A Java/XML Distributed Environment for Statistical Computing

E. James Harner, Hengyi Xue, Lingyi Zheng, and Jun Tan

West Virginia University, Department of Statistics

P.O. Box 6330, 423 Hodges Hall

Morgantown, WV, 26506 USA

{jharner, hxue, lzheng, jtan}@stat.wvu.edu

1. The Client/Server Architecture

A distributed Java application, called JavaStat, has been developed for doing statistical analyses (http://www.ci.tuwien.ac.at/Conferences/DSC-2001/Proceedings/). It is designed to run on multiple computers connected by the Internet. The data model and the associated statistical logic of JavaStat reside on the server while plots, reports, and their controllers are located on the clients. JavaStat can be used as a standalone application to perform traditional data analyses or it can be used for collaborative research among a group of users.

The software has a client-server structure. It is organized around the concept of a session. A Session object stores information about a client, including its name and the data it uses. The data of a Session is stored in a DataSet object which derives from an XML data file. A DataSet is a collection of variables with state information. The server module of the software runs on a server machine and it manages sessions (using SessionManager), performs complex computing tasks on behalf of the clients (using StatEngine), and coordinates interaction among collaborative objects (using TaskManager).

The principal responsibility of the client is to access and display DataSet views (spreadsheets, reports, and plots). Clients are lightweight in the sense that they rely on the server module for complex computing algorithms and operations. The classes in the client are organized into 'modules' according to their functionality. For example, a histogram view of a dataset derives from the HistogramModule.

The client communicates with the server using Sun's Java RMI (Remote Method Invocation) protocol. This protocol allows complex objects to be passed among different Java Virtual Machines (JVMs). A client acts as a remote observer for a DataSet, which can have many remote observers. In collaborative mode, a privileged client can make changes to the DataSet and these changes will be propagated to all registered remote observers.

After a client acquires a DataSet from the server, it can display it in either of two views: the table view gives a user a spreadsheet that is used for viewing, changing, and inserting data; the variable-list view is used for data modeling. The user can see which variables are categorical and which are numerical and can assign roles to the variables (explanatory, outcome, and conditioning variables). After the roles are assigned, the user can perform data analyses. This is done through the `Plot` menu. Each of these menu items will do different default data analyses based on the characteristic of the variables. For example, if both the x and y variables are categorical, then a mosaic plot is created by default and a contingency report is generated.

2. R as a Computing Engine

JavaStat allows the user to do standard analyses and advanced plotting, but it was not designed for complex modeling tasks. Instead, R will be used as a backend computing engine for JavaStat. The use of R within JavaStat in not intended to replicate the full functionality of R. Rather, R will be used to enhance JavaStat—initially by using the advanced modeling functions of R. The R Interface to Java (http://www.omegahat.org) by Duncan Temple Lang makes this possible. The interest here is to allow R functions to be used to implement Java interfaces. With this interface, we can access R modeling functions from JavaStat. The result of invoking an R function is an R object—typically a list. When this is passed back to the Java method that called it, standard mechanisms for converting R objects to Java objects are applied. Reports and plots can then be generated from the converted R-Java objects.

3. Future Work

JavaStat is designed for doing statistical analyses. A companion component for manipulating mathematical expressions, called JavaMath, is also being developed. JavaStat and JavaMath both have the same underlying structure based on a collaborative session. Therefore, the SessionManager and TaskManager of JavaStat are being moved to a higher level in a system called JEMS (Java Environment for Mathematics and Statistics). Currently, JavaMath is in its initial stage of development and is principally used for manipulating mathematical expressions dynamically as represented in a MathML-based tree structure (http://www.w3.org/Math/).

RÉSUMÉ

Le logiciel collectif Javastat distribué par Java a été developpé pour conduire des analyses statistiques. Le logiciel a été developpé pour opérer sur plusieurs ordinateurs connectés via l'internet. Les banques de données et la logique statistique du programme se retrouvent sur la serveur alors que les graphiques, rapports et controlleurs sont localiss chez les abonnées. Le logiciel Javastat peut être utilisé comme application individuelle pour l'analyse conventionelle de données ou encore, dans le cadre d'une recherche collective par une groupe d'utilisateurs. Le logiciel permet à l'utilisateur de faire des analyses conventionelles graphiques jumellées a des projections graphiques avancées mais n'a pas été conu pour de taches complexes de modéllisation. Plutôt, 'R' sera utilisé comme ordinateur de support pour de logiciel Javastat.