The SCAPE Platform

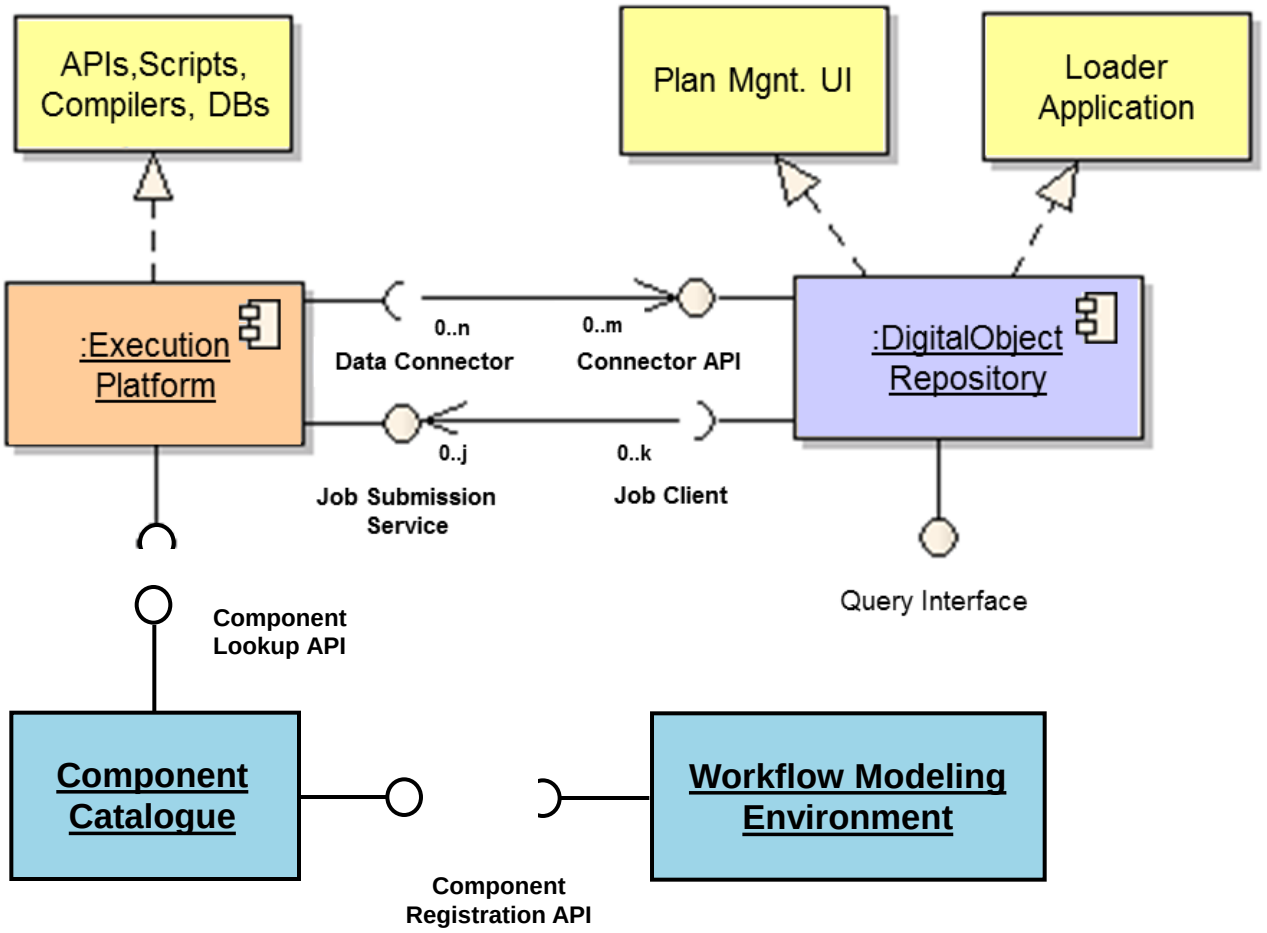
Overview

Rainer Schmidt
SCAPE Information Day
May 5th, 2014
Österreichische Nationalbibliothek

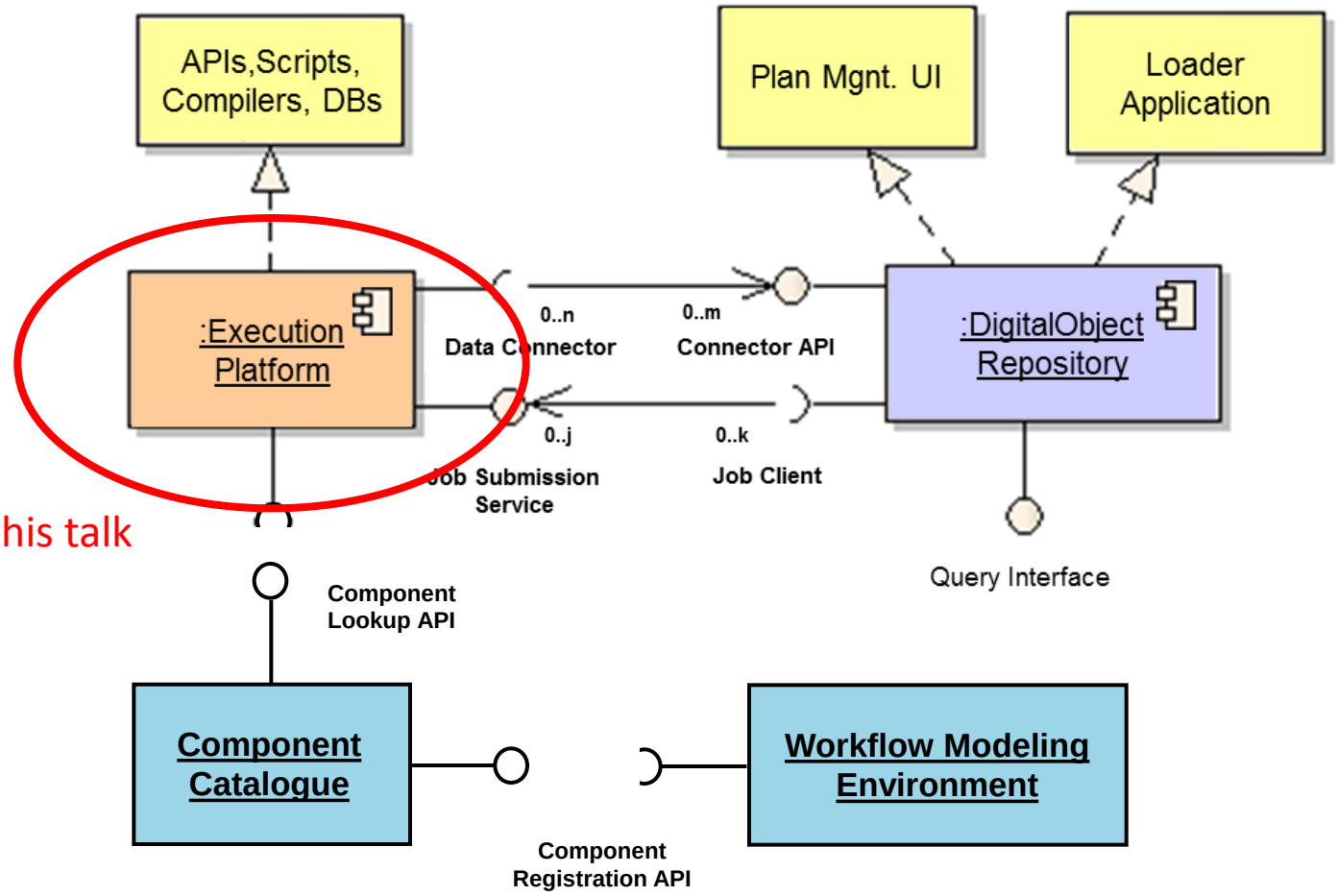
The SCAPE Platform Sub-project

- Develops an integrated environment that is based a set of mature technologies including
  - Apache Hadoop
  - Taverna Workflow Stack
  - Fedora Commons
- Aims at enabling the execution of preservation workflows in a scalable fashion. Supporting different preservation tools, workflows, and various data sets.
- An infrastructure that has been deployed in different configurations and at various SCAPE institutions.
- Part of the SCAPE environment which also involves planning, packaging, scenario development, …

Architectural Overview (Core)



Architectural Overview (Core)



Focus of this talk

Execution Platform

- Wrapping Sequential Tools
  - Using a wrapper script (Hadoop Streaming API)
  - PT's ToMaR supports pre-defined patterns (based on toolspec language)
  - Works well for processing a moderate number of files
    - e.g. applying migration tools or FITS.
- Writing a custom MapReduce application
  - Much more powerful and usually performs better.
  - Suitable for problems at larger-scale and/or handling complex file formats (e.g. Web archiving).
- Using a High-level Language like Hive and Pig
  - Very useful to perform analysis of (semi-)structured data, e.g. characterization output.

- Wrapping Sequential Tools
  - Using a wrapper script (Hadoop Streaming API)
  - PT's ToMaR supports pre-defined patterns (based on toolspec language)
  - Works well for processing a moderate number of files
    - e.g. applying migration tools or FITS.
- Writing a custom MapReduce application
  - Much more powerful and usually performs better.
  - Suitable for problems at larger-scale and/or handling complex file formats (e.g. Web archiving).
- Using a High-level Language like Hive and Pig
  - Very useful to perform analysis of (semi-)structured data, e.g. characterization output.

Available Tools

- Preservation tools and libraries are pre-packaged so they can be automatically deployed on cluster nodes
  - SCAPE Debian Packages
  - Supporting SCAPE Tool Specification Language
- MapReduce libs for processing large container files
  - For example METS and (W)arc RecordReader
- Application Scripts
  - Based on Apache Hive, Pig, Mahout
- Software components to assemble a complex data-parallel workflows
  - Taverna, Pig, and Oozie Workflows

MapRed Tool Wrapper - ToMaR

Hadoop Streaming API

- Hadoop streaming API supports the execution of scripts (e.g. bash or python) which are automatically translated and executed as MapReduce applications.
  - Can be used to process data with common UNIX filters using commands like *echo*, *awk*, *tr*.
- Hadoop is designed to process its input based on key/value pairs. This means the input data is interpreted and split by the framework.
  - Perfect for processing text but difficult to process binary data.
- The steaming API uses streams to read/write from/to HDFS.
  - Preservation tools typically do not support HDFS file pointers and/or IO streaming through stdin/sdout.
  - Hence, DP tools are difficult to use with streaming API

Tool-to-MapReduce (TOMAR)

- Hadoop provides scalability, reliability, and robustness supporting processing data that does not fit on a single machine.
  - Application must however be made compliant with the execution environment.
- Our intention was to provide a wrapper allowing one to execute a command-line tool on the cluster in a similar way like on a desktop environment.
  - User simply specifies toolspec file, command name, and payload data.
  - Supports file references and (optionally) standard IO streams.
- Supports the SCAPE toolspec to execute preinstalled tools or other applications available via OS command-line interface.

Tool Specification Language

- The SCAPE Tool Specification Language (toolspec) provides a schema to formalize command line tool invocations.
  - Can be used to automate a complex tool invocation (many arguments) based on a keyword (e.g. ps2pdfs)
- Provides a simple and flexible mechanism to define tool dependencies, for example of a workflow.
  - Can be resolved by the execution system using Linux packages.
- The toolspec is minimalistic and can be easily created for individual tools and scripts.
  - Tools provided as SCAPE Debian packages come with a toolspec document by default.

Ghostscript Example

```
<?xml version="1.0" encoding="utf-8" ?>
<tool xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http:/
  <operations>

    <operation name="ps-to-pdfa">
      <command>/usr/bin/gs -dPDFA -dBATCH -dNOPAUSE -sDEVICE=pdfwrite -sOutputFile=${o
      <formats in="eps,pdf,ps" out="pdfa"/>
      <inputs ...
    </operation>

    <operation name="ps2pdf">
      <description>Converts ps to pdf files</description>
      <command>/usr/bin/ps2pdf ${input} ${output}</command>
      <inputs>
        <input name="input" required="true">
          <description>Reference to input file</description>
        </input>
      </inputs>
      <outputs>
        <output name="output" required="true">
          <description>Reference to output file</description>
        </output>
      </outputs>
    </operation>
    <operation name="s-ps2pdf">
      <description>Converts ps to pdf files with streaming</description>
      <command>/usr/bin/ps2pdf - -</command>
      <inputs>
        <stdin>true</stdin>
      </inputs>
      <outputs>
        <stdout>true</stdout>
      </outputs>
    </operation>
  </operations>
</tool>
```

Suitable Use-Cases

- Use MapRed Toolwrapper when dealing with (a large number of) single files.
  - Be aware that this may not be an ideal strategy and there are more efficient ways to deal with many files on Hadoop (Sequence Files, Hbase, etc.).
  - However, practical and sufficient in many cases, as there is no additional application development required.
- A typical example is file format migration on a moderate number of files (e.g. 100.000s), which can be included in a workflow with additional QA components.
- Very helpful when payload is simply too big to be computed on a single machine.

Example – Exploring an uncompressed WARC

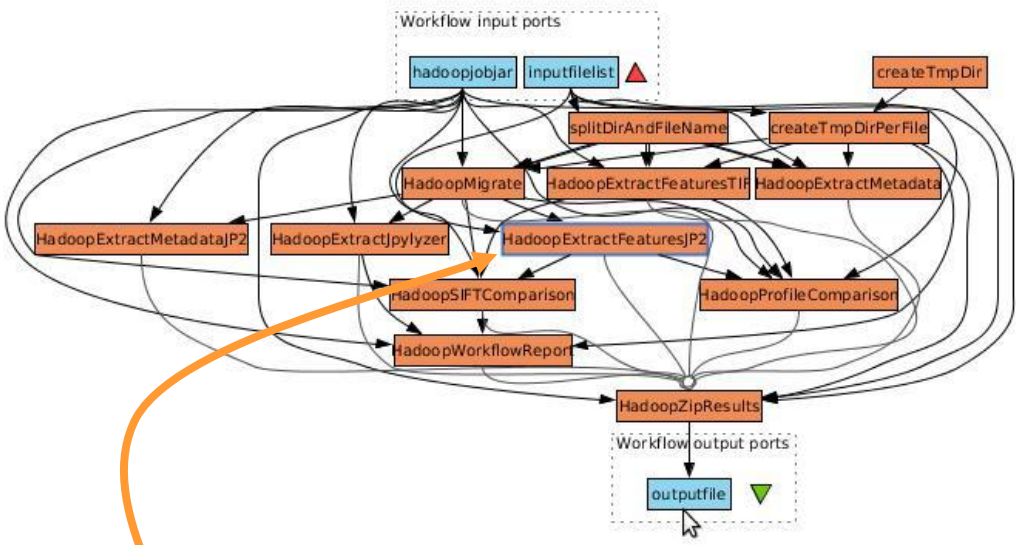
- Unpacked a 1GB WARC.GZ on local computer
  - 2.2 GB unpacked => 343.288 files
  - `ls` took ~40s,
  - count *.html files with `file` took ~4 hrs => 60.000 html files
- Provided corresponding bash command as toolspec:
  - <command>if ["$(file ${input} | awk "{print \$2}")" == HTML]; then echo "HTML" ; fi</command>
- Moved data to HDFS and executed pt-mapred with toolspec.
  - 236min on local file system
  - 160min with 1 mapper on HDFS (this was a surprise!)
  - 85min (2), 52min (4), 27min (8)
  - 26min with 8 mappers and IO streaming (also a surprise)

Workflow Support

What we mean by Workflow

- Formalized (and repeatable) processes/experiments consisting of one or more activities interpreted by a workflow engine.
  - Usually modeled as DAGs based on control-flow and/or data-flow logic.
- Workflow engine functions as a coordinator/scheduler that triggers the execution of the involved activities
  - May be performed by a desktop or server-sided component.
- Example workflow engines are Taverna workbench, Taverna server, and Apache Oozie.
  - Not equally rich and designed for different purposes: experimentation & science, SOA, Hadoop integration.

ToMaR and Taverna Workbench



Command: hadoop jar mpt-mapred.jar -j $jobname -i $infile -r toolspecs

- No significant changes in workflow structure compared to sequential workflow.
- Orchestrating remote activities using Taverna's Tool Plugin over SSH.
- Using Platform's MapRed toolwrapper to invoke cmd-line tools on cluster

ToMaR and MapReduce workflows

- ToMaR utilized as a stand alone MapReduce applications or chained with other Hadoop applications.
- Apache Oozie provides a server (cluster) sided workflow engine and scheduler for Hadoop jobs.
  - Supporting ToMaR as MapReduce actions
- ToMaR is very well suited to help implementing ETL workflows dealing with binary content (e.g metadata extraction).
  - Apache PIG provides a widely used ETL tool for Hadoop implementing a script based query language.
  - Implemented UDF that enables one to use ToMaR within PIG scripts.
  - Enables use of data flow mechanisms and data base abstractions to process CLI-baed tool output.

Ongoing Work

- Source project and README on available on Github
  - https://github.com/openplanets/tomar
- Presently required to generate an input file that specifies input file paths (along with optional output file names).
  - Input binary directly based on input directory path
  - Enable Hadoop to take advantage of data locality
- Input/output steaming and piping between toolspec commands has already been implemented.
- Support for Hadoop file types like SequenceFiles.
- Integration with Hadoop Streaming API and PIG.

Thank you! Questions?