GWD-R.xx
SAGA-WG
Version: 0.1 June 28, 2011

SAGA Resource Management API

Status of This Document

This document provides information to the grid community, proposing a standard for an extension to the Simple API for Grid Applications (SAGA). As such it depends upon the SAGA Core API Specification [1]. This document is supposed to be used as input to the definition of language specific bindings for this API extension, and as reference for implementors of these language bindings. Distribution of this document is unlimited.

Copyright Notice

Copyright © Open Grid Forum (2007). All Rights Reserved.

Abstract

This document specifies a Resource Management API extension package for the Simple API for Grid Applications (SAGA), a high level, application-oriented API for distributed application development. This Resource Management (RM) API is motivated by a number of use cases collected by OGF's SAGA Research Group in GFD.70 [2], and by requirements derived from these use cases, as documented in GFD.71 [3]). Also, the SAGA community has been receiving additional new use cases, in particular related to virtualized resources.

The RM API extensaion allows to interface to resource discovery systems, to instantiate resources on the fly, and to perform reservation upon discovered or created resources. The resulting resources and resource reservations can be consumed by the SAGA Core job API package, to be used for job instantiation.

Contents

1 Introduction **3**

1.1 Notational Conventions 3

1.2 Security Considerations 3

2 SAGA Resource API **4**

2.1 Introduction 4

2.2 Specification 4

2.3 Specification Details 7

3 Intellectual Property Issues **8**

3.1 Contributors 8

3.2 Intellectual Property Statement 8

3.3 Disclaimer 8

3.4 Full Copyright Notice 8

References **10**

1 Introduction

A significant number of SAGA use cases [2] ask for the possibility...

1.1 Notational Conventions

In structure, notation and conventions, this documents follows those of the SAGA Core API specification [1], unless noted otherwise.

1.2 Security Considerations

As the SAGA API is to be implemented on different types of distributed middleware systems, it does not specify a single security model, but rather provides hooks to interface to various security models – see the documentation of the `saga::context` class in the SAGA Core API specification [1] for details.

A SAGA implementation is considered secure if and only if it fully supports (i.e. implements) the security models of the middleware layers it builds upon, and neither provides any (intentional or unintentional) means to by-pass these security models, nor weakens these security models' policies in any way.

2 SAGA Resource API

2.1 Introduction

For dynamic provisioning scenarios, `saga::job::service` needs state, and some new methods. To get that, the `saga::job::service` gets extended to `saga::job::manager`. A new class `saga::job::resource` is a factory for `manager` instances, and can produce `manager` instances for different use cases (IaaS, Grid, advanced res, pilot job).

2.1.1 Classes

The SAGA Job-Resource API consists of three classes: a `manager` class, which represents an entity which provides (i.e. finds, creates, destroys) `job_service` instances – such `job_service` instances represent the second class of this API. Multiple managers can be combined into a manager `job_service_pool`, which extends the `job_service` class by some pool management methods, but otherwise behaves equivalently.

`job_service` instances are created according to a respective resource description, which defines the properties of the resources a job `job_service` instance is interfacing to. To cater to the wide range of use cases this API targets, several different description types are available: a plain `resource_description`, a `iaas_description`, a `reservation_description`, and a `pilotjob_description`.

2.2 Specification

```
package saga.job
// extension of the existing saga::job package
// FIXME: not sure about name spacing / versioning
package resource
{
  class  resource_description : implements saga::attributes
  {
    // attribs need to be as uniform as possible over the different
    // types.  One could possibly introduce a base type with a seto
    // of generic attributes, and derive specific description types
    // from it.  that makes the API somewhat more complex, but also
    // somewhat easier^H^H^H^H^H^H cleaner to expand if needed.  For
    // example, one could introduce a completely free-form resource
```

```
// description...

type : simple  (see saga-core v1.1, saga-sd v1.0)
         url:          old fashioned resource manager url
         glue:         glue sql query, first match or pool is returned

       reservation (see DRMAA v2.0)
         string        name (any, all, wildcard)
         time          start_time
         time          end_time
         time          duration
         int           slots
         array<string> users
         array<string> candidate_machines
         long          phys_memory
         enum          machine_os;
         enum          machine_arch;

       pilot_job (see BigJob v2.0)
         string        type = advert, diane, ...
         string        executable
         array<string> args
         array<string> env
         string        queue
         time          start_time
         time          end_time
         time          duration
         int           slots
         array<string> candidate_machines
         long          phys_memory
         enum          machine_os
         enum          machine_arch

       iaas (see OCCI v1.0)
         // IaaS:
         string       vm_id        // id of VM image to be used
         ...???
         // compute:
         enum         architecture // CPU Architecture of instance.
         int          cores        // #cores assigned to the instance.
         string       hostname     // FQHN for the instance.
         int          speed        // CPU Clock in gigahertz.
         int          memory       // Min RAM in gigabytes for instance.
         enum         state        // active, inactive, suspended
         // network:
         int          vlan
```

```
            string       label
            enum         state
            string       address
            string       gateway
            enum         allocation
            // storage:
            int          size
            enum         state
  }

  enum state
  {
    Unknown = 0,
    Pending = 1,  // will become active eventually
    Active  = 2,  // accepting jobs
    Closed  = 3,  // closed by user
    Expired = 4,  // closed by system
    Failure = 5   // disappeared etc.
  };


  class  manager : implements saga::object,
                   implements saga::task::async
  {
    CONSTRUCTOR  (in    session          session,
                  in    string           url = 0,
                  out   manager          obj);
    DESTRUCTOR   (in    manager          obj);

    // return job service for a specific resource
    request      (in    res_description  rd,
                  out   job_service      m);

    // return a job service for all matching resources
    request_pool (in    res_description  rd,
                  out   job_service_pool p);
  }


  class job_service : implements saga::job::service
  {
    get_state       (out state           s);
    close           (in  bool            drain); // drain before close?
    submit          (in  string          jsdl);
    get_description (out rsource_des     d);
    get_manager     (out manager         m);
```

```
    // get a new reservation for the job service's resource:
    // new_job_service = job_service.get_manager ().request (reservation);
  }


  class job_service_pool : implements saga::resource::job_service
  {
    add_job_service    (in  job_service         js);
    remove_job_service (in  job_service         js);
    list_job_services  (out array<job_service>  js);

    // set scheduler policy, such as
    // default, round_robin, random, load, ...
    set_scheduler      (in  string              s="");
  }

  // FIXME:
  //  - There should be a way to bind a data item with a task in a
  //    pool.  Granularity?  How to specify data items?
}
```

2.3 Specification Details

3 Intellectual Property Issues

3.1 Contributors

This document is the result of the joint efforts of several contributors. The authors listed here and on the title page are those committed to taking permanent stewardship for this document. They can be contacted in the future for inquiries about this document.

3.2 Intellectual Property Statement

The OGF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the OGF Secretariat.

The OGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this recommendation. Please address the information to the OGF Executive Director.

3.3 Disclaimer

This document and the information contained herein is provided on an "As Is" basis and the OGF disclaims all warranties, express or implied, including but not limited to any warranty that the use of the information herein will not infringe any rights or any implied warranties of merchantability or fitness for a particular purpose.

3.4 Full Copyright Notice

Copyright (C) Open Grid Forum (2007). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its

implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the OGF or other organizations, except as needed for the purpose of developing Grid Recommendations in which case the procedures for copyrights defined in the OGF Document process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the OGF or its successors or assignees.

References

[1] T. Goodale, S. Jha, H. Kaiser, T. Kielmann, P. Kleijer, A. Merzky, J. Shalf, and C. Smith. A Simple API for Grid Applications (SAGA). Grid Forum Document GFD.xx, 2007. Global Grid Forum.

[2] A. Merzky and S. Jha. A Collection of Use Cases for a Simple API for Grid Applications. Grid Forum Document GFD.70, 2006. Global Grid Forum.

[3] A. Merzky and S. Jha. A Requirements Analysis for a Simple API for Grid Applications. Grid Forum Document GFD.71, 2006. Global Grid Forum.