Tree-structured data transformation framework

iTrac xRools

Problematic.

We needed a DSL for mappings between different formats with:

- Grammar close to business documentation
- Scalability (for complex transformations)
- Both XML and POJO input/output representations
- Simple grammar, not so verbose as XSLT
- Extensibility
- Validation support
- Performance

Features

Implemented using Scala:

- specialized DSLs (for analysts and developers), which are compiled to the unified and extensible rule model
- JDom (XML) and JxPath (POJO) rule processor implementations
- extensible grammar
- branching and looping (finite loops only)
- rules composition and grouping (for scalability)
- bidirectional transformations
- xpath validation by XSD
- auto-positioning for target elements by XSD (order of rules not affect order of tags)
- groovy support
- XSLT support
- documentation generation (wiki or csv)

Most used Scala features:

- Parser Combinators
- Pattern Matching
- Implicits
- Macroses (prototype only)
- Futures

Internal DSL. Examples.

JIRA

Description

Map notional from every "/SomeFormat/SWAP/Leg (property with name 'notional') to every /FpMl/swap/swapStream (path: notionalStepSchedule/initialValue). Notional must be formatted as %*.*f.

Dates

Created:

12/Aug/13 6:11 PM

Documentation

Name	From	Function	To
Notional	/SomeFormat/SWAP/Leg[x]/property[name='notional']	format('%8.8f')	/FpMl/swap/swapStream[x]/notionalStepSchedule/initialValue

```
val transformer = new RuleBuilder  {
 "Notional Mapping" :=
    "/FpMl/swap/swapStream[x]/notionalStepSchedule/initialValue"
      <- (notional => notional.toDouble.formatted) <-
         "/SomeFormat/SWAP/Leg[x]/property[name='notional']"
}
val outputXml = transformer(inputXml)
```

Internal DSL. Bidirectional transformation.

```
val transformer = new RuleBuilder  {
   def direct (s: String) = s.split(“;”).reverse.mkString(“,.”))
   def reverse (s: String) = s.split(“,.”).reverse.mkString(“;”)

   “BidirectionalRule” := ”/Root/Right” <- (direct * reverse) <- ”/Root/Left”
}

val outputXml = transformer(inputXml)
val recoveredInput = transformer.reverse(outputXml)
```

```
<Root>
  <Left>1;2;3</Left>
</Root>
```

```
<Root>
  <Right>3,.2,.1</Right>
</Root>
```

Internal DSL. Composition&Inheritance.

```
import xrools.{RuleBuilder, RuleHelpers}

class RootRules extends RuleBuilder  with Helpers {
        "ProductType".v <- calculateProduct <- "//trade"
        "Mappings"  += SwapRules test "ProductType".v == "swap"
        "Mappings"  += FraRules test "ProductType".v == "fra"
}

class CommonRules extends RuleBuilder  with Helpers {
        "CommonRule1" :=  …
}

class FraRules extends CommonRules {  /** Fra-specific rules */ }

class SwapRules extends CommonRules {    /** Swap-specific rules */    }

trait Helpers extends RuleHelpers {
        def calculateProduct(s: String) { … }
}
```

External DSL.

"My Rule" := /path/to/destination <= f1(), f2(), f3<= /path/to/source

Dashboard DEV | iKeeper | Environments | Events | Logs | Metrics | Tools | \konddmy

ROOT / ARTIFACTS / DEFAULT / TRANSFORM / FIXML-FPML / RULES.XRL

Save and Close | Save

rules.xrl version: **5**; modified: **Aug 16, 2013** Text | Table

```
17 "tw:PartyRefCtpyVar"                    := $CptyId <=
18   /fixml:FIXML/$MarketId/Instrmt/SecXML/$MarketFpmlNS:FpML/party
19     [partyId!='$DBParty' and (@id='$PayerId' or @id='$ReceiverId')]/@id
20 "MessageId"                             :=
21   /ifpml:executionAdvice/header/messageId[@messageIdScheme='urn:x-schemas-**-com:***base/message-id']
22   <= random('over9000')
23 "SentBy"                                :=
24   /ifpml:executionAdvice/header/sentBy[@messageAddressScheme='urn:x-schemas-**-com:***base/originating-system'] <= `*****`
25 "CreationTimestamp"                     := /ifpml:executionAdvice/header/creationTimestamp
26   <= timestamp('yyyy-MM-dd/T/HH:mm:ss.SSS/Z/')
27 "IsCorrection"                          := /ifpml:executionAdvice/isCorrection <= `false`
28 "ExecId"                                := $ExecID <= /fixml:FIXML/$MarketId/@ExecID
29 "CorrelationId"                         := /ifpml:executionAdvice/correlationId
30   [@correlationIdScheme='urn:x-schemas-db-com:***base/correlation-id'] <= $ExecID
31 "TBD_SequenceNumber"                    := /ifpml:executionAdvice/sequenceNumber <= '1'
32 "OrigTradeRef"                          := /
33   ifpml:executionAdvice/trade/tradeHeader/partyTradeIdentifier[@xsi:type='instructions:PartyTradeIdentifier']/partyReferenc
34 "OrigTradeId1"                          := $OrigTradeId <= replace('[\.\:]','') <= /fixml:FIXML/$MarketId/Instrmt/SecXML/$
35 "ign:TBD_VersionedTrade"                := /ifpml:executionAdvice/trade/tradeHeader/partyTradeIdentifier[partyReference[@h
36 "ign:TBD_TradeVersion"                  := /ifpml:executionAdvice/trade/tradeHeader/partyTradeIdentifier[partyReference[@h
```

External DSL. Table view

- For analytics & QA
- Used as actual documentation

rules.xrl version: **5**; modified: **Aug 16, 2013** Text Table

	ID	From	Function	…	To
8	PartyRefCtpyRec…	/fixml:FIXML/fixml:$MarketId/fixml:Instrmt/fixml:SecXM…			$ReceiverId
9	PartyRefCtpyPay…	/fixml:FIXML/fixml:$MarketId/fixml:Instrmt/fixml:SecXM…			$PayerId
10	bbg:PartyRefCtpy…	/fixml:FIXML/fixml:$MarketId/fixml:Instrmt/fixml:SecXM…			$CptyId
11	tw:PartyRefCtpyVar	/fixml:FIXML/fixml:$MarketId/fixml:Instrmt/fixml:SecXM…			$CptyId
12	MessageId	/	random('9')		/ifpml:executionAdvice/ifpml:header/ifpml:messageId[@mess
13	SentBy	'ITRAC'			/ifpml:executionAdvice/ifpml:header/ifpml:sentBy[@message
14	CreationTimestamp	/	timestamp('yyyy-MM-dd/T/HH…		/ifpml:executionAdvice/ifpml:header/ifpml:creationTimestamp
15	IsCorrection	'false'			/ifpml:executionAdvice/ifpml:isCorrection
16	ExecId	/fixml:FIXML/fixml:$MarketId/@ExecID			$ExecID
17	CorrelationId	$ExecID			/ifpml:executionAdvice/ifpml:correlationId[@correlationIdSche
18	TBD_SequenceN…	'1'			/ifpml:executionAdvice/ifpml:sequenceNumber
19	OrigTradeRef	$MarketType			/ifpml:executionAdvice/ifpml:trade/ifpml:tradeHeader/ifpml:pa
20	OrigTradeId1	/fixml:FIXML/fixml:$MarketId/fixml:Instrmt/fixml:SecXM…	replace('[\.\:]','')		$OrigTradeId
21	ign:TBD_Version…	$OrigTradeId			/ifpml:executionAdvice/ifpml:trade/ifpml:tradeHeader/ifpml:pa
22	ign:TBD_TradeV…	'1'			/ifpml:executionAdvice/ifpml:trade/ifpml:tradeHeader/ifpml:pa
23	PartNum	/fixml:FIXML/fixml:$MarketId/@MultilegPartNum			$PartNum
24	tw:LinkIDVar	/fixml:FIXML/fixml:$MarketId/@TrdID			$LinkId
25	bbg:LinkIDVar	/fixml:FIXML/fixml:$MarketId/@HstCxID			$LinkId
26	bbg:OrdLinkIDVar	/fixml:FIXML/fixml:$MarketId/@OrdLinkID			$OrdLinkId
27	bbg:OrdIDVar	/fixml:FIXML/fixml:$MarketId/@OrdID			$OrdId
28	tw:LinkId	$LinkId	test('PartNum');replace('[\.\:]','')		$packageId
29	bbg:LinkId	$LinkId	test('LinkId');replace('[\.\:]','')		$packageId
30	packageId	$packageId	test('packageId')		/ifpml:executionAdvice/ifpml:trade/ifpml:tradeHeader/ifpml:pa

External DSL. Decision Table

Books.table:

```
TraderName              |   Tenor        | Currency
*                       |   [1,20]       | USD          => BOOK1
not match("petrov")     |   *            | RUB          => BOOK2
match(".*@.*\.com")     |   [1,3],[5,8]  | USD, EUR     => BOOK3
```

Rule:

```
$trader; $tenor; $ccy => decide('Books') => $book
```

External DSL. Parser Examples.

Rule Parser

```
def rule = """.*:="""".r.? ~ expr.? ~ "=>" ~ functions ~ "=>".? ~ expr ~ comment.? ^^ {
  case id ~ exprFrom ~ _ ~ functions ~ _ ~ exprTo ~ cmt =>
    Rule(exprFrom, exprTo, transformers, id, comment)
}

def expr = repsep(ruleValue, ";") ^^ Expression

def ruleValue = variable | strValue | path
```

Decision Table Parser

```
private def decisionTable = header ~ rule.+ ^^ {
  case h ~ rs => DTable(h, rs)
}

private def header = repsep(value, "|")

private def rule = key ~ "=>" ~ col ~ comment ^^ {
  case k ~ _ ~ v ~ _ => Rule(k, v)
}

private def comment = opt("#" ~ value)

private def key = repsep(col, "|")

private def col = repsep(nullColumn | expr | range | link | stringColumn, ",") ^^ {
  case r: Column => r
  case List(s: StringColumn) => s
  case l => ListColumn(l)
}
```

Fluent DSL (experimental).

Fluent DSL example

```
def aaa(s1: String, s2: String, s3: String, s4: String) = s1 + s2 + s3 + s4
def bbb(s1: String, s2: String) = (s1, s2)
def zzz(s1: String, s2: String, s3: String) = (s1, s2)

val transform = xrools {
  val a1 = "/aaaa/bbb".x
  "/aaaa/bbb".x = "/zzz/kkk".x
  "/aaaa/bbb".x = a1
  val a3 = aaa("/aaaa/bbb".x, "/zzz/kkk".x, a1, "bbb")
  val a4 = if (exists(a3)) "/zzz/kkk".x else "bbb"
  val (a5, a6) = bbb("/aaaa/bbb".x, "/uuuu/zzz".x)
  val (a7, a8) = zzz("/aaaa/bbb".x, "/uuuu/zzz".x, "/uuuu/yyy".x)
}
```

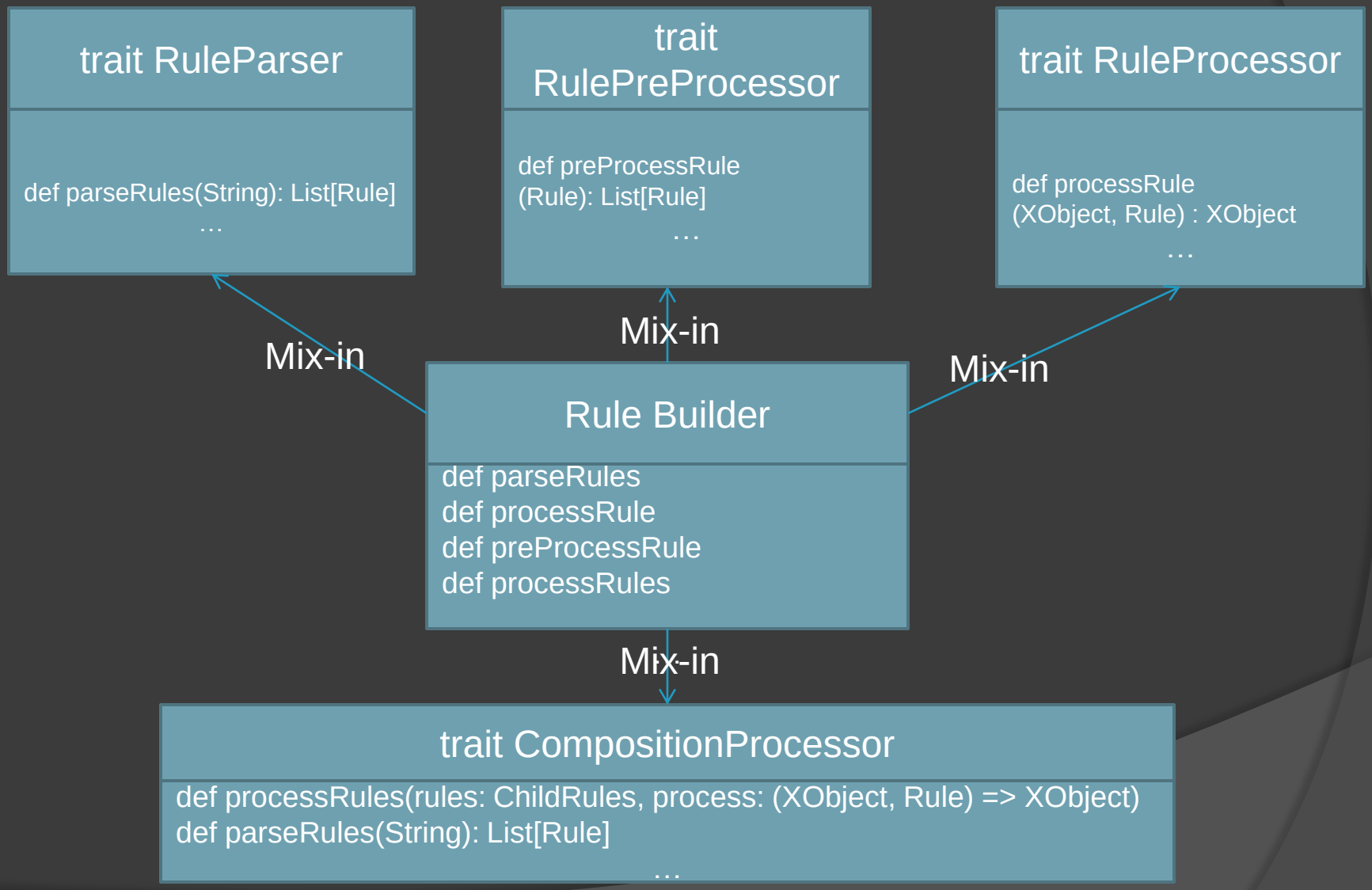
External DSL equivalent

```
$a1 <= /aaaa/bbb
/aaaa/bbb <= /zzz/kkk
/aaaa/bbb <= $a1
a3 <= aaa() <= /aaaa/bbb; /zzz/kkk; $a1; 'bbb'
$a4 <= test('a3') <= /zzz/kkk
$a4 <= test('!a3') <= 'bbb'
 $a5; $a6 <= bbb() <= /aaaa/bbb; /uuuu/zzz
 $a7; $a8 <= zzz() <= /aaaa/bbb; /uuuu/zzz; /uuuu/yyy
```

fragment of macros, which converts fluent DSL to external DSL

```
lazy val process: Tree => String = {
  case b@Block(x, _) => b.children.map {
    case ValDef(_, name, _, Value(value)) => "$" + name.toString() + " <= " + value
    case ValueAssign(path1, Value(path2)) => path1.toString + " <= " + path2.toString
    case ValDef(_, name, _, FunctionCall(functionCall)) => s"$name <= $functionCall"
    case ValDef(_, name, _, If(Apply(Exists(), ValueList(cond)), Value(valthen), Value(valelse))) if cond(0) == '$' =>
      "$" + s"$name <= test('${cond.tail}') <= $valthen \n" + "$" + s"$name <= test('!${cond.tail}') <= $valelse"
    case ValDef(_, tempVarName, _, Match(Typed(FunctionCall(functionCall), _), _)) =>
      s" ${getAllUntupledValues(b, tempVarName.toString).map("$" + _).mkString("; ")} <= $functionCall"
    case t: Tree if !hasNoSynthetic(t) => ""
    case t: Tree => showRaw(t)
  }.filter(_.nonEmpty).mkString("\n")
  case t: Tree => showRaw(t)
}
```

Extensibility. Mixins.



Validation

- Grammar validation using parsers
- Rule validation by xsd
  - checks if xpath may exist in xml (using xsd-schema)
- Difference with expected output
  - checks if rule work as expected

To improve

- fluent DSL
- variables incapsulation in External DSL
- xpath autocomplition

Links

- Parsers: http://eprints.nottingham.ac.uk/237/1/monparsing.pdf
- JxPath: http://commons.apache.org/proper/commons-jxpath/
- Jdom: http://www.jdom.org/
- Xerces: http://xerces.apache.org/
- Macros: http://docs.scala-lang.org/overviews/macros/overview.html