Outside the box: Tinderbox XML tools

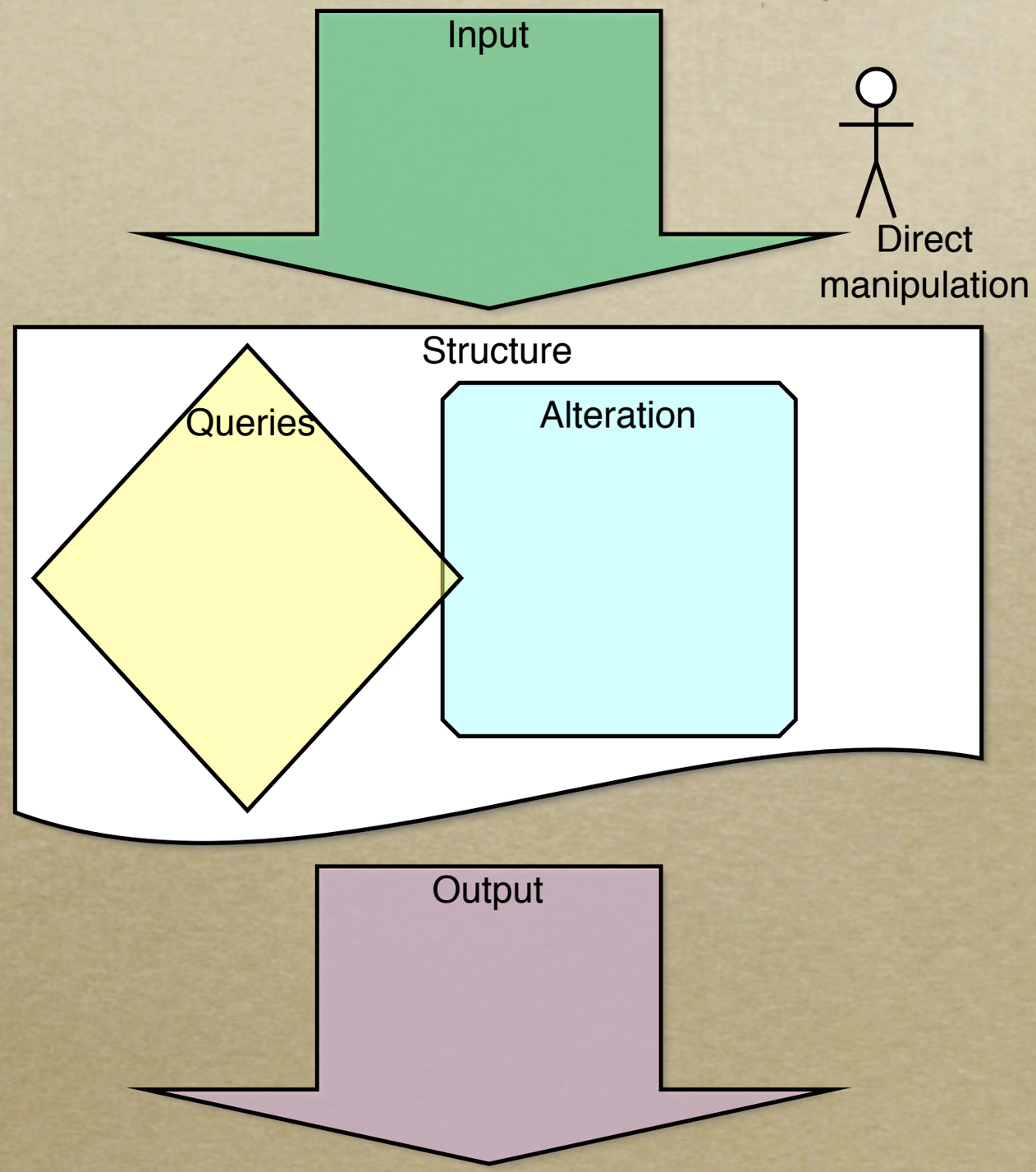
Tinderbox as a data analysis tool

What are we trying to achieve?

- *Hypertext: Making semantic structure explicit*
  - *with links and attributes*
- *Explore and manipulate structural properties*

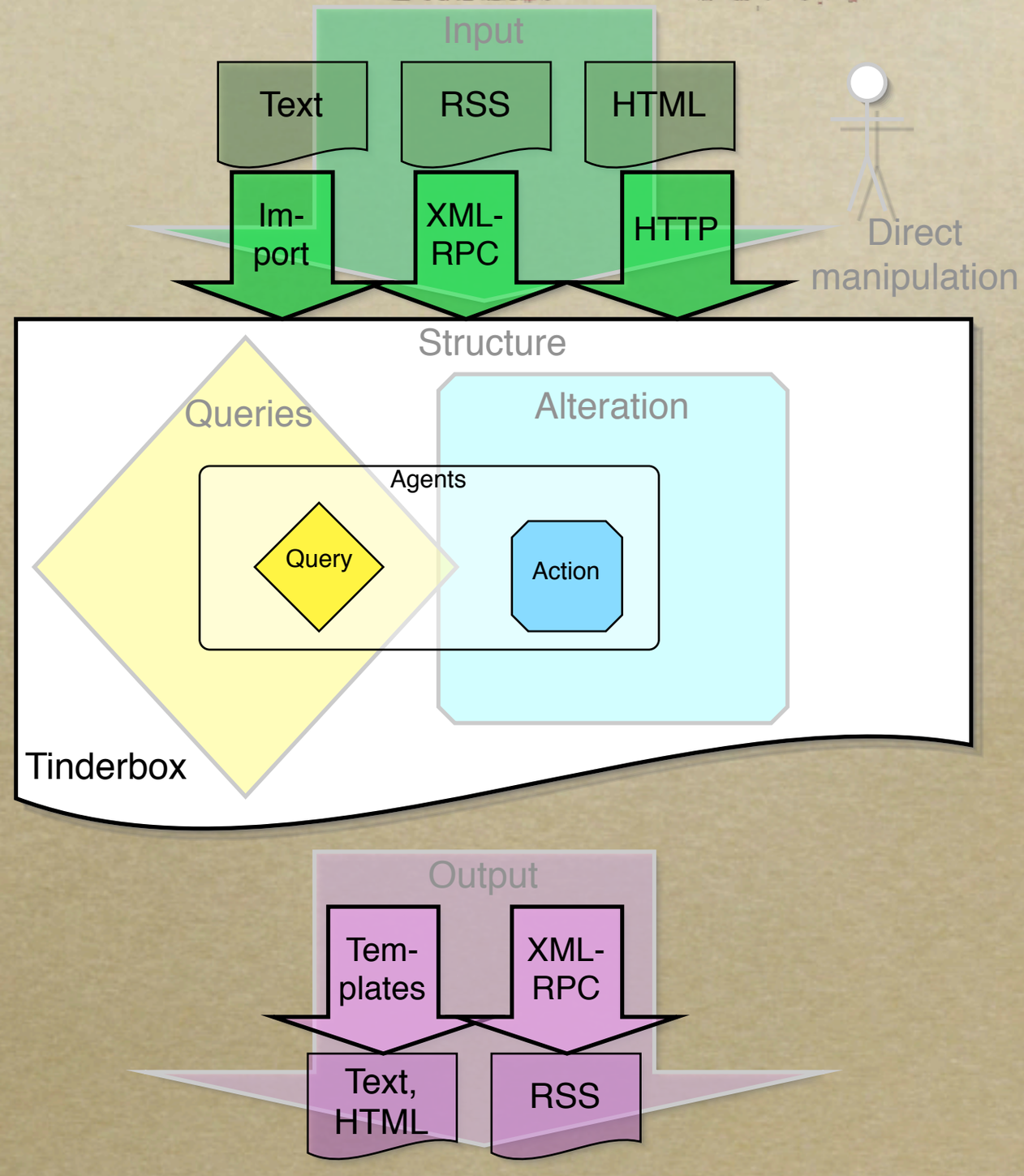
What we need for data analysis

- *Data input*
- *Structure the data*
- *Structural queries*
- *Programmatic alteration of the structure*
- *Data output*



What Tinderbox offers for data analysis

- *Input: HTML, Text, RSS*
- *Structure: Children, Links, Attributes*
- *Agent queries*
- *Agent action*
- *Output: Templates, RSS*



Tinderbox limitations

- *Other inputs*
- *Complex queries*
- *Recursion*
- *Variables*
- *Adornments*
- *Structural actions*

XML: A bird's eye view

- *Hierarchical elements*
- *Attributes*
- *Document Type Definition*

```
<rootElementName >
  <emptyElementName />
  <subElementName attributeName="attributeValue">
    <anotherName>Some text</anotherName>
  </subElementName>
</rootElementName>
```

The tinderbox XML format

- *item and attributes*

```
<tinderbox version="2" revision="2" >
    <attrib Name="Creator" parent="General" editable="0"
     visibleInEditor="1" kind="1" canInherit="0" default="" />
    <colors />
    <menu name="Value" kind="stamps" />
    <linkTypes />
    <item ID="3159019231" Creator="system" >
        <attribute name="Created" >7 Feb 2004 17:20:31</attribute>
        <attribute name="Name" >Root node</attribute>
        <item ID="3161029641" Creator="maparent" >
            <attribute name="Created" >1 Mar 2004 23:47:21</attribute>
            <attribute name="Name" >Child node</attribute>
        </item>
    </item>
    <links />
    <windows />
    <macros />
</tinderbox>
```

The tinderbox XML format

- *links*

```
<tinderbox version="2" revision="2" >
    <attrib Name="Creator" parent="General" editable="0"
     visibleInEditor="1" kind="1" canInherit="0" default="" />
    <colors />
    <menu name="Value" kind="stamps" />
    <linkTypes />
    <item ID="3159019231" Creator="system" >
        <attribute name="Created" >7 Feb 2004 17:20:31</attribute>
        <attribute name="Name" >Root node</attribute>
        <item ID="3161029641" Creator="maparent" >
            <attribute name="Created" >1 Mar 2004 23:47:21</attribute>
            <attribute name="Name" >Child node</attribute>
        </item>
        <item ID="3161029642" Creator="maparent" >
            <attribute name="Name" >NodePrototype</attribute>
            <attribute name="IsPrototype" >true</attribute>
        </item>
    </item>
    <links >
        <link name="prototype" sourceid="3161029642" sourcecreator="maparent"
        sstart="-1" slen="0" destid="3161029641" destcreator="maparent"/>
    </links>
    <windows />
    <macros />
</tinderbox>
```

The tinderbox XML format

- *aliases*

```
<tinderbox version="2" revision="2" >
    <attrib Name="Creator" parent="General" editable="0"
     visibleInEditor="1" kind="1" canInherit="0" default="" />
    <colors />
    <menu name="Value" kind="stamps" />
    <linkTypes />
    <item ID="3159019231" Creator="system" >
        <attribute name="Created" >7 Feb 2004 17:20:31</attribute>
        <attribute name="Name" >Root node</attribute>
        <item ID="3161029642" Creator="maparent" >
            <attribute name="Name" >NodePrototype</attribute>
            <attribute name="IsPrototype" >true</attribute>
        </item>
        <item ID="3161029643" Creator="maparent" >
            <attribute name="Name" > Child node Alias</attribute>
            <attribute name="Alias" >-1133937655</attribute>
        </item>
    </item>
    <links />
    <windows />
    <macros />
</tinderbox>
```

The tinderbox XML format

- *styleruns*

```
<tinderbox version="2" revision="2" >
    <attrib Name="Creator" parent="General" editable="0"
      visibleInEditor="1" kind="1" canInherit="0" default="" />
    <colors />
    <menu name="Value" kind="stamps" />
    <linkTypes />
    <item ID="3159019231" Creator="system" >
        <attribute name="Created" >7 Feb 2004 17:20:31</attribute>
        <attribute name="Name" >Root node</attribute>
        <item ID="3161029641" Creator="maparent" >
            <attribute name="Created" >1 Mar 2004 23:47:21</attribute>
            <attribute name="Name" >Child node</attribute>
            <text >This is the text of the node</text>
            <styles >
              <tstyle font="Geneva" bold="0" italic="0" underline="0"
             start="0" size="10" height="13" ascent="10" color="#000000"/>
              <tstyle font="Geneva" bold="0" italic="0" underline="0"
             start="10" size="10" height="13" ascent="10" color="#000000"/>
            </styles>
        </item>
    </item>
    <links />
    <windows />
    <macros />
</tinderbox>
```

XPaths: A bird's eye view

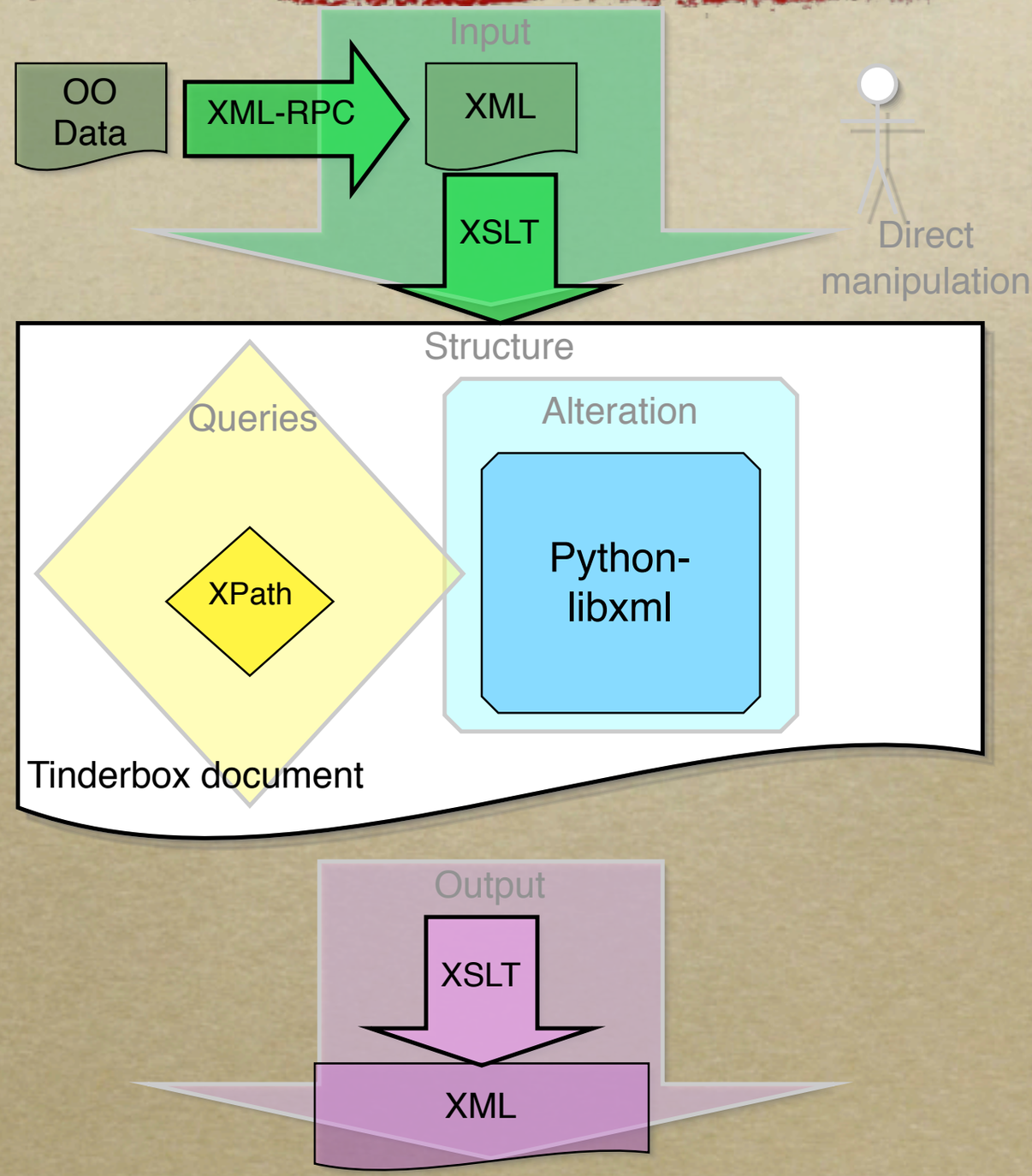
- *Select sets of elements using a path of names*
- *@attributes and text()*
- *element[conditions]*

```
<root attr="v0">
  <sub1 att1="v1">
    <str>text</str>
  </sub1>
  <sub1 att1="v2">
    <str>Some other text</str>
  </sub1>
</root>
```

```
root/sub1/str -> 2 str elements
root/@attr -> "v0"
root/sub1[1]/text() -> "text"
root/sub1[@attr='v1']/str/text()
-> "Some other text"
```

TinderToolBox

- *Data manipulation commands*
  - *MoveNotes, CreateLink, SetText...*
- *Target, Value, Parameter... are XPaths*
- *XPath extensions*
  - *links(), property(), prototype()...*



TinderToolbox simple examples

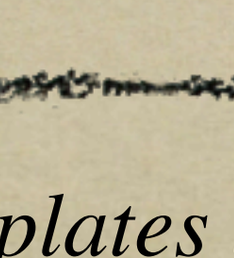
XSLT: A universal language of XML transformation

- *Tinderbox uses XML as data format*
- *XSL allows to convert between XML data formats*
  - *Solution for Input and Output problems*
- *An XML Syllogism:*
  - *All data is an object*
  - *All object is XML*
  - *So all data is XML....*

XSLT: A bird's eye view

- *Templates based on XPaths*

```
<root attr="v0">
  <sub1 att1="v1">
    <str>text</str>
  </sub1>
  <sub1 att1="v2">
    <str>other text</str>
  </sub1>
</root>
```



```
<newRoot>
  <v>v1:text</v>
  <v>v2:other text</v>
</newRoot >
```

```
<xsl:stylesheet>
  <xsl:template match="root">
    <newRoot>
      <xsl:apply-templates/>
    </newRoot>
  </xsl:template>
  <xsl:template match="sub1">
    <v>
      <xsl:value-of select="@att1"/>
      :
      <xsl:value-of select="str/text()"/>
    </v>
  </xsl:template>
</xsl:stylesheet>
```

Simple Input examples

An advanced input example: IMAP through XML-RPC

- *XML-RPC is used for RSS*
- *Combined with XML Object Marshalling*
- *< 1 page of Python code*

```python
#!/usr/bin/python
import DocXMLRPCServer,xmlrpclib,getpass, imaplib, email, email.Parser

#Login
M = imaplib.IMAP4()
M.login(getpass.getuser(), getpass.getpass())

parser = email.Parser.Parser() # parses RFC822 into objects

#Allow none in XML-RPC Marshalling
dumps_orig = xmlrpclib.dumps
def dumps(params, methodname=None, methodresponse=None, encoding=None, allow_none=1):
    return dumps_orig(params, methodname, methodresponse, encoding, allow_none)
xmlrpclib.dumps = dumps

def fetchMailbox(mbox):
    (ok, count) = M.select(mbox, True)
    if not ok: raise "Could not select"
    count=int(count[0])
    messages = []
    for i in range(1,count+1):
        (ok, t) = M.fetch(i, '(RFC822)')
        if not ok: raise "Could not fetch"
        messages.append(parser.parsestr(t[0][1]))
    return messages

server = DocXMLRPCServer.DocXMLRPCServer(("localhost", 8009))
server.register_function(fetchMailbox)
server.register_introspection_functions()
server.serve_forever()
```

A look at XML-RPC results

- *Not a good format, but XML-RPC format are not usually ours to control*

```
<?xml version="1.0"?>
<methodResponse>
  <params>
    <param>
      <value>
        <array>
          <data>
            <value>
              <struct>
                <member>
                  <name>_headers</name>
                  <value>
                    <array>
                      <data>
                        <value>
                          <array>
                            <data>
                              <value>
                                <string>From</string>
                              </value>
                              <value>
                                <string>"Marc-Antoine Parent" &lt;maparent@acm.org&gt;</string>
                              </value>
                            </data>
                          </array>
                        </value>
                        <value>.....</value>
                      </data>
                    </array>
                  </value>
                </member>
                <member>
                  <name>_payload</name>
                  <value>
                    <string>This is the body of the email....</string>
                  </value>
                </member>
              </struct>
            </value>
          </data>
        </array>
      </value>
    </param>
  </params>
</methodResponse>
```

Convert to Tinderbox XML format

```
<item Creator="system" ID="">
  <attribute name="Name">RE: Graphics in Leo's body pane</attribute>
  <attribute name="From">"Marc-Antoine Parent" &lt;maparent@acm.org&gt;</attribute>
  <attribute name="To">"Edward K. Ream" &lt;edream@tds.net&gt;</attribute>
  <attribute name="Subject">RE: Graphics in Leo's body pane</attribute>
  <attribute name="Date">Mon, 20 Mar 2000 09:25:12 -0500</attribute>
  <attribute name="Message-ID">&lt;NDBBLLJMNMIDGJGCCCPPKEJECGAA.maparent@acm.org&gt;</attribute
  <attribute name="MIME-Version">1.0</attribute>
  <attribute name="Content-Type">text/plain;
      charset="iso-8859-1"</attribute>
  <attribute name="Content-Transfer-Encoding">7bit</attribute>
  <attribute name="In-Reply-To">&lt;000601bf9204$2f722f00$923cabd0@xyzzy.tds.net&gt;</attribute
  <text>&gt; Thinking out loud here:  What about embedding images in text?
&gt; Leo uses the XML escape conventions,
&gt; as it must for the file format to be valid XML, so embedding an
.....
  </text>
</item>
....
```

What transformation would look like

Integration issues

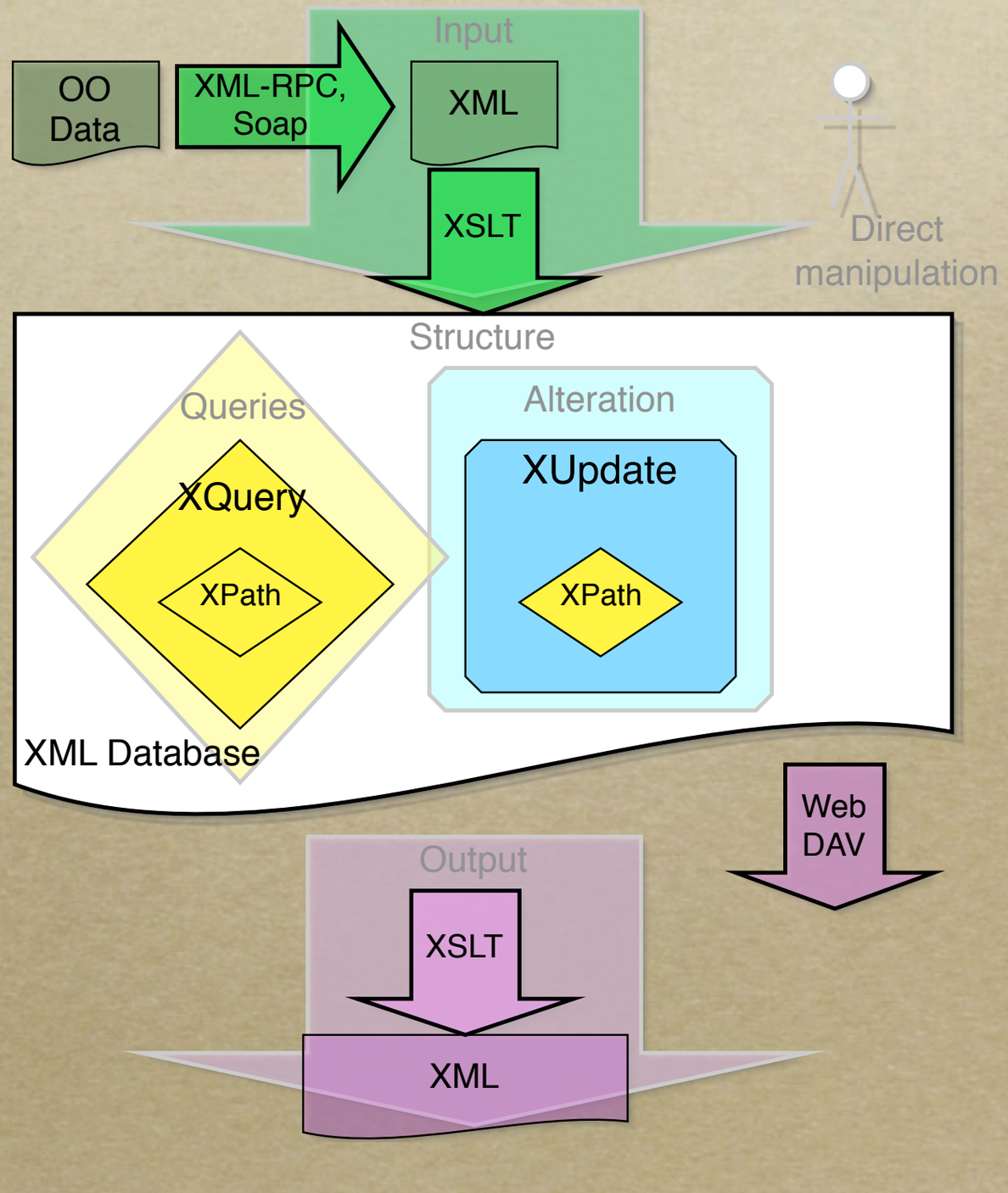
- *Paste in the right place*
- *Tinderbox unique ID*
- *Link to prototype*
- *Duplicates if repeated import*

Visualization

- *SVG*
- *GXL and dot*

A XML Technology portfolio

- *XPath*
- *XSLT*
- *XUpdate*
- *XQuery*
- *XML-RPC*
- *SOAP*
- *XML Object Marshalling*



XML databases: Distributed Tinderbox?

- *eXist*

Conclusion