DRAGONFLY: A JAVA-BASED IAMC CLIENT PROTOTYPE

WEIDONG LIAO, PAUL S. WANG
Institute for Computational Mathematics
Department of Mathematics & Computer Science
Kent State University
Kent, Ohio 44242, U.S.A.
E-mail: wliao@mcs.kent.edu, pwang@mcs.kent.edu

Dragonfly is an evolving prototype client for *Internet Accessible Mathematical Computation* (IAMC). Developed in Java, Dragonfly is a protocol-compliant end-user client for making mathematical computing readily available on the Web/Internet. Features of Dragonfly include remote help, input editing, command template, history, textbook-style mathematical expression display, as well as interactive display and manipulation of 2-D and 3-D plots of mathematical functions. To represent graphing data, an XML-based MathML extension for graphing is proposed. Dragonfly also features an architecture for easy extension. A system demonstration supplements the paper presentation.

1 Introduction

The Internet and the World-Wide Web make many kinds of information and services easily accessible. Ad-hoc methods have been used to make mathematical computing available on the Internet/Web. On SymbolicNet for example, one can find demonstrations of mathematical computations ranging from differentiation, integration, polynomial factorization, to plotting of curves and surfaces and LATEX code generation. Such access is made through direct TCP/IP link, remote X connection, CGI scripts, or Java Servlet/Applet.

The Institute of Computational Mathematics (ICM) at Kent State University, together with collaborators at other institutions, is pursuing the IAMC (Internet Accessible Mathematical Computation) research project to address a number of critical issues for interactive access to mathematical computing on the Internet. The goal of IAMC is to make all kinds of mathematical computations easily accessible on the Web and the Internet. This topic is a focal point of discussion at the 1999 *IAMC Workshop* part of the *International Symposium on Symbolic and Algebraic Computation* (ISSAC'99), held late July in Vancouver, Canada. For more background and related activities, the reader is referred to the online Proceedings of the IAMC'99 Workshop and the IAMC homepage [a], and the Workshop on *The Future of Mathematical Communication* (FMC Dec. 1999)[b].

[a] `http://horse.mcs.kent.edu/icm/research/iamc`
[b] `http://www.msri.org/activities/events/9900/`

The design and implementation of an IAMC client prototype, Dragonfly, is the topic here. Among the many IAMC components, the client prototype is the most important in making the system appealing and attractive to end-users. A short review of user interface systems for scientific computing and an overview of IAMC set the stage for presenting the design and features of Dragonfly.

1.1 Scientific User Interfaces

A well designed user interface can make scientific systems not only more appealing and convenient but also more powerful to users. On top of the the usual requirements, a scientific user interface must understand and support the display and manipulation of mathematical symbols, expressions and formulas, as well as plots of curves and surfaces. An important trend in the scientific user interface research is to separate user interface from the back-end compute engines.

Different user interface features have been explored: two-dimensional editing of mathematical formulas in experimental systems such as MathScribe, GI/S, or SUI and later in mathematical assistants such as Milo and Theorist; quality curve and surface plotting in Axiom, Maple, Mathematica, and SUI; separating the user interface from the compute engine in Maple Polyith, CaminoReal, MathStation, SUI, and CAS/PI[8].

As an example of a user interface for compute engines connected over a LAN, GI/S by Young and Wang [3] explored many sophisticated features. Designed for the Unix/X11 environment, GI/S employs a window manager to allow multiple overlapping windows, popup menus, command line editing, history mechanism, and 2 and 3 dimensional mathematical function plotting. A distinct feature of GI/S is its use of display data structure that is not necessarily related to the expression data structure used by the underlying compute engine. This helps the separation between user interface and compute engines.

The successor to GI/S, SUI (Scientific User Interface) [2], improved the GI/S in several aspects. SUI specifies a SUI-to-engine interface, which defines several categories of requests: display requests, requests to another engine, requests to SUI, requests for generic services. These make SUI a more formalized client-server system. SUI also adds concurrent use of the different servers and provides buttons to interrupt computations from the user interface. For more information and references on user interfaces for computer algebra systems, the reader is referred to the survey article[9].

In the Internet area, the IBM digital publishing group has released the experimental Techexplorer, a Web browser plug-in that dynamically formats and

displays documents containing scientific and mathematical expressions coded in Tex/Latex. Some MathML is also supported. Techexplorer also allows a user to send expressions to a fixed compute server for evaluation.

WebEQ [15] from Geometry Technologies Inc. is a Java class package that can be used to display WebTex and MathML in a Java application or applet. The W3 Consortium's Amaya allows users to browse/edit Web pages containing mathematical expressions. Together with the rest of the Web page, these expressions are manipulated through a WYSIWYG interface.

2 IAMC Overview

Dragonfly is a prototype client for IAMC, a distributed system to make mathematical computing easily accessible and interactively usable on the Internet [11]. The overall IAMC architecture is shown in Figure 1. In summary, IAMC consists of the following components:

1. IAMC client (Icl) — The end-user agent for accessing services provided by any IAMC server.
2. IAMC server (Isv) — The server providing computational services through the IAMC protocol layers. An Isv may or may not employ one or more external compute engines to perform mathematical computations.
3. Protocol layers — A series of protocols used for connecting IAMC clients and servers. The IAMC protocol layers are shown in Figure 2.
4. Mathematical Data Encoding — Standard or user-defined mathematical data encoding formats.
5. The External Engine Interface (EEI) — A specification and API implementation for binding existing compute engines to IAMC servers.

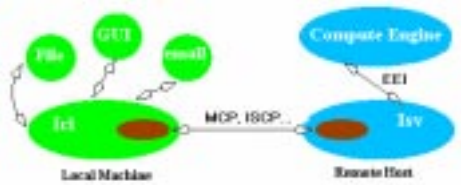

Figure 1: IAMC Architecture

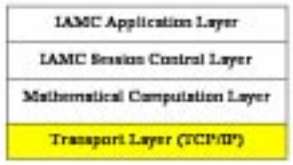

Figure 2: IAMC Protocol Model

2.1 Mathematical Data Encoding

Mathematical data encoding is an active research topic in mathematics and the symbolic computation community has lead the way in this area [11]. OpenMath [1]

is a standard for representing semantically rich mathematical objects and making them exchangeable between mathematical and non-mathematical systems. MP [4] is another mathematical data encoding format that uses binary-encoded annotated parse trees for efficient exchange of mathematical expressions. For the Internet/Web, MathML [7] defines a hypertext-based standard for both presentation and content markup of mathematical expressions.

IAMC supports MathML and MP encodings by default. MathML is expected to be widely used in Web pages to include mathematical content and its rendering software will become widely and freely available. Dragonfly interfaces to WebEQ as rendering software for mathematical expression display.

2.2 A Graphing Extension to MathML

The SIG [13] package defines a simple but effective method to represent graphs. The disadvantage for this method is that its data representation is ad-hoc. In an open environment like IAMC, a more structured encoding format is desired. XML [16] is a good choice for this purpose. XML can encode both presentation and semantic information, which is essential to the IAMC framework. XML is aslo becoming an encoding standard for data exchange. Many Application Programming Interfaces (API) are available to make XML encoding/decoding easy.

Because MathML is also an application of XML, it won't be necessary to define another markup language. Extending MathML seems a good way to include mathematical graphing information. Several new tags have been added to MathML, including <math-graph>, <xrange>, <yrange>, <coordinates>. This representation for mathematical graphing data is called MathML/Graph. For detailed information about MathML/Graph, The readers are referred to ICM technical report in ICM home page [6].

Dragonfly is part of the IAMC prototyping effort [10] that also includes an MCP protocol library, a server (Isv), and an external engine interface (EEI). Hereafter we will focus on Dragonfly.

3 Dragonfly Capabilities and Organization

As a prototype implementation of IAMC clients, the functional requirements for Dragonfly can be outlined as follows:

1. Connect to any user-specified IAMC server
2. Obtain capability and usage documentation from the IAMC server and make them available to the user
3. Receive computational, control and help commands from the user and send them to the server

4. Parse infix mathematical expressions entered by the user
5. Receive results from the server
6. Display mathematical symbols and expressions in textbook-like fashion
7. Plot mathematical curves and surfaces
8. Present command templates from server to the user as required
9. Display server dialog and relay user data thus obtained back to the server
10. Encode/decode mathematical and graphical data
11. Connect and communicate with server using the MCP protocol
12. Provide an architecture and organization to allow easy extensions and plug-ins to add features and functionalities as the prototype evolves

Corresponding to the above requirements, the main components and the corresponding Java class packages for Dragonfly are listed as follows.

- *org.icm.IAMC.icl* - The Overall Control Classes for Dragonfly
- *org.icm.IAMC.icl.ui* - User Interface Package
- *org.icm.IAMC.icl.render* - Mathematical Expression Rendering Package
- *org.icm.IAMC.icl.plot* - 2-Dimensional and 3-Dimensional Plotting
- *org.icm.IAMC.icl.parser* - Parser for mathematical expression
- *org.icm.IAMC.icl.help* - Help library
- *org.icm.IAMC.icl.util* - Utilities
- *org.icm.IAMC.icl.MCP* - MCP Protocol Library
- *org.icm.IAMC.icl.ISCP* - IAMC Session Control Protocol Library

Dragonfly is developed on the Java 2 platform, with the WebEQ Java library and IBM XML4J as supporting packages. Swing/Java 2D in Java 2 platform are employed to implement the user interface, the HTML-based help facility, and 2-dimensional and 3-dimensional mathematical function plotting and manipulation. WebEQ is used for rendering mathematical expressions in textbook style. XML4J helps encoding/decoding of MathML/Graph data.

4 Using Dragonfly

When Dragonfly is initially invoked, a browser window appears that contains a *Location* Box, an *Output Panel* and a input *Command* Box. The user can select or type a desired IAMC URL in the Location Box, which identifies an IAMC server possibly anywhere over the Internet. Dragonfly then connects to the target IAMC server. If the connection succeeds, a welcome message from the IAMC server (Isv) appears in the Output Panel. Otherwise, an error message appears, stating what went wrong with this connection.

The *Command* Box is for typing user commands, which will be echoed in the Output Panel simultaneously. This command typed in *Command* Box will be sent to the Isv and the result obtained will be shown in the *Output Panel* in 2-dimensional format. A button labelled *Remote Help* next to the *Command*

Figure 3: *M*ain User Interface

Box is for getting help about computational commands and their usage from the Isv.

A detailed sample Dragonfly session can be found in the technical report on Dragonfly that can be found in ICM home page [6]. In general, the usage of Dragonfly follows these steps:

1. Start Dragonfly
2. Enter IAMC URL in Location Box
3. Check available commands and their usage for the current Isv by clicking on the Remote Help button
4. Enter commands and view results;
5. Close the connection to the current Isv.

5 Remote Help

While all IAMC servers can perform basic computational operations, their capabilities and usages vary. Providing a way to check the commands supported by an Isv and to obtain their usages becomes essential. The Dragonfly Remote Help is a solution.

The *Remote Help* facility can be triggered by clicking on a button labelled

Remote Help. Once Dragonfly connects to an Isv, the Remote Help button is enabled. Clicking on this button will send a message to the Isv to obtain the Isv-specific help document that is in HTML or other MCP-accessible formats. The help information will be displayed in a pop-up Help Window.

The Remote Help button is context-sensitive. If the Command Box is empty when Remote Help button is clicked, a list of available commands will appear in the Help Window. If the Command Box contains a command, the Remote Help button retrieves a detailed description about this command.

6 User Commands

The current version of Dragonfly provides two facilities for users to enter commands. A *Command Box* in the Dragonfly main workspace, as shown in Figure 3 can accept the commands in text format. Besides, a *Command Template* is also available from the Dragonfly Menu to help enter many useful commands correctly. Later on Dragonfly will introduce a two-dimensional editor to enable users to edit mathematical expressions in a visual way.

Commands typed in the Command Box will be echoed in the Output Panel as well. The Command Box supports convenient text editing for modifying commands and correcting typing errors. The Command Template is a mechanism to guide the user to enter commands and arguments correctly. Simply put, a command template is a dialog for users to fill in the variables, expression, and parameters corresponding to a mathematical operation.

Figure 4: *Dragonfly* Command Template Window

Figure 4 shows a template for an integration operation. Not all mathematical operations have templates available. The commands with templates available can be found in Template menu and their command templates can be invoked there too.

The commands entered through Command Box or Command Template are saved in a history buffer for later reference. The commands previous typed in Command Box or Command Template window can be tracked by using history command. All these commands are organized in a history list in which each command is assigned a number. This number can be used to invoke the corresponding command, which leads to a generalized redo facility.

7 Output Panel and Mathematical Expression Rendering

The *Output Panel* in Dragonfly provides full features for users to track computational sessions. It not only outputs the results, but also echoes the commands entered in Command Box. The content in Output Panel can also be saved in disk files, in MathML or MP format, for later use or for transmission to another Isv. The Output Panel displays the mathematical results are in 2-dimensional graphical format by defaulti, but the users can also to select other output formats including infix, prefix, postfix, MathML, and MP from the *Output* menu.

Dragonfly leverages the WebEQ class library to render mathematical expressions in 2-dimensional form. WebEQ renders mathematical expression by using a *rendering tree*. The nodes of a rendering tree consists of "schema" classes which correspond to different mathematical notation layout schemas, such as fractions, superscripts, square roots, etc. To render an expression, the expression is first parsed to create a rendering tree. The rendering tree is then installed in a so-called Equation Panel that performs the actual 2-D rendering.

8 Mathematical Graphs

Dragonfly supports 2-dimensional and 3-dimensional plotting. It plots 2D/3D graphics in a separate window and allows interactive manipulation of the graphics displayed. The Dragonfly plotting module takes MathML/Graph (Section 2.2) encoded data from any Isv and performs rendering and interactive manipulation. The functionalities are powered by Java 2D.

1. The user enters the plot command in the Command Box
2. The plot command is sent to the Isv
3. The Isv forwards the plot command to a compute engine
4. The coordinate data is returned to the Isv
5. The Isv encodes the coordinate data set, by XML, in MathML/Graph
6. The Isv compresses the MathML/Graph encoded data before sending back to the Icl
7. The Icl decompresses the result

8. The Icl decodes the MathML/Graph dat by using XML4J
9. The graphics is rendered in a Dragonfly Graph Window

Figure 5 shows the 2-D and 3-D Graph Window.

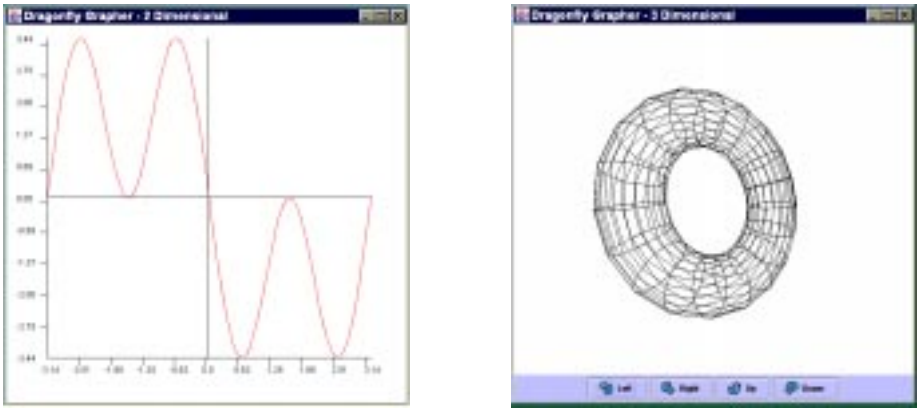

Figure 5: *2*-d and 3-d Plotting

9 Future Work

Dragonfly demonstrates a prototype IAMC client. It will surely evolve as the IAMC architecture moves forward. Continuing work include the support of MP-encoding format and a sophisticated mathematical expressino editor. Efforts are also ongoing integrate Dragonfly with a Web browser, as well as other Internet client software. Dragonfly can open the door for students, teachers, scientists, and engineers to access a wide variety of services involving mathematics on the Web and Internet.

Acknowledgements

Work reported herein has been supported in part by the National Science Foundation under Grant CCR-9721343 and the Ohio Board of Regents Computer Science Enhancement Funds.

References

1. ABBOTT, J., DIAZ., A. and SUTOR, R. S. *Report on OpenMath.* ACM SIGSAM Bulletin (Mar. 1996), 21-24.

2. DOLEH, Y. and WANG, P. S. *SUI: A System Independent User Interface for an Integrated Scientific Computing Environment.* In Proc. ISSAC 90 (Aug. 1990), Addison-Wesley (ISBN 0-201-54892-5), pp. 88-95.
3. Young, D. and WANG, P. S. *GI/S: A Graphical User Interface for Symbolic Computation Systems.* In Journal of Symbolic Computation. Vol 4. No. 3. 1987. pp. 365-380.
4. GRAY, S., KAJLER, N. and WANG, P. *MP: A Protocol for Efficient Exchange of Mathematical Expressions.* In Proc. ISSAC'94 (1994), ACM Press, pp. 330-335.
5. GRAY, S., KAJLER, N. and WANG, P. *Design and Implementation of MP, A Protocol for Efficient Exchange of Mathematical Expressions.* Journal of Symbolic Computation 25 (Feb. 1998), 213-238.
6. ICM Home Page. `http://horse.mcs.kent.edu/icm`.
7. ION, P., MINER, R., BUSWELL, S., S. DEVITT, A. D., POPPELIER, N., SMITH, B., SOIFFER, N., SUTOR, R. and WATT,S. *Mathematical Markup Language (MathML) 1.0 Specification.* (www.w3.org/TR/1998/REC-MathML-19980407), Apr. 1998.
8. KAJLER, N. *CAS/PI: a Portable and Extensible Interface for Computer Algebra Systems.* Proceedings of ISSAC'92, ACM Press 1992
9. KAJLER, N. and SOIFFER, N. *A Survey of User Interfaces for Computer Algebra Systems.* , Journal of Symbolic Computation 25 (Feb. 1998), 127-159.
10. LIAO, W. and WANG, P. S. *Building IAMC: A Layered Approach.* Proc. International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'00). pp. 1509-1516.
11. WANG, P. S. *Design and Protocol for Internet Accessible Mathematical Computation.* In Proc. ISSAC'99 (1999), ACM Press, pp. 291-298.
12. WANG, P. S. *Internet Accessible Mathematical Computation.* In Proc. 3rd Asian Symposium on Computer Mathematical (ASCM'98) (Aug. 1998), pp. 1-13.
13. WANG, P. S. *A System Independent Graphing Package for Mathematical Functions.* Proceedings, International Symposium on the Design and Implementation of Symbolic Computation Systems, DISCO'90, Capri, Italy, April 1990, Springer-Verlag Lecture Notes in Computer Science, Vol. 429, pp. 245-254.
14. WANG, P. S., GRAY, S. and ZHANG, B. *Mathematical Computational Protocol Specification.* Initial Draft. Oct., 1999.
15. WebEQ Home Page. `http://www.webeq .com/`.
16. XML Home Page. `http://www.xml.org/`.