GUMTREE DECODED

T. Lam, N Xiong, P. Hathaway, N. Hauser, ANSTO, Sydney, Australia

Abstract

During the construction of 8 new Australian neutron beam instruments, the software team from Bragg Institute (ANSTO) has developed a novel software system, codename GumTree, which unifies data acquisition and analysis under a single user application. GumTree is a Java-based system that builds on Spring, OSGi and Eclipse RCP framework. It provides many application building blocks for creating different kinds of scientific applications, including control system connector, data processing engine, reduction algorithm libraries, OpenGL data visualisation toolkit, workflow support and script engine connectivity (cPython, JavaScript, etc). With tight integration of above components, users can script and plan their experiments in an interactive way, and let GumTree automates the experiment based on the automatically analysed raw results. The main benefit of this approach is to increase the effective use of instrument, saving instrument time and cost for long running experiments. GumTree system can run as either desktop application or middleware server mode. The simplified web client version that uses Adobe Flex and AJAX are also under development.

I. CONCEPT

What is GumTree

GumTree is an enabling software technology for supporting various kind of neutron scattering experiments. It covers instrument control, experiment planning, live data reduction and simple data analysis in the desktop client area; web status monitoring, data management and database connectivity in the server side area. The GumTree project is originally funded by the Australian Nuclear Science and Technology Organisation (ANSTO), as part of the neutron beam instrument project for constructing eight new state-of-the-art instruments.

Project Approach

The direction of the GumTree project is driven by fulfilling the needs of the project's stakeholders:

- **Instrument Users** - The main part of this project is to provide an integrated scientific workbench to centralise everything that an instrument user will need during their experiment. The design of the workbench is catered for novice and expert users. This is achieved by providing guided visual assistance for inexperienced users and powerful in-application scripting capability for users with great confidence.
- **Software Developers** - GumTree is designed to be customisable and extensible. Each GumTree based application runs on top of a small dynamic runtime platform, named OSGi, with the choice of a number of reusable components. Developers only need to configure the platform and write a small portion of customised code for their instruments.
- **Facility Operators** – We have planed to support facility operators in the area of data management and system monitor by providing a number of add-on components that will run on top of the GumTree platform.

Open Source Technology

GumTree is built on the foundation of a number of Java based open source technologies. There are small part of the system that is written in Python script and other JVM supported scripting languages. The core of GumTree is a micro-kernel that runs on top of the Equinox OSGi runtime [1]. OSGi is a small Java based application container that allows part of the Java code to be installed or unloaded dynamically at runtime. The GumTree runtime kernel is also hybridised with the Spring Dynamic Module (DM) dependency injection container for supporting application configuration via XML.

The desktop side of GumTree is created with Eclipse Rich Client Platform (RCP) technology. Eclipse RCP provides many useful API (such as docking window, abstract file system, help system and data binding) and extensions (XML editing support, remote system communication and graphical editing framework) to help developers to create high level and professional looking applications. Many Eclipse extensions are freely available to the open source community, but some extensions like Mathematica and IDL workbench are proprietary software. GumTree is designed to support as much Eclipse extensions as possible, for example, GumTree can mix with IDL to create a vanilla data analysis application.

Table 1: Leveraged open source technologies in GumTree

Component	**Technologies**
Platform Runtime	Java, Equinox OSGi, Spring DM
Windowing Toolkit	SWT
Visualisation	JFreeChart, OpenGL
Data Format	netCDF, HDF
Web Communication	Restlet, Jetty
Persistence	Db4o
Logging	SLF4J, Log4J
XML Processing	xStream, dom4j
Code Generation	EMF-SDO

II. APPLICATION

GumTree Workbench

At the initial construction phase of the neutron beam instrument project, we have made two important decisions on our software system [2]:

1. All instruments, if possible, should be standardised on common hardware, drivers and software choice.
2. Instrument software stack will adopt the client-server model.

On the server side, the Swiss SICS control system from Paul Scherrer Institut (PSI) has been chosen as the common instrument control system, leaving the client side to be an in-house development effort. At the beginning of 2004, Andy Götz, the co-founder of TACO and TANGO, has visited Australia to started a new Eclipse RCP based desktop project, codename GumTree, to be the graphical front-end for ANSTO's SICS server. Over the years, GumTree has evolved from a simple instrument control program to an integrated scientific workbench, with the ability to automate experiment and data reduction process.

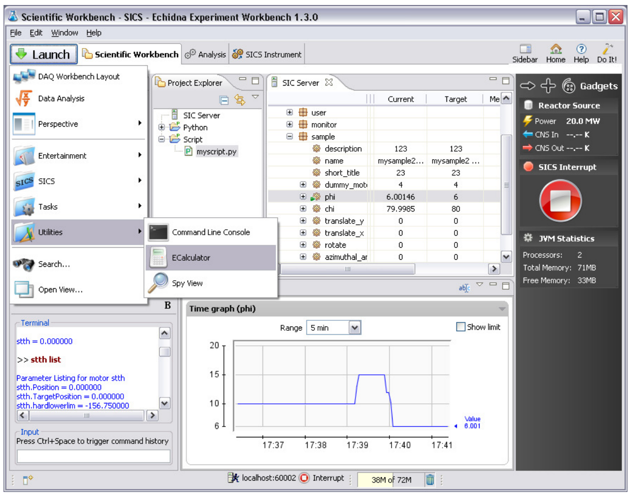
Instrument Control

GumTree provides two type of access to the SICS control system: direct command via socket and treepath based control [3]. The latter one, also known as Hipadaba, supports the object oriented way of instrument control in SICS. The underlying protocol is wrapped in JSON format, which is a light weight data format compares to XML. Through Hipadaba, GumTree can create a variety of graphical components, like table tree, text box and synoptic widget, to drive and display instrument state. The transfer of the tree structure is done via the Service Data Object (SDO) [4], which is implemented with Eclipse Modeling Framework (EMF).

GumTree does not limit to SICS. Early experimental work has been done to demonstrate the possibility to support different control systems, such as TANGO and EPICS, via their Java client APIs.

Integrated Experiment Environment

Many modern day desktop applications come with advanced windowing system for managing user interface in a clean and centralised way. As more and more graphical user interface components are developed during the project, we saw the need of using an application framework that can provide a sophisticated windowing system. The Eclipse Rich Client Platform (RCP) was almost the only choice we have at that time [5]. Due to the power of Eclipse RCP, we expanded our vision of GumTree to more than just a control system front-end. Since then we introduced a concept called *integrated experiment environment*, where any thing to do with neutron beam experiment (such as instrument control, monitoring, data file viewing, data file access database connection, report generation, note editing) can be done within a single application [6]. As the result, each component can interoperate and share data easily because they live on a single process.



SIC Server

	Current	Target	Me
user			
monitor			
sample			
description	123	123	
name	mysample2...	mysample2 ...	
short_title	23	23	
dummy_mot	4	4	
phi	6.00146	6	
chi	79.9985	80	
translate_y	0	0	
translate_x	0	0	
rotate	0	0	
azimuthal_ar	0	0	

Terminal

```
stth = 0.000000

>> stth list

Parameter Listing for motor stth
stth.Position = 0.000000
stth.TargetPosition = 0.000000
stth.hardlowerlim = -156.750000
```

Input
Press Ctrl+Space to trigger command history

Figure 1: GumTree Workbench

Scripting

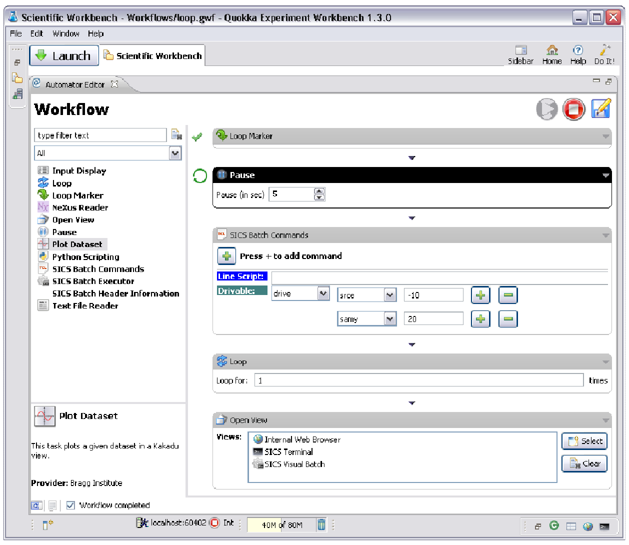
Scripting ability was one of the most wanted features requested by our expert users. It was a technical challenge for GumTree until the release of Java 6, where Sun Microsystem has introduced the dynamic scripting API (JSR-223) on top of the Java virtual machine. With this scripting API, users can call and automate any functionality available in GumTree at runtime, such as instrument control and workbench manipulation. In particular, we have chosen cPython as our scripting language to GumTree via JEPP, so that we can leverage a large number of numerical and scientific libraries from the community.

Since GumTree is based on Eclipse, this makes GumTree as a primary platform for editing scripts. For example, we embed PyDev, one of the open source Python development tools available to Eclipse, for editing Python scripts. Eclipse RCP also comes with API for managing files and code completion. There is very little work we need to support scripting in GumTree.

Workflow

Workflow is a very common concept for performing a very complex task by composing a series of well defined small tasks. UNIX pipeline, BPEL business process, Yahoo! Pipes and Mac OSX Automator are good examples of workflow. This concept was discussed within the context of Web 2.0 [7], as an important technique for service mashup (application hybrid).

GumTree provides various workflow task blocks for instrument control, data manipulation, and workbench control. Users can visually mix and compose those task blocks in a workflow for creating their own GumTree automation script.



Pause (in sec) 5

Drivable: drive srce -10

samy 20

Loop for: 1 times

This task plots a given dataset in a Kakadu view.

Provider: Bragg Institute

Figure 2: GumTree Workflow

Data Analysis

Data analysis is the most sophisticated part within GumTree. The data analysis capability was driven by the need of reducing and analysing live data during the user experiment. This provides instant feedback on the quality of their data and provides information to adjust their experiment strategies if necessary. Neutron scattering experiments usually require hours for data collection, so this feedback technique could save time and money because neutron beam time is generally expensive.

The data analysis respect of GumTree is built upon the following foundation blocks:

- **GumTree Numeric** – a Java based numeric and data format library
- **Kuranda visualisation toolkit** – an SWT OpenGL based 1D/2D data plotting library
- **Cicada processing framework** – a processor based data analysis engine. It can be extended for data acquisition and simulation [8].
- **Kakadu data analysis suite** – a graphical front-end for the Cicada framework. It supports auto GUI generation based on the content provided by Cicada.

GumTree Server

GumTree server is the headless version of GumTree (normal GumTree application without user interface components), with embedded HTTP server and communication protocol component equipped. It serves as a middleware layer between client and instrument control system for monitoring and remote access, using HTTP ReSTful web services [9]. Control system information can then be accessible via normal web browser or Rich Internet Application (RIA) such as Adobe Flex or AJAX. For all SICS controlled neutron instruments at ANSTO, we have developed flex clients to display live data and instrument status.

III. PROGRAMMING MODEL

Architectural Style

GumTree has a distinguish architecture under the influence of Eclipse and Spring architecture style. Each GumTree application is built on 4 layers:

1. **Application platform** – a very generic application container for running any desktop or server application.
2. **Application framework** – a set of generic framework for building feature rich application.
3. **Scientific framework** – provides instrument control / data analysis application building blocks.
4. **Instrument customisation** – instrument specific logic and algorithm for supporting very specific user groups.

Each GumTree application layer contains a set of plug-ins (Java module), which provides coherent features and extension point to allow other plug-ins to contribute.

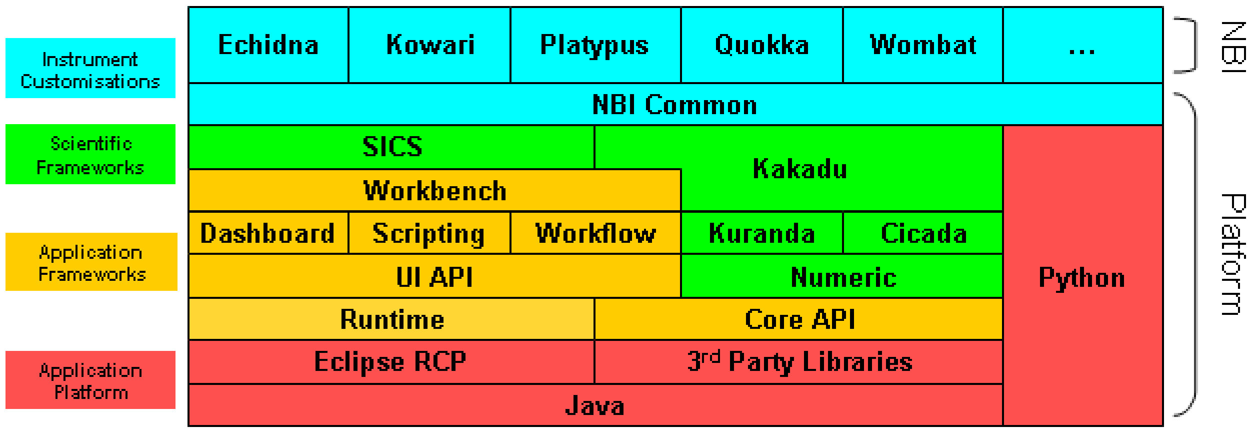


Figure 3: The architectural layer for the GumTree application

This layered and modular approach in GumTree is the natural extension to the object oriented architecture (OOA). The design of platform also involves two other architectural approaches to make GumTree more scalable:

- Service oriented architecture (SOA) – application logics are writing in form of simple objects, known as services, so that they can be injected and reused within the application easily.
- Resource oriented architecture (ROA) – any information that can be named is abstracted to "resource", for examples: document, image, database record, device, application state and functionality (service). Each resource has unique identifier in form of URI. This architectural style provides a simple and uniform way of resource discovery and manipulation. The GumTree server uses this extensively for providing instrument remote access.

Design Philosophy

The design of GumTree follows 3 important principles:

1. **Minimal infrastructure** – design APIs based on only few but powerful concepts, because less means less for developers to learn and maintain. In order to achieve this, concepts need to be simple but be able to span into various application. For example, the GumTree Cicada process framework is powerful enough for the use of data analysis or data acquisition.
2. **User Experience** – providing initiative user features are measurement of project's success. Software is useless unless users can use it within their comfort zone. Many GumTree components are designed and customised to keep user to focus on their tasks.
3. **Customisations** – this is about leveraging the building blocks (principle 1) and make them available to fulfil user requirements (principle 2).

IV. CONCLUSION

GumTree has already been used to collect data for high resolution power diffractometer (**Echidna**), residual stress diffractometer (**Kowari**), small angle neutron scattering (**Quokka**), and high intensity power diffractometer (**Wombat**). Few more constructing and commissioning instruments at ANSTO and NECSA (South Africa) have been planned to use the GumTree system. The future direction of the project will focus on providing a more mature scripting support for data analysis, and data management integration. GumTree is an active open source project hosted at Codehaus (http://gumtree.codehaus.org/). Facility collaboration on the project is welcome.

REFERENCES

[1] I. Skerrett, "Creating Micro-platforms", http://ianskerrett.wordpress.com/2006/05/23/creating-micro-platforms/

[2] A. Götz and N. Hauser, "Grad Unified Model for Control System", NOBUGS 2004 Conference, http://lns00.psi.ch/nobugs2004/papers/paper00125.pdf

[3] M. Könnecke and M. Zolliker, "Treepath Based Instrument Control", NOBUGS 20008 Conference, http://www.nbi.ansto.gov.au/nobugs2008/papers/paper132.pdf

[4] "Introduction to Service Data Objects", http://www.ibm.com/developerworks/java/library/j-sdo/

[5] T. Lam, A. Götz, F. Franceschini, N. Hauser. "GumTree - A Java Based GUI Framework for Beamline Experiments", Journal of Neutron Research (2006)

[6] T.Lam, N.Hauser, et al., "GumTree - An Integrated Scientific Experiment Environment", Physica B, 385-386 (2006)

[7] I. Forrester, "Pipelines: Plumbing for the next web", http://www.princexml.com/howcome/2007/xtech/papers/output/0082-32/index.xhtml

[8] N. Xiong and P.Hathaway, "Multi-platform Processor Framework for Data Analysis, Data Acquisition and Simulation", ICALEPCS 2009 Conference, THP076

[9] T. Lam, "Controlling instrument in the RESTful way", NOBUGS 2008 Conference, http://www.nbi.ansto.gov.au/nobugs2008/talks/157.ppt

[10] J. Thelin, "A Comparison of Service-oriented, Resource-oriented, and Object-oriented Architecture Styles", http://www.thearchitect.co.uk/presentations/arch-styles/3-arch-styles.pdf