SAMP for GWS?

Mark Taylor (Bristol)

IVOA Interop,
Victoria
20 May 2010

$Id: gws.tex,v 1.4 2010/05/20 19:02:53 mbt Exp $

Introduction

Outline

- Purpose of this talk
- What is SAMP?
  - Overview
  - Applications perspective
  - Architectural components
- Is SAMP useful in a GWS context?

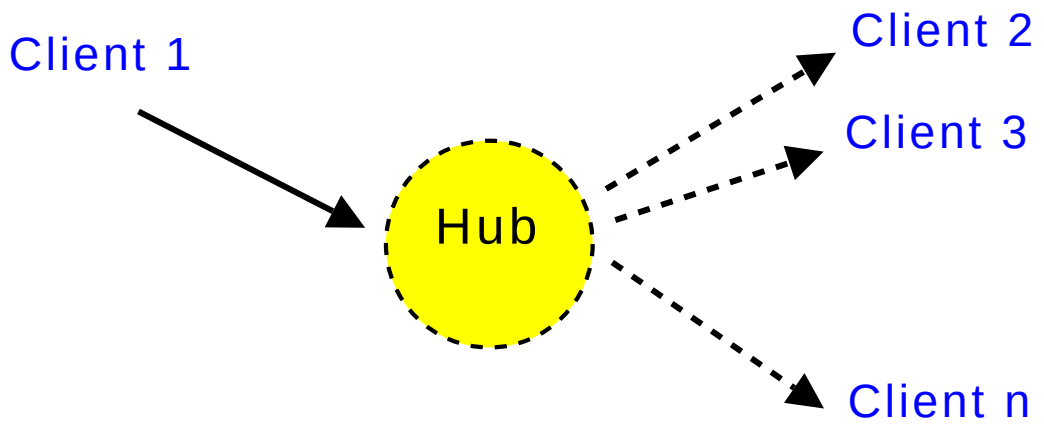
SAMP Overview

Simple Applications Messaging Protocol

- Built on earlier protocol PLASTIC
- SAMP 1.11 REC 2009-04-21
- Relatively mature; only minor changes under consideration for 1.2
- Libraries/Toolkits available in 5+ languages
- Used in 20+ applications, including most/all major VO & non-VO astro GUI tools

Hub-based, peer-to-peer messaging framework

- Hub is an independent daemon
- Clients may send and/or receive messages



SAMP Applications Perspective

SAMP is supposed to facilitate the following things:

- Communication between a (small) community of applications
- Loose coupling between applications
- Extensible message semantics
- Language-neutral communications
- Allow tools to concentrate on what they do best, using external tools for non-core functionality
- Good-enough messaging
- Does what you want, most of the time, without a lot of developer effort

Typical SAMP sequence:

- Image viewer and table viewer start up
- Table viewer sends a catalogue to the image viewer (`table.load.votable` MType)
- Image viewer plots markers over image accordingly
- User activity to select marker(s) in image view highlights corresponding row in table viewer and vice versa (`table.select.rowList` MType)

SAMP Architecture

SAMP provides:

- Basic RPC mechanism
  - Fundamentally asynchronous, with synchronous façade available
  - Options to receive or ignore return values
  - Options to direct to one, or broadcast to all, willing recipients
- Abstract definition of message passing syntax
  - Hierarchical message semantics labelling (MType)
  - Small string-based type system (string, list, map),
    oriented towards portability not expressiveness or efficiency
  - Named parameters and return values
  - Error reporting mechanism
- Centralized communications brokering
  - Centralised subscription management for publish/subscribe messaging
  - All messaging communication is client↔hub, not client↔client
  - Bulk data exchange may be client↔client (usually via exchanged URLs)
- Peer discovery
  - Polling or callbacks for changes in peer group
  - Framework for peer description

SAMP Architecture

The Way of SAMP:

- Language- and transport-neutral framework
  - Abstract protocol defined separately from transport layer ("Profile")
  - "Standard" XML-RPC-based Profile defined
  - Other profiles not precluded
- Pervasive use of extensible vocabularies
  - Well-known required or optional keys with defined semantics
  - Custom/private keys always permitted, ignored if unrecognised
- Relaxed attitude
  - Didn't work? Probably will next time
  - Can't do exactly what's requested? Do something similar
  - Implementor simplicity more important than exact predictability or 100% reliability

SAMP Architecture

- SAMP Restrictions:
  - Limited reliability
    - Delivery in send sequence not guaranteed
    - Delivery not guaranteed at all
    - Messages "crossing in the mail" tolerated
  - Minimal, closed data type system
    - Only string, list, map basic types defined for message and reply content
    - But extensible recommendations for string-encoding int, float, boolean etc
    - No explicit provision for binary data (though could base-64 encode)
  - Limited inline message length
    - Bulk data usually moved by exchanging URLs
- Standard Profile restrictions (*other profiles could work round these*):
  - Limited security
    - Client private-keys passed in plain text
  - Hub discovery requires access to local filesystem
    - samp-secret password in read-protected file `~/.samp` by default

SAMP and GWS?

SAMP features:

- Abstract definition of message passing syntax ☑
- Basic RPC mechanism ☑
- Language- and transport-neutral framework ☑
- Pervasive use of extensible vocabularies ☑
- Centralized communications brokering ☒
- Peer discovery ☒
- Relaxed attitude ?

SAMP restrictions:

- Limited reliability ?
- Basic data type system ?
- Limited inline message/response length ?
- Limited security ?

Conclusions

- My thoughts:
  - Some patterns, approaches, design from SAMP relevant to GWS
  - Would be possible to share some API, data type definitions etc
    - RPC model, Type system, MTypes, extensible vocabularies, XML-RPC, . . .
  - SAMP not suitable for direct use in web service context
    - Most significant issue is centralized peer-to-peer-via-hub model
  - Structural modification (complication) of SAMP to fit other paradigms is undesirable
- Your thoughts:
  - . . . ?