

An Architecture for NA3

Next Steps

January 2007
Paris Workshop

Before I begin

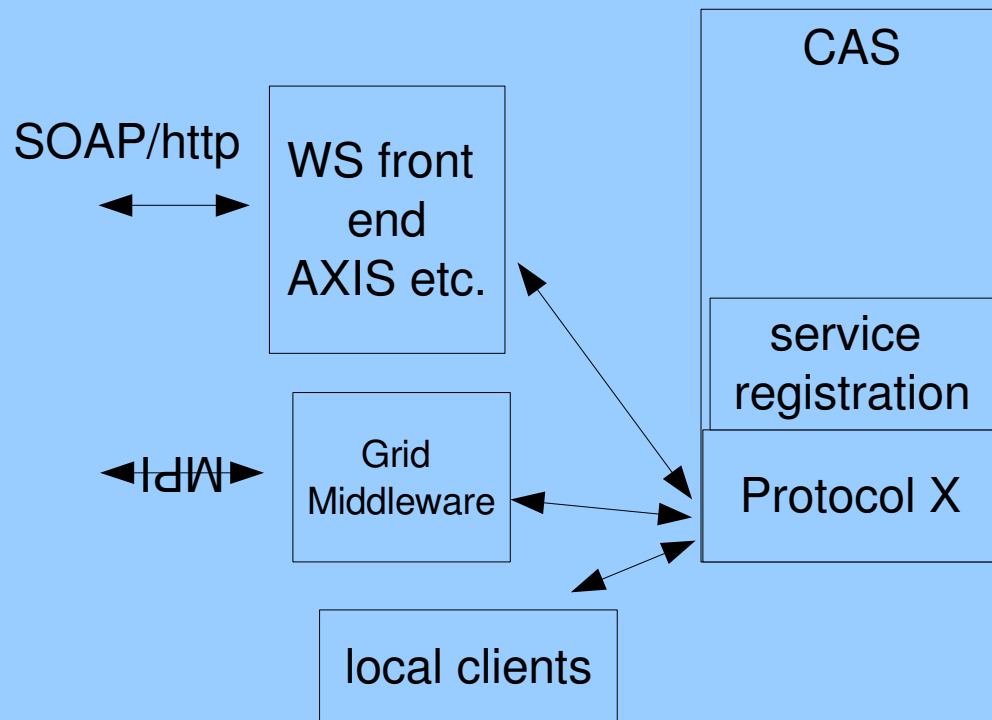
- A few reminders
 - Use the Wiki!
 - talks from this meeting
 - interesting links
 - useful documents
 - anything else you want to share
 - Send Bernd information for the Website!
 - more complete partner information
 - descriptions of activities
 - public documents
 - Prepare for the annual report!
 - send me information on “use and dissemination”
 - get ready for the accounting deadlines.

- **Easy, robust and reliable** way for users to create and export services implemented in any of our (or other) systems.
 - might be generic “simplify this OM object” service or even “execute this string” or specialised “look this up in your database”
- **Equally easy** way to use services you know about
 - less concerned about discovery/broking
- Service providers and consumers might be:
 - different systems running locally
 - multiple copies of the same system distributing a computation
 - a local system and a remote service with specialised capabilities
- Even though we're talking about Web services, the Web may not be involved.

- We're committed to Web services (ie SOAP/http)
 - this is a good idea in many circumstances
 - But it's too heavy in other circumstances
 - cooperating processes on same machine
 - closely coupled parallel computing
- So handle Web Services features in a proxy
 - define a protocol for the proxy to talk the CAS which is
 - lightweight (as simple as possible but no simpler)
 - sufficient to be used without the Web services proxy for the other applications.
 - similar to the MapleNet architecture.

Architecture Overview

- service registration associates CAS functions with service names
- WS front end is common component.



Ideas Towards Protocol X

1. It should be XML based

It makes no sense to do a text protocol which isn't.

2. Basic operation is a function call

pass fn name and argts return result.

Arguments can be basic:

string, byte stream, int

or complex

OM, MathML

- We already have an XML application which can represent a function call, basic types and complex types: OpenMath
- So we proposed that each fn call (and result) would be an OM Object
 - but, if the service designer chooses, that OM Object might just be an OMString or OMBytes containing the data in some other format
- Can include MathML by a hack (attribution).

- As I see the interface on the server-side, the service provider
 - starts the CAS
 - registers one or more CAS functions as services, specifying service names and some information about argument and return types
 - this presumably from a CAS program run at startup
 - tells the CAS instance to become a server, specifying ports, etc.
 - possibly starts a WS proxy
 - or maybe that started the CAS and read a file which did the above

Example

- Suppose I'm doing parallel processing in GAP
- I want to have lots of slave processors doing a specific computation for which I have written a GAP function foo
- So each slave registers a service called (say) “do-foo” and tells the GAP interface that calls to do-foo should be passed to the GAP function foo.
- The arguments to do-foo might as well be the GAP ASCII (or even binary) representations, so as far as the Web services layer is concerned do-foo just takes string arguments and returns strings.
- So on the master, in GAP, I convert my data to strings, issue a remote call to do-foo and then convert the results back.

Example Ctd

- The OM object passed to one of the slaves for a request would be something roughly like:

```
<omobj> <oma> <oms name="SCIENCE-rpc" />  
  
  <omstring>do-foo</omstring>  
  <omstring>Group((1,2,3))</omstring/>  
  <omstring>78</omstring/>  
</oma></omobj>
```

Other issues in Protocol X

- Resource Limits
 - we proposed yesterday that we need to support resource and other limits in Protocol X to allow for implementation of security decisions made in a proxy.
- Interrupts
 - it was agreed in Brussels that you need a way for the client to interrupt an ongoing computation on the server
 - This means that protocol X needs to support some out-of-band messages
 - This is a pain. If we had time limits, would we need interrupts?
- Well known function names
 - we might allow `<oms>` instead of `<omstring>` for the function name to allow for uniquely defined well-known functions

How does it look as a Web Service?

- Do we just have a single Web service?
 - A method called something like **SCIENCE-rpc** taking an OpenMath object (ie XML) and returning the same
 - perhaps another service for interrupt?
 - This has the advantage that service providers don't need to worry about WSDL or anything at that level
 - there is a common WSDL description of all our services
 - and the disadvantage that many tools that automatically make local interfaces to remote web services will not be able to see the actual service names or argument types
 - We can use a well-known service name to return details of the services offered on a server
 - but is this just reinventing WSDL etc.?

What do we need at OpenMath level defining the messages in protocol X

- Basic operations:
 - Basic procedure call
 - *Call and Store result*
 - *Recover stored value*
 - Results of Successful procedure call
 - Error report of unsuccessful procedure call
 - *possibly interrupt*
- Other things
 - *symbols for different kinds of resource limit annotations*
 - a symbol which says “the real data is in a non-OpenMath annotation”
 - allows MathML
 - *A symbol which references a stored value by cookie*
 - symbols for errors
 - maybe symbols for well-known procedures

What to Achieve this afternoon

- Some decisions:
 - do we need interrupts?
 - what do we expose to the WSDL layer?
 - will someone write the WSDL document
 - what new OpenMath symbols do we need
 - who will write the CD(s)?
- Simple applications to demonstrate initial connectivity
 - ideas