

IAMC Framework
Architecture and Prototyping
A Progress Report

P. Wang, S. Gray, N. Kajler, D. Lin, W. Liao, X. Zou

Contents

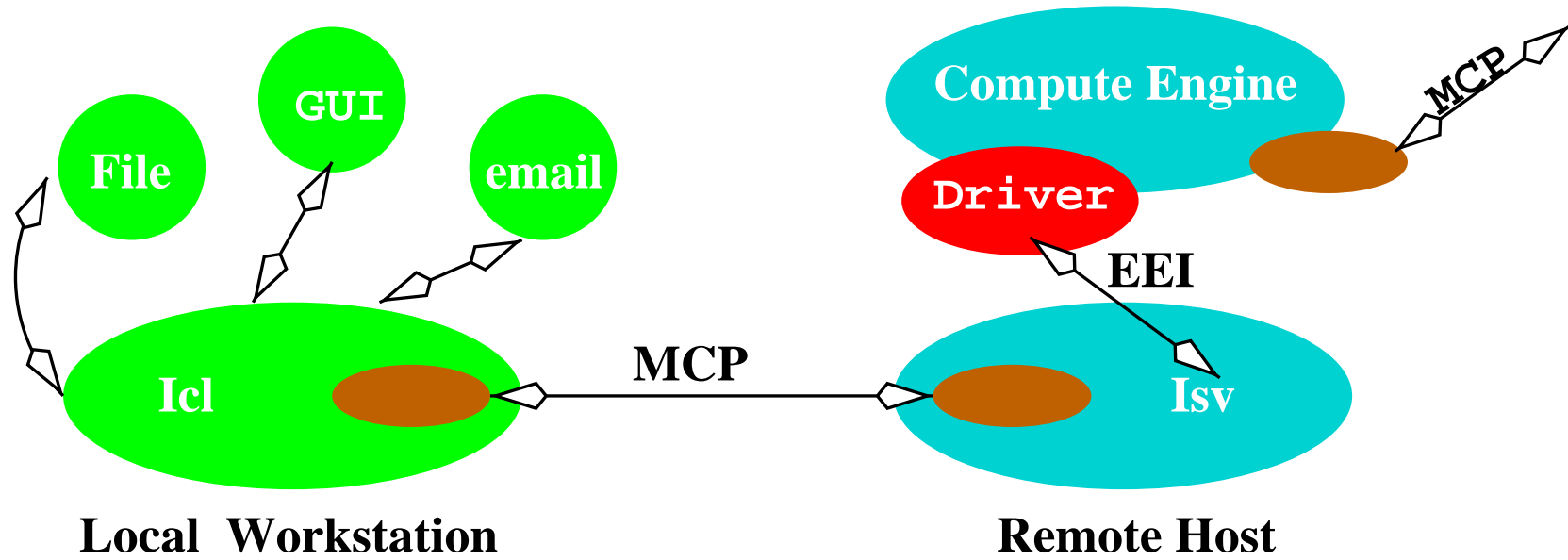
- IAMC Framework goals
- Architecture overview
- Dragonfly – an IAMC client prototype
- MCP – mathematical computation protocol
- MathML/Graph
- Starfish – an IAMC server prototype
- External engine interface
- Web accessibility
- Further work

IAMC Framework goals

The IAMC framework is an effort to establish a protocol-based, platform, programming language, and mathematical encoding independent, solution for serving mathematical computation over the Web/Internet.

- To make math-oriented data and services easily and widely accessible on the Internet in many contexts – directly, via the Web, by email, for distance learning, etc.
- To support interactive use of user-designated remote *compute servers*.
- To provide efficient and effective communication of mathematical data over the Internet.
- To allow exchange and further processing of computational results among different compute servers.

IAMC Architecture



IAMC Framework Components

1. IAMC client — An end-user agent for accessing services provided by any IAMC server.
2. IAMC server — A program to provide mathematical computation powers through the MCP protocol.
3. Protocol — An effective request-response protocol to link clients and servers and to support one-time transactions and interactive sessions.
4. Mathematical Data Encoding — Standard and user-defined mathematical data encodings can be used.
5. External Engine Interface (EEI) — A specification and API implementation for binding compute engines to IAMC servers.


Dragonfly Functionalities

- Connect to and communicate with any user-specified IAMC server via the MCP protocol.
- Obtain capability and usage documentation from the IAMC server and make them available to the user.
- Receive computational, control and help commands from the user and send them to the server.
- Parse infix mathematical expressions entered by the user.
- Receive results from the server.
- Display mathematical symbols and expressions in textbook-like fashion.

- Select subexpressions with the mouse.
- Save mathematical results in files under well-defined formats such as MP, MathML, and OpenMath.
- Plot mathematical curves and surfaces.
- Present command templates from the server to the user when requested.
- Display server dialog and relay user data thus obtained back to the server.
- Encode/decode mathematical and graphical data.

Dragonfly Implementation

- Written in Java
- Swing to implement the GUI and the HTML-based help facility.
- Java 2D to support 2-dimensional and 3-dimensional mathematical function plotting and manipulation.
- WebEQ to render mathematical expressions in MathML presentation code.
- XML to help encode/decode graphing data.


Integration Template

Example: integration(sin(x),x,0,pi/2)

Integrate (, integrand $f(x)$
 , variable x
 , lower limit a
) upper limit b

Clear Fields

Enter Command

The MCP Protocol

- One-time requests and persistent computation sessions.
- No restrictions on content types or mathematical encodings.
- Supporting the special needs of mathematical computations.
- Permitting both server and client to send requests and to return responses.
- Both synchronous and asynchronous message exchanges.
- Distinguishing protocol control from computation control.
- Simple, effective, and extensible.

MCP Message Classes

- *Initialization* — The initialization class supports session creation and configuration right after client-server connection.
- *Control* — The control class supplies MCP protocol control and management methods for both the client and server side.
- *Computation* — A class of server-side operations to perform application supported computations.
- *Dialog* — A class of client-side methods to solicit information from the end user.

Initialization Message

Request Initialization S1

Method: canDo

Version: MCP/1.0

Server-name: PolyFactor

Content-length: 128

"Calculus"=integrate diff taylor...

"Linear Algebra"=vector matrix determinant ...

"Complex Analysis"=absValue conjugate realPart ...

Response Initialization S1 100 OK

"Area_name"=*command1 command2* ...

Mathematical Data Encoding

IAMC Framework is Mathematical data encoding neutral. The Dragonfly-Starfish prototype knows about infix, MathML, and MP.

1. Dragonfly parses infix input into MathML content encoding.
2. MathML content data is sent through MCP with automatic MP compression/decompression.
3. Starfish sends MathML content encoding to a compute engine.
4. Starfish receives MathML content and/or MathML presentation result from engine driver.
5. Starfish sends the result back to the Icl via MCP.
6. MathML (presentation) is rendered for end-user.

MathML/Graph

Graphing tags for:

$$y = \cos(x), \quad 0 \leq x \leq \pi$$

```
<mathGraph name="mycosine"
  type=rectangular
  dimension=2 plotcolor=black
  bgcolor=white>
<equations type=normal>
  <apply> <eq/> <ci>y</ci>
    <apply> <cos/> <ci>x</ci> </apply>
  </apply>
</equations>
```

```
<range>
  <var type=independent> <ci> x </ci> </var>
  <lower> <cn type="integer">0</cn> </lower>
  <upper> <apply> <times/>
            <cn type="integer">2</cn>
            <ci type="constant">&pi;</ci>
          </apply>
  </upper>
</range>
...
```

```
<coordinates type="rectangular" dim="2"  
    valuetype="float"  
    point="x,y" points=40>  
<cn type="integer">0</cn>  
<cn type="integer">1</cn>  
<cn>0.16041128085776021</cn>  
<cn>0.98716167535309518</cn>  
    ...  
</coordinates>  
</mathGraph>
```

Server Prototype: Starfish

iamc://host:port/isv_name

- Be MCP compliant and easily customizable.
- Maintain and manage computation sessions.
- Launch/control external compute engines dynamically.
- Support synchronous/asynchronous calls, callbacks, and client-generated interrupts.
- Supply server administrative functions.
- Allow plug-in modules for extensibility.

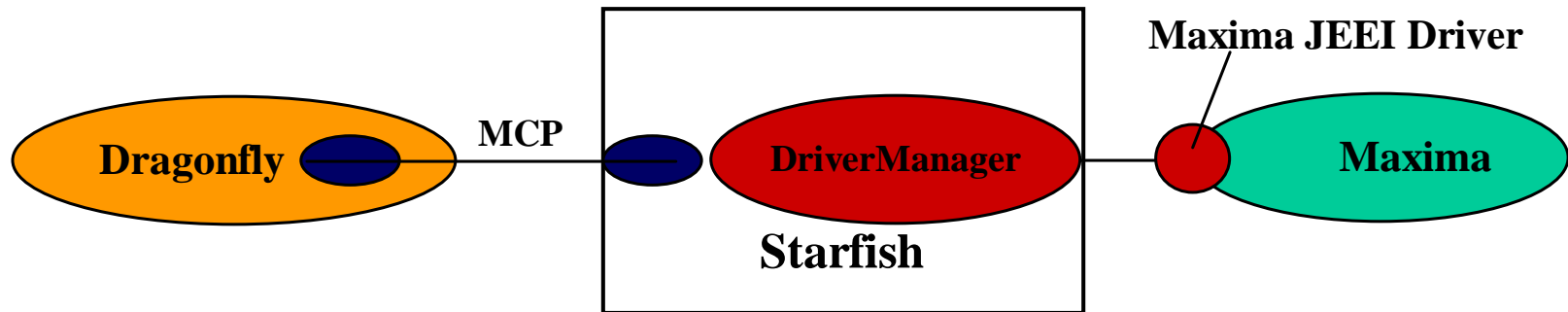
External Engine Interface

- perform initializations before receiving user commands. Usually, the initialization sets up the input/output modes, processes customization/configuration parameters, etc.
- execute in several different modes such as normal mode and debug mode.
- send a prompt when it is ready for input. Different compute engines use different prompts. The prompt also indicate the current engine status/mode.
- require a terminating character for each command.
- return a result or an error message for a command. Results may

contain text, mathematical expressions, and graphics. The beginning and end of the result are well-defined.

- ask for extra information from the user in order to complete a particular computation.
- support several types of user-generated interrupts.
- keep track of commands and results in generated variables.
- supply help and documentation information directly or from some other source.

IAMC Prototyping Structure



Accessibility from the Web

Dragonfly is a stand-alone application. Ways to access an IAMC Framework client from the Web:

- An IAMC-aware browser may launch its IAMC helper (Dragonfly) when following an IAMC URL.
- A Web server-side content type

Content-Type: application/x-iamcurl

can launch a user-configured client.

- Converting Dragonfly into a browser plug-in will help embed interactive computations within Web pages.
- Library modules can be developed to enable Web server-side programs to function as compact IAMC clients and easily request and obtain mathematical results from IAMC servers.

Further Work

- XML implementation of MathML/Graph
- an extensible mathematical expression encoding converter
- Converting Dragonfly into a valid plug-in to be used with Netscape
- Creating a large collection of reusable Java classes covering all aspects of the IAMC implementation
- Supporting the rapid development of clients, servers and external engine interfaces.
- Collaborating with many people from all parts of the world.