Basic Execution Management Services (BES): Roadmap

Karl Czajkowski, Ian Foster and Steve Tuecke for the Globus Alliance

karlcz@univa.com, foster@mcs.anl.gov, tuecke@univa.com

Draft of 9 March 2005

This document is a *rough* roadmap for the development of BES standards and complementary services. This roadmap proposes a staged set of activities that can:

(a) define initial standards and services early, so as to get early value from basic execution services—while producing durable specifications that can persist into the future; and

(b) address the more difficult long-term issues required for robust, scalable, interoperable EMS systems, without predicating deployment on having all the difficult problems solved.

Overview

Please see the accompanying "BES Approach" document which summarizes a number of important issues to be considered for BES.

We point out that the primary issues involved in BES are (in order of decreasing difficulty): resource modeling, job description, job submission and management. Conversely, these issues are probably best addressed in the reverse order to get early functionality deployed.

Modeling of site capabilities is probably the hardest issue. In order to support automation of execution management in a Grid, we need to address the issue that different schedulers/sites offer different capabilities. These differences are due to a combination of local resource capability, local management systems, and local policy. We cannot dictate these differences away, but rather need to describe them so that clients/brokers (with the help of discovery systems) can differentiate providers and route requests appropriately.

Stage 1

A reasonable BES specification could combine WS-Agreement[1] with JSDL[2] for the term language, resulting in a system to create Agreements that represent an obligation to (a) execute one Actor and (b) provide a management interface to control it. Thus, we obtain:

- Actor instance description: simple JSDL
- Actor creation and management: WS-Agreement

[1] http://www.gridforum.org/Meetings/GGF11/Documents/draft-ggf-graap-agreement.pdf

[2] http://www.epcc.ed.ac.uk/~ali/WORK/GGF/JSDL-WG/

- Modeling: only basic WS-Agreement template advertisement

Issue: What job-control features are needed in stage 1?

- WS-Agreement provides some basic annotations about lifecycle.
- Extended BES portType can easily add more, using WSRF model with more WS-ResourceProperties.
- GT 4.0 GRAM[3] has a “release hold” operation that allows the client to “unpause” the job lifecycle after it has paused due to a "hold request" field in the initial job description. This is used primarily to synchronize effects on the resource with other external activities.
-

Issue: What, if any, data staging and credential delegation features are needed in Stage 1?

  - Issue: are the JSDL terms for staging desirable, or should Container-specific mechanisms be composed through extension?
  - Do we want to support credential delegation, or is that best left as a composition with proprietary services and creation request extensions?

Stage2 and Later

There are a number of areas where more discussion is needed to even make a preliminary evaluation of problem depth. These areas might be addressed in one or more additional stages after the Stage 1 work, or they may be pie-in-the-sky problems that may not be addressed for years.

- **Richer negotiation**: WS-Agreement scoped away more complex interactions to allow client and provider to “find” a solution that they can accept. WS-Agreement views these as possibly being replacements for the AgreementFactory creation or possibly as a precursor that is finalized with an AgreementFactory creation exchange

- **Richer job description languages**: Piecemeal extensions may be required to support some kinds of QoS constraints, etc. In addition, some significantly different syntax may be needed to capture advanced features of some scheduler systems such as job arrays, inter-job dependencies, or multi-job workflows.

- **Richer provider capability modeling**: Separate activities might yield better Container Annotations, or advertisements, that would help with discovery of Containers capable of supporting the complex BES scenarios of particular clients. It is not clear whether one generic solution would address all scenarios, or whether some communities would use different extensions than others.

- **Delegation**: A difficult part of credential provisioning for Actors is being able to discover local environment requirements. This is related to the other

[3] http://www-unix.globus.org/toolkit/docs/development/4.0-drafts/execution/wsgram/WS_GRAM_Public_Interfaces.html

discovery/modeling problems and it intersects with environment-specific semantics. For example, a job that wants to access certain files may need to obtain additional rights from the submitting user, or an application may require license allocation etc.

- **WSDM**: How should BES be integrated with other management processes? Is it more important to allow management *of* BES services through other systems like WSDM[4] or management of BES Actors through those systems?

[4] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsdm