

The purpose of this presentation:

- To collate some of the key ideas emerging from the project
- To propose generalised models for the Integra environment
- To suggest some areas for further discussion

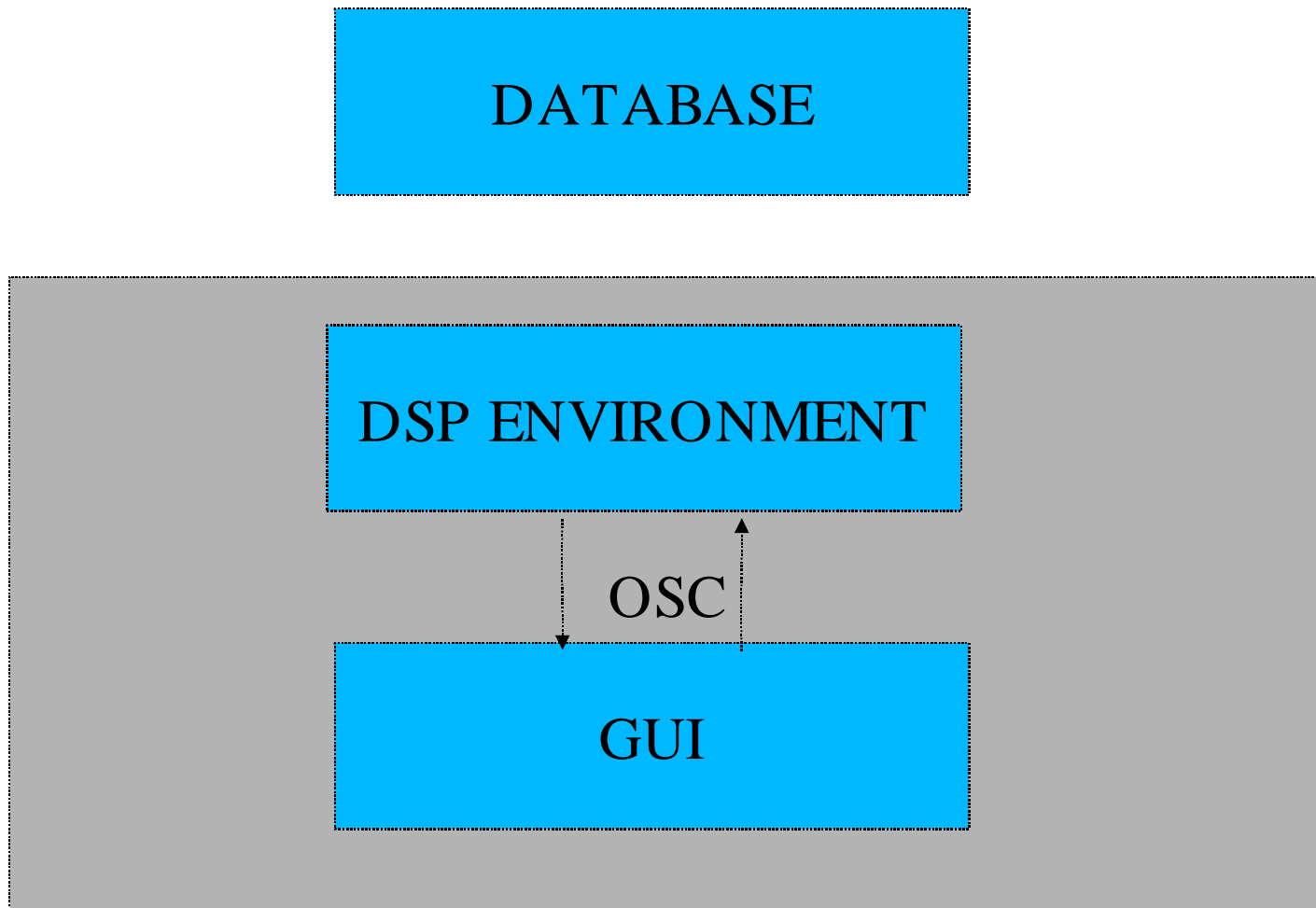
Emergent ideas

- The Integra environment should provide facilities for working with a range of existing DSP environments, in particular common Music-N derivatives
- The core of the Integra system should be an OSC namespace implementation that facilitates language-neutral communication between environments, and between modules within environments.

Emergent ideas

- The system should provide common environment-neutral GUI tools for the development and performance of musical works
- Documentation and performance data should be stored in a database, possibly in a format that reflects the class hierarchy implemented in the namespace

Simplified Model



Extending the model: Database

- Should we aim to be able to dynamically build patches from data stored in the database?
- Should performance parameters/patch data be stored in the same database as general documentation?
- Should the database be accessible from within the DSP environments?

Extending the model: Database

- How should we store performance parameters – (neutral, follow class model)?
- If we follow the namespace class hierarchy with the database representation, should we use an OORDMS e.g. ZODB, ozone, eXist?

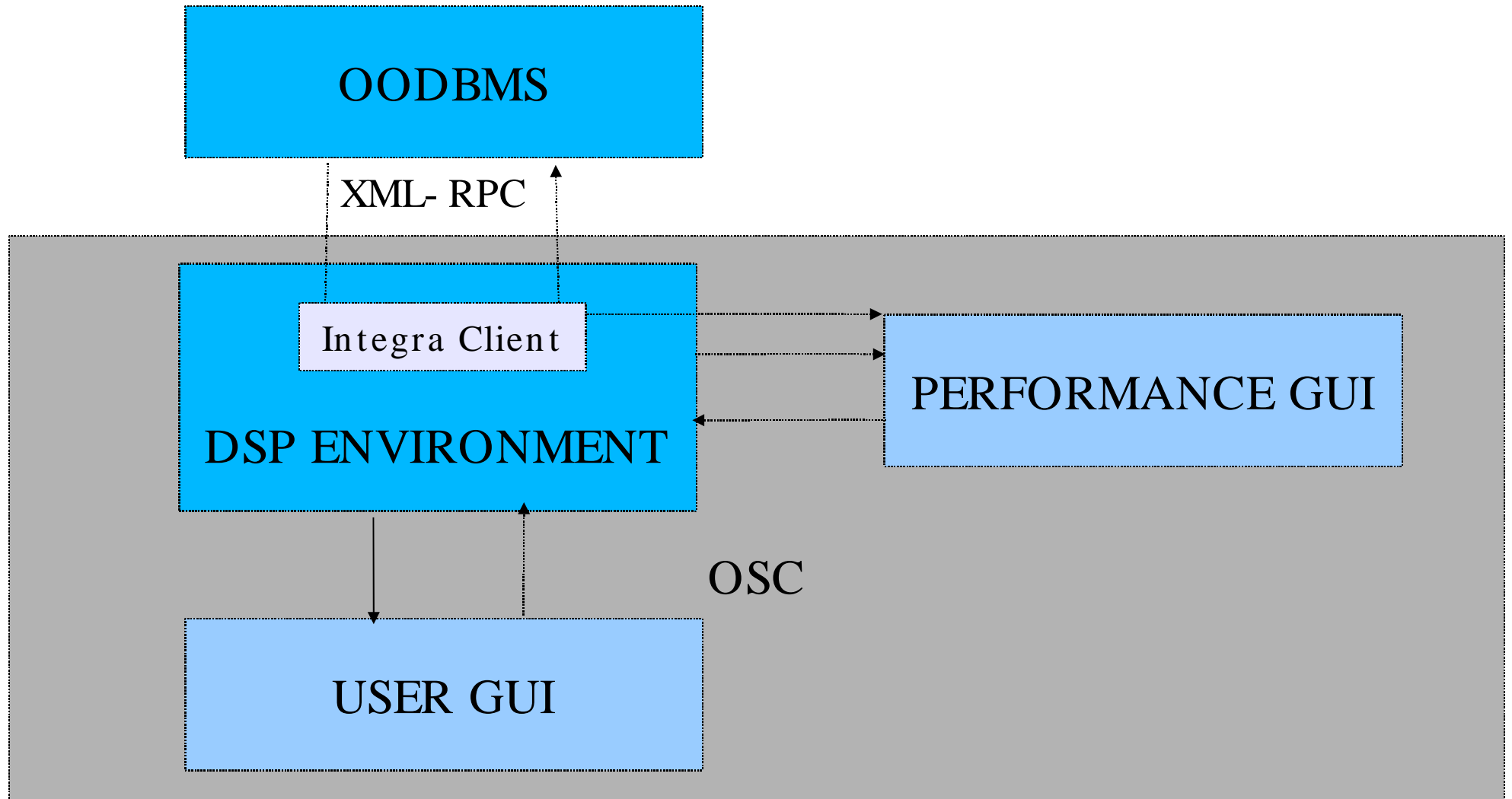
Extending the model: GUI

- Should the 'development GUI' and 'performance GUI' provide two separate front ends for a common set of tools?
- Could we build the 'performance GUI' as an executable file – unique to each work?

Extending the model: GUI

- Could we build the development GUI using tools provided by existing environments (e.g. Max/MSP), or should we use lower level tools (e.g. GTK)?
- Exactly what tools and functionality are we going to provide in the 'development GUI'?

Extended Model 1



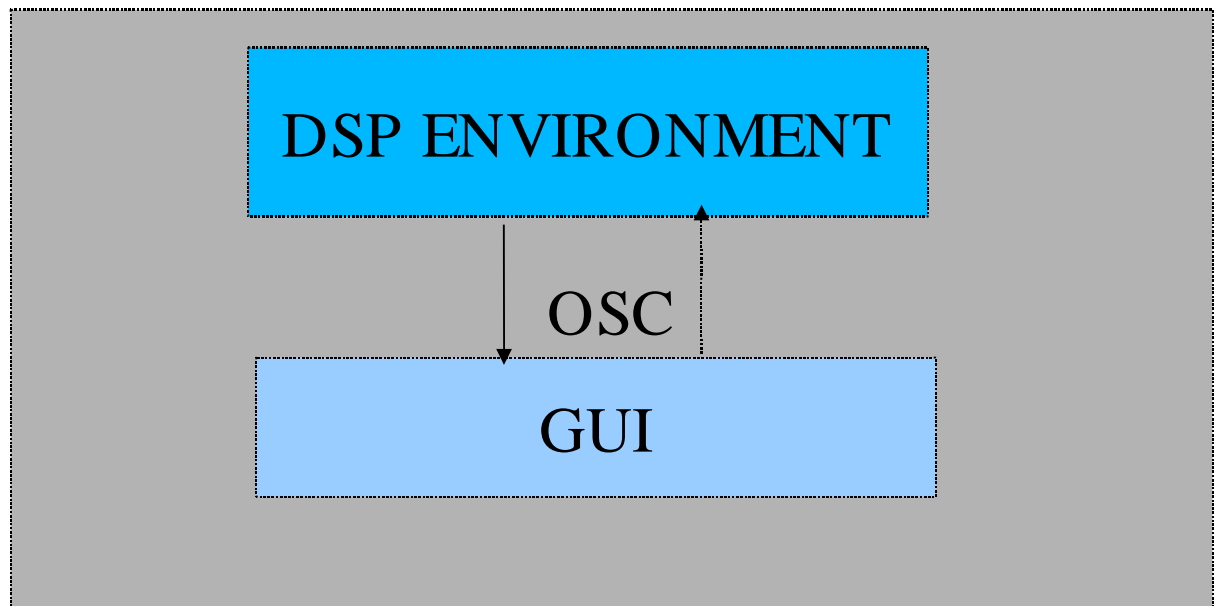
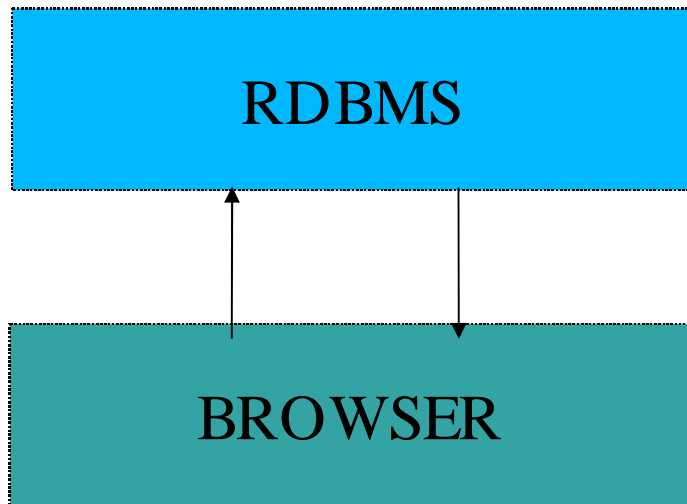
The extended model 1

- Performance and patch data is stored in a database
- An Integra client is loaded at runtime by the DSP environment, and is responsible for:
 - Obtaining patch data from the database
 - Building the patch
 - Building the GUI
 - Loading parameters
 - Writing patch data and parameters back to the database

The extended model 1

- The client should communicate with the database using a simple protocol that can work over http. XML-RPC?
- The client should exist as a library with wrappers for the different environments we want to support

Extended Model 2



The extended model 2

- Performance and patch data is stored in a database
- Patch data is downloaded before runtime
 - User can request an existing patch, or request a certain set of tools
 - GUI built on the server and downloaded?
 - User downloads a bundle/installer containing patch, datafiles, GUI?
 - User responsible for 'installing' this on environment workstation

The extended model 2

- User responsible for uploading data to the database via browser?
- Form-based mechanism for ensuring database consistency and forcing documentation details to be entered