Contribution ID: **54** Type: **Poster**

Workflow processing with gLite L&B: medical imaging case study

Monday, 11 April 2011 09:00 (8 hours)

Conclusions

We developed extensions to the L&B service which allow tracking execution of a generic workflow, described in terms of UML activity diagram/XMI document. The workflow can consist of nodes representing both jobs run on the grid and user actions. The implementation is fully neutral wrt. the actual engine responsible for execution of the workflow; according to the L&B design, L&B is responsible for tracking the execution, not its run, it is not yet another workflow engine.

The development was done on demand of the medical imaging group, fulfilling its immediate needs. However, we keep the design of the middleware layer to be reasonably generic and we plan to offer it to cover other similar requirements.

The implementation is being tested on NGI CZ resources.

Overview

The gLite Logging and Bookkeeping service (L&B) was designed to track user jobs on the grid. The tracking is decoupled from the services that are responsible for the actual job execution (WMS and CE in particular). Besides simple jobs L&B supports WMS DAGs (directed acyclic graphs) and job collections.

In this work we extend the set to more generic workflows which may contain, besides automatically executed jobs, also user actions. We demonstrate feasibility of the extensions on a specific usecase in the medical imaging application area.

Impact

The described development was driven by the demand of the group providing the recognised collection of pathology images http://atlases.muni.cz. The workflow starts with acquiring a gigapixel-sized raw image from a microscope. The image is split up into approx. 30,000 tiles which are processed individually (colour correction, sharping etc.) as parallel jobs. Then the tiles are composed back into a huge image. After this phase the user's expert assessment is required to provide further parameters of additional colour corrections, cropping, and geometric transformation. Those operations are applied on the huge image, and it can be repeated several times with different parameter settings eventually. Besides the intentional tuning some of the operations may fail and require

re-run with different parameters. Finally, the image is tiled again and scaled down for presentation at the web site.

The group runs the image processing workflows on NGI CZ resources. Recently they extended their equipment, they are able to produce significantly higher number of raw images so that their production starts overlapping with the processing, and the entire process becomes difficult to manage manually. This was the primary motivation, the L&B, which tracks the users jobs anyway, gets extended to become the user's experiment "logbook" replacing tedious and error prone manual work. In the current prototype the user is given a graphical tool which queries L&B and displays currently active workflows, highlighting those where user action is required. The actiond themselves are done with the original tools then. Even this loose integration improves the user's work significantly because of fully offloading the need to keep track of dozens of active workflows.

On the other hand, the extensions to L&B were designed in a reasonably generic way so that those should be reusable in other application scenarios as well.

Description of the work

First, the L&B is extended to store a template of the user's workflow. The template is a UML activity diagram, expressed in terms of XMI document, having uniquely identified nodes (executed actions) and edges (dependencies).

When a workflow instance is ready to be run, it is registered in L&B, bound to a specific template there, and assigned a unique job (workflow) identifier.

Workflow nodes are instantiated by individual jobs which declare themselves that they belong to a particular workflow id, and they implement its specific node. Technically this is done by a new L&B event which is logged upon the job submission. The necessary information, id of the workflow node in particular, is extracted from the job description (Torque submit file or gLite JDL) where it must be provided by the user. For actions taken by the user, the information is provided explicitly, typically in a single call to an L&B API wrapper which tells L&B "I've done this".

On the other hand, queries to L&B are used to find out which actual jobs represent nodes of a workflow instance and what is their state (not started yet, running, done, essentially). Moreover, the L&B state computation logic is extended to understand the workflow definition and to be able to identify "pending" workflow nodes, i.e. their prerequisites are satisfied but the nodes have not been instantiated yet.

Finally, through the standard L&B user tags, both the whole workflow and its nodes can be annotated, and the annotations used to narrow down further queries.

By design, L&B does not replace a workflow engine, which is left with the responsibility to start runnable jobs, deal with eventual job failures etc. However, similarly to gLite WMS, the workflow engine can rely on the information retrieved from L&B. In the usecase described bellow trivial job dependencies are handled by the Torque batch system, while the user-assisted operations are based on L&B information fully.

Primary authors: KŘENEK, Aleš (CESNET); VESELÝ, David (Masaryk University); HEJTMÁNEK, Lukáš (Masaryk University)

Presenter: KŘENEK, Aleš (CESNET)

Session Classification: Posters

Track Classification: Poster