iLab Batch LabServer Architecture Overview

Draft - 7 Sep 2009 - LJP

The architecture consists of two parts:

1. **LabServer** – This contains the mechanisms for receiving an experiment specification, scheduling and running an experiment according to that specification. Communication with the LabServer is through a web service which implements the iLab ServiceBroker-to-LabServer API.

2. **LabEquipment** – This implements the control of the physical equipment or hardware for the experiment. This may or may not exist depending on the type of experiment.

ServiceBroker – This is not part of the LabServer architecture but needs to mentioned because a LabClient can only communicate with a LabServer via a ServiceBroker.

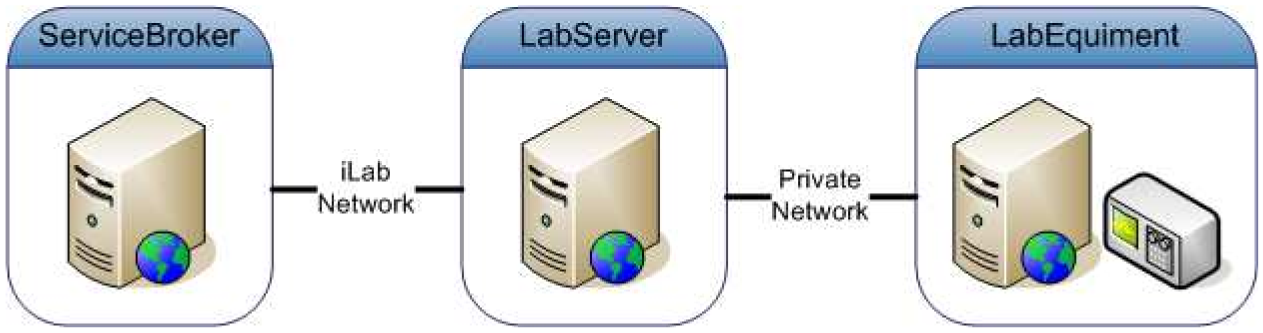

Figure 1. LabServer Architecture.

LabServer

The LabServer is a computer running a web service. The computer may be real or virtual. The operating system installed may be Microsoft Windows, Linux or Apple Macintosh (just to name a few). The LabServer source code may be written in any programming language that supports web services.

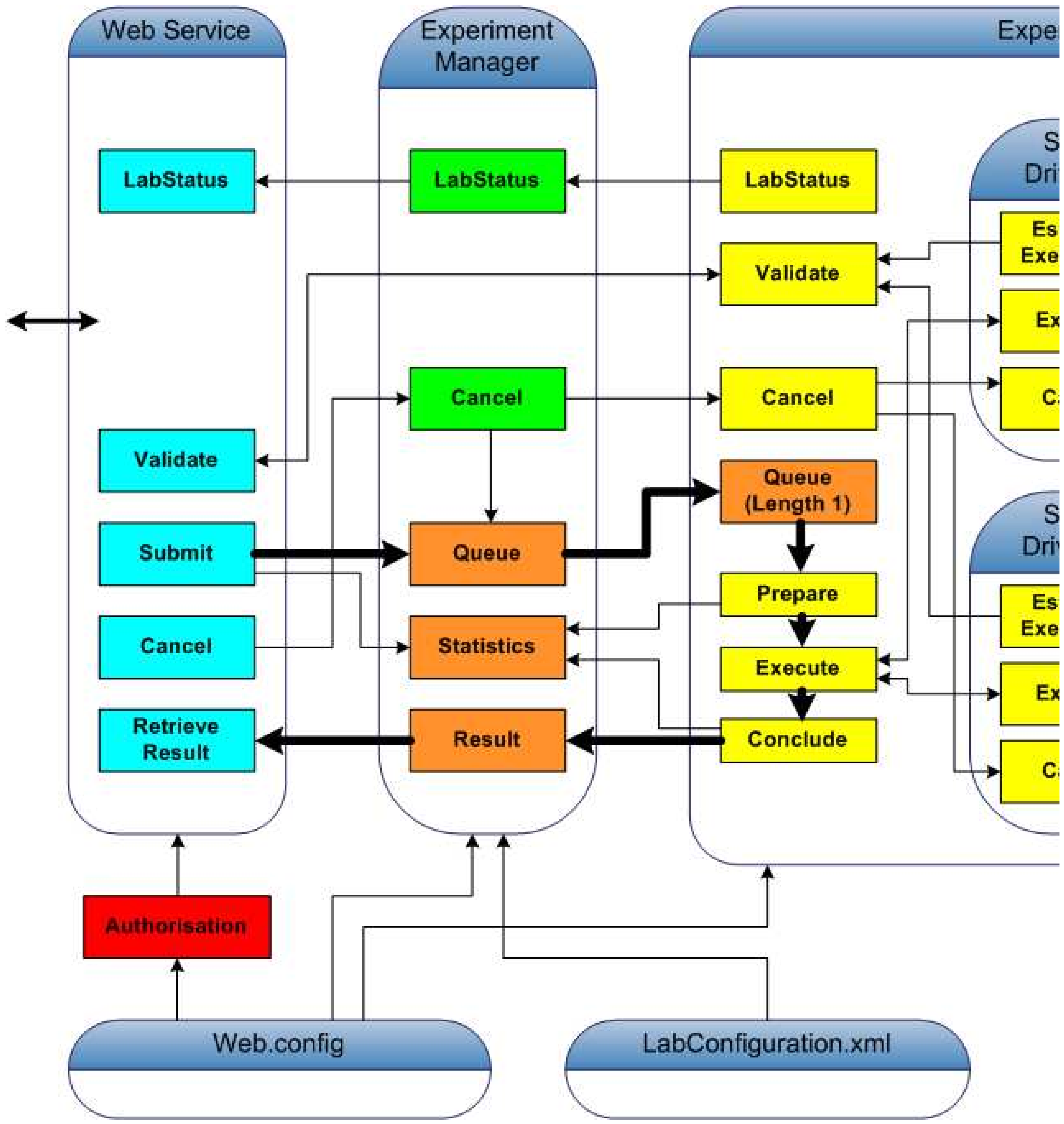

Figure 2. LabServer.

The LabServer consists of a number of parts:

1. **Web Service** - When a web service method is called by a ServiceBroker, an authorisation check is carried out. The SOAP header contains the ServiceBroker's GUID and a passkey which are checked against entries in the LabServer's configuration file. If the ServiceBroker has authorisation, the web service method passes the call onto the Experiment Manager for processing.

2. **Experiment Manager** – Responsible for processing web service method calls. It determines the configuration of the LabServer. It provides the status of the LabServer, validation results of an experiment specification, and takes a submitted experiment specification and places it on a queue ready for processing by an Experiment Engine.

3. **Experiment Engine** – Processes and runs an experiment specification. It validates an experiment specification by calling upon Setup Drivers, provides the status of the currently running experiment, and saves the experiment results for retrieval by a ServiceBroker.

4. **Setup Drivers** – Determine the way in which the experiment specification will be run on the Experiment Engine. They also provide an estimate of the time required to run the experiment. Typically, there are multiple setup drivers, e.g., measurements over time and measurements over distance. The setup drivers call upon Equipment Drivers to operate the equipment and to get an estimate of the time required to carry out those operations.

5. **Equipment Drivers** – Provide the interface to the LabEquipment. Operations required by the setup drivers are converted to commands by the equipment drivers and sent to the LabEquipment for processing. The commands may be either be sent to a LabEquipment's web service or by a TCP/IP connection to the equipment.

LabEquipment

The LabEquipment is a computer running a web service. It has experiment specific hardware attached and is connected to a private network. The operating system installed on the computer will depend on the hardware being used.

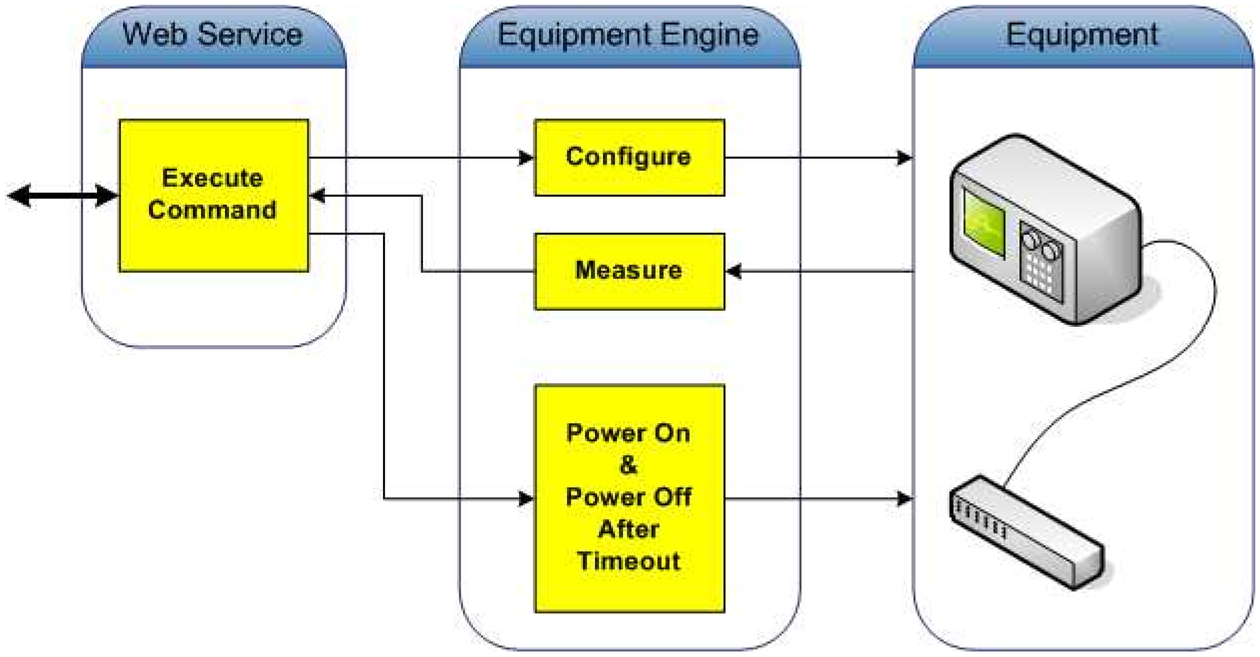

Figure 3. LabEquipment.

The LabEquipment consists of the following:

1. **Web Service** - Receives commands from the LabServer and processes them. The commands may control the equipment or request an estimate of the time required to control the equipment.

2. **Hardware Drivers** – These are drivers for manufacturer specific equipment such as an adapter card that plugs into the computer. Equipment could also be attached to the computer with a USB, serial or network connection.

3. **Power On/Off** – When the equipment needs to perform some function, it can be powered up and initialised. Then after a period of inactivity, the equipment can be powered back down again. This is optional, but helps to prevent the equipment from "wearing out" when it only gets used once or twice per day/week.

Experiment Farming

Suppose that there are multiple identical sets of equipment and they are operated in exactly the same way, it is not necessary to use seperate LabServers to drive each set of equipment. If multiple Experiment Engines were created with each servicing one set of equipment then the Experiment Manager could pass submitted experiment specifications to each Experiment Engine as they became available to run an experiment. Effectively, the Experiment Manager creates an array of Experiment Engines and shares the submitted experiment specifications amongst them. There will still be one Equipment Service for each Experiment Engine though.

Validation of an experiment specification still needs to be carried in some part by an experiment engine, but which one? Well, the first one because it will always be present.

Experiment farming has the effect of reducing the user wait times during peak periods and also allow the LabServer to continue operation without downtime should any one set of equipment fail.

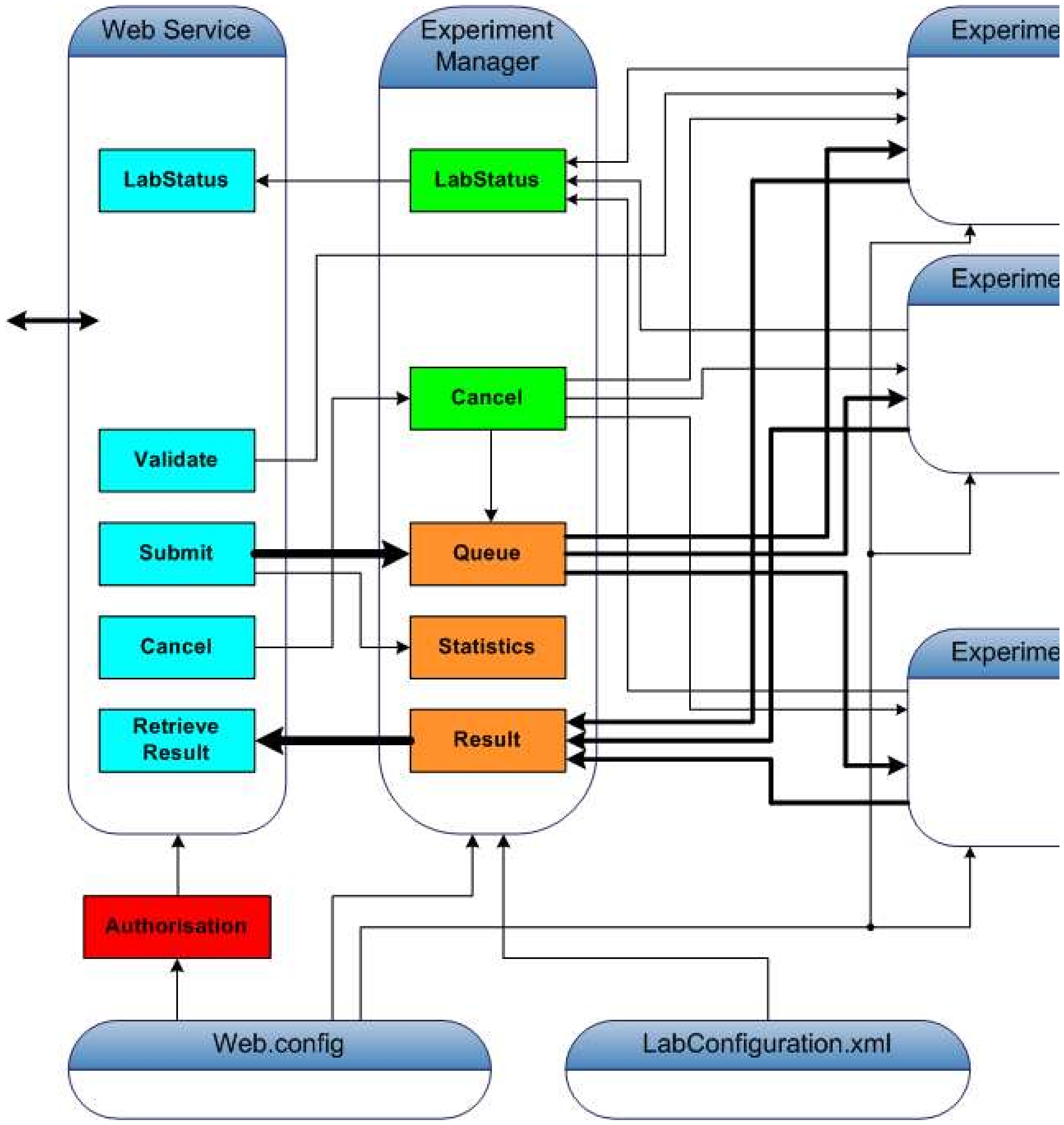

Figure 4. LabServer Farm.