Toward an protocol for Internet accessible mathematical computation

Andrew Solomon

June 1999

A report on work in progress at

The University of St. Andrews,

- Member of the ESPRIT OpenMath Consortium
- GAP Development Headquarters

GAP is both a command line calculator and a fully featured software development environment for computational abstract algebra.

Internet Accessible Mathematical Computation

- One of the aims of OpenMath is to support Internet accessible mathematical computation;
- this requires a **standard protocol** for client-server computation.

A conundrum

- To specify a protocol is *outside the scope* of OpenMath.
- To have a standard protocol is *essential* to OpenMath.

The solution

OpenMath developers must:

- work towards using a single generic protocol amongst themselves;
- *recommend* this generic protocol to potential OpenMath developers.

The purpose of this talk is to describe the protocol which is emerging from St. Andrews' work within the ESPRIT consortium.

Motivating examples

- a generic abstract algebra compute server based on GAP. in the first instance this will serve:
  - group character tables to a web page client;
  - Galois group computations to Axiom;
  - simple integer computations to IDA.
- machinery for GAP to be a client of servers, including:
  - Axiom – for polynomial factorization, Gröbner basis;
  - perhaps Maple?

Terms of Reference

- *interoperability* the ability to use different mathematical software tools together on the same problem. It is the result of *tool integration*.
- *Internet accessible mathematical computation* (IAMC) we use this as a blanket term covering all aspects of (possibly remote) interoperability.
- *math encoding* examples are OpenMath, content MathML and MP. These are standards or conventions for representing mathematical objects. They are intended to be neutral with respect to the syntax of mathematical software packages and aim to express mathematical objects rather than computational behaviour.
- *mathematical computation protocol (MCP)* a protocol to

coordinate client-server interaction for IAMC.

- *compute engine* program used by an MCP server to actually perform the required mathematical computation. Typically this will have its own syntax and data structures.

Issues and Principles

Issues and principles which guided the design of an MCP:

Issue There are multiple math-encodings which a compute server can reasonably expect to encounter.

Principle (encoding neutrality) A standard MCP should not exhibit a bias towards a particular math encoding.

Issues and Principles

Issue Attempting to request a computation in a single expression quickly leads to very complex expressions arising.

Issue Input which is heterogeneous (with respect to math encodings) cannot be achieved in a single expression.

Principle (accessible context) The server can have a continuously evolving state, or *context*, and the client must be able to refer to objects in the context of the server.

Issues and Principles

Issue Several different IAMC servers may provide the same computations.

Issue Several different IAMC clients may require the same computation, but with the result expressed differently to suit the clients' capabilities.

Principle (client-server anonymity)

- Servers which perform the same operations should be transparently interchangeable regardless of how they are implemented and what compute engine they use.
- Clients must be able to specify, in a generic way, the form which the result is to take without the server knowing 'who' the client is.

Issues and Principles

Corollary (of the principle of client-server anonymity) *commands* to the server should not be encoded in the language of the compute engine.

Principle (small instruction set) there should be a small but complete set of instructions in the MCP.

The proposed MCP *in abstractio*

Assumption (assign and retrieve are sufficient)

We claim that a sufficiently general archetype for a computation session is a number of repetitions of the following sequence of instructions:

```
a1 := Obj1;
a2 := Obj2(a1);
...
ak := Objk(a1, a2, ..., ak-1);
Retrieve(ak, MCPFormObject);
```

That is to say, `assign` and `retrieve` together with being able to refer to objects in the server context is a 'complete' instruction set.

Assumption (reasonable math encoding) A math encoding used for MCP should be expressive enough to describe the required form of an object in that encoding, or it should be so restrictive that there is a definite canonical choice in the matter.

The MCP *in carnae*

There are several choices for a basic protocol in which to realize our abstract MCP.

- Knowledge Querying and Manipulation Language (KQML);
- Foundation for Intelligent Physical Agents' Agent Communication Language (FIPA ACL);
- Wang's MCP.

Example using Wang's MCP

Pseudo-code of the session between WebCharacter and PermGroupServer:

```
g := Group((1,2,3));
cg := CharacterTable(g);
m := MatrixOfCharacterValues(cg);
Retrieve(m, OMMCPFormMatrixOfCyclotomics);
```

Client requests computation session with server

```
Control
ControlRequest: C1-initialization
Version: MCP/1.0
User-agent: WebCharacter
Accept: text/OpenMath
```

Server responds to client

```
Control
ControlResponse: C1
Version: MCP/1.0
Status: normal
Server-name: PermGroupServer
Greeting: Computes with permutation groups.
Accept: text/OpenMath
CanDo: basic.cd group.cd mpc.cd
Accept: text/GAP
CanDo: Group, Elements, Size, CharacterTable, ListPerm, PermList
HasState: yes
```

An assign message

Fast forward to the command

```
m := MatrixOfCharacterValues(cg);
```

This is expressed as:

```
Command
Sequence: 3
Command-literal: assign
Content-type: text/OpenMath
SendResult: no
MCPHandle: "m"
```

```
<OMOBJ>
  <OMA>
    <OMS cd="group" name="MatrixOfCharacterValues"/>
    <OMA>
      <OMS cd="mcp" name="handle"/>
      <OMSTR "cg" \>
    </OMA>
  <OMA>
</OMOBJ>
```

The client retrieves the matrix

The client sends a retrieve message, stipulating the form of the response in the message body.

```
Command
Sequence: 4
Command-literal: retrieve
Retrieved-Content-type: text/OpenMath
Content-type: text/OpenMath
SendResult: yes
MCPHandle: "m"

<OMOBJ>
  <OMS cd="mcp" name="FormMatrixOfCyclotomics"/>
</OMOBJ>
```

The server sends the matrix

```
Result
Sequence: 4
Status: normal
Content-type: text/OpenMath

<body contains a matrix of cyclotomics in OpenMath encoding>
```