Brokering Service for Supporting Problem-Oriented Grid Environments

Anastasia Shamakina

Supercomputer Simulation Laboratory,
76, Lenin Ave, South Ural State University, 454080 Chelyabinsk, Russia
E-mail: {sham2004}@bk.ru

Nowadays many planners in Grid environment support scheduling are based on a workflow. However, none of currently existing tools uses additional information concerning specifics of a problem area and the representation of a workflow. This paper describes a scheduling algorithm and architecture of a CAEBeans resources broker. It considers the above aspects as well as utilises resource reservation, thus managing hardware, software and licenses. This broker can be used for an effective search of resources in problem-oriented grid environments. The CAEBeans Broker is implemented in Java as a UNICORE service. This approach involves component independence from a computing platform and provides full information about a current state of a service, and supports secure and reliable performance, lifetime management, dispatch change notifications, management policy of access to the resources and access control certificates.

1 Introduction

Grid computing[1] intends to solve many tasks in various fields, e.g. medicine, engineering design, nanotechnology, climate prediction and others. In the present paper we consider a CAEBeans system[2] that provides access to CAE software (Computer Aided Engineering[3]). The CAEBeans system allows to carry out task decomposition, search for demand resources; submit tasks; monitor tasks execution; and provide results mapping for the user.

In common, any CAE-task solution includes the following technological steps: geometry creation, computing grid generation, definition of boundary conditions, simulation execution, visualization and solution analysis. The CAE-tasks have some peculiarities that should consider characteristics of resources, a set of engineering packages, the number of licenses available and so on.

Many grid environments support complex applications with a workflow[4]. They are usually modelled by a directed acyclic graph (DAG). Such tools as Condor DAGMan[5], CoG[6], Pegasus[7], GridFlow[8] and ASKALON[9] can be listed. Some additional information about problem area specifics and workflow representation can significantly improve the efficiency of planning resources methods. However, none of the currently existing tools consider this peculiarity.

The purpose of this work is to create methods and algorithms for scheduling a workflow, as well as the development of a resource broker based on it. The broker is applied to search for optimum resources in problem-oriented grid environments.

The article is structured as follows. Section 2 contains the main concepts of problem-oriented environments; section 3 describes some use cases regarding the resource broker; section 4 provides the architecture of the resources broker; section 5 considers the process of resource allocation in the CAEBeans Broker component; and section 6 gives a scheduling algorithm. The main results are summarised in the concluding part of this paper.

2 Basic Concepts of Problem-Oriented Environments

Here, we give some definitions of the basic concepts that are necessary to determine a problem-oriented environment.

Task describes the process the transformation of input parameters into output parameters.

Job is a set of Tasks organised as a workflow aimed to achieve a useful result. Job determines the order of tasks execution, conditions under which this or that task will be started, the mutual synchronisation of tasks and information flows between tasks.

Resource is hardware, software and licenses required to perform a task.

Service is a specification of resources to solve a specific class of Tasks. Service defines the format of the input and output data.

Abstract workflow is a set of a Job and requirements for resources.

Activity is an allocation of necessary resources and launch of a specific service with respect to specific recourses for executing a specific Task.

Concrete workflow is a workflow of activities aimed at implementation of a specific Job.

Distributed Problem-Oriented Environment is a set of resources, services, software and middleware focused on the implementation of Concrete workflows for a specific problem domain.

3 The CAEBeans Broker of Resources

The CAEBeans Broker is a component of the CAEBeans system that receives a job from clients, accords resource requirements and finds the most suitable computing nodes for each task from a workflow. We can distinguish the following main tasks of a broker: processing a resource database and analysing resource requests from a CAEBean Server[10]; gathering and providing information about the current state of a problem-oriented environment.

The CAEBeans Broker is implemented in Java as a UNICORE service. This approach involves component independence from a computing platform; provides full information about the current state of the service, and supports secure and reliable performance, lifetime management, dispatch change notifications, management policy of access to resources and access control certificates.

Software component CAEBeans Broker is a service that provides an interface for selecting the most appropriate resources. The search of required resources proceeds for multiple jobs once a resource broker works. In addition, an auxiliary component for data collection interacts with the resource broker. A use case diagram for the CAEBeans Broker is shown in Figure 1.

In the use case “Allocate Resources” sends a request from a client to the resource broker. The CAEBeans Server works as a client and responds to the execution of tasks and their monitoring. The use case begins when a CAEBeans Server specifies requirements for resources which are necessary for the job. The use case “Allocate Resources” includes that of “Set Reservation”, which allows one to set the reservation of selected resources.

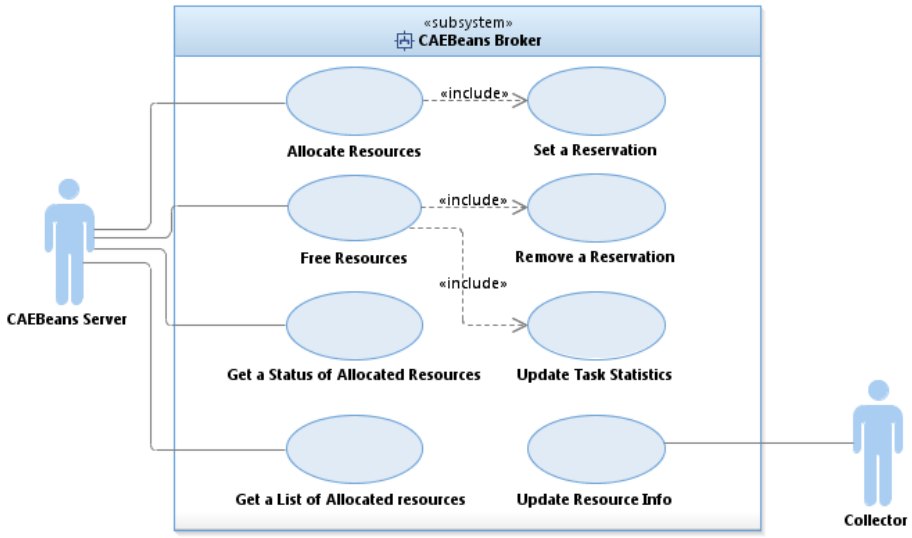

Figure 1. A use case diagram of a CAEBeans Broker subsystem

The use case "Free Resources" notifies the resource broker of a necessity for releasing some chosen resources. This use case also assumes the removal of a resources reservation request[11]. A corresponding removal request is formed by the use case "Remove Reservation".

The use case "Get a Status of Allocated Resources" asks for information about resources allocation. When resources allocation succeeds, a CAEBeans Server asks for a list of allocated resources using the case "Get a List of Allocated Resources".

Use cases "Update Resource Info" and "Update Task Statistics" update the information about characteristics of resources and statistics of tasks performance in the database of the resource broker.

4 The Architecture of the CAEBeans Broker

According to the use case diagram shown in the Figure 1, the following architecture of the resource broker CAEBeans Broker was designed (Figure 2). The CAEBeans Broker consists of the following components:

- The Master accepts requests from a CAEBeans Server and creates an instance of a WSResource that is called a Brokered Workflow.
- Each Brokered Workflow processes a request. It generates a list of required resources to perform a workflow and make a reservation.
- The Component Resource Manager manages the Resource Database that contains information about target systems and reserved resources.

- The Component Statistics Manager manages the Statistics Database that contains information about tasks statistics.
- The Collector works independently from the CAEBeans Broker and carries out the collection of information for the Resource Database.

In the Figure 2 the components of the UNICORE platform that intend to cooperate with the CAEBeans system are marked with red colour.

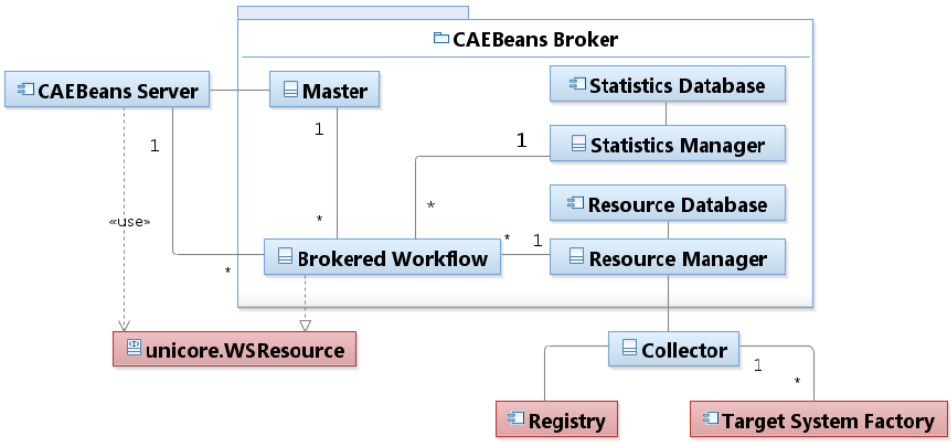

Figure 2. The architecture of the CAEBeans Broker

5 Resource Allocation of the CAEBeans Broker Component

According to the use case diagram shown in the Figure 1 the resource broker provides some methods for resource allocation and information gathering.

Consider how a CAEBeans Broker service ensures the process of finding necessary resources for the job (Figure 3).

A CAEBeans Server component provides access to the main service Master. Then the CAEBeans Server calls for the "Allocate resources" method for workflow resource allocation, as well as passes an abstract workflow as an input parameter. The abstract workflow contains requirements for resources.

The Master creates an instance of a Brokered Workflow, representing a WSResource in the terms of UNICORE. The Master initialises a concrete workflow, obtaining necessary information from the abstract workflow. It writes down the concrete workflow by the Ehcache framework in a cache at first and then on a disk. The Master returns the unique identifier of the created instance to the CAEBeans Server.

The Brokered Workflow gets a run-time estimate for each task and starts a search for demanded resources in the Resource Database. If the search for the Brokered Workflow succeeds, a list of resources is received. The selected resources are reserved.

The CAEBeans Server checks the status of resource allocation by directly addressing the instance of the Brokered Workflow. In case of a successful resource allocation, the CAEBeans Server requests for a list of the allocated resources for its workflow. After the workflow execution the CAEBeans Server sends a request for releasing the resources to the Master.

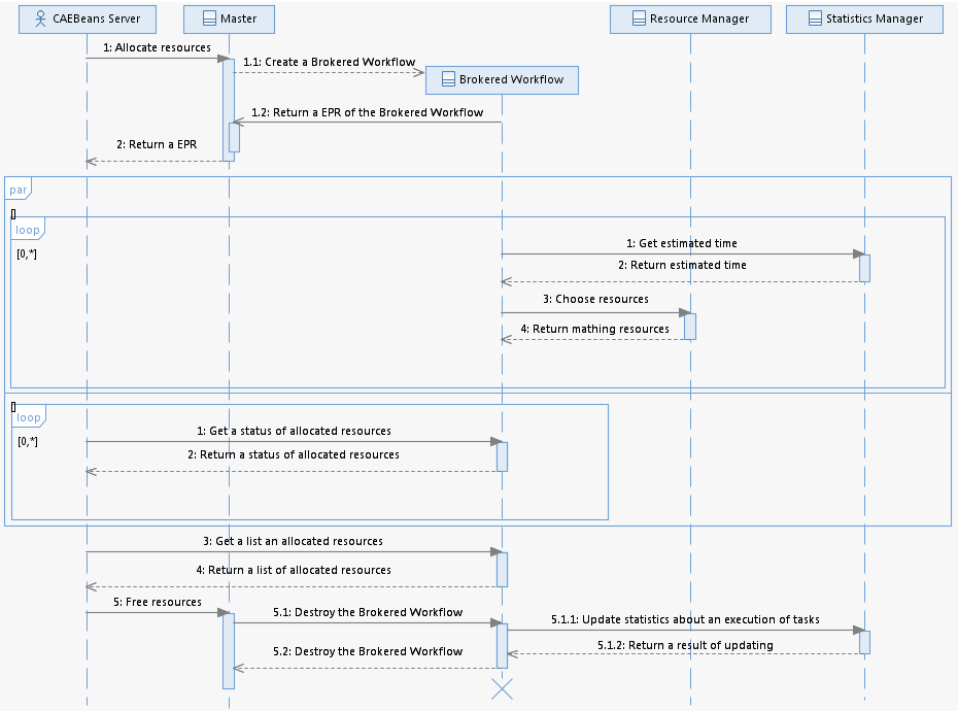

Figure 3. A sequence diagram of resource allocation by the CAEBeans Broker

The process of information gathering for the Resource Database is presented in the Figure 4. Updating steps repeat themselves consistently during the whole time of the Collector's execution. Updating information about resources signifies gathering data concerning hardware, software and licenses that is available for the CAEBeans Broker. The statistics of task performance includes information about the real time of tasks performance with specific parameters and on specific computing nodes.

6 Scheduling Algorithm of the CAEBeans Broker

The present work suggests developing a scheduling algorithm for distributed problem-oriented computing environments and considers some additional information regarding specifics of a problem area and workflow representation. It discusses using resource reservation and managing hardware, software and licenses in distributed computing environments.

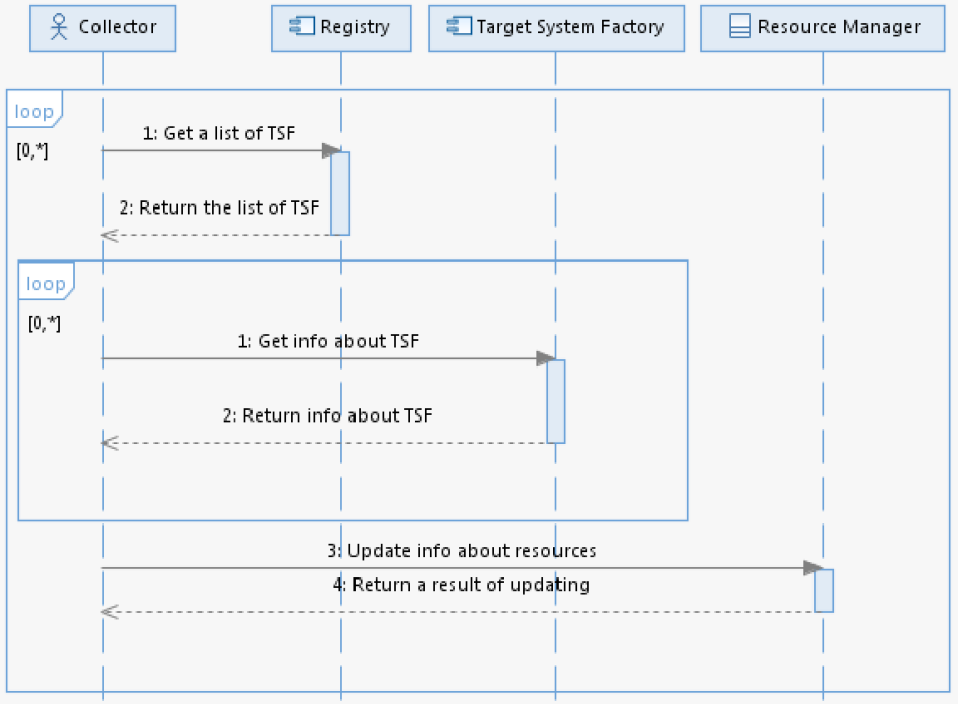

Figure 4. A sequence diagram of information gathering by the CAEBeans Broker

Here, the following methods are used:

- the method of two-phase resource reservation;
- the method of dominant sequence clustering;
- the accounting of task problem parameters for estimating its run-time performance.

The method of two-phase resource reservation. Reservation of resources in distributed computing environments is now a complex challenge, as it demands for existence of a component for a centralized scheduler of resources and for instant decision-making regarding reservation of resources in a task control system. The method of two-phase resource reservation is offered for the solution of this problem. This method allows carrying out preliminary reservation (marking) of demanded resources at the first phase, at the same time prospecting an optimum schedule for a workflow. Some time, confined by the priority of workflow performance, is necessary for making a plan of a workflow. After this timeout, the first phase finishes. Within the second phase, a final workflow resource allocation is completed or a release of all marked resources is carried out. In both cases, the allocation/release operations are carried out atomically.

The method of dominant sequence clustering. The workflow mapping to computing resources is implemented by the adapted method of dominant sequence clustering (DSC[12]). The essence of the DSC method lies in the minimisation of parallel time spent on workflow

performance. However, in the original DSC method scheduling is carried out in two steps. At the first step, there proceeds an association of several tasks into clusters (groups), at the second step, mapping of task clusters to real computing resources is performed. The adaptation of this method means the usage of preliminary resource reservation at the step of the association of clusters: a search for the free slots on computing resources will be done directly when workflow scheduling is accomplished.

The accounting of task problem parameters for estimating of its run-time performance. The time of the performance of separate tasks in a workflow is specified by a user default. However, in most cases users tend to overestimate the time of execution. The task statistics is stored in a Statistics Database to provide more exact assessment of time required for task performance. The database is managed by the Statistics Manager. The Statistics Manager provides information about the exact time of task execution based on the stored data concerning task parameters and the architecture of computing nodes that this task was executed on.

In the Figure 5 a scheduling algorithm of the CAEBeans Broker is presented. At the initial step, a job is presented in the form of a workflow. For each task from the job create a separate group. Every available resource has its own colour. It is necessary to paint of tasks in colours of available computational resources.

```
// Phase I
T = 0;                      // Time of the first phase
V_uncol = V;                // A set of uncolored nodes
R = ∅;                      // The set of reserved slots
while (T < T_lim){
  colored_method(G);        // The method of coloring graph G
  T++;
}
// Phase II
if (V_uncol ≠ ∅){
  for (i = 1; i = |V_uncol|; i++){
    n_i = V_nocol[i];
    s_i = eft(n_i);
    S = {s_i};
    reserve(S);
    R = R ∪ S;
  }
}
allocate(R);                // Allocate the reserved slots
```

Figure 5. A scheduling algorithm of the CAEBeans Broker

In the first phase a search of an optimal schedule to run a workflow and a resource reservation using a method of graph colouring. However, the first phase has a limit time T_{lim}. The limit time is based on the fact that a problem of finding of an optimal schedule is NP-complete.

During the execution of the second phase is searched nodes, which there were no reserved slots on computing resources. The set of uncoloured nodes is V_{uncol}. We allocate the first appropriate free slots for each node from the set V_{uncol} on computing resources. We perform a final resource allocation for all nodes.

Here a procedure $colored\_method(G)$ produces the graph colouring by the method described below. A function $eft(n) = s$ will search a slot s for the node n using an algorithm Earliest-Finish-Time.

```
st = 0;                    // A step of the method
Vuncol = V;                // A set of uncolored nodes
while (Vuncol ≠ ∅){
  DSst = domseq(Gst)       // Find a dominant sequence
  PTst = partime(Gst);     // Calculate a parallel time
  EDSst = {eij = (ni,nj), где ni,nj ∈ DSst};
  EDSst = EDSst \ {eij = (ni,nj), where ni and nj ∉ Vuncol};
  sort(EDSst);             // Sort weights of edges
  for (m = 1; m = |EDSst|; m++){
    eij = EDSst[m];
    S = search(ni,nj);  // S = {si,sj}
    if (S ≠ ∅){
      Vuncol = Vuncol \ {ni,nj}; // Subtract colored nodes
      reserve(S);              // Reserve slots
      eij = 0;
      break;
    }
  }
  st++;
}
```

Figure 6. A method of graph colouring of the CAEBeans Broker

In the Figure 6 the method of graph colouring is presented.

Here a function $domseq(G) = DS$ returns a dominant sequence, containing at least one uncoloured node. A procedure $sort(EDS)$ sorts weights descending, a procedure $reserve(S)$ reserves slots from a set S on the computational resources. A function $search(n_i, n_j) = S$ searches slots s_i and s_j, satisfying the requirements n_i, n_j respectively at the same computational resources.

7 Conclusion

In this paper, a scheduling algorithm in a distributed problem-oriented computing environment is described. It considers some additional information about specifics of a problem area and workflow representation; using resource reservation; managing hardware, software and licenses of distributed problem-oriented computing environments. Some use cases of a resource broker are presented and the process of resource allocation and architecture of the CAEBeans Broker are proposed.

The further work includes the following: conducting computational experiments for evaluating the effectiveness of a scheduling algorithm on supercomputers “SKIF-SUSU Aurora” and “SKIF Ural” at South Ural State University.

Acknowledgements

The present work is supported by grants given by the Council of President of the Russian Federation (Project MK-1987.2011.9) and the Russian Foundation for Fundamental Research (Project No.11-07-00 478-a).

References

1. I. Foster, C. Kesselman, *The Grid 2, Second Edition: Blueprint for a New Computing Infra-structure*, Morgan Kaufman, San Francisco 2003.
2. G. I. Radchenko, *A service-oriented approach of integration of engineering design and analysis systems in distributed computing environments (in Russian).* In: Parallel Computing Technologies (PaVT'2011): Proceedings of the International Scientific Conference (Moscow, March 28 - April 1, 2011), pp.606–616, SUSU Publishing House, Chelyabinsk 2011.
3. B. Raphael, I. F. C. Smith, *Fundamentals of computer aided engineering*, John Wiley, 2003.
4. J. Yu, R. Buyya, *A Taxonomy of Workflow Management Systems for Grid Computing* Grid Computing **3**, 171–200, 2005.
5. Condor `http://www.cs.wisc.edu/condor/`.
6. G. Laszewski, I.Foster et al. *CoG Kits: A Bridge between Commodity Distributed Computing and High-Performance Grids* // Proceedings of the ACM Java Grande 2000 Conference, pp. 97–106, CA, USA, June 2000.
7. E. Deelman, J. Blythe et al. *Pegasus: Mapping Scientific Workflows onto the Grid* // Proceedings of Grid Computing: Second European AcrossGrids Conference (AxGrids 2004), pp. 11–26, Nicosia, Cyprus, January 2004.
8. J. Cao, S. A. Jarvis et al. *GridFlow: Workflow Management for Grid Computting* // Proceedings of the 3rd International Symposium on Cluster Computing and the Grid (CCGrid03), pp. 198–205, Tokyo, Japan, May 2003.
9. M. Wieczorek, R. Prodan, T. Fahringer *Scheduling of Scientific Workflows in the ASKALON Grid Environment* ACM SIGMOD Record **34**, 56–62, 2005.
10. R. S. Fedyanina, *CAEBeans Server: a runtime environment of problem-oriented shells over engineering packages (in Russian).* In: Parallel Computing Technologies (PaVT'2010): Proceedings of the International Scientific Conference (Ufa, March 29 - April 2, 2010), pp. 621–628, SUSU Publishing House, Chelyabinsk 2010.
11. G. Mateescu, *Quality of Service on the Grid via Metascheduling with Resource Co-Scheduling and Co-Reservation* International Journal of High Performance Computing Applications **17**, 209–218, 2003.
12. T. Yang, A. Gerasoulis, *DSC: Scheduling Parallel Tasks on an Unbounded Number of Processors* IEEE Transactions on Parallel and Distributed Systems, Vol. 5, No. 9, pp. 951–967, 1994.