LoCuS (Laboratory fOr CompUter Science) System is instructional software for teaching computer science theories through experiments. Experiments within the context of LoCuS are laboratory excursions akin to those found in other sciences like biology, chemistry, and physics. A locus (Latin for “place”) is a “collection of points which share a property” or “the path through which a point moves to satisfy a given condition.” The LoCuS logo is a Boy’s surface, named after its founder, Werner Boy. Boy’s surface is a particularly intricate locus which is described by a system of complex equations. Just like the Mobius strip, Boy’s surface has only a single side. Our goal is for LoCuS to define a new place to emphasize the science of computation as well as a new path for computer science, in the form of labs.
Contents

1 Introduction 1
2 Release Management 3
3 LoCus Build Scripts 5
4 Manual JNLP Deployment 7
5 Standalone jar Deployment 9
6 LoCus Labs Website for Teachers 11
   6.1 Implementation .................................................. 11
   6.2 Connecting to the MySQL Database Directly .................. 11
7 Past Issues and Solutions 13
Chapter 1

Introduction

This document provides information specific to the LoCuS project. For a more general overview of the Chief Programmer’s responsibilities, please refer to the TAU Chief Programmer’s Guide. Please also read the Getting Started Guide for the LoCuS Project before reading this document. We assume that the reader is already familiar with the project setup procedure and has access to the source code as well as the SVN repository.
Chapter 2

Release Management

Since LoCuS is hosted on the department’s website, the terms release and deploy mean the same thing: uploading updated jar files to the web and updating links on the website. Deployments are located at /cs/cgi/projects/locus.

The LoCuS version checking procedure is described as follows. LoCuS is released together with appropriate versions of apparatuses and labs. When run, LoCuS compares modification dates of apparatuses on the server with those stored locally on the client, and updates apparatuses if needed. In order for LoCuS to be able to check the versions of apparatuses correctly, several steps must be performed before deployment.

1. In the LoCuS Ant script build.xml, the path to the deployment location must be changed to include the new version of the platform, e.g.
   <property name="jnlp.deployment.path" value="/cs/cgi/projects/locus/version-5-0/labs"/>

2. The version variable must be updated in the edu.arizona.locus.core.Metadata class.

3. Download paths for apparatuses, labs and JNLP files must be set in
   locus/dist/resources/settings.properties, e.g.,
   PLUGINS_URL = http://cgi.cs.arizona.edu/projects/locus/version-5-0/labs/jnlp/plugins/
   LABS_URL = http://cgi.cs.arizona.edu/projects/locus/version-5-0/labs/jnlp/labs/
   JNLP_URL = http://cgi.cs.arizona.edu/projects/locus/version-5-0/labs/jnlp/

4. The first property DEPLOYMENT_TYPE in the settings.properties file controls how LoCuS locates apparatuses and labs. This property accepts two values.
   ECLIPSE_DEPLOYMENT sets LoCuS to look for labs and plugins in the locus/dist/ directory in the workspace folder.
   JNLP_DEPLOYMENT sets LoCuS to look for labs and plugins in the ~/locus/ directory.

   For the purposes of development, in SVN DEPLOYMENT_TYPE should always be set to ECLIPSE_DEPLOYMENT. Before release, please change this line to DEPLOYMENT_TYPE = JNLP_DEPLOYMENT.

For details on the deployment procedure, see the Chapter 3.

After deployment, scripts in /cs/cgi/projects/locus/labs/ must be updated to point to a new location. Specifically, in get_jnlp.php one line must be updated to specify the new deployment location in the codebase attribute.
<jnlp spec="1.0+" codebase="http://cgi.cs.arizona.edu/projects/locus/version-5-0/labs/jnlp"/>.

The same changes must be applied to get_jnlp_minimal.php.
Chapter 3

LoCuS Build Scripts

We use the ant build system for all our Java projects. Each Java project directory has its own ant script in its project directory (named build.xml). The ant script of the LoCuS project consolidates all the other ant scripts so building and deploying is simple.

It will be very useful to understand how to write simple ant scripts. Please browse through the available ant scripts (named build.xml) in each project directory.

The ant script of the LoCuS project provides the following ant tasks. To run a particular ant task, go to the locus project directory and run the command ant task-name, where task-name is one of the following.

1. **deploy** – uploads everything so that LoCuS can be downloaded and run off the web. This might take quite a while if you are uploading off-campus. Use partial upload tasks when possible.
2. **docs** – create javadoc HTML files.
3. **makeplugins** – make all available plugins (for new plugins please add an entry on this task).
4. **jnlp** – only create the jnlp deployment.
5. **uploadlib** – upload only the dependency libraries (jar files under locus/lib/).
6. **uploadlocusjar** – upload the main LoCuS jar file (use this to save upload time when only the main LoCuS project has changed).
7. **uploadlabs** – upload only the labs.
8. **uploadplugins** – upload all the plugin jars.

Note that often it is not necessary to deploy everything—for example, just a single apparatus might have changed, or only the main LoCuS jar has changed. In such cases, use the more granular ant tasks, namely, uploadlib, uploadlocusjar, uploadlabs and uploadplugins.

Deploying will be a lot faster if done directly from one of the school’s servers, rather than from your personal computer (even slower if done from off-campus). Make sure to tag the current repository whenever you release a new deployment (see the SVN chapter of the Tau Getting Started Guide).

Deployed releases can be found in the /cs/cgi/projects/locus directory.
Chapter 4

Manual JNLP Deployment

Use the following procedure only when deploying using the ant script fails.

1. Go to the LoCuS project directory (which contains the file build.xml)

2. Run the command ant jnlp. This should create a folder called dist_ant.

3. Edit the file dist_ant/locus.jnlp and make sure the codebase attribute points to the web folder where you wish to deploy the jnlp application. Currently the jnlp application is deployed in /cs/cgi/projects/locus/labs/jnlp/ and /cs/www/projects/locus/jnlp. The former has been used for teachers/schools and the other is for sharing among us (and other contributors elsewhere).

4. Make a copy of the jnlp file for each lab and set the argument tag in application-desc to point to the appropriate lab file. Put all the copies of the jnlp files in the dist_ant folder as well.

5. Upload the contents of dist_ant to your jnlp web folder.

6. Copy all the plugin jar files to the plugin folder in your web folder in (/cs/cgi/projects/locus/locus-version/jnlp/plugins). If plugin jar files need to be created, please see the Apparatus Developer’s Guide.

7. Distribute links to your jnlp files for people to use.

   If you want to update the current certificate (for example you want to change the signer’s identity), please follow these steps.

   1. Go to the files directory inside the LoCuS project directory.

   2. Remove the old key:

   \r
   \texttt{rm keystore}

   3. Run

   \begin{verbatim}
   python genkey.py
   \end{verbatim}

   Use locuspocus as the password or modify locus/build.xml accordingly.

   4. Follow the steps in creating the jnlp deployment given above.
Chapter 5

Standalone jar Deployment

Use these steps to create a “standalone” deployment of LoCuS which can be run on a computer without internet access (unlike the jnlp deployment we have been using).

1. Go to the LoCuS project directory (which contains the file build.xml)

2. Run the command
   
   ant onejar
   
   This should create a folder called dist.ant which should contain everything needed to run LoCuS.
Chapter 6

LoCuS Labs Website for Teachers

We have a simple makeshift website for teachers to conduct LoCuS lab exercises in their classrooms. Currently this website is available at [http://cgi.cs.arizona.edu/projects/locus/labs/](http://cgi.cs.arizona.edu/projects/locus/labs/).

The Naval Post Graduate School is responsible for building the final more comprehensive and scalable web application. In the meantime we needed something to field test LoCuS on local schools.


6.1 Implementation

The website is a **php** application that uses a **mysql** database. The website materials are located at `/cs/cgi/projects/locus/labs/`. You can access it by logging into `lectura`.

Scripts for managing the **mysql** database are located at `/cs/cgi/projects/locus/labs/dbscripts/`.

All **php** scripts perform database operations (add/remove/fetch) via the class `DBManager.php`. Modify this class to add or change database operations as needed.

The **mysql** database has four tables. The script for creating this database structure is available at `/cs/cgi/projects/locus/labs/dbscripts/source.txt`. *Do not execute this script on the current database, or all the data in the database will be destroyed.*

6.2 Connecting to the MySQL Database Directly

Sometimes it is necessary to make changes to the database directly (for example, if we want to change its schema, to backup, or to make changes not supported by the web interface). If you make any structural changes, please update the database script at `/cs/cgi/projects/locus/labs/dbscripts/source.txt` to make sure we can create the correct database in the future when needed.

To connect to the database, execute the following command on `lectura`:

```
$ mysql -h mysql.cs.arizona.edu -pOS5hurts -u locus rts locus
```

Or, simply use the provided script:

```
$ cd /cs/cgi/projects/locus/labs/dbscripts/
$ ./mysqlconnect.py
```
Chapter 7

Past Issues and Solutions

1. CVS and SVN permission issues (access denied etc.)
   If everything you try fails, ask Rick to ask lab to execute the following commands:
   cd /cs/svn/tau
   chown rts locus -R && chmod 660 locus -R && chgrp locus locus -R

2. CVS lock issue
   If a cvs command fails because of a lock issue ("waiting for lock on ...", "lock held by ..." and so on), and no one knows how to unlock, then log into lectura and remove all the files listed by executing the following command:
   find /cs/projects/cvsroot/tau/locus -name ".cvs.*"
   (perhaps we should back up these files before removing them). Note that we are using SVN now.

3. Permission denied issue when accessing files in JNLP deployment
   JNLP grants privileges granted by the user (like file read/write privileges) only to the main application thread. So if you need to load a resource, use the main classloader:
   Thread.currentThread().getContextClassLoader().getResourcesAsStream(...).
   Or, use the functions available in the edu.arizona.locus.core.Utilities class.

4. Loading the LoCuS application from Eclipse fails. Lots of “not found” errors
   Make sure that your working directory is set to locus/dist where all the resources (plugins, configuration files, etc.) are located. To do this, right click on the locus project in the Package Explorer (or Project Explorer), and click Run Configuration under Run As menu.
   Under Arguments tab set Working Directory to $workspace_loc:locus/dist
   Click Apply, click Run.

5. Integrating existing ant scripts in Eclipse GUI
   Thus far we have not found a convenient way to integrate our ant scripts in Eclipse. One easy way to run ant tasks from eclipse is to go to Window menu, Show view → Ant, and add all the build.xml files. Then you can double click on a task to run it.