

EQUIP: the Equator Universal Platform

Introduction

Chris Greenhalgh

1st April 2001

v.1.1. last updated 11th April 2001.

Overview

EQUIP is being developed within the Equator IRC as a proposed integration platform and middleware solution for the various activities and projects within Equator. The first iteration of EQUIP is being led by Chris Greenhalgh at the University of Nottingham, with contributions anticipated from a number of other partners.

EQUIP aims to:

- provide a common run-time infrastructure able to address multiple application and deployment domains, including (but not limited to) application integration, realtime 3D graphics, wireless networked systems and heterogeneous platforms;
- support easy interoperability between Java and C++, with cross-language and cross-platform type definitions (using a subset of CORBA IDL) and serialisation formats, and a number of services and facilities implemented natively in both languages;
- support easy extensibility, including run-time class loading in both Java and C++ (using Bamboo [Bamboo]).

In outline EQUIP may be viewed as a middleware platform, plus a number of (relatively) standard services and modular facilities or libraries.

EQUIP is organised as a number of modules (c.f. Java packages). These can be found in Equator/Modules/equip. The initial modules are:

- `equip_runtime` (`equip.runtime`): the core system runtime facilities, including memory management and common base classes.
- `equip_net` (`equip.net`): cross-platform networking classes, including support code for simple clients and servers, and a standard trader/nameservice.
- `equip_net_Trader`: the trader itself (initially C++ only).
- `equip_data` (`equip.data`): a common data sharing service supporting event, shared data item and tuplespace style operations
- `equip_data_Server`: a networked (client-server style) data server (initially C++ only).
- `equip_data_Browser`: a Java graphical browser for a data server; currently very partial implementation.
- `equip_data_Test`: a test module and example for the data service (good place to start looking). This is documented in the Equip Data service introduction.
- `equip_math` (`equip.math`): point, quaternion and matrix classes and standard operations (initially mainly 3D).
- `equip_daw3d` (`equip.draw3d`): abstract interface classes for renderable nodes in a simple scene graph (inspired in part by Maverik [Maverik]).
- `equip_draw3d_dgl` (`equip.draw3d.dgl`): implementation of `equip_draw3d` for the University of Nottingham's DGL graphics library [DGL] (only actually implemented in C++).

A number of other C/C++ systems and packages have also been ported to the Bamboo build/runtime environment so that they can be used more easily with EQUIP:

- `crg`: a subset of the University of Nottingham's CRGSHARE libraries (developed for MASSIVE-2), including the DGL renderer and supporting libraries.
- `massive3`: a subset of the University of Nottingham's MASSIVE-3 collaborative virtual environment system (this may be ported to use EQUIP in the future).

- vrjuggler: a port of the University of Iowa's VRJuggler [VRJuggler] toolkit, with particular support for multi-display immersive VR interfaces (e.g. CAVEs and the like).

Each of these typically comprises a number of related modules, reflecting the underlying organisation of the original system.

This set of documents aims to explain the structure and operation of EQUIP.

References

[Bamboo] <http://www.watsen.net/Bamboo>

[Maverik] <http://aig.cs.man.ac.uk/maverik/download.htm>

[VRJuggler] <http://www.vrjuggler.org/>