm […] that expresses the logic of a computation without describing its control flow. […] For example:

- A high-level program that describes what a computation should perform.
- Any programming language that lacks side effects.
- A language with a clear correspondence to mathematical logic.

WIKIPEDIA
The Free Encyclopedia

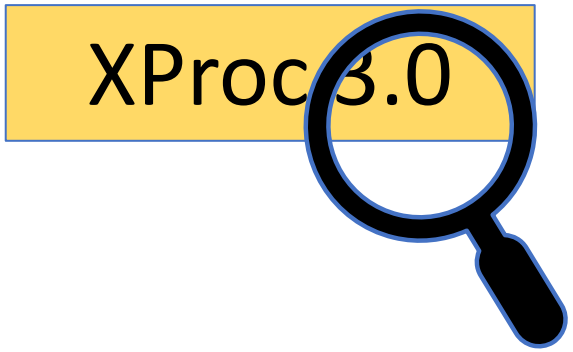
What do we mean by “declarative”?

[…] declarative programming is a programming paradigm […] that expresses the logic of a computation without describing its control flow. […] For example:

- A high-level program that describes what a computation should perform.
- Any programming language that lacks side effects.
- A language with a clear correspondence to mathematical logic.

WIKIPEDIA
The Free Encyclopedia

What do we mean by “declarative”?

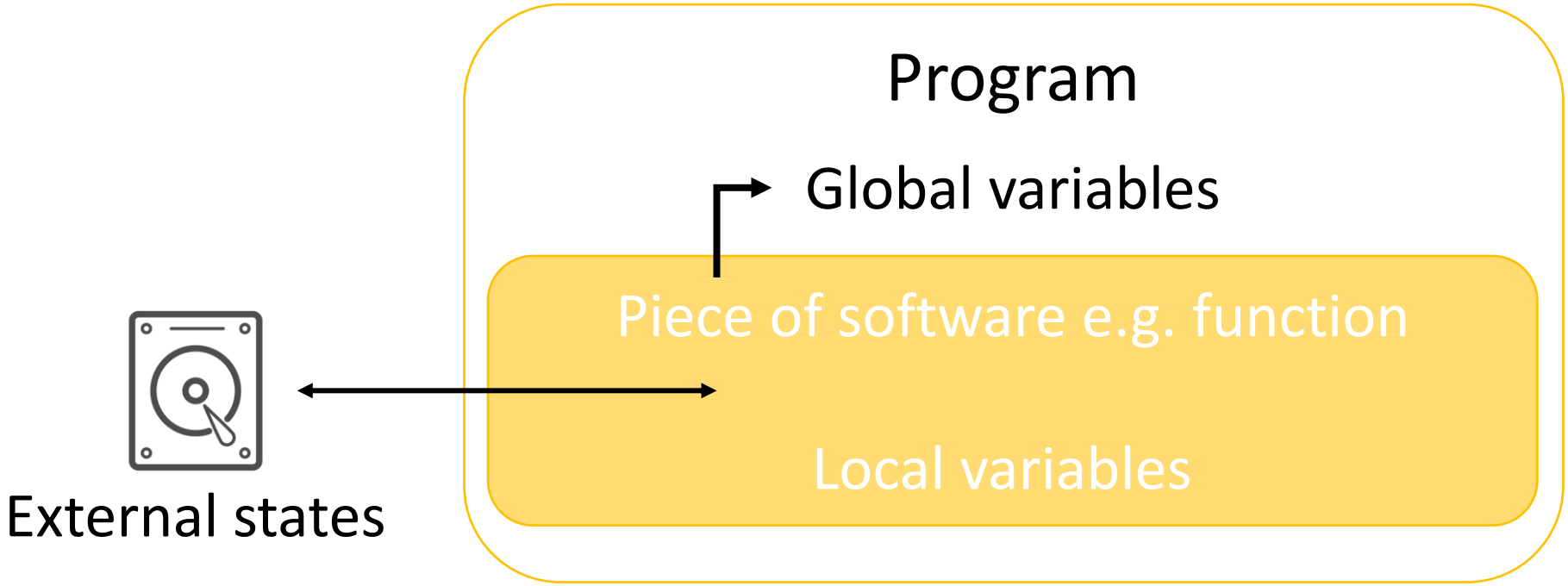


- No side effects.
- No flow control.
- Describe what a computation should perform.

Declarative languages

What do we mean by “declarative”?

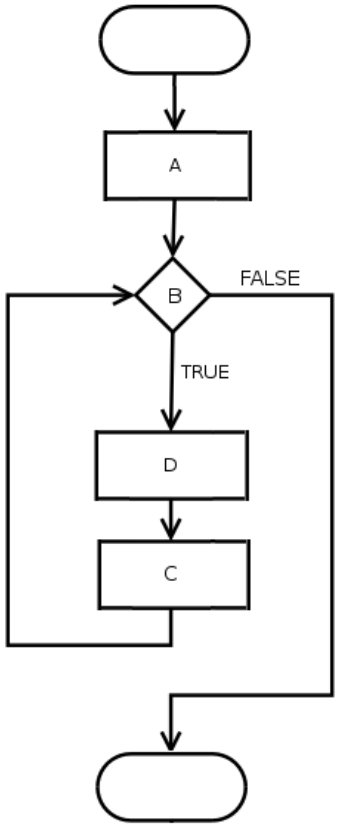
- No side effects?



Excluding side effects by design makes software easier to debug or maintain and allow better implementations. ➜ Deterministic functions in XPath.

What do we mean by “declarative”?

- No flow control?



- Imperative languages are different ways to express flow control.
- Declarative language (try to) avoid flow control:
  - First order logic, relational algebra, function calculus, etc.
- The implementations have to figure out the flow control for themselves.

What do we mean by “declarative”?

- Describe what a computation should perform?

- In other words: “describe the desired results without explicitly listing commands or steps that must be performed.”
- The result can be described in natural language or a domain specific expert language.
- The translation into a computer language is done by the implementation.
- However: Describing steps to archive a result is a part of natural languages and expert languages as well.

What do we mean by “declarative”?

- Describe what a computation should perform?

! Describing steps to produce a result can be on different levels of abstraction.

Prepare a Lasagna!

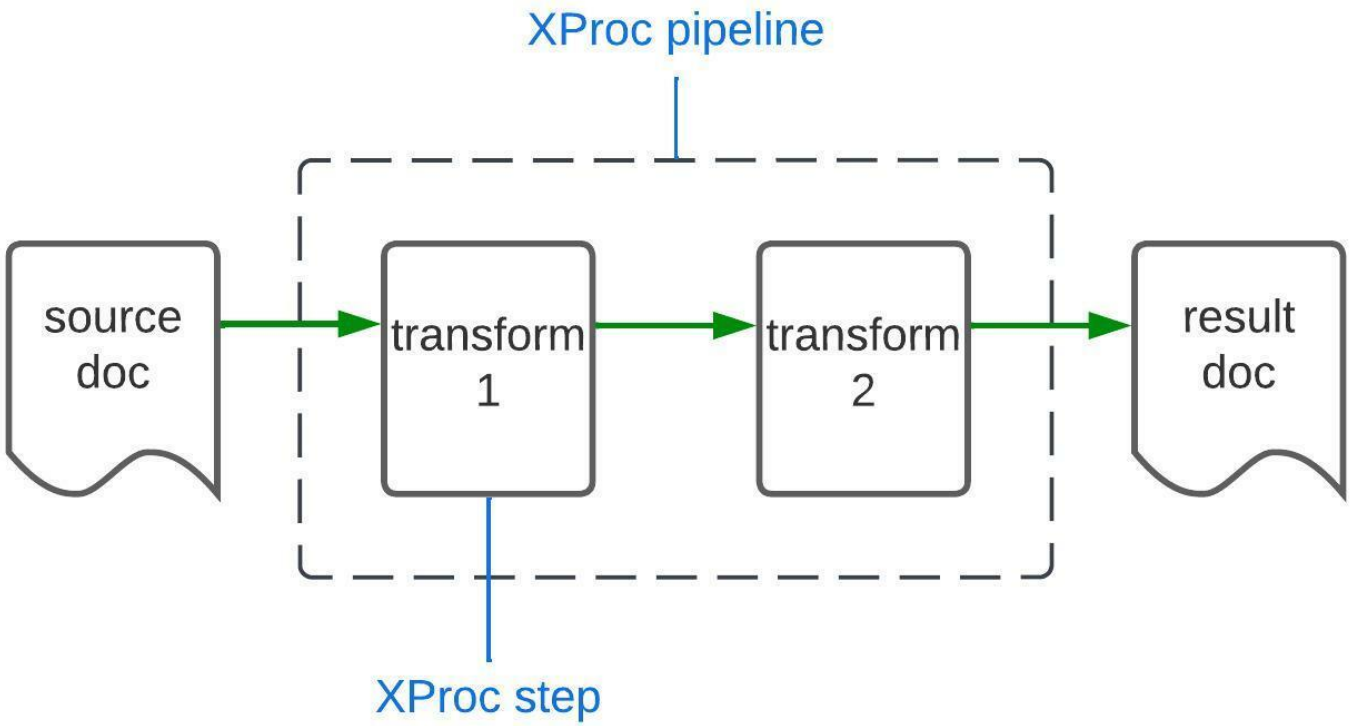
1. Prepare a Bolognese and a Bechamel.
2. Layer them with noodles and top with cheese.
3. Bake for 40 minutes at 200C.

Lasagna recipe from beginners cookbook

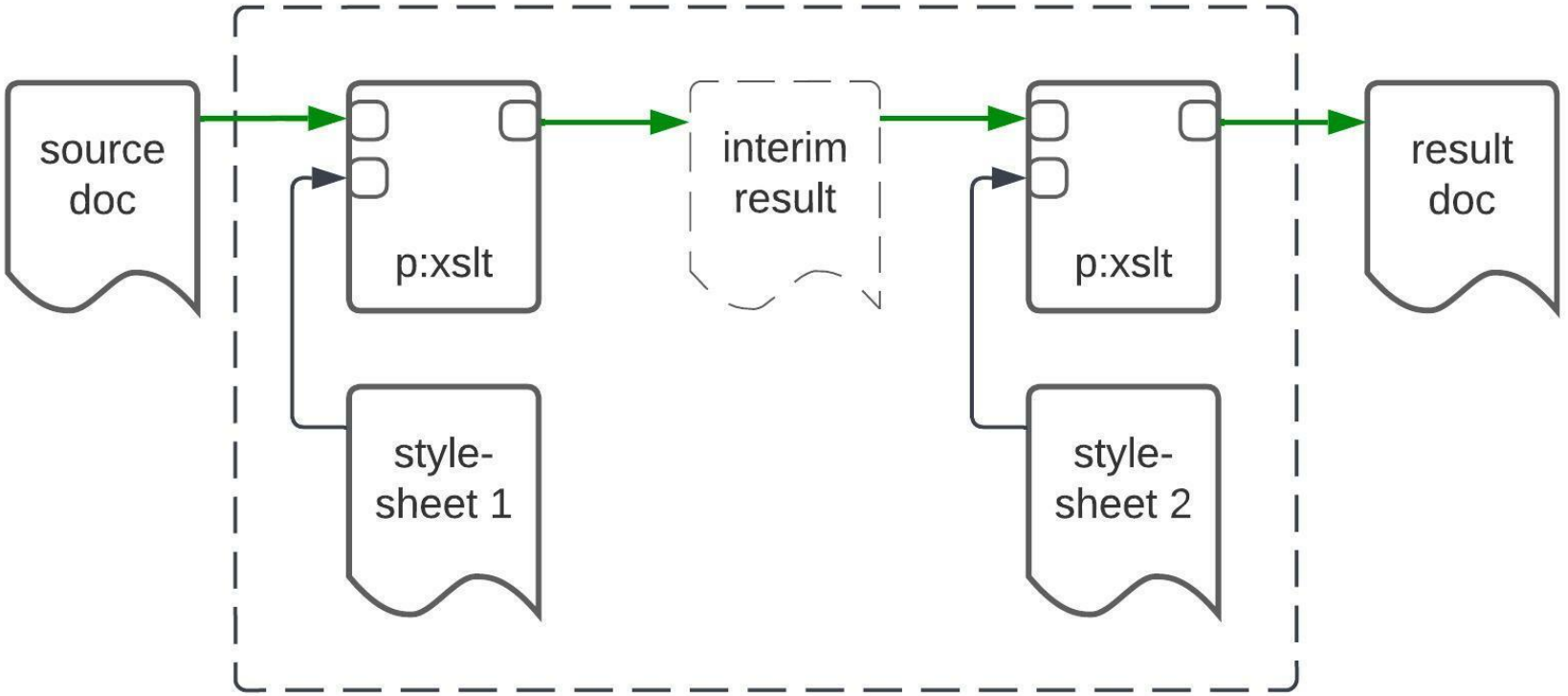
A (very) quick look at XProc 3.0

- XProc 3.0 is “a language for describing operations to be performed on documents“. (spec)
  - Pipelines specifying a sequence of operations
  - Programming language
  - XML based syntax
- XProc 3.0 has a formal specification as a result from a Community Group effort

Simple example (functional)



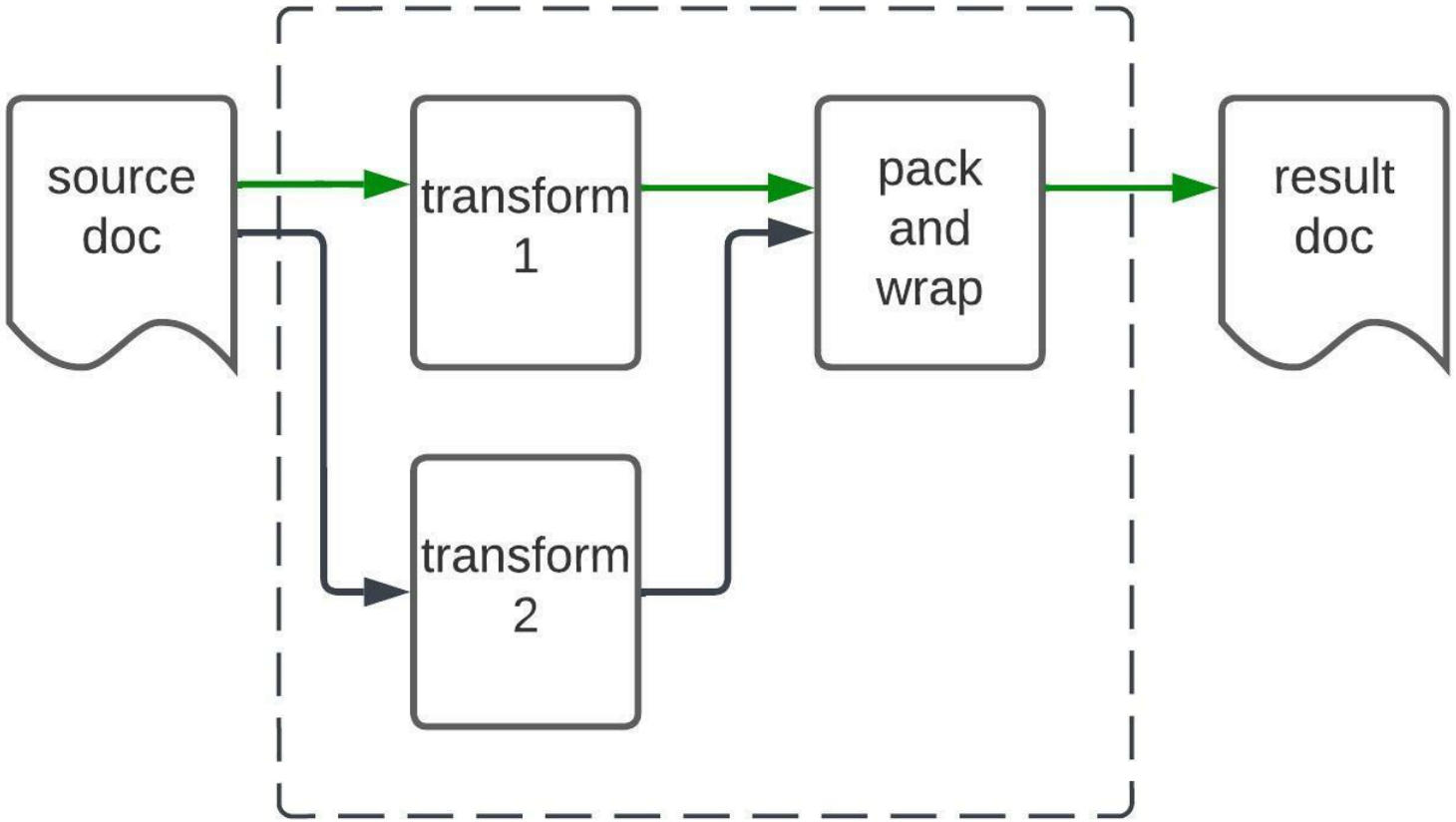
Simple example (technical)



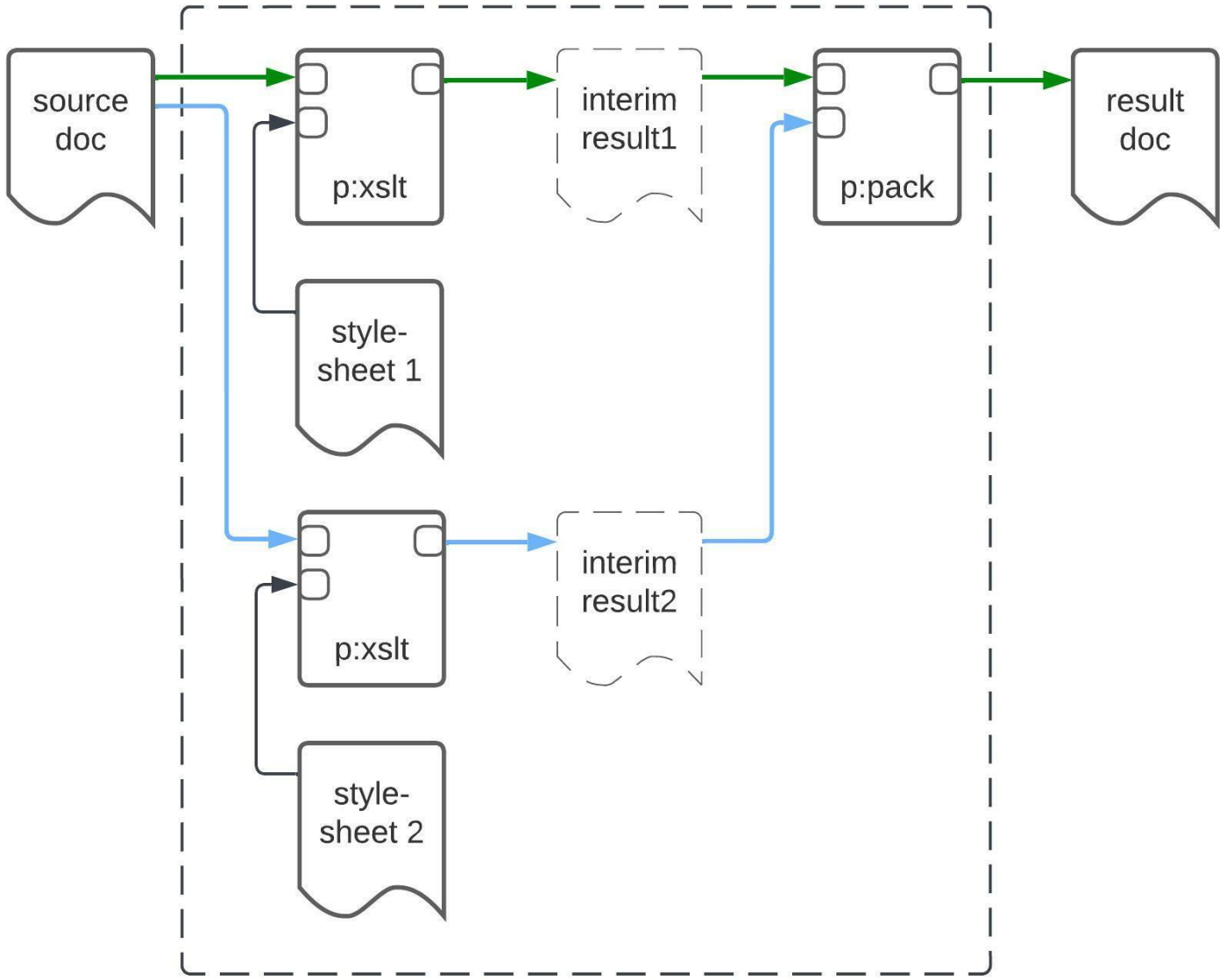
Simple example (code)

```
<p:declare-step xmlns:p="http://www.w3.org/ns/xproc"
version="3.0">
    <p:input port="source"/>
    <p:output port="result"/>

    <p:xslt version="3.0">
        <p:with-input port="stylesheet" href="stylesheet1.xsl"/>
    </p:xslt>

    <p:xslt version="3.0">
        <p:with-input port="stylesheet" href="stylesheet2.xsl"/>
    </p:xslt>

</p:declare-step>
```

Somewhat more evolved example (functional)



Somewhat more evolved example (technical)

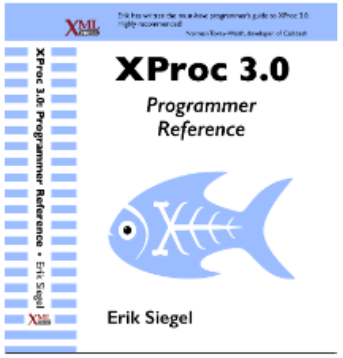


Somewhat more evolved example (code)

```
<p:identity name="src"/>

<p:xslt version="3.0" name="tfx2">
   <p:with-input port="stylesheet" href="stylesheet2.xsl"/>
</p:xslt>

<p:identity>
   <p:with-input pipe="@src"/>
</p:identity>

<p:xslt version="3.0">
   <p:with-input port="stylesheet" href="stylesheet1.xsl"/>
</p:xslt>

<p:pack wrapper="pack">
   <p:with-input port="alternate" pipe="@tfx2"/>
</p:pack>
```

References

- CG report (formal specification)
  - https://spec.xproc.org/3.0/xproc/
- Standard step library
  - https://spec.xproc.org/3.0/steps/
- Community Group
  - https://www.w3.org/community/xproc-next/
- Mailing list
  - xproc-dev-request@w3.org
- Introduction (entry point)
  - https://xproc.org/introduction.html



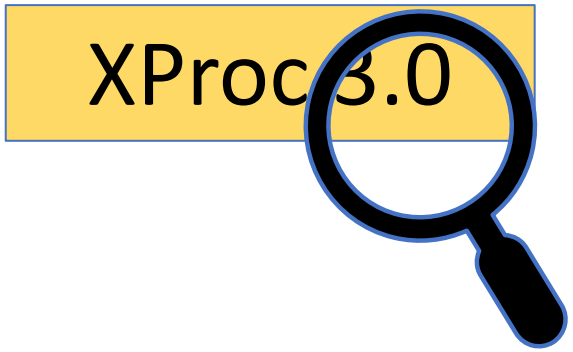
Implementations (with published test results)

- MorganaXProc-III
  - https://www.xml-project.com/morganaxproc-iii/
  - Status test coverage: 100% *
- XML Calabash
  - https://xmlcalabash.com/
  - Status test coverage: 80% *

* https://test-suite.xproc.org/implementation.html

Declarative aspects of XProc 3.0

Recap:



- No side effects.
- No flow control.
- Describe what a computation should perform.

Declarative languages

Declarative aspects of XProc 3.0

No side effect?

- There are a lot of steps producing external side effects:
  - p:store, p:http-request, p:file-delete, …
  - Steps can even consume external side effects of other step with p:store href="x" and a later p:load href="x".
- Deterministic XPath functions yield different result when called on different steps, i.e. are open to side effects.

Declarative aspects of XProc 3.0

No side effect?

- But: No side effects from changing global variable states!
- XProc 3.0 has global variable states, but the are immutable.
- You can declare a new variable with the same name and a different value,
- but this new value will disappear at scope’s end.

Declarative aspects of XProc 3.0

No side effect?

Imperative, e.g. Java

```
int var = 4;
for (int i=1; i < 3; i++)
        var = var +1;
System.out.println(var); /* var is 6 */
```

Declarative aspect in XProc 3.0

```
<p:variable name="var" select="4" />
<p:for-each> /*Iterate over a sequence of 2 docs*/
  <p:variable name="var" select="$var +1" />
</p:for-each>
/*var is 4 */
```

Declarative aspects of XProc 3.0

No flow control?

- Imperative: Layout flow control.
- Declarative: Implementation creates flow control.

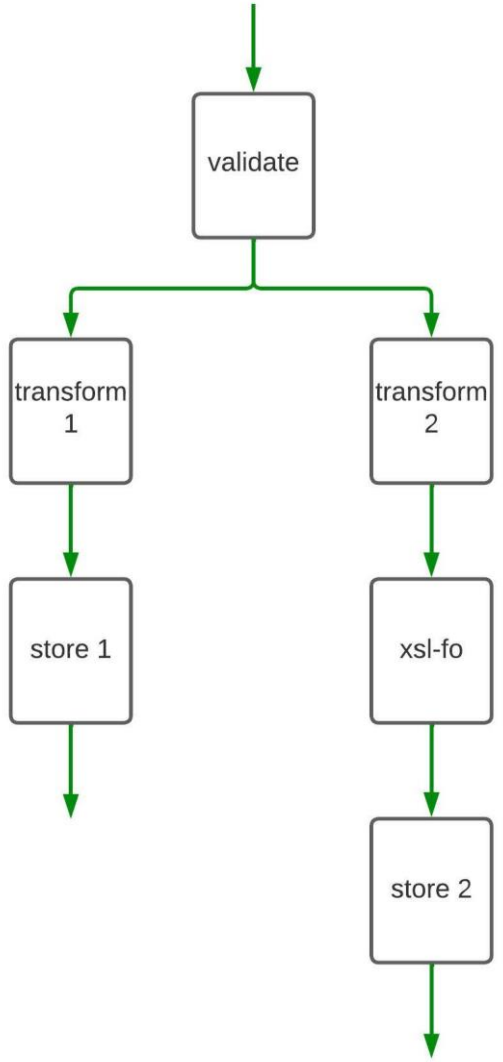
“XProc is a language in XML which provides a set of commands for a flow control in order to generate XML oriented workflows.”
https://www.data2type.de/en/xml-xslt-xslfo/xproc

Case closed?

Declarative aspects of XProc 3.0

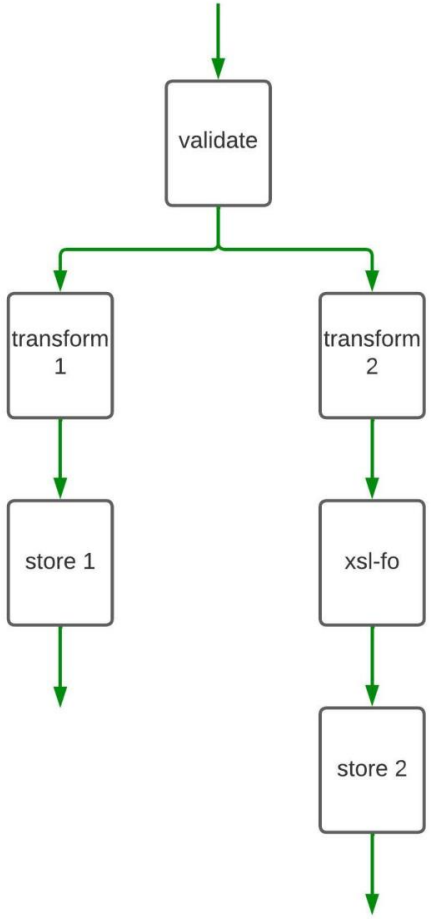
No flow control?

```
<p:validate-with-xml-schema name="schema-validation">
   <p:with-input port="schema" href="schema.xsd" />
</p:validate-with-xml-schema>
<p:xslt name="xslt1">
   <p:with-input port="stylesheet" href="style1.xsl" />
</p:xslt>
<p:store name="store1" href="result1.xml" />
<p:xslt name="xslt2">
   <p:with-input pipe="@schema-validation" />
   <p:with-input port="stylesheet" href="style2.xsl" />
</p:xslt>
<p:xsl-formatter content-type="application/pdf" />
<p:store name="store2" href="result.pdf" />
```



Declarative aspects of XProc 3.0

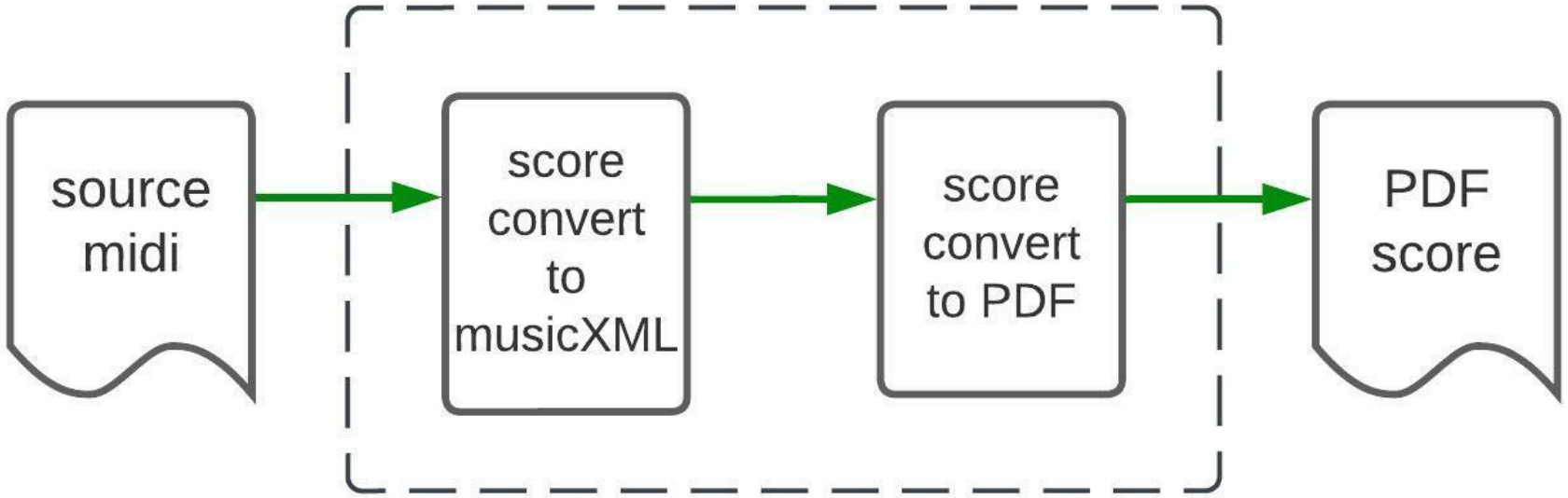
No flow control?



- XProc's flow is not execution order!
- Connections only restricts execution order.
  - validate >> transform1 >> store1
  - validate >> transform2 >> pageformat >>store2
- Two concepts of "flow control":
  - Connection between steps vs.
  - Computers execution plan.
- XProc engines evaluate execution plan(s).

Declarative aspects of XProc 3.0

Describe what a computation should perform?



Declarative aspects of XProc 3.0

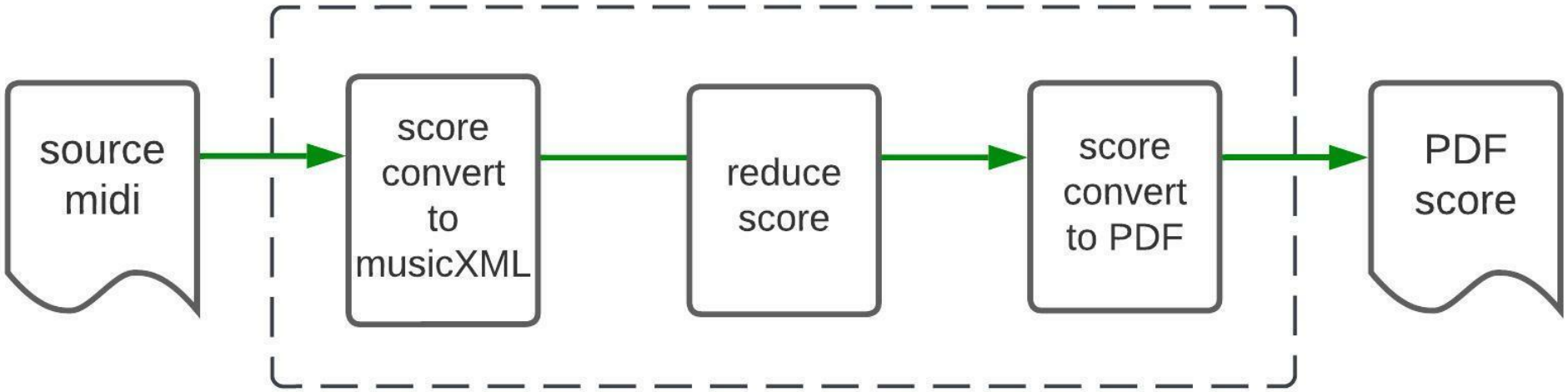
```
<p:import href="lib-music.xpl"/>

<p:load  href="dueling-banjos.mid" content-type="audio/midi"/>

<mox:convert-score content-type="application/xml"/>

<mox:convert-score content-type="application/pdf"/>

<p:store href="score.pdf"/>
```

Declarative aspects of XProc 3.0

```
<p:library xmlns:p="http://www.w3.org/ns/xproc"
    xmlns:mox="http://www.xml-project.com/morganaxproc"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    version="3.0">

    <p:declare-step type="mox:convert-score"
    mox:class="mox.morganaxproc.midi.MidiHandler">
        <p:input port="source" content-types="xml audio/midi"/>
        <p:option name="content-type" required="true" as="xs:string"/>
        <p:output port="result"/>
    </p:declare-step>

</p:library>
```

Declarative aspects of XProc 3.0

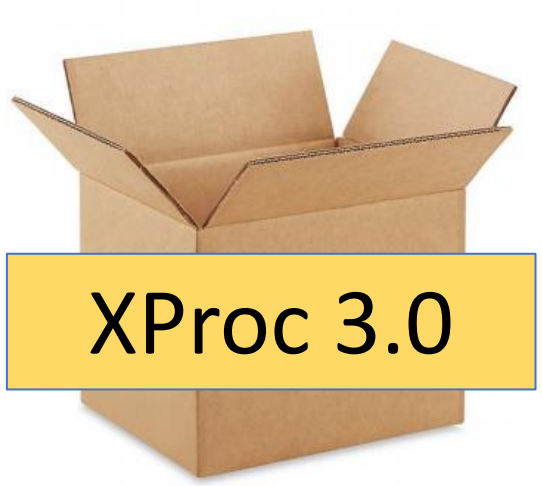
Describe what a computation should perform?



Declarative aspects of XProc 3.0

```
<p:import href="lib-music.xpl"/>
<p:load href="dueling-banjos.mid" content-type="audio/midi"/>

 <mox:convert-score content-type="application/xml"/>

 <p:xslt version="3.0">
    <p:with-input port="stylesheet" href="reduction.xsl"/>
    <p:with-option name="parameters" select="map{'parts' : ('P1', 'P2')}"/>
 </p:xslt>

 <p:store href="score-reduced.xml" serialization="map{'indent' : true()}"/>

 <mox:convert-score content-type="application/pdf"/>
 <p:store href="score-reduced.pdf"/>
```

Take aways



Declarative languages

- No side effects. +/-
- No flow control. ✓
- Describe what a computation should perform. ✓

Why this talk?

- Add some body to the claim XProc has a declarative nature
- Provide a handle to claim
  - XProc 3.0 helps solving problems in a short amount of time
  - XProc 3.0 is easy to learn

Thank You
Questions?

Achim Berndzen
www.xml-project.com
<xml-project />

Geert Bormans
www.c-moria.com