Scalable Preservation Workflows

in the SCAPE project

Rainer Schmidt
AIT – Austrian Institute of Technology

13th Int. PASIG Meeting
September 16-18, 2014 ,
Karlsruhe, Germany

Project Overview

- European Commission FP7 Integrated Project
  - 16 Organizations, 8 Countries
  - 44 months: February 2011 – September 2014
  - Budget: 11.3 Million Euro (8.6 Million Euro funded)
- Consortium: data centers, memory institutions, research centers, universities & commercial partners
  - Extended to involve HPC computing centers.
- Platform-SP: Supporting scalable processes for preserving content from different communities.
  - Integration of tools, workflows, and repositories.
  - Scientific + medical data, Web archives, digitized materials.

Example Content

- Danish broadcast radio and TV archives (500TB)
- Web archiving snapshots of public Internet (10s of TB) containing hundreds of millions of resources
- Millions of high-resolution images from digitized books + metadata (10s of TB)
- 100.000s of data items generated by scientific instruments.

Example Use-Cases

- Detection of image duplicates and cropping error in millions of scanned book pages.
  - Matchbox: Computer Vision (OpenCV) based approach
- Preserving Scientific Data: Conversion from RAW to NeXus (a Proton and Neutron Community format).
  - Raw data, metadata, HDF5.
- Deep characterization of Web archives (millions of resources)
  - Generating metadata/features from files
  - Formats, encodings, validity, size, …
- Quality Assurance for Workflows involving Audiovisual Content
  - Comparison of browser snapshots, Audio cross correlation, Images-based quality assurance (migration).

Need for Scalability

- Strong need for executing tools and workflows in a scalable fashion to deal with data volumes beyond test examples.
  - Local workflows, command-line tools, Matlab, …
- Workflows IO and CPU bound
  - Data Intensive Computing technology desired
- Workflows highly dependent on third party tool/libraries
  - Many legacy applications, often depending on content-specific libraries (ffmpeg, OpenCV, scientific libraries).
  - Mainly required for generating data sets from binary content (or binary-to-binary transformation)
- Re-engeneering these individual applications to fit a particular programming model / execution environment not feasible.

Motivation for ToMaR

- Hadoop provides scalability, reliability, and robustness supporting processing data that does not fit on a single machine based on the MapReduce model.
- Our intention was to provide a generic wrapper allowing our users to execute a command-line tool on the cluster in a similar way like on a desktop environment.
  - User specifies tool, command, and payload data.
- Handling pre-installed legacy applications on Hadoop
  - Focus on command-line interface
  - Support for file references and standard IO streams.
  - Extensibility wrt. file systems and invocation mechanisms, for example HBase, HDInsight blob storage, API calls.

Toolspec Example

```
<?xml version="1.0" encoding="utf-8" ?>
<tool xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://scape-project.eu/tool
    https://raw2.github.com/openplanets/scape-toolspecs/master/toolspec.xsd"
    xmlns="http://scape-project.eu/tool"
    xmlns:xlink="http://www.w3.org/1999/xlink" schemaVersion="1.0" name="fits" version="1.0.1"
    homepage="http://bla.org/">
    <operations>
        <operation name="identify">
            <description>Identifies a file</description>
            <command>/root/fits-0.6.1/fits.sh -i ${input} -o ${output}</command>
            <inputs>
                <input name="input" required="true">
                    <description>Reference to input file or directory</description>
                </input>
            </inputs>
            <outputs>
                <output name="output" required="true">
                    <description>Reference to output file or directory</description>
                </output>
            </outputs>
        </operation>
        <operation name="j-identify">
        <description>Fits</description>
        <command>/usr/bin/java edu.harvard.hul.ois.fits.Fits -i ${input} -o ${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 zipped output file</description>
            </output>
        </outputs>
    </operation>
    </operations>
</tool>
```

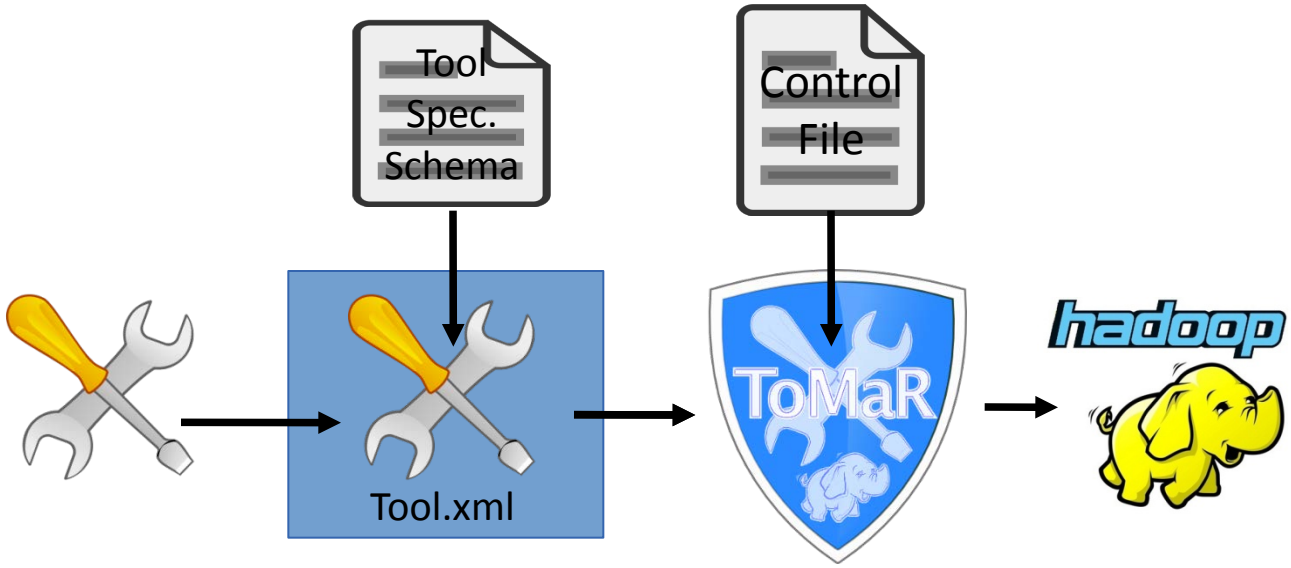
Data Generation and Decompostion

Input Split

```
openjpeg image-to-jp2 –input=“hdfs://myFile1.tif“ --
output=“hdfs://myFile2.jp2“
openjpeg image-to-jp2 –input=“hdfs://myFile2.tif“ --
output=“hdfs://myFile2.jp2“
openjpeg image-to-jp2 –input=“hdfs://myFile3.tif“ --
output=“hdfs://myFile3.jp2“
```

Record

…



Handling Data Locality

- Control Lines provide file pointers to payload data
  - Information on data locality implicit but not exploided per se.
  - A record (= 1 process) may (recursive) process 1 - N files.
- Data Locality supported by ToMaR's ControlLineInputFormat
  - Can be dynamically set using a generic option
- Algorithm:
  - Identifies the required data blocks (replicated) for each record (i.e. a Control Line).
  - Generates a sorted list of hosts for each Control Line
  - Generates a new **sorted** Control File with respect to data locality
  - Generates sorted data splits which carry information about regarding their location

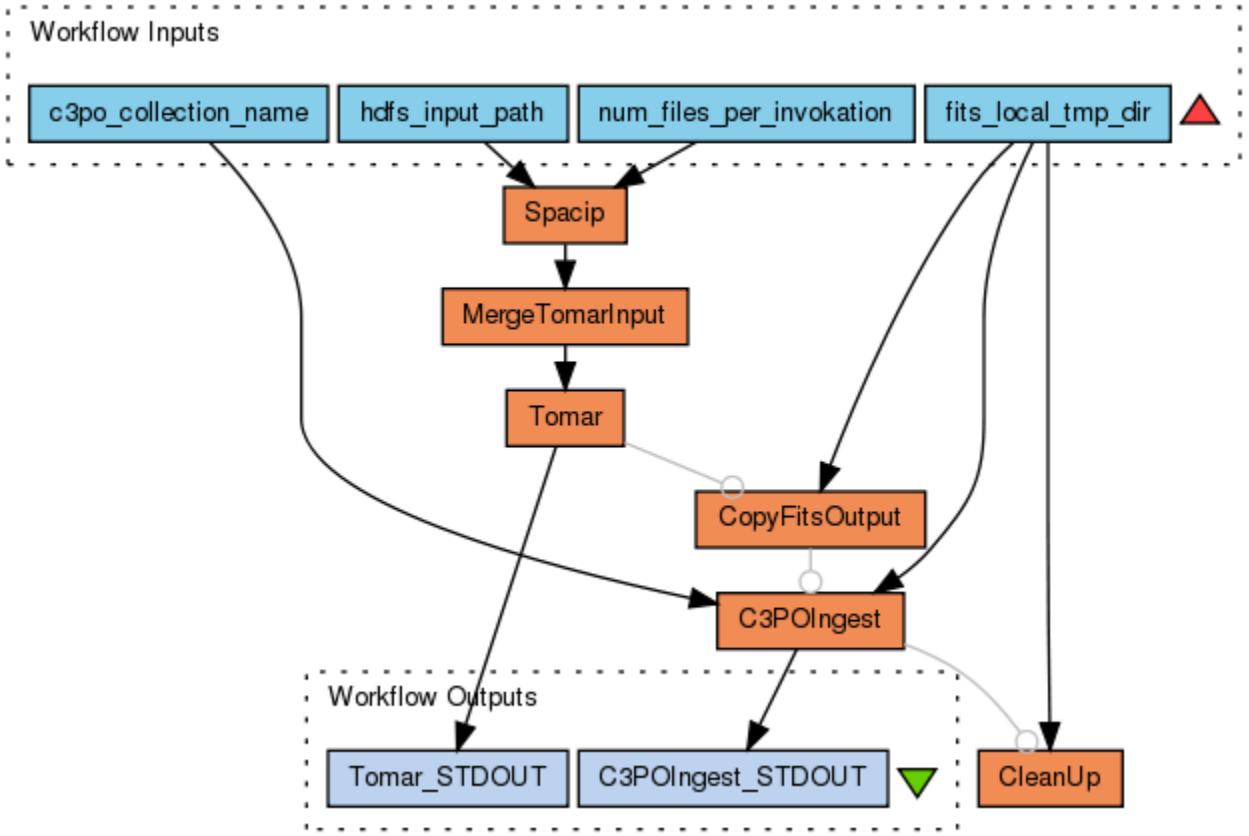
Example – Austrian Web Archive

- The Austrian National Library (ONB) collects the web site within the .at domain.
  - About 1.3 x 10^9 resources, 32TB compressed data
  - 10 – 100 thousands of flat files per container
- ONB interest in analyzing the content of the Web archive.
  - Content profiling using FITS (Harvard Univ. Library)
- FITS unifies output of wrapped third-party open source tools.
  - Although Java, very difficult to ship on Hadoop cluster
- Employed ToMaR to organize data handling and execution of FITS on the cluster
  - Tuning of split size and record size crucial for achieving reasonable performance.

Workflow Integration #1

- ToMaR can be used within an eSciene workflow
  - Orchestrated using Taverna or Oozie on the cluster
  - For example to implement migration, validation, and comparison of AV content
- Many content-based use-cases targeted towards data management applications.
  - Extract-Transform-Load (ETL), data analytics, database building
  - Extracting features, loading result data into databases, performing analysis tasks…
  - Desire to make use of the data management tools available within the Hadoop Ecosystem (HBase, Pig, Hive).

Taverna Workflow Example



Workflow Integration #2

- ToMaR often required within ETL workflows
  - Well suited to extract data sets from binary content
- Apache Pig provides a language and platform suited for implementing such processes on Hadoop
- ToMaR UDF provides the necessary hooks for integrating ToMaR with Apache Pig
  - Enables users to utilize legacy apps. directly within a PIG script.
  - Data can be directly loaded from/into Pig relations.
    - No need to generate Control File and/or intermediate result files.
- Data flow directly handled through the Pig data analytics platform.
  - Delegates utilization of MapReduce platform almost entirely.

Pig Example: FITS + XPath UDF

```
REGISTER tomar-1.4.2-SNAPSHOT.jar;

DEFINE ToMarService eu.scape_project.pt.udf.ControlLineUDF();
DEFINE XPathService eu.scape_project.pt.udf.XPathFunction();

%DECLARE toolspecs_path 'toolspecs';
%DECLARE xpath_exp1 '/fits/filestatus/valid';

/* STEP 1 load content */
image_paths = LOAD '$image_paths' USING PigStorage() AS (image_path: chararray);

/* STEP 2 FITS with ToMaR */
fits = FOREACH image_paths GENERATE image_path as image_path, ToMarService('$toolspecs_path',
  CONCAT(CONCAT('fits stdxml --input="hdfs:///user/rainer/files/', image_path), '"'))
  as xml_text;

/* STEP 3 XPATH */
fits_validation_list = FOREACH fits GENERATE image_path,
  XPathService('$xpath_exp1', xml_text) AS node_list1;

fits_validation = FOREACH fits_validation_list GENERATE image_path, FLATTEN(node_list1) as node1;
fits_validation_filtered = FILTER fits_validation BY node1 == 'true';

STORE fits_validation into 'output/fits_validation';
```

Tomar PIG UDF: FITS | XPath

Hadoop map task list for job_201405230747_0001 on jobtracker

All Tasks

Task	Complete	Status	Start Time	Finish Time	Errors	Counters
task_201405230747_0001_m_000006	100.00%		23-May-2014 07:50:48	23-May-2014 07:54:49 (4mins, 1sec)		18
task_201405230747_0001_m_000007	100.00%		23-May-2014 07:50:48	23-May-2014 07:54:50 (4mins, 2sec)		18
task_201405230747_0001_m_000000	95.99%		23-May-2014 07:50:48			18
task_201405230747_0001_m_000001	95.99%		23-May-2014 07:50:48			18
task_201405230747_0001_m_000002	89.99%		23-May-2014 07:50:48			18
task_201405230747_0001_m_000003	95.99%		23-May-2014 07:50:48			18
task_201405230747_0001_m_000004	89.99%		23-May-2014 07:50:48			18
task_201405230747_0001_m_000005	85.98%		23-May-2014 07:50:48			18
task_201405230747_0001_m_000008	4.00%		23-May-2014 07:54:48			0
task_201405230747_0001_m_000009	4.00%		23-May-2014 07:54:49			0
task_201405230747_0001_m_000010	0.00%					0

FITS | XPath: Standalone vs. Pig

- 50.000 random files processed (1000 splits).
- File Input –> FITS | Java XPATH -> HDFS output.
  - Realized using standalone ToMaR capabilities
  - Realized in PIG Latin using ToMaR UDF and XPATH UDF
- Execution Environment: 9 nodes on Eucalyptus/Xen environment, physical partitions mounted, 1 dedicated master and 8 worker nodes.
- ToMaR (standalone): 9.02h using piped cmd-Line calls, 4.21h with additional Java application shipping.
- Pig (ToMaR UDF): 9.32h using 2 UDF calls, 4.42h with additional Java application shipping.

Summary

- ToMaR motivated by need for processing large volumes of (binary) content using data intensive technology.
  - Refactoring applications can become a major problem for both domain users and application developers.
- ToMaR provides a generic approach for using legacy applications on Hadoop clusters
  - Supports streaming of input/output data, as well as within tasks to reduce map overheads.
  - Additionally, support for Java application shipping and
  - Integration with Apache Pig through UDF
  - Hadoop InputFormat supporting Data Locality

Thank you!

https://github.com/openplanets/tomar