Cyberintegrator

Rob Kooper
Chris Navarro
Liana Diesendruck
Jong Lee
Luigi Marini

National Center for Supercomputing Applications
University of Illinois at Urbana-Champaign
Outline

• Definitions
  • Scientific Workflow
  • Cyberintegrator

• Examples
  • KISTI (CFD)
  • TX Water Management (RAPID)
  • WSSI (RHESSys)

• Cyberintegrator
  • Architecture
  • Technologies used
  • Rest interface

• PAW
• Deployment, how to get it
• Future work
  • Integration with Medici
Scientific Workflow

• A scientific workflow system is a specialized form of a workflow management system designed specifically to compose and execute a series of computational or data manipulation steps, or a workflow, in a scientific application.

• A workflow consists of a sequence of connected steps where each step follows without delay or gap and ends just before the subsequent step may begin. It is a depiction of a sequence of operations, declared as work of one or more simple or complex mechanisms.
Cyberintegrator

- Workflows as a communication mechanism
  - Make workflows documented and sharable
  - Separate science from ‘logistics’
- Enable integration of independent tools
  - Keep models, algorithms, data in open formats accessible from outside the scientific workflow system
- Expose workflow as a service
  - The model encapsulated by a workflow can be exposed a restful service
Definitions

- Input = data that is used by algorithm
- Output = data that is created by algorithm
- Parameter = controls the algorithm executed
- Tool = the encapsulation of the algorithm
- Step = a single execution of a tool
Definitions

- **Workflow** = a sequence of steps
- **Executor** = code to execute a type of tool
- **Engine** = code to execute a workflow
USE CASES
Cyberintegrator Use Cases

• **KISTI**
  • Execute complex CFD on HPC systems
  • Used in university courses with hundreds of students

• **Texas Water Management**
  • Execute RAPID model
  • Uses Cyberintegrator service from inside ArcGIS
  • Used by Microsoft as a demo at AGU

• **WSSI**
  • Execute RHESsys model
KISTI Use Case

- Working with KIST super computer center in Korea
- Allow users to run complex solvers
- Upload their models
- Run solvers on HPC
- Parameter sweeps
- Visualize results
Solver Selection
Parameter Selection
### Execution List

<table>
<thead>
<tr>
<th>Name</th>
<th>Date</th>
<th>Status</th>
<th>Action</th>
<th>Detail</th>
</tr>
</thead>
<tbody>
<tr>
<td>N/A</td>
<td>2012-11-22 17:41</td>
<td>FINISHED</td>
<td>Cancel</td>
<td></td>
</tr>
<tr>
<td>test_by Junhyung</td>
<td>2012-11-22 09:17</td>
<td>FINISHED</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Create**:
  - Angle of Attack=10
  - Angle of Attack=11
  - Angle of Attack=12
  - Angle of Attack=13
  - Angle of Attack=14
  - Angle of Attack=15
  - Angle of Attack=16
  - Angle of Attack=17
  - Angle of Attack=18
  - Angle of Attack=19
  - Angle of Attack=20
  - Angle of Attack=21
  - Angle of Attack=22
  - Angle of Attack=23
  - Angle of Attack=24
  - Angle of Attack=25

- **Monitor**: Select
- **Logout**: Select
Visualization
TEXAS WATER MANAGEMENT
Texas Water Management

- Working with UT-Austin and Texas Commission on Environmental Quality (TCEQ)
- Goal: Building a decision support system for water management
- Utilizing the river flow model called RAPID
Building a Cyberintegrator Workflow for RAPID

- Download NLDAS data
- Execute RAPID model
- Generate visualization (images) of the model results
Web Application

Real-Time Water Decision Support System v0.1

This prototype real-time modeling system downloads Noah-MP Land Surface model data, which forecast runoff, soil moisture, evapotranspiration, and water table levels given land surface features. These results are then used by a river model called RAPIDay to forecast stream flows. Model forecasts are visualized as a Web application for students and decision makers to understand the impacts of drought and flood conditions on streamflows. Users can adjust model parameters to predict the impacts of alternative curtailment scenarios or weather forecasts.

Setting up Workflow

Input the following parameters to run the workflow:

NLAS start date: 2012-01-26
NLAS end date: 2012-01-29
Viz start date: 2012-01-26
Viz end date: 2012-01-29

Run Reset

Model Results

Collaboration with Center for Research in Water Resources, Texas Commission on Environmental Quality (TCEQ)
USE CASE: WSSI
Creating CI Workflow

• You can create a CI workflow by using CI Desktop
• CI workflow can wrap the command line tools
• Example: simplified run.sh script to run RHESSys
  • 3 inputs: worldfile.zip, tecfile, flowfile
  • Unzip worldfile.zip, Run RHESSys, Zip the results

```bash
#!/bin/bash
unzip $1
/home/jonglee/rhessys/rhessys -st 1990 1 1 1 -ed 1993 10 1 1 \ 
  -b -t $2 -w ./worldfile -r $3 -s 12.0880 14.2677 \ 
  -sv 2.1529 83.7472 -gw 0.4108 0.0823
zip results.zip result_*
```
Simple Web Application

![WebRHEsys interface](image-url)
Cyberintegrator Architecture

- Plugin based
- Executor types
  - Local (on local machine)
  - Remote (remote service)
- Example executors
  - Java (local)
  - Command Line (local)
  - HPC (remote)
Technologies Used

• JAVA
• Spring Framework, especially spring-data, well established, been around long time.
• Hibernate, used as ORM
  • Data is stored in MySQL (tested), but can be any relational database.
Local vs Remote Executors

• Local Executors
  • Run on same machine as Cyberintegrator
  • Cyberintegrator controls what executors is running
  • Limited number of parallel processes

• Remote Executors
  • Run on different machine
  • Process Management is done outside of Cyberintegrator
  • All executors are started if possible
JAVA Executor

- Local executor
- JAVA code is run in Cyberintegrator VM
- Need implementation of JAVA interface
  - setInput
  - setParameter
  - Execute
  - getOutput

- System.exit() is a bad call!
Java Wizard

- Add JAR files with tools
  - Including any additional jar files needed
- Select tools that needs to be imported

- Wizard will use interface to get
  - Name and description
  - Inputs and outputs
  - Parameters
Command Line Executor

- Local executor
- Execute command line tool
- Sets working folder to a temp folder
- Can capture stdout and stderr
- Will add copy of inputs in temp folder
  - Prevents modification
- Will copy outputs back to database
Command Line Wizard

• Point to executable
• What inputs, outputs are needed
• What parameters are needed
  • Flags, options etc.
• Any additional files needed
• Set environment variables
HPC Executor

- It is a RemoteExecutor that uses SSH Channels to communicate with various queuing systems.
- Similar to Command Line Executor except the execution line (executable, flags, inputs, etc) for the tool are appended to a script and submitted to a queuing system.
- An XML definition file must be provided to the tool with information about the HPC (e.g. location of submit, terminate, status commands), a script to append the execution line to, and the regex for parsing job status messages.
- Queuing systems tested
  - PBS
  - Loadleveler
  - SGE (minimally tested)
HPC Tool

- A Wizard guides users through tool creation process
- User provides XML host definition file and executable
- The wizard allows user to specify program arguments and inputs that will be used to build the tool’s User Interface
- Each HPC Tool requires the following information, which is added dynamically to the tool definition and will be part of the tool’s UI
  - Username on target machine
  - Userhome on target machine
  - SSH URI for target machine
Cyberintegrator Applications

• Server Application
  • Exposes Cyberintegrator as restful service
  • Allows uploading/downloading workflows/data
  • Allows execution of workflows on server

• Workflow Editor
  • Web based
  • Work online/offline
  • Allows for creation/editing of workflows on server

• Tool creator
  • Temporary tool to allow creation of tools on server
Cyberintegrator Server

- Standard REST endpoint
  - Results are JSON
- Same engine/executors as Desktop
- Can execute workflows on demand
  - Workflows as a service!
- Can upload datasets for workflow
- Can specify parameters for workflow
Cyberintegrator REST

- People [GET, POST]
  - http://<host:port>/persons/{pid}
- Workflows [GET, POST]
  - http://<host:port>/workflows/{wid}/
  - http://<host:port>/workflows/{wid}/zip
  - http://<host:port>/workflows/{wid}/executions/{eid}
- LogFiles [GET]
  - http://<host:port>/logfiles/{lid}/
- Datasets [GET, POST]
  - http://<host:port>/datasets/{did}/
  - http://<host:port>/datasets/{did}/zip
  - http://<host:port>/datasets/{did}/{fid}
  - http://<host:port>/datasets/{did}/{fid}/zip
Workflow Editor

- Web based
- Create tools
- Create workflows
- Execute workflows
- View past executions
- Upload/download datasets/results
Workflow Editor
PAW
PAW

- Published Active Workflow
- Workflows can have many steps, many inputs and many parameters, not all should be exposed to user or as service.
- Allows single widget to control multiple parameters
- Associates UI widgets with parameters.
PAW Editor

- Web based (HTML5) tool for interactively publishing workflows. The tool allows you to:
  - Publisher can specify which workflow fields to expose to users
  - Guides user through process of mapping Web UI widgets (Text, Int, Float, Custom) to one or more exposed workflow fields
  - Add Metadata about workflow tools
- Review panel allows user to review/modify JSON before publishing
PAW Editor – Field Mapping
PAW Editor – Review JSON

Published Workflow JSON

```
{
  "title": "eAIRS-2D CFD",
  "description": "",
  "wfTitle": "",
  "wfId": "39eb553e-3cd9-4ecc-838a-887941b1659f",
  "userId": "cmnnavarr@illinois.edu",
  "inputs": [
    {
      "title": "Target SSH",
      "description": "",
      "defaultValue": "ssh://ranger.tacc.teragrid.org:22",
      "id": "675e363e-7189-eecd-5641-82b234210048",
      "widgetId": "edu.illinois.ncsa.edison.cfd.client.view.TextInputView",
      "type": "STRING",
      "min": "",
      "max": "",
      "enumerations": []
    },
    {
      "parametersMap": [
        {
          "paramId": "0",
          "parameters": [
            {
              "step": "eAIRS Results",
              "title": "Target SSH"
            }
          ]
        },
        {
          "paramId": "91d478f47-6072-4e0e-81f7-da55e52b1dce",
          "parameters": [
            {
              "step": "eAIRS-CFD-Tachyon-MPI",
              "title": "Target SSH"
            }
          ]
        }
      ]
    }
  ]
}
```

Publish Close
Getting Started with Cyberintegrator

- Download Cyberintegrator app
- Create tools
  - Use toolcreator for now
- Create workflow
- Execute workflow on server
- Check results
Future Work

• Finish Web based editor
  • Allow for tool creation
  • Add authentications (openID)

• Data integration with Medici
  • Right now data stored in filesystem
  • Data can be stored in Medici

• More executors for Cyberintegrator
  • MATLAB
  • R

• PAW editor
  • Allow selection of widgets
  • Publish PAW as a web application
Cyberintegrator FAQ

- **Source Code**
  - [https://opensource.ncsa.illinois.edu/stash/projects/CBI](https://opensource.ncsa.illinois.edu/stash/projects/CBI)

- **Bugs**
  - [https://opensource.ncsa.illinois.edu/jira/browse/CBI](https://opensource.ncsa.illinois.edu/jira/browse/CBI)

- **Documentation**
  - [https://opensource.ncsa.illinois.edu/confluence/display/CBI](https://opensource.ncsa.illinois.edu/confluence/display/CBI)

- **Application Downloads**
Questions

• Feel free to contact us

• http://isda.ncsa.illinois.edu

• isda@ncsa.illinois.edu
Cyberintegrator Demo

• Software URLs:
  • https://opensource.ncsa.illinois.edu/bamboo/browse/CI-SERVER
  • Download latest build:
    • cyberintegrator-webapp-all.zip
    • cyberintegrator-tool-creator.zip

• Source URL:
  • https://opensource.ncsa.illinois.edu/stash/scm/~cnavarro/grep-demo.git
Cyberintegrator Start

- Unzip cyberintegrator-webapp-all.zip
- Launch bin/cyberintegrator-service
- Open webbrowser
  - http://localhost:8888/persons (Should return [])

- Unzip cyberintegrator-tool-creator.zip
- Launch bin/tool-creator
Create First Tool

- Add Person
- Add Command-Line tool
  - netstat
    - Executable is netstat
    - Capture stdout
    - Add parameter
      - Name is options
      - Default value is –an
      - Can be empty
Create Second Tool

- Build in eclipse
  - Clone git repository
    - https://opensource.ncsa.illinois.edu/stash/scm/~cnavarro/grep-demo.git
  - Import projects
  - Run->As Maven package

- Build in command line
  - git clone https://opensource.ncsa.illinois.edu/stash/scm/~cnavarro/grep-demo.git
  - cd grep-demo/grep-tool
  - mvn package
Add Second Tool

• Add Java tool
  • Add files point to target/grep-tool-example-0.0.1-SNAPSHOT.jar
  • Select GrepTool
Create Workflow

• In browser go to [http://localhost:8888/editor](http://localhost:8888/editor)
  • Login is email address of user created
  • Password can be blank (for now)
  • Editor should show 2 tools
  • Create new workflow
    • Either plus under workflows or on tab page
    • (known bug of invalid first workflow page CBI-468)
• Drag netstat and grep on canvas
  • Connect stdout of netstat to input file of grep
• Save workflow
Execute Workflow

- Click on Execute
- Open workflow just created
- Fill in workflow
  - Title, description
  - Options = -an
  - Regex = .*LISTEN.*
Workflow History

- Click on History
- Select on execution just created
  - See how long a step took (milliseconds)
  - Download results