SAMP document overview

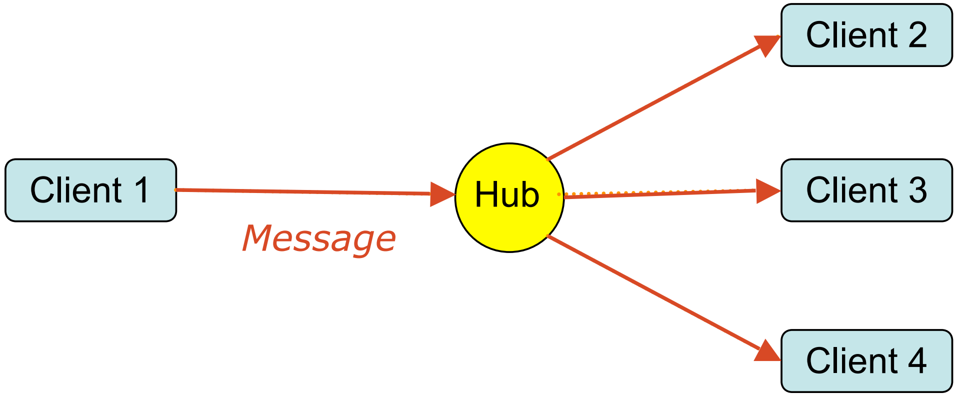
Thomas Boch [CDS]

Document outline

- Introduction
- Architectural Overview
- Abstract APIs and Data Types
- Standard Profile
- MTypes

Hub-based architecture

- Hub : broker service routing messages between clients



MType

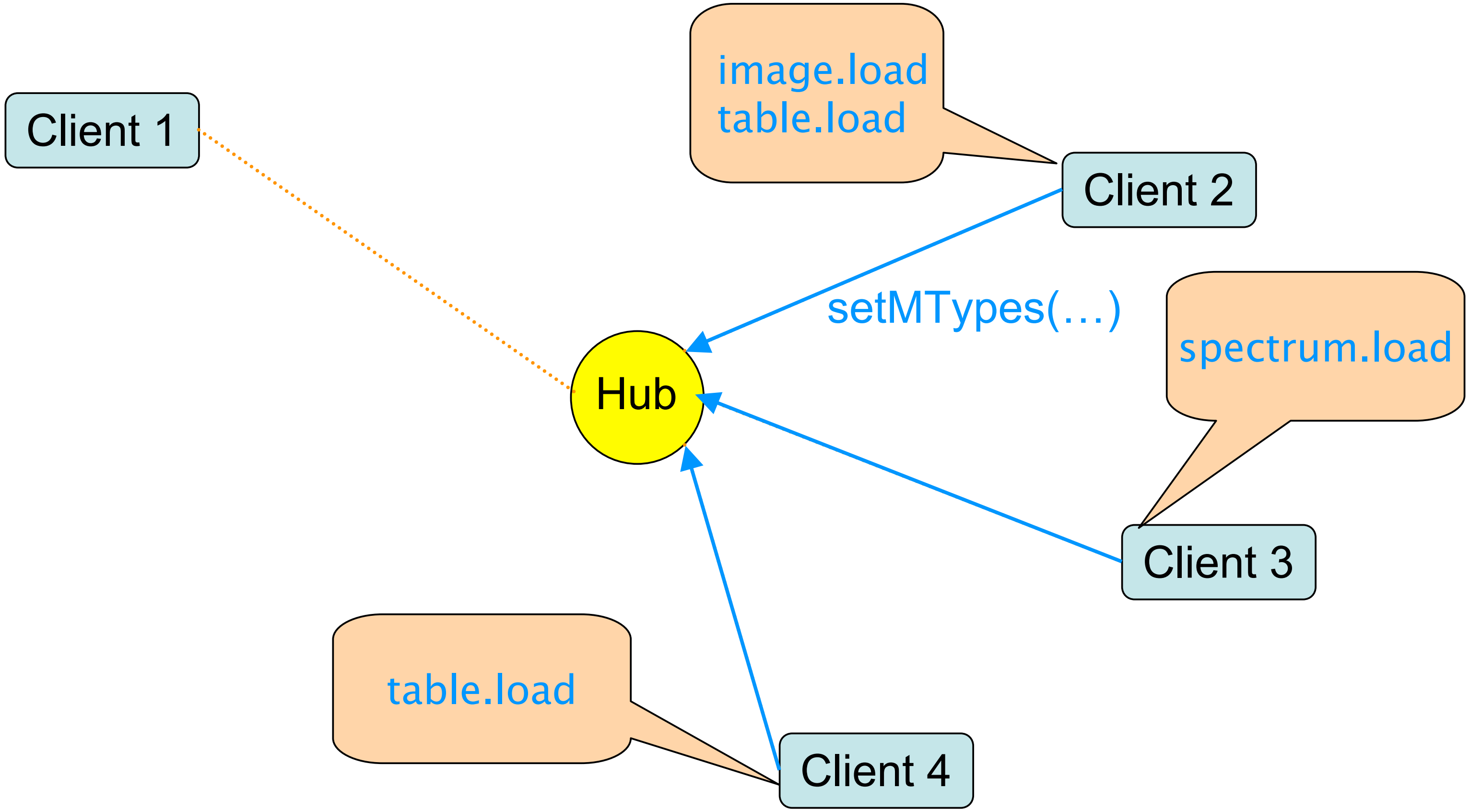
- Message = MType + parameters
- MType : defines the Semantics of a Message and of its arguments and return values
  - Eg: ‘display an image’ --> *image.load(imname)*
  - Loose definition : exact behaviour to a given Mtype is specific of each SAMP client

Message subscription (1/2)

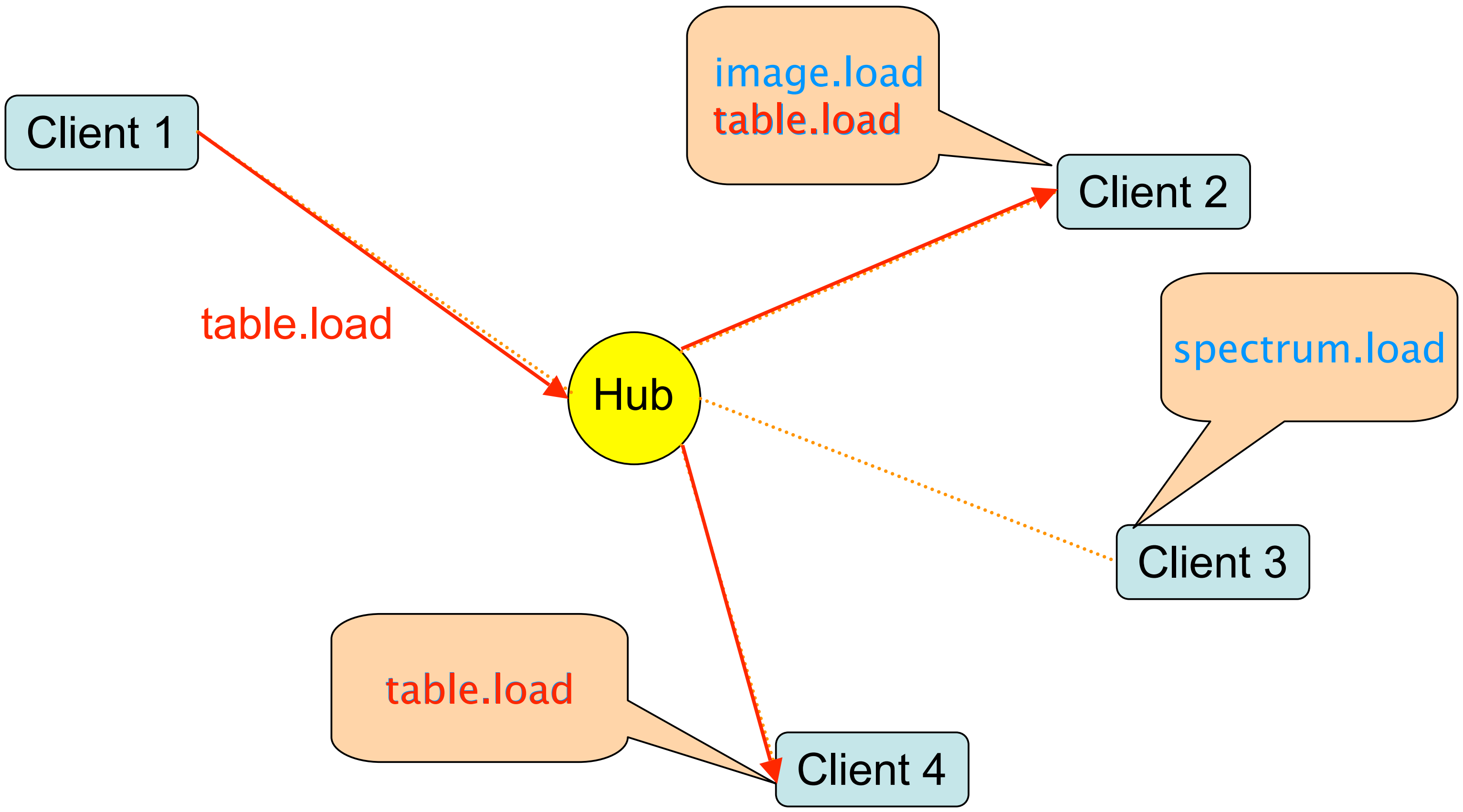
- Each SAMP client subscribes to the MTypes it wants to receive

- A Message is delivered only to the clients subscribed to the corresponding MType

Message subscription (2/2)

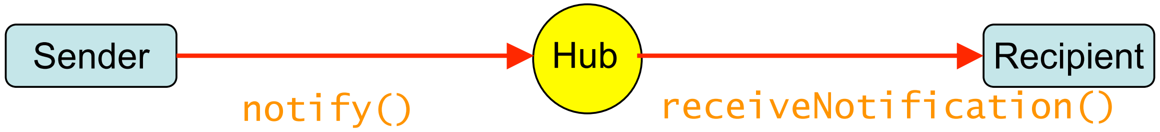


Message subscription (2/2)



Delivery Patterns

- Notification



Sender

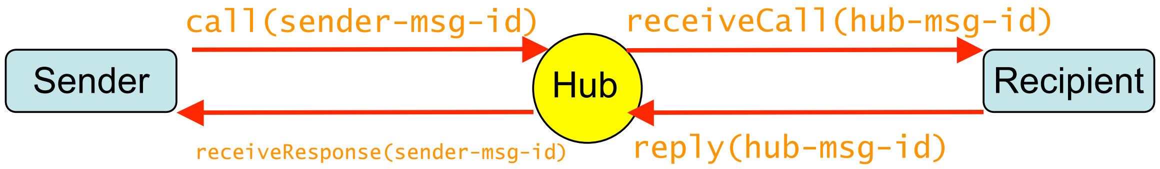
notify()

Hub

receiveNotification()

Recipient

- Asynchronous Call/Response



call(sender-msg-id)

receiveCall(hub-msg-id)

Sender

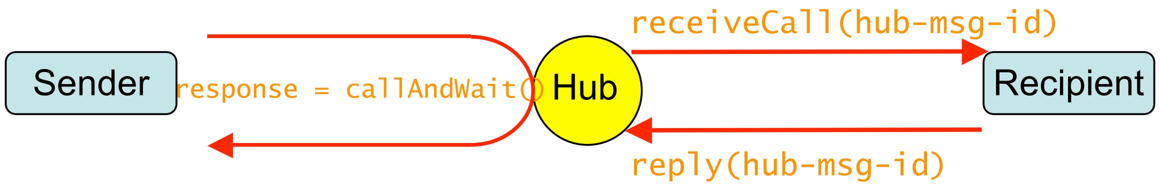
Hub

Recipient

receiveResponse(sender-msg-id)

reply(hub-msg-id)

- Synchronous Call/Response



receiveCall(hub-msg-id)

Sender

response = callAndWait()

Hub

Recipient

reply(hub-msg-id)

Abstract APIs

- Provides a high level description of the messaging protocol
  - Independent from the transport mechanism
  - Stable core, robust to changes
- Includes
  - Hub discovery mechanism
  - SAMP data types (string,list,map)
    + scalar type encoding convention (SAMP int, SAMP float, SAMP boolean)
  - Hub API
  - Client API

Hub API

- Operations that a hub must support
  - register(), unregister()
  - setMetadata(map metadata), getMetadata(...)
  - setMTypes(list mtypes), getMTypes(client-id)
  - getRegisteredClients()
  - getSubscribedClients(list mtypes)
  - notify(...), notifyAll(map message)
  - call(...), callAll(map message)
  - response = callAndWait(map message)
  - reply(...)

Client API

- Operations which may be called (by the hub) on a callable client
  - receiveNotification(…)
  - receiveCall(…)
  - receiveResponse(…)

Profiles

- A profile is a set of rules defining how abstract interfaces (APIs) are mapped to specific network operations
- Gives rooms for other Profiles if needed in future

Standard Profile

- Relies on XML-RPC transport layer
- Mapping rules from abstract APIs to Standard profile
  - Hub discovery : lockfile in a well-known location
  - Data type mappings
  - API mapping
    - Hub/Client methods prefixed with *samp.hub/samp.client*
    - New method setXmlrpcCallback() to inform the hub of the XML-RPC endpoint of the client
    - New method isAlive() to ping the hub, and checks whether it is responding. Callable by non-registered applications

SAMP client sample session

- Reading lock file
- Registering with the hub
- Setting metadata
- Sending a notification
- Setting MTypes
- Declaring XML-RPC callback
- Processing a message

SAMP lockfile

```
# SAMP lockfile written at 2008-05-
  16T22:26:23+0000
# Hub implementation by Alasdair Allan
  <alasdair@babilim.co.uk>
# Required keys:
samp.secret=HyP0flAQVZZxqmHla26W
samp.hub.xmlrpc.url=http://10.37.129.2:8001/
samp.profile.version=1.0
```

XML-RPC request for registering

```
<?xml version='1.0'?>
<methodCall>
  <methodName>samp.hub.register</methodName>
  <params>
    <param><value>
      <string>HyP0flAQVZZxqmHla26W</string>
    </value></param>
  </params>
</methodCall>
```

XML-RPC response from the hub

```
<?xml version="1.0">
<methodResponse>
  <params>
    <param><value>
      <string>app-id:p435Zregi5WhQjPB2j</string>
    </value></param>
  </params>
</methodResponse>
```