SwinDeW-G
(Swinburne Decentralised Workflow for Grid)

System Architecture

Authors: SwinDeW-G Team

Contact: **{yyang, jchen}@swin.edu.au**

Date: 05/08/08

1. Introduction

SwinDeW-G is a scientific workflow management system running on GT4 compatible grid infrastructures or testbeds. It combines p2p (peer to peer) to simulate the enactment of scientific processes in a decentralised manner. This prototype functions using the JXTA platform (www.jxta.org) for p2p and the Globus Toolkit for grid.

This report is intended to provide an explanation of how to use and develop on SwinDeW-G. It is assumed that the reader has a basic understanding of workflow and process enacting.

2. Requirements

Before commencing it is required that SwinDeW-G and its related software is installed. An explanation of how to do this is available in the SwinDeW-G Setup Guide, http://www.swinflow.org/docs/Setup_Manual.pdf.

3. System Architecture

The overall system architecture is depicted in Fig. 1 below. The following sub-sections will describe each of these layers in more detail. As we can see, SwinDeW-G sits on a grid testbed.

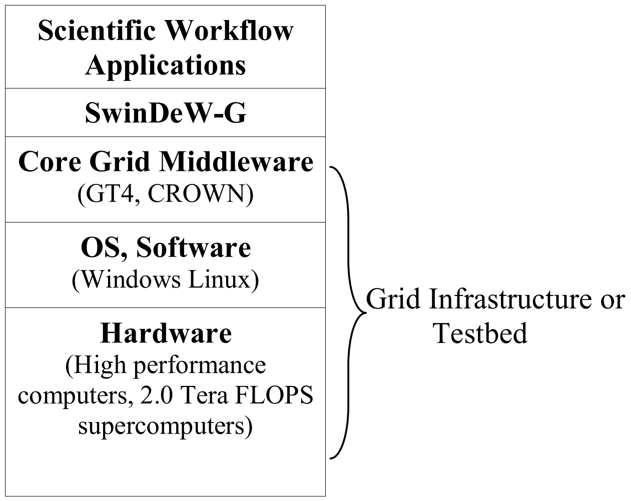

Scientific Workflow Applications
SwinDeW-G
Core Grid Middleware
(GT4, CROWN)
OS, Software
(Windows Linux)
Hardware
(High performance computers, 2.0 Tera FLOPS supercomputers)
Grid Infrastructure or Testbed

Fig. 1. Overall System Architecture of SwinDeW-G Environment

3.1 Grid Testbed

An overall picture of a Grid testbed is depicted in the bottom plane of Fig. 2 which contains many grid nodes distributed in different places. Each grid node contains many computers including high performance PCs and/or supercomputers composed of many computing units. The primary hosting nodes include the Swinburne CS3 (Centre for Complex Software Systems and Services) Node, Swinburne ESR (Enterprise Systems Research laboratory) Node, Swinburne Astrophysics Supercomputer Node, and Beihang CROWN Node in China. They are running Linux, GT (Globus Toolkit) 4.04 or CROWN grid toolkits 2.5 [CROWN 2006] where CROWN (**C**hina **R**&D Environment **O**ver **W**ide-area **N**etwork) is an extension of GT4.04 with more middleware, hence compatible with GT4.04. Besides, the CROWN

Node is also connected to some other nodes such as those in Hong Kong University of Science and Technology, and University of Leeds in UK. The Swinburne Astrophysics Supercomputer Node is cooperating with PfC (Platforms for Collaboration), VPAC (Victorian Partnership for Advanced Computing) and so on.

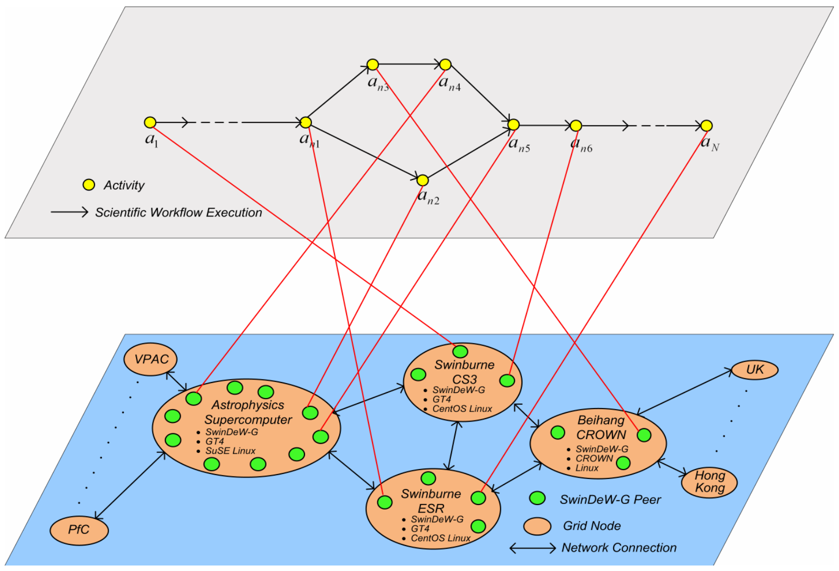


Fig. 2. Physical Network Layout of SwinDeW-G

3.2 SwinDeW-G

Currently, SwinDeW-G is deployed at all primary hosting nodes in Fig. 2. It is planned to extend to other nodes in the near future. SwinDeW-G is a peer-to-peer based scientific workflow management system. A scientific workflow is executed by different peers that can be distributed at different grid nodes. Different peers communicate with each other directly in a peer-to-peer fashion. As shown in Fig. 2, each grid node can have a number of peers. A peer can be simply viewed as a capable grid service.

At build-time, scientific workflow specifications are defined. Then, at run-time, they are executed by different peers which facilitate the computing and resource sharing power of underlying grid testbed.

3.3 Sample Scientific Workflow Execution in SwinDeW-G

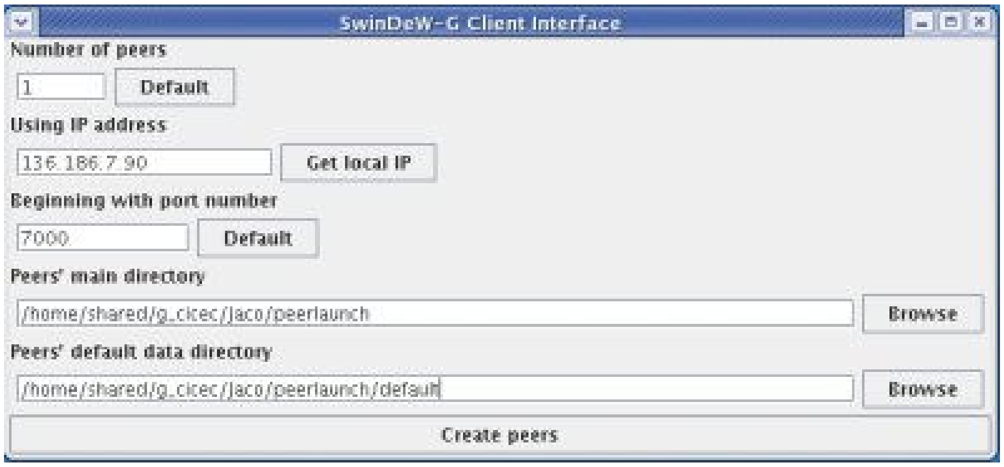
The top plane of Fig. 2 shows a sample scientific workflow execution in SwinDeW-G. Scientific workflow activities are executed by different peers located in different grid nodes.

When a scientific workflow is executed, each activity is assigned to a peer. This assignment is based on which peer is the most suitable to execute the activity. To be suitable the peer must first be capable of executing that activity and not be busy with other tasks. Once all the activities have been assigned to a peer, the scientific workflow is then executed from start to finish. Each peer that has a task assigned to it will communicate with the other peers so that the scientific workflow executes in the correct order.

4. Launching Peers

4.1. SwinDeW-G Client

SwinDeW-G peers can be launched automatically via a java client. This client is located in the 'clients' directory under the name … . The running of this client executes the shell script … When used it allows the user to easily launch any number of peers via a GUI shown in Fig. 3.



Number of peers: 1
Using IP address: 136.186.7.90
Beginning with port number: 7000
Peers' main directory: /home/shared/g_cicec/jaco/peerlaunch
Peers' default data directory: /home/shared/g_cicec/jaco/peerlaunch/default

Fig. 3. The SwinDeW-G peer launching client

- **Number of peers** – This defines how many peers will be launched. Each peer will be opened in its own command window. For a standard desktop computer it is recommend that no more then 6 peers be launched. Exceeding this number will result in severe system slow down and a possible crash.
- **Using IP Address** - This is the IP address that the launched peers will use. This should be the IP address of the computer that the peers are launched on. Clicking the 'Get local IP' button will automatically detect the computer's IP address.
- **Beginning with port number** – The first port that will be assigned. Each peer requires 2 ports. The first port is for the globus container and the second is for the JXTA p2p communications. Even ports will be assigned to the container while odds are assigned to p2p. Ports will be assigned incrementally starting from the one that is specified.
  Note: Please ensure that the ports that SwinDeW-G uses are not blocked by a local firewall. They need to be open for communication on the local machine.
- **Peers' main directory** – This directory will be used to store data about the peers. Each peer will have its own directory. A description of the files and uses of the peer directory can be found in the following section.
- **Peers' default directory** – This is the directory that contains the default setup files for each peer. A description of this can be found in the following section.

When the 'Create peers' button is pressed a data directory will be created for each peer. These peers will then load within their own command window. This command window will display an output of the peer's actions. A description of this output can be found in section 5.

4.2. Directory Structure

Each peer will automatically have a directory created for it. The peer uses this directory to access its configuration and capabilities files and to cache data for JXTA communication. When the 'Create peers' button is pressed, the program creates a unique directory for each peer and then copies the files from the 'defaults' directory into it. The data directories are only created for peers that do not already have one. In other words if peer1 already has a data directory containing its configuration files then no changes will be made to it when peer's are launched.

The file 'platformConfig' is the configuration file for JXTA. JXTA uses this to get information about the peer before it loads. This information includes the peer's name, ip address, port and login data. The SwinDeW-G client modifies the 'platformConfig' file for each peer to dynamically set their configuration data based upon the user input.

The file 'log4j.properties' configures how the output will be displayed.

4.3. Configuring Capabilities

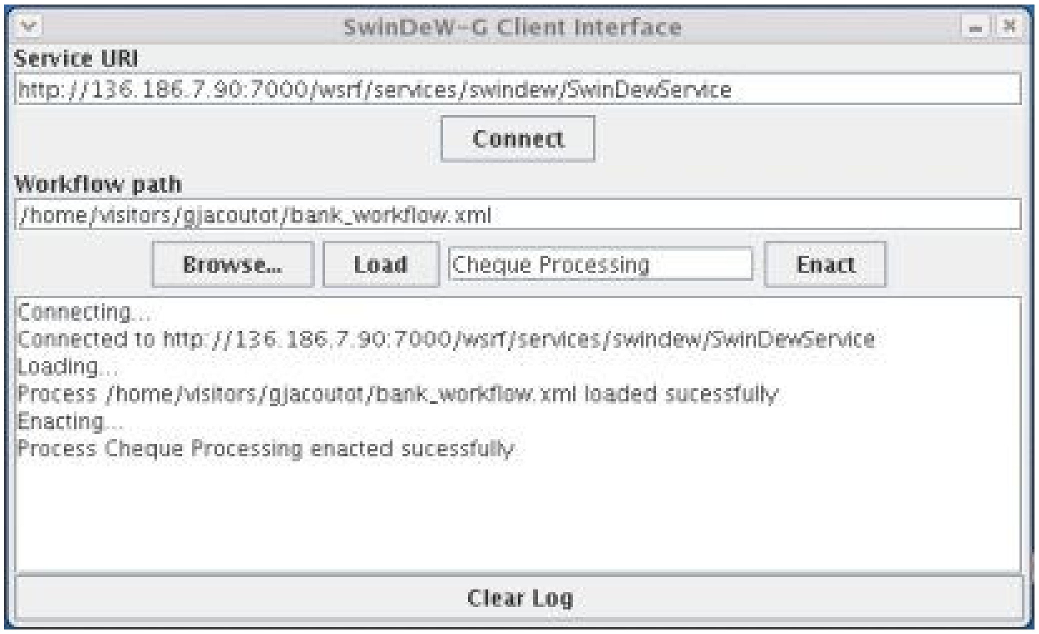
Another file that is copied into each peer's data directory is 'capabilities.xml'. This file contains a list of the capabilities that the peer will have. By default each peer will be assigned all the capabilities in the Cheque Processing workflow. These are Scan Cheque, Analyse Image and Access Accounts.

To further experiment with SwinDeW-G it is possible to limit or change the capabilities that each peer has. This can be done by modifying 'capabilities.xml' and reloading the peer.

5. Uploading and Enacting Processes

The SwinDeW-G client is also able to upload and enact workflow processes. This can be done once a number of peers have been launched.

5.1. Client GUI



Service URI: http://136.186.7.90:7000/wsrf/services/swindew/SwinDewService

Workflow path: /home/visitors/gjacoutot/bank_workflow.xml

Cheque Processing

```
Connecting...
Connected to http://136.186.7.90:7000/wsrf/services/swindew/SwinDewService
Loading...
Process /home/visitors/gjacoutot/bank_workflow.xml loaded sucessfully
Enacting...
Process Cheque Processing enacted sucessfully
```

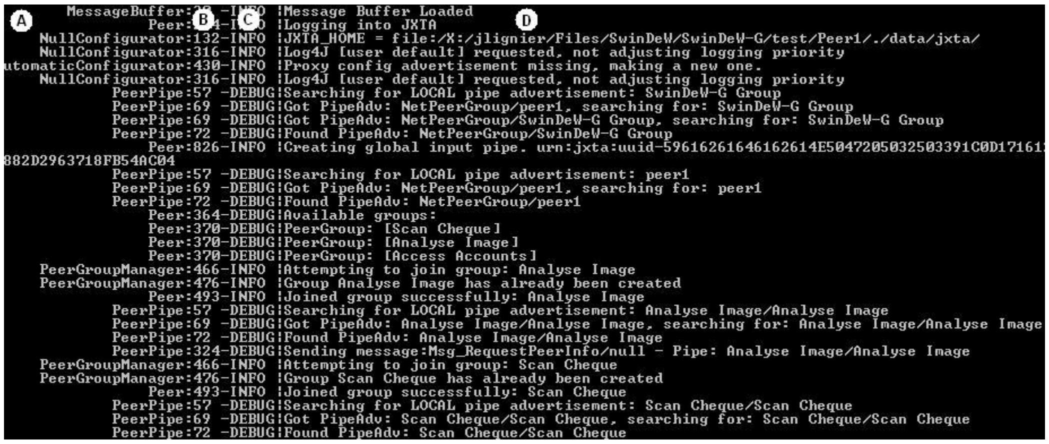
Fig. 4. The SwinDeW-G Workflow Process Client

- **Service URI** – As shown in Fig. 4, this is the URI of the peer that you want to upload and/or enacted a process on. The URI can be found in the peer's command window when globus loads. The default URI will be that of the first peer that is loaded. Once the URI has been entered click connect to establish a connection to this peer. If the URI is invalid or the peer can not be located, then an error message will be displayed in the output window.
- **Workflow path** – This is the where the xml workflow file is located on the local machine. The browse button will open a file browser window to locate the file. The load button will upload the workflow to the connected peer. When a peer receives a workflow file, it will distribute the tasks in the workflow to peers that have the correct capabilities. Included with the SwniDeW-G distribution is 'bank_workflow.xml' which is located in the 'client' directory.
- **Enact process** – To enact a process on the connected peer, type in the name of the process and press the 'Enact' button. This peer will then try to locate this process and enact it. The process in the 'bank_workflow.xml' file is 'Cheque Processing'.
- **Output textbox** – This shows the of the different actions performed within the process client. To clear the text simply press the 'Clear log' button.

6. SwinDeW-G Output

Each SwinDeW-G peer will have its own command window. This window will show the output from the different tasks that the peer is performing. An example of this output is shown below.

6.1. Output Description



```
    MessageBuffer:[illegible] -INFO |Message Buffer Loaded
            Peer:[illegible]4-INFO |Logging into JXTA
  NullConfigurator:132-INFO |JXTA_HOME = file:/X:/jlignier/Files/SwinDeW/SwinDeW-G/test/Peer1/./data/jxta/
  NullConfigurator:316-INFO |Log4J [user default] requested, not adjusting logging priority
utomaticConfigurator:430-INFO |Proxy config advertisement missing, making a new one.
  NullConfigurator:316-INFO |Log4J [user default] requested, not adjusting logging priority
        PeerPipe:57 -DEBUG|Searching for LOCAL pipe advertisement: SwinDeW-G Group
        PeerPipe:69 -DEBUG|Got PipeAdv: NetPeerGroup/peer1, searching for: SwinDeW-G Group
        PeerPipe:69 -DEBUG|Got PipeAdv: NetPeerGroup/SwinDeW-G Group, searching for: SwinDeW-G Group
        PeerPipe:72 -DEBUG|Found PipeAdv: NetPeerGroup/SwinDeW-G Group
            Peer:826-INFO |Creating global input pipe. urn:jxta:uuid-59616261646162614E504720503250339 1C0D17161
882D2963718FB54AC04
        PeerPipe:57 -DEBUG|Searching for LOCAL pipe advertisement: peer1
        PeerPipe:69 -DEBUG|Got PipeAdv: NetPeerGroup/peer1, searching for: peer1
        PeerPipe:72 -DEBUG|Found PipeAdv: NetPeerGroup/peer1
            Peer:364-DEBUG|Available groups:
            Peer:370-DEBUG|PeerGroup: [Scan Cheque]
            Peer:370-DEBUG|PeerGroup: [Analyse Image]
            Peer:370-DEBUG|PeerGroup: [Access Accounts]
  PeerGroupManager:466-INFO |Attempting to join group: Analyse Image
  PeerGroupManager:476-INFO |Group Analyse Image has already been created
            Peer:493-INFO |Joined group successfully: Analyse Image
        PeerPipe:57 -DEBUG|Searching for LOCAL pipe advertisement: Analyse Image/Analyse Image
        PeerPipe:69 -DEBUG|Got PipeAdv: Analyse Image/Analyse Image, searching for: Analyse Image/Analyse Image
        PeerPipe:72 -DEBUG|Found PipeAdv: Analyse Image/Analyse Image
        PeerPipe:324-DEBUG|Sending message:Msg_RequestPeerInfo/null - Pipe: Analyse Image/Analyse Image
  PeerGroupManager:466-INFO |Attempting to join group: Scan Cheque
  PeerGroupManager:476-INFO |Group Scan Cheque has already been created
            Peer:493-INFO |Joined group successfully: Scan Cheque
        PeerPipe:57 -DEBUG|Searching for LOCAL pipe advertisement: Scan Cheque/Scan Cheque
        PeerPipe:69 -DEBUG|Got PipeAdv: Scan Cheque/Scan Cheque, searching for: Scan Cheque/Scan Cheque
        PeerPipe:72 -DEBUG|Found PipeAdv: Scan Cheque/Scan Cheque
```

Fig. 5. Command window output from a SwinDeW-G peer

The output of SwinDeW-G is divided into four columns. These columns are labelled in the image shown in Fig. 5 and are described below.

A. The java class file that the message is being generated in. All the source files for SwinDeW-G are located in the 'src' directory and can be matched against the class names in this column. Note: not all the message in the output window originate from SwinDeW-G some are also from JXTA and Globus.
B. This is the line number of the class file that the message is being generated on.

C. The type of message that is being generated. There are four types of message. These are Debug, Info, Error and Fatal.
D. The output message. The describes a certain actions that has taken place within SwinDeW-G

6.2. Log File

All the output that is displayed in the command window is also output to a text file. This text file can be found in the … directory under the name … . If unknown errors occur it is recommend to email the generated log files to the SwinDeW-G development team.

Note: The log files are emptied every time a peer loads. So each log file only contains the output of the last peer execution.

7. Test Case

An example workflow has been included in the SwinDeW-G distribution to demonstrate the prototype's functionality. The file that contains this workflow can be located under 'clients/bank_workflow.xml'. The following section describes how to upload and enact this process along with a description of the workflow process.

7.1. Workflow Process Description

Fig. 6 is a description of the workflow that we will use in our test case.

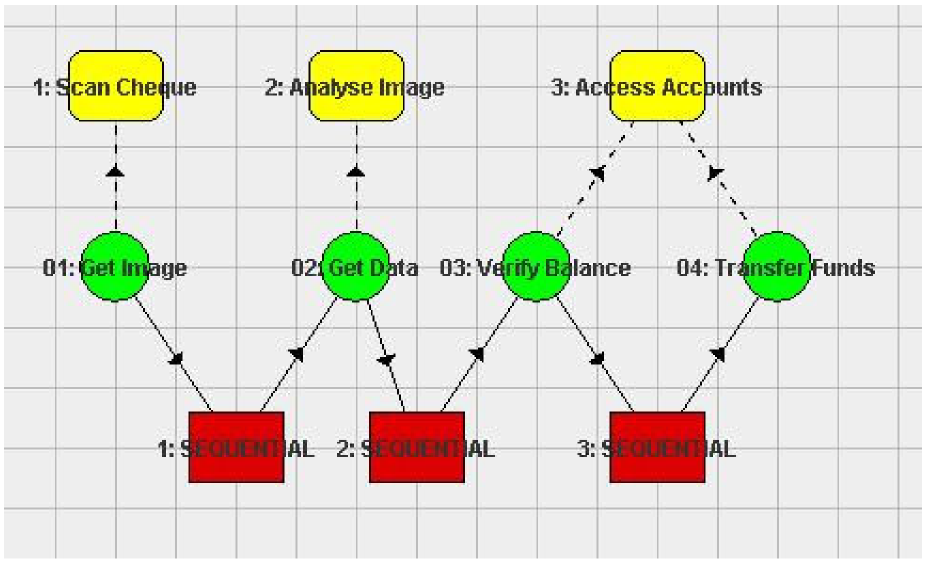


Fig. 6. Cheque Processing Workflow (A Large Number of Cheques)

This is a graphical representation of the workflow contained in 'bank_workflow.xml'. It is intended as a simplified version of how cheques are processed in a bank once they have been received. A description of the different elements in the diagram is provided bellow.

- **Yellow boxes** – These are the capabilities that represent the work that will be performed when a task is enacted. These capabilities are generally able to be performed on a diverse set of peers as defined in their 'capabilities.xml' file.

- **Green circle** – These are the tasks of the workflow. In this workflow they execute one after the other in sequential order until the end is reached. When each task is enacted its capability is also accessed.
- **Red rectangle** – This is the logic of the workflow. This controls how the process will flow from one task to the next.

7.2. Launch peers

To start with we need to load 3 peers on which the workflow will be enacted. To do this open the SwinDeW-G client and open the 'Peer Launcher' tab. Use the following parameters in the input fields:

- **Number of peers** – 3
- **Using IP address** – Local machine IP address. This can be found by clicking the 'Get local IP' button
- **Beginning with port number** – This should be left as default unless one of the ports that will be assigned (7000 – 7005) is already in use.
- **Peers' main directory** – By default the 'peer_data' directory under 'clients' can be used. If you wish the peer data to be stored elsewhere then specify this directory here.
- **Peers' default data directory** – This will be the 'defaults' directory under 'clients'.

Once these parameters have been entered then press the 'Create peers' button.

Three command windows should now open which will show each peer loading and joining groups according to their capabilities.

7.3. Upload workflow

The workflow now needs to be uploaded onto one of the peers. It does not matter which peer we use because when a SwinDeW-G peer receives a workflow it will automatically distribute it amongst its neighbours. In this example we will upload the workflow to peer1.

First, open the SwinDeW-G client and open the 'Client' tab. Now enter the URI of peer1 and click 'Connect'. Next, browse to where 'bank_workflow.xml' is located. Once this is done, click the 'Load' button.

In peer1's command window you should now see the workflow being loaded and distributed to peer's 2 and 3.

7.4. Enact process

Processes should only be enacted on peers that have the first capability in the workflow. This does not affect our example as all three peers have all the capabilities of the workflow.

To enact the process first enter the process name "Cheque Processing" into the text box. Then click the 'Enact' button. Be sure that the client is still connected to peer1.

In the command window for each of the peers you should see the process enacting from start to finish.

7.5. Further Experimentation

To further experiment with the functionality of SwinDeW-G you may want to run this test using additional peers. Also it is possible to limit the capabilities that each peer has so that certain tasks in the workflow can only enact on some of the peers.