

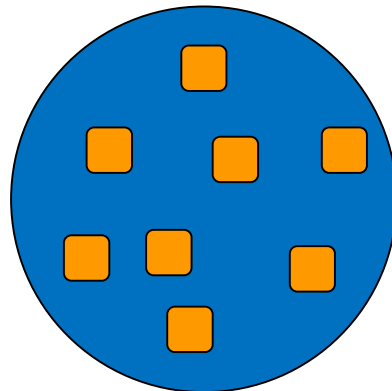
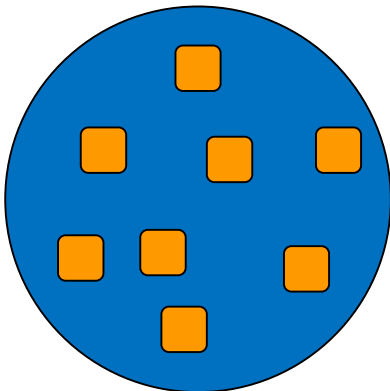
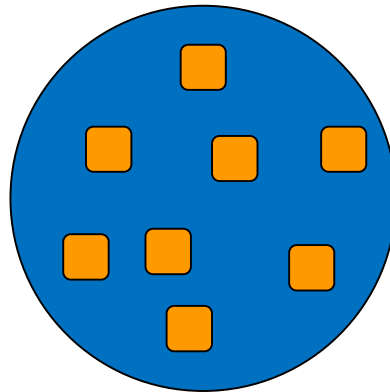
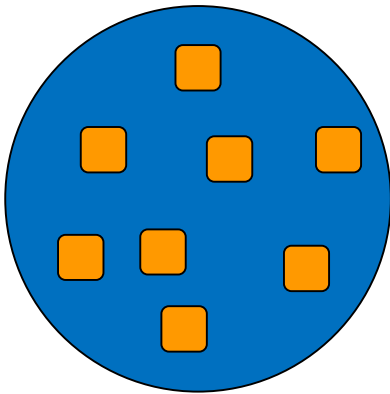
Topic tools

- a brief introduction.

Hans-Jürgen Rennau, Traveltainment GmbH
February 13, 2015



Aggregation of functionality is brain-friendly.



2015-02-13

Topic Tools

**Class /
Method**

**Module /
Function**

**Service /
Operation**

**TopicTool /
Operation**



Topic tool = command-line tool

*Single point of access
to a range of functionality
dedicated to a particular topic.*

xsdv - XSD reporting

logspy - logdata analysis

mockMachine - test generator



Topic Tools

*a lightweight-framework
(in fact a topic tool)
helping you to write*

topic tools





What else?

Generic command-text model.

operation

parameters

Boolean parameters

*tree?xsd=srv.xsd, enames=*RQ, noprefix, ~skipAnno*

=

*tre?x=srv.xsd, en=*RQ, nop, ~skipa*

Syntactic shortcuts

Anything else?



*Uniform **help** operation.*

*Uniform, aggregated **error reporting***

*Uniform set of **smart parameter types***

User perspective: a simple example ...



- Command-text syntax
- Help
- Error reporting
- Smart parameters
- Syntactic shortcuts

Topic Tools, the ...



Developer perspective



Operation = anno + code



```
(:~@operations
  <operations>
    <operation name="foo" type="..." func="foo">
      <param name="boo" type="..." />
      <param name="bar" type="..." />
    </operation>
  </operations>
:)
```

```
declare function f:foo($request as element())
  as ... {
  let $boo := tt:getParam($request, 'boo')
  let $bar := tt:getParam($request, 'bar')
  return f:whatnot($boo, $bar)
};
```



Workflow

- Instantiate `"request=new?dir=/tta/foo
mod=xyz, ops=fooOp barOp" ttools.xq`

● *Write code & edit annotations*

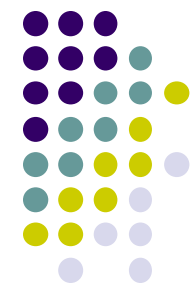
- Build `"request=build?dir=/tta/foo" ttools.xq`
- Test `"request=fooOp?barPar=1" xzy.xq`

- Extend `"request=add?dir=/tta/foo
mod=xyz, ops=booOp beeOp" ttools.xq`

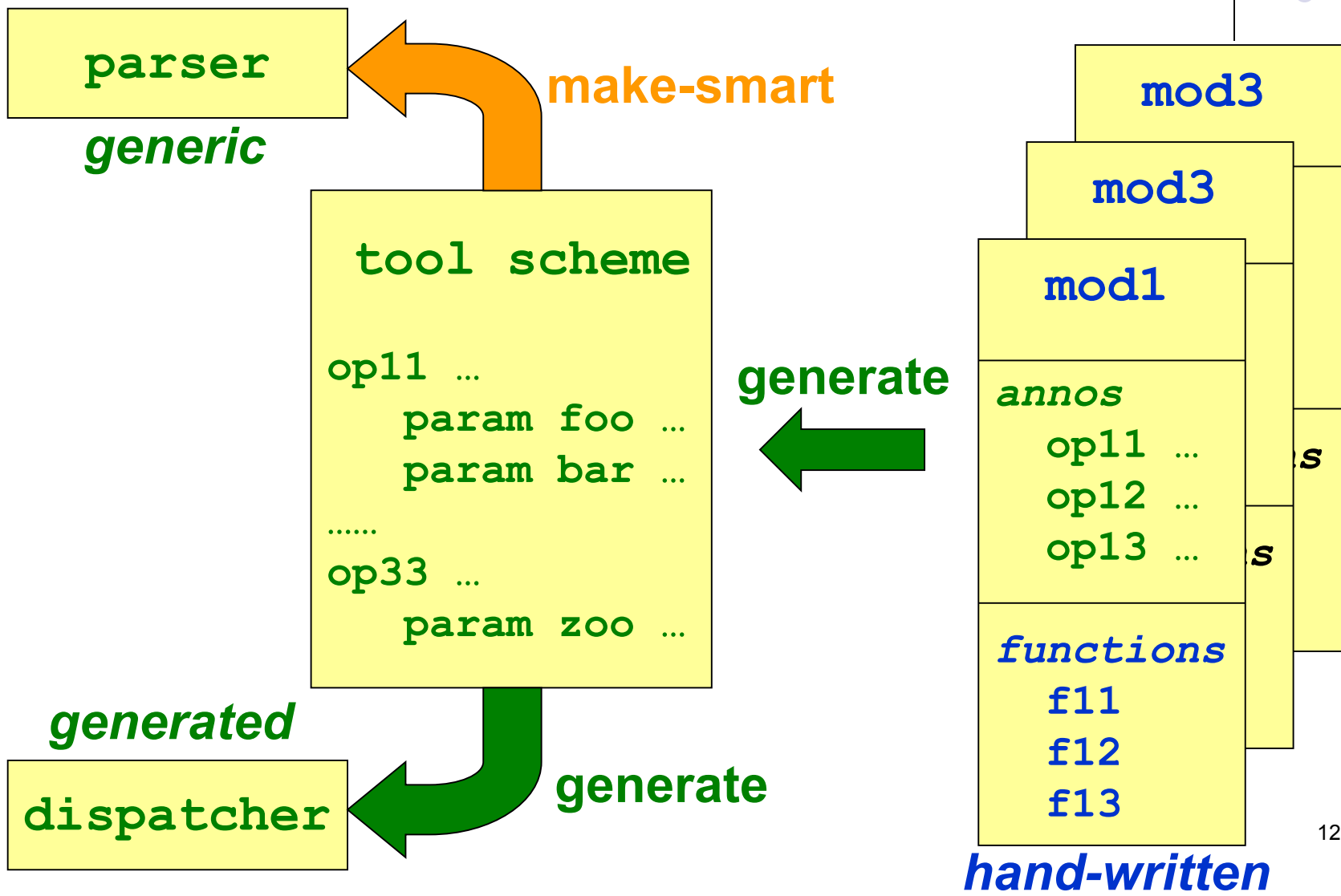
Workflow example ...



(little demo)



Architecture



Things to consider



- Automated **integration** of operations
- CL-interface modelled by **annotations**
- Input supplied to app code as a **message**
- Input **validated** (against types & facets)
- Input augmented (**smart parameter** types)
- Built-in **utility operations** (e.g. `_doctypes`)
- Built-in **help** and input **error reporting**

Where?



<https://github.com/hrennau/TopicTools>

Tutorials:

doc/gettingStartedWithTopicTools.pdf

doc/gettingStartedWithPcollections.pdf

User manual:

under construction ...

(expected release: during March)

Thank you, ...

