A Framework for Mobile Grid Environments based on Semantic Integration of Ontologies and Workflow-based Applications

A.P.C. SILVA[1]
V.C.M. BORGES[1]
M.A.R. DANTAS[1]

UFSC - Federal University of Santa Catarina
INE - Informatic and Statistics
LaPeSD - Laboratory of Research in Distributed Systems
88040-900 Florianopolis (SC) - Brazil
[1] (parra,vcunha,mario)@inf.ufsc.br

Abstract. One of objectives of the mobile grid infrastructure is become possible that the mobile user can use the computational power available in computational grids for solving complex problems. However, the majority of researches that search the interaction between mobile devices and grid computing do not provide a mechanism more automated and coordinated for submission and monitoring of several tasks that work cooperatively to solve a single problem. Additionally, none of these researches consider different forms that resources are described for the virtual organizations that compose the grid to select the resources more appropriate to execute these tasks. Therefore, in this paper is presented a framework that it treats these aspects. This framework uses the approach of integration semantics of multiple ontologies and the workflow mechanism, allowing a selection more omnibus of resources appropriate for execution of tasks of an automatized and controlled way more in the grid. A case study and experimental results demonstrated that framework enables: a selection of the resources more appropriate of different organizations for the execution of an application submitted from a mobile device and less power consumption of battery of these devices during the application submission and monitoring.

Keywords: Ontology, Workflow, Resource Selector, Semantic Matching, Grid, Applications Submission and Monitoring

(Received Received September 26, 2007 / Accepted Accepted December 08, 2007)

1 Introduction

The grid computing technology is an approach that proposes the reduction of problems related to the resources sharing and coordination among different organizations, also called virtual organizations (VOs). The sharing process comprises not only file transfer, but mainly the access to computer, software packages, services and special devices. These resources can have different access policies and are known differently by their characteristics inside a specific organization [12].

However, restrictions at the mobile devices difficult the provision of applications or services that allow resolution of complex problems which requisite high performance environments. A recent grid infrastructure, called as mobile grid [3, 22], proposes the integration of mobile devices with grid environments. This proposal has two forms of interaction with the grid.

As [3, 22] cites, the first form show the device as an access interface to a grid (e.g., grid portal). The second method is characterized by mobile devices as resources for a grid, providing resources of processing and other resources facilities (e.g., GPS receivers, sensors and cameras). The computing power of mobile devices has presenting a growing rate of improvements in recent years. However, the actual processing capacity and storage still does not enable the resolution of complex problems inside these devices, as observed in [30]. Therefore,the first form becomes more interesting.

Many researches are developing to allow interac-

tions between mobile devices and grid environments. An example of a novel function is to hide complex details from grid *middleware* operations for the mobile user (e.g., resources selection, security and load balancing). Recent related works in mobile grid are tackling the following aspects: [14] shows a mechanism based on ontology that enable to access services, that are available in mobile devices through the gateway; the proposal architecture presented in [30] provides a uniform method to manage QoS, ensuring interactions between mobile devices and grid in a dynamic and transparent form to users; in [10] it is presented a generic mobility model that extends models of conventional prediction, selecting resources in environments of mobile grid.

A common characteristic of these related works [3, 10, 14, 22, 30] is that these proposals only allow submission and monitoring of a task per time from the interface of a mobile device. Usually, mobile users require to control the submission of several tasks that work cooperatively, solving a single problem in a grid configuration. Therefore, it becomes necessary the use of a mechanism that makes possible the execution of these tasks in an more automatic and coordinated fashion.

Moreover, an engine for submission more automated can reduce the traffic of the wireless network. It is worth noting that a reduction in the access to wireless network can provide a smaller dissipation of battery energy of these devices, i.e., increasing its lifetime [23]. Extending the battery lifetime is one of the most critical and challenging problems in theses devices [28]. Therefore, this is one approach that provides more automatization for applications submissions, as it occurs in our architecture; it sends fewer submission requests and consequently, it may reduce the dissipation of battery energy of these devices.

Another aspect to be consider is related to resources, where these tasks must be submitted. An approach of application submission and monitoring in a automated and coordinate way through workflow concept is proposed in [2]. However, this approach and others mentioned previously [3, 14, 22, 30] does not concern about selection of resources in which application tasks will be submitted. As [7] observed, **resource matching** is the capacity for selecting resources more adequate of a grid configuration in accordance to requirements of each task of an application. [10] considers an approach of resources selection, which authors developed a selection algorithm based in parameters of mobile and fixed resources, as mobility rate and predicted time of availability. A prior and automatic selection of resources more adequates for application execution could enable more transparency in the interaction among mobile users and grid.

None related work does not take into consideration that different organizations have different autonomy. This fact brings more difficult to resources selection, because these resources can present syntactically distinct descriptions, but with same semantics. In a classic example, two distintic organizations refer to characteristics of memory size respectively as *physical_memory_size* and *main_memory_size*. Considering that one query for resources show that it desires a resource that has *physical_memory_size* $>=$ 1024 MB, only those resources that satisfy the restriction and that has the term *physical_memory_size* will be returned. Therefore, the use of a mechanism of resources selection from grid becomes more flexible and extensible, when it does not only consider the syntax of the descriptions of the resources, but also the meaning involved in these descriptions.

The main goal of this article is to present a framework proposal to allow a coordinate and automated fashion of submission and monitoring of several tasks to a grid environment from mobile devices employing workflow. Additionally, providing the selection of resources to submit each task in a grid, considering different ways that resources were described, using a selection based on the semantics integration of ontologies.

The paper is organized as follows. In section 2 we show concepts semantics integration. We present the proposed framework and their main components in the section 3. In section 4 we present a case study in respect of resource selector. In section 5 describes the experimental results about comparison of energy consumption between our framework and other approaches. Finally, in the section 6 we present our conclusions and future works.

2 Ontologies Semantic Integration

Following [15], ontology can be expressed as a formal and explicit specification from a shared concept. Studer el. al [32] mentions that the concept is related to a abstract model of a phenomenon which identifies relevant aspects of the phenomenon itself. On the other hand, the formal means that the ontology can be interpreted by a computer and the shared term intend to express that the ontology captures knowledge of a group of individuals.

As [13] observes, a relevant aspect referring to application of ontologies is the integration of existing systems/databases. This requirement indicates clearly that the interoperability is an important feature because it allows different computational systems exchanging information.

One direction to reach interoperability between two different systems (or databases) is to execute ontologies mapping, finding out semantic correspondences between these ontologies. However, as several research works indicate [13, 25, 9], automatic semantic mapping ap-

proaches are not able to identify the majority semantic correspondences. In other words, it could be expected that many semantic equivalences will be not considered. Erigh & Sure [9] coment that an automatic mapping could lead to incorrect results. Therefore, there is a common acceptance that a human interaction in this process is an important feature.

The global, multiple and hybrid approaches are found in the literature [13] for the integration of information sources based on ontologies. Casare & Sichman [5] mention that the hybrid method has the advantage of not limit the diversity of models that describe a same domain, due to the fact that each model has your own particular ontology. In addition, this paradigm does not consider ontologies complex mapping. The hybrid approach considers only defined terms from different ontologies that are related semantically with terms of shared global ontology.

3 Proposal Framework

In this section are described the main components of the proposal framework, which are: **Workflow Manager** (WM) and **Resource Selector** (RS), as shown in Figure 1. In item (a), information providers of the VOs publish their information resources in grid and their resource ontologies in the framework.

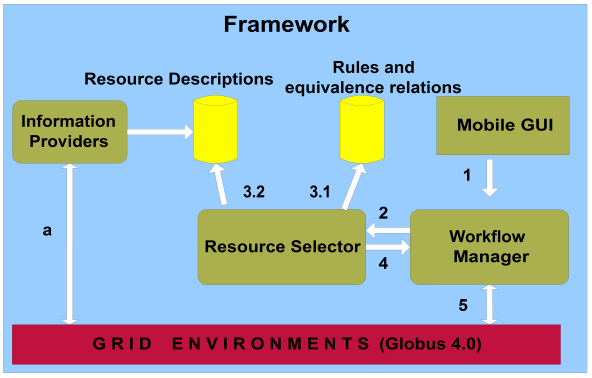

Figure 1: Proposal Framework

On step (1), Mobile GUI component was designed so that mobile users can submit and monitor application executions in grid environments from the own device. Each submission is forwarded to the **WM** component. Next, on step (2), **WM** sends queries to the **RS** which searches appropriated computing resources for executing each application task. When **RS** receives queries, it extends semantically queries based in semantic rules (step (3.1)). According to extension and equivalence relations of the terms (originating of the ontologies semantic integration). The **RS** searches in the resource descriptions, looking for resources that satisfy requirements of each task (step (3.2)). After search, **RS** returns the resources to the **WM** in which tasks should be submitted (step 4). Lastly, on step (5), **GW** controls submission of all tasks to the grid resources selected by the **RS**. **SR** and **WM** components are localized in the wired networks.

3.1 Workflow Manager (WM)

Due to the appearance of the grid computing, which provides several services, resources and the capacity to solve complex problems, the application execution with multiple tasks for the resolution of a unique problem has become more and more common. In general, this aggregate of tasks has interactions and dependence, requiring the use of some computational tools (e.g., software and/or database) that are shared by virtual organizations of the grid.

These tasks represent a work flow whose data are sent/received among tasks, obeying certain rules. Therefore, the utilization of a mechanism to control, organize and have some automatization of tasks execution flow become necessary. For this purpose, the workflow concept is employed in our approach, similar to some grid research projects [19, 6, 29]. This concept presents thin granularity and a generic solution for the definition of grid resources used in each stage of application execution in an automated and coordinated manner. The classic concept of workflow is presented in [16].

Besides this component processing and managing requests coming from mobile devices to execute in a grid environment, it also collects related information to task execution, performing all these functionalities of a transparent way to a mobile user. Last, it provides automatization for mobile users, dispatching all tasks to grid resources without any necessity of a user's interaction. Therefore, it allows more agility in the execution of tasks that work together to solve a problem. As it is shown in Figure 2. This component comprises three modules, as follows: **Controller**, **Engine** and **Collector**.

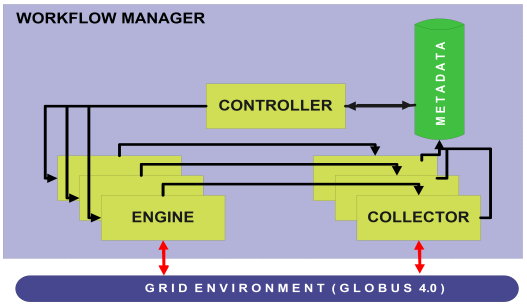

Figure 2: Workflow Manager [2]

Each submitted application has its proper **Engine** instance, **Collector** instance, and a unique Identifier.

The Java CoG Kit package [18] was used for interacting this component with Globus 4.0. The **Controller** module is responsible for receiving requests that arrive from mobile devices and order these requests, commanding and controlling their order through the Identifier. When this module receives a submission request, it creates an instance of the **Engine** module and it directs the request to this instance.

The **Engine** is the main module of this component. This module sends a resource request of the application to the **RS**. Subsequently, the **RS** returns a list from machines for the **Engine** that contains necessary grid resources for the execution of each application task. Next, the **Engine** module specifies machines selected in the workflow definition file of the submitted application, interpreting this definition file (or workflow script). Definition files are described in ***Karajan*** workflow language [17] of the *Java CoG Kit* package.

These scripts specify invoked programs, grid resources, data and flow control of each application. In addition, this module can interpret different workflow scripts for knowing how it must be submitted and controlled in the execution of each task flow of different applications. In this component, each application has defined its proper workflow script for specifying its task flow. Therefore, our architecture enables the execution of different applications. Thereafter, the submission of each task is done through GRAM grid service [11]. Finally, while these tasks are submitted, their status are informed through events generated during the execution. Thus, the **Engine** creates an instance called **Collector** for capturing this information and for storing them in checkpoint files (i.e., XML files) that report the status that each task is currently found.

The measure that tasks are submitted, their status are informed through events generated during the execution of the workflow. Therefore, the *Engine* creates an instance called *Collector* for capturing these information and to store them in checkpoint files (i. e., XML files) that reports the status that each task is currently. When the *Controller* receives a request from status from PDA, it extracts this information in the checkpoint files and it returns to the PDA.

Therefore, this component can provide advantages to mobile users, for example: due to the automatization provided for the **WM**, it allows more agility in tasks submission that work together to solve a problem; it minimizes errors in the ordination of these tasks submitted; execution of processes in which can coordinate distributed resources in multiple virtual organizations, also it obtains specific capacities of processing through the integration of multiple teams involved in different parts of the workflow and lastly, it also provide the necessary transparency to mobile user to utilize the grid services of the Globus middleware.

3.2 Resource Selector (RS)

The proposal selector was designed and implemented, using Semantic Web [1] tecnologies, to provide a grid resource matching based on the semantic integration of ontologies. All grid resources are described using different ontologies to supply application requirements. In this research, we adopt the meaning used by the terms to describe resources and not only a simple consideration of the syntax.

Figure 3 illustrates the framework conceived to build the selector prototype. The framework Jena [4] was employed, because it easily allows creating semantic web applications using Java language. In addition, this framework has interesting features such as: a facility to manipulate OWL ontologies and capability to infer information from the knowledge modelled in the ontologies using an inference engine (i.e. based on rules).

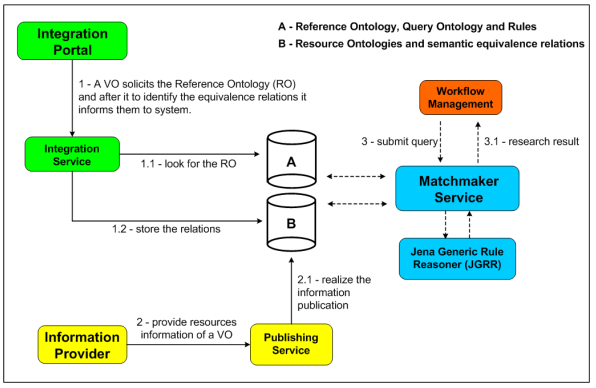

Figure 3: The framework proposal

In figure 3, there is the sketch of the grid resource matching framework proposal. The three main components of framework have the following characteristics:

- **Ontologies Integration Portal**: this component represents the interface used by a virtual organization to realize the semantic integration of its own ontology inside the matching system. The ontology developer is responsible for resources description of a VO. Adopting the concept of a Reference Ontology (RO), it influences the developer to consider semantic equivalences between terms from its VO ontology and the RO. Moreover, this interface was not prepared for the mobile users make the integration of the multiple ontologies, i.e., defining equivalences. Only specialists users (ontology developers of the VO) is that have knowledge necessary and suficient for interacting with knowledge systems. Therefore, in this approach, the developers are responsible to establish the semantic equivalencies and inform them to the semantic matching system. This interface was only

developed on desktops computers, enabling an interface less restricted and more flexible for this functionality;

- **Information Providers**: this component represents the resources information collectors of the VOs. Each collector publishes resources information from its own system, through Publication Service, as figure 3 shows;

- **Matchmaker**: the semantic matching operation is the main function of this component. The operation considers semantic equivalences coming from ontologies semantic integration and also considers restrictions defined in the queries that are submitted by users to matchmaker.

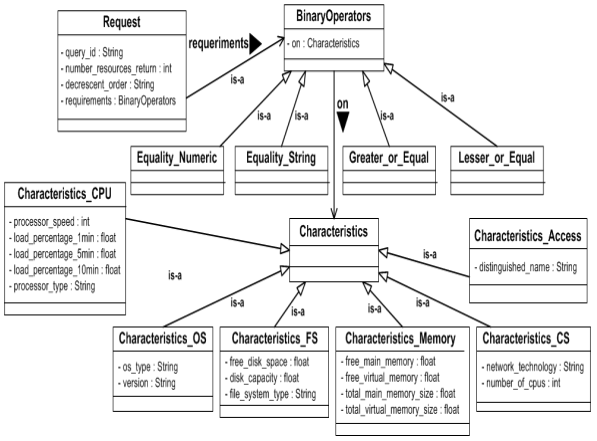

Figure 4: The Query Ontology

3.2.1 Reference and Query Ontology

The integration of different ontologies from different virtual organizations in our research work was achieved developing a shared common ontology called Ontology Reference (OR). This ontology was based on Core Grid Ontology (CGO) [33]. This proposal is a high level framework, where grid domain experts can represent all grid concepts semantically in a coeherent and consistency form. CGO was developed in OWL language for capturing and modeling the basic concepts and knowledge about the grid domain [33]. In the web context, the ontologies construction language OWL [21] is considered as a pattern by the W3C. The CGO features, flexibility and extensibility, allow this ontology be used for grid information integration, resources discovery, and resources allocation management [33].

The OR developed based on part in CGO ontology and the former was extended for modeling the basic elemets that integrate computational resources. The model these elements based on CIM model [8]. The Query Ontology (QO), detailed in Figure 4, through a class diagram it was built as an ancillary query language, that it is recognized from the developed matchmaker. The QO utilizes terms were previously defined in the OR that denote the main characteristics of grid resources.

Special concepts were conceived to help users to precisely describe grid resources requisites. These concepts were created using the *BinaryOperator* class. In other words, these operators indicate to the matchmaker component how to compare values attributed to terms defined in QO with the values attributed to terms that represent grid resources characteristics defined in different ontologies of VOs.

Inside the *Request* class it was defined the following properties, as figure 4 shows: ***query_id*** (query identifier); ***decrescent_order*** (a directive that indicates to the matchmaker which order criterion (decrescent order) resources will be returned. This instruction is based on resource characteristics [numeric value] defined inside the subclasses of *Characteristics* class); ***number_resources_return*** (this directive specifies the maximum number of resources the matchmaker should return) and ***requirements*** property that creates a relation between *Request* class and *BinaryOperators* class instances.

The *BinaryOperators* class has the property *on* which connects operators to the characteristics from the resources defined by the *Characteristics* class, as figure 4 illustrates. This property characterizes restrictions that resources should match. Subclasses from the *Characteristics* class specialize characteristics from the resources for providing more accurate information upon each type of resources. Examples are *Characteristics_CPU*, *Characteristics_OS* and *Characteristics_Memory* classes which aggregate characteristics of the following elements that compose the computational resources: processors, operating system and main memories.

3.2.2 Matchmaker

This component, as it was previoulsy mention, has the function to make the semantic matching between the published resources and queries. The following steps illustrate how the matching operation is realized.

(A) Query Expansion semantically:

This phase realizes the query expansion of semantic form in accord with the knowledge structure and information modeled in the RO. One interesting example to illustrate how this module works, can be verified when an operating system equal to Unix is requested. In the proposal system, the operating system concept was expressed in *OperatingSystem* class that it was specialized in *Windows*, *Unix*, and *MacOS* classes, as the Figure 5 shows. The *Unix* class, was specialized in *Linux* class. In the *Unix* class were created instances for representing the operating systems *Solaris*, *AIX* e *FreeBSD*. In

Linux class: *Debian*, *Slackware* and *Fedora Core*.

Through the rules, it is possible to capture these and others knowledge that exist inside the RO. Therefore, the query to the come to **Matchmaker** is expanded according to the valuation of a rule on the query and RO. After valuation, the *Matchmaker* will return not only resources with operating system *Unix*, but those with *Solaris*, *AIX* and *FreeBSD*. Moreover, the knowledge and information modeled inside the RO about the operating system *Unix*, captured by same rule, indicate to **Matchmaker** that returns resources with operating systems *Debian*, *Slackware* and *Fedora Core* too. This occurs, because the subclass relation between *Linux* and *Unix* classes is transitive. In other words, this relation expresses the knowledge that operating systems *Linux* are *Unix* too.

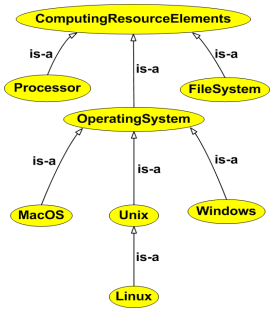

Figure 5: Part of Reference Ontology developed

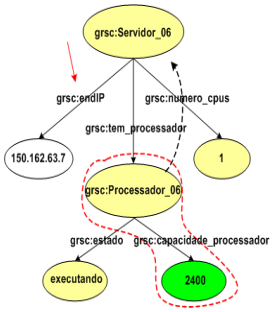

Figure 6: Resource describes in ontology A

(B) Resources search:

After the query expansion, the matchmaker searches which resources satisfy the restrictions defined in the queries. A more clear visualization of this step is exemplified in figure 6.

This figure illustrates a resource in form of a RDF ((Resource Description Framework) graph. Assuming that a user has defined in a query the following restriction *processor_capacity* >= 1800 MHz. In other words, the resources to be returned must have at least 1800 MHz of processing capacity. The matchmaker first searches in the equivalence relations stored in the system which terms are equivalents to the term *processor_capacity*. In the ontology A, this term is *capacidade_processador*.

The term *processor_capacity* is found like predicate in the triple <*grsc:Processador_06, grsc:capacidade_processador, 2400*>, that we denote triple base. This triple is indicated by the red traced line in figure 6. The procedure will verify if the value 2400 (attributed to *capacidade_processador*) attends this condition. The next step, if this value satisfies, it will be the search of the triple from the RDF graph that represents the computational resource that satisfies the restriction. The triple that has the predicate denoting IP address is that one that identifies the desired resource. We choose IP address, therefore it is a single qualifier of a resource in grid environments.

The matchmaker initiates the search for the subject *grsc:Processador_06* of the triple base, because this triple has as predicate the characteristic that satisfies the restriction. The matchmaker checks if this subject has predicate that means IP address. However, as the subject *grsc:Processador_06* does not present such predicate, the matchmaker verifies between the triples that have *grsc:Processador_06* as object which of them have this predicate. The triple <*grsc:Servidor_06, grsc:tem_processador, grsc:Processador_06*> is the only one that it has *grsc:Processador_06* as object and therefore matchmaker checks if its subject *grsc:Servidor_06* has the predicate that denotes IP address.

Based on semantic integration of the ontology A with the OR, matchmaker it verifies that the term *endIP* (indicated for the red arrow in figure 6) express the characteristic IP address in ontology A. In the result of integration is known that the term *endIP* is equivalent to *addressIP* (defined inside the RO for expressing IP address). Thus, the **Matchmaker** stores this triple that indicates the resource that satisfy this restriction. For each restriction existent in the query, the **Matchmaker** searches the triples that represent the resources that satisfy the restriction. These triples are stored in a set. This procedure occurs for all the restrictions defined in the query. The next step realizes the intersection of these sets to determine which resources satisfy all the restrictions, forming what we call of set solution. Case the query has directives, these are applied on the set solution and the result is returned to the **WM**. Otherwise, the own set solution is returned to the **WM**.

4 Case Study

This section describes the case study that was developed to test our framework. This study ocurred in a scenario of mobile grid that has the following configuration: a wired network where are grid resources and the access point that allows the mobile user utilizes grid resources. Therefore, it works as a bridge between the wired network and wireless network. The specification of the wireless network is 802.11b/g WLAN (Wireless LAN).

In our case study was used the workflow developed in the Genoma Project [20]. This workflow aims at the DNA sequencing of the Gluconacetobacter diazotrophicus bacterium. This workflow is composed for 7 tasks. The detailed description of each task can be visualized in [20]. The workflow definition was realized using the Karajan workflow language. For each task was elaborated a query for searching resources that attend requirements necessary to execute them, as illustrated in

the Figure 7. These queries were defined using the QO and associates to the workflow.

Directives/Restrictions	T1	T2	T3	T4
JR.request_id	query_T1	query_T2	query_T3	query_T4
JR.owner	vinicius	vinicius	vinicius	vinicius
JR.decrescent_order	total_main_memory	-	processor_capacity	processor_capacity
JR.number_resources_return	1	1	1	1
JR.Equality_String.**os_type**	-	Unix	-	Unix
JR.Greater_or_Equal.**total_main_memory**	>= 1024 MB	-	-	-
JR.Greater_or_Equal.**processor_capacity**	-	-	>= 3000 MHz	>= 3000 MHz
JR.Equality_String.**software_id**	S1	S2	S3	S4
JR.Equality_String.**database_id**	-	-	-	-

Directives/Restrictions	T5	T6	T7
JR.request_id	query_T5	query_T6	query_T7
JR.owner	vinicius	vinicius	vinicius
JR.decrescent_order	-	total_main_memory	processor_capacity
JR.number_resources_return	1	1	1
JR.Equality_String.**os_type**	-	-	Unix
JR.Greater_or_Equal.**total_main_memory**	-	>= 2048 MB	-
JR.Greater_or_Equal.**processor_capacity**	-	-	>= 3000 MHz
JR.Equality_String.**software_id**	S5	S6	S7
JR.Equality_String.**database_id**	-	DB1	DB2

Figure 7: Restrições e diretivas de pesquisa definidas para cada tarefa

The grid environment is composed of 3 virtual organizations, called as OV_1, OV_2 and OV_3, which have its proper resources ontologies. The use of 3 ontologies allow to show the diversity of the visions that the grid resources can present in organizations different. Organizations OV_1, OV_2 and OV_3 had respectively its resources described by ontologies developed in [31] (in the English language), [26] and [27] (in the Portuguese language), all constructed using OWL language. Each organization published 10 descriptions of resources, totalizing thirty grid resources. Semantic equivalences were informed jointly with resources established by each VO when integrating its ontology in the system. These equivalences are shown in the Figure 8. The first line in the Figure 8 shows the terms *os_type*, *SO* and *nomeSO* that they were defined respectively in the ontologies of OV_1, OV_2 and OV_3 organizations. These terms are equivalents to the *os_type* term defined in the RO, and therefore, in the QO denotes operating system.

Query Ontology	OV_1's Ontology	OV_2's Ontology	OV_3's Ontology
os_type	os_type	SO	nomeSO
max_number_of_processes	max_number_of_processes	numero_maximo_processos	-
total_virtual_memory_size	swap_memory_size	total_swap_MB	-
login	login	nome_conta	nomeConta
number_of_processes	number_of_processes	-	-

Figure 8: Equivalence relations

The WM sends queries of this *workflow* to RS when it receives a submission from a mobile user desiring to execute this workflow. The RS verifies which resources attend each query, basing it in the equivalence relations among terms of the QO and resources ontologies integrated. For instance, the T7 task, shown in the Figure 7, it needs a resource that can be to access by the *vinicius* user (*owner* term), that it has operating system *Unix* (*os_type* term), that it has at the least of *3000 MHz* (*processor_capacity* term) of clock processor speed, it also has *software S7* installed (term *software_id*) and the database *DB2* installed (*database_id* term). Amongst resources that have these characteristics, the RS must returns those that will have the biggest capacity of processing (criterion attributed to the directive *decrescent_order*).

Resources Characteristics	Resource (a)
Unitary_Computer_System.ip_address	150.162.56.12
Unitary_Computer_System.authorized_account.**distinguished_name**	vinicius;caetano
Unitary_Computer_System.running_os.**os_type**	SunOS
Unitary_Computer_System.has_software.**software_id**	S3, S5, S7

Grid Resource (a) shared for the OV_1 that satisfies requirements of T2 and T5 tasks

Resources Characteristics	Resource (b)	Resource (c)
Host.endIP	140.68.107.10	140.68.87.50
Host.contas_autorizadas.**nome_conta**	parra, mario, vinicius, caetano	parra, alex, vinicius
Host.tipoSO.**SO**	Fedora Core	Fedora Core
Host.**total_memoria_MB**	16384.0 MB	3062.0 MB
Host.**velocidade_cpu**	2400.0 MHz	3000.0 MHz
Host.tem_software.**nome_software**	S1, S5, S6	S2, S7
Host.tem_base_dados.**nomeBD**	BD1, BD2	BD2

Grid Resources (b) and (c) shared for the OV_2 that satisfies requirements respectively of T1 and T6, and T7 tasks

Resources Characteristics	Resource (d)	Resource (e)
Host.ipHost	147.160.50.37	147.160.12.19
Host.permissoesUsuarios.**idConta**	anubis, vinicius, guilherme	vinicius, parra
Host.temSistemaOperacional.**nomeSO**	Debian	Debian
Host.temCPU.**clockCPU**	3200.0 MHz	3200.0 MHz
Host.temSoftware.**identificador_software**	S1, S3, S4	S1, S3, S4, S6, S7
Host.tem_base_dados.**identificador_base_dados**	-	BD1, BD2

Grid Resources (d) and (e) shared for the OV_3 that satisfies requirements of T3 and T4, and T7 tasks

Figure 9: Resources that satisfy the tasks requirements

The resource (e), presented in the Figure 9, it was the resource returned for the RS in accordance with query_T7. (c) and (e) resources attend restrictions defined in the query_T7 (requisite of the T7 task). However, the resource (e) was chosen, due it presents greater processing capacity, thus satisfying, the classification criterion defined in query_T7 (*processor_capacity*). It is important to detach semantic matching realized by the RS among terms that express characteristics of the resource (e) and the ones that was used in the query. Equivalence relations among terms resultants from the semantics integration of the OV_3 ontology and the RO are: *idConta* and *owner*; *nomeSO* and *os_type*; *clockCPU* and *processor_capacity*; *identificador_software* and *software_id*; and *identificador_base_dados* and *database_id*. In addition, it is important to detach the query expansion allowed to the RS recognizing that the resource (e) has operating system Unix. This knowledge was defined inside the RO and it was captured through rule.

5 Experimental Results

This study ocurred in a scenario of mobile grid that has the following configuration: a wired network where are

grid resources and the Access Point (AP) which works as a bridge between the wired network and the wireless network, enabling the mobile users to make use of grid resources. The configuration chosen for the experimental tests was a real scenario in a WLAN. This environment is projected to reach one real evaluation of the application execution in the mobile grid and related cases. Characteristics of the PDA are: Palm Tungsten C executing the operating system Palm 5.2.1 with processor 400MHz, 64 MB RAM, Built-in Wi-Fi (802.11b).

The **Framework** allows the submission and monitoring of applications through a single submission request. In other words, *Workflow Manager* controls the execution flow and it invokes the necessary computational tools without requesting interaction of users for performing all steps. This is a differential aspect when compared to the **Other** approach presented in [3, 10, 14, 22, 30]. In these others research works, the user controls the submission order of each task that work together cooperatively to solve the same problem. In the **Other** approach implemented the user submits all application tasks, a task each time through the interface.

In the graph of Figure 10 a comparison of the mean battery power consumption for submitting and monitoring tasks for resolution of a problem is presented. The comparison considers the use of **Framework** in contrast to the **Other** approach implemented in [3, 10, 14, 22, 30]. The result shows that **Architecture** reached an average saving of 31% when compared to the **Other** approach. **Framework** allows a larger number of executions, i.e., on average 12 more application executions than the **Other**, as it can be seen in the graphic.

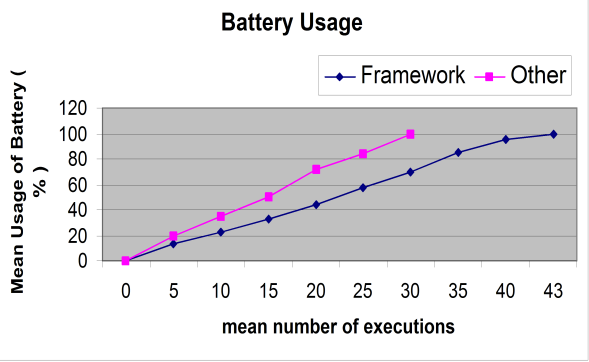

Figure 10: Battery Usage

It was performed 24 experiments in each approach, each experiment represents the percentage of the battery consumption of five application executions of the sequencing problem. In each application execution, the consumption percentage for submitting and monitoring each task in the **Other** approach was summed up and the consumption percentage for submitting and monitoring the application execution in the **Framework** approach was verified too. Besides, in each experiment, grid resources were exclusively dedicated for tests and the time interval of sending notifications was equal in both approaches. Therefore, these variables do not influence the result.

The average and confidence interval of battery consumption for each experiment was respectively 11% and ***(+/-)*** 1,6603% in **Framework**, while in the **Other** approach was 16,17% and ***(+/-)*** 1,6323%. A ***2-Sample T*** Test [24] was used to compare if there is a significant difference between the **Framework** and the **Other** approach. In a significance level of 95%, the ***2-Sample T*** test evidenced a significant difference between the two samples, due to the resultant ***P-value*** = 0,0002, as [24] observes, when ***P-value*** $\leq$ 0,05, it indicates a significant difference between two samples. In other words, **Framework** saves significantly more battery energy than the **Other** approach for submitting and monitoring an application.

6 Considerations and Future Works

In this article, we presented a framework for submission and monitoring of several tasks to a grid environment through mobile devices, selecting resources more appropriated previously to execute these tasks. Framework components became possible the resolution of complex problems using the workflow mechanism and the semantics integration of ontologies that describe the grid resources. Therefore, our framework provide to a mobile user a semantics form to select resources more adequate for execution of these tasks in an automated and coordinated style in the grid configuration.

Experiments from a case study indicate that the Resource Selector component allowed a more appropriate resource selection from different VOs for the execution of one workflow submitted from mobile device. It is important to mention that these executions considered different descriptions that these resources have inside each VO. Moreover, the experimental results indicated that the proposed framework enabled to consume less battery power of mobile devices for submitting applications.

As future works, the capacity of restriction of queries will be extended, extending the RO, so that, they express other binary operators (e.g., different [!=] and multiplicity [*]); it allow the automatic adjustment of the execution flow in cases of disconnections of mobile devices and to incorporate a method of semi-automatic mapping in framework for decreasing the ontologies developers effort of VOs in integration of yours ontologies in the system.

References

[1] Berners-Lee, T., Hendler, J., and Lassila, O. The semantic web. *Scientific American. Disponível em: http://www.w3.org/2001/sw*, 5(284):34–43, May 2001.

[2] Borges, V. C. M. and Dantas, M. A. R. Uma abordagem de submissão e monitoração de múltiplas tarefas para ambientes de grade computacional utilizando dispositivos móveis. *XXXIII SEMISH*, pages 403–418, 2006.

[3] Bruneo, D., Scarpa, M., Zaia, A., and Puliafito, A. Communication paradigms for mobile grid users. *3rd (CCGrid'03)*, pages 669–676, May 2003.

[4] Carroll, J. J., Dickinson, I., Dollin, C., Reynolds, D., Seaborne, A., and Wilkinson, K. Jena: implementing the semantic web recommendations. *13th WWWC*, pages 74–83, 2004.

[5] Casare, S. and Sichman, J. S. Using a functional ontology of reputation to interoperate different agent reputation models. *JBCS - Special Issue on Ontology Issues and Applications*, 11(2):19 – 94, November 2005.

[6] Chen, J. and Yang, Y. Dynamic setting, verification and adjustment of upper bound constraints in grid workflow systems. *IEEE Computer Society, 2th SKG'06*, 0:10, 2006.

[7] Czajkowski, K., Foster, I., Kesselman, C., Sander, V., and Tuecke, S. A protocol for negotiating service level agreements and coordinating resource management in distributed systems. *8th JSSPP*, pages 153–183, July 2002.

[8] DMTF. Common information model (CIM) standards, Agosto 2006. Disponível em: http://www.dmtf.org/standards/cim.

[9] Ehrig, M. and Sure, Y. Ontology mapping - an integrated approach. In *5th ESWS*, volume 3053, pages 76–91. Springer-Verlag, 2004.

[10] Farooq, U. and Khalil, W. A generic mobility model for resource prediction in mobile grids. *CTS, IEEE Computer Society*, 0:189–193, 2006.

[11] Foster, I. Globus toolkit version 4: Software for service-oriented systems. *IFIP NPS*, pages 2–13, 2005.

[12] Foster, I., Kesselman, C., and Tuecke, S. The anatomy of the grid: Enabling scalable virtual organizations. *IJSA*, 3(15), 2001.

[13] Freitas, F., Stuckenschmidt, H., and Noy, N. F. Ontology issues and applications guest editors' introduction. *JBCS - Special Issue on Ontology Issues and Applications*, 11(2):5–16, November 2005.

[14] Grabowski, P., Kurowski, K., Nabrzyski, J., and Russell, M. Context sensitive mobile access to grid environments and vo workspaces. *MDM, IEEE Computer Society*, 0:87, 2006.

[15] Gruber, T. R. Toward principles for the design of ontologies used for knowledge sharing. *IWFO*, 1993.

[16] Hollingsworth, D. Workflow management coalition. reference model and api specification. *WfMC-TC00-1003*, 1996.

[17] Laszewski, G. and Hategan, M. Workflow concepts of the java cog kit. *Journal of Grid Computing*, 3(3-4):239–259, 2005.

[18] Laszewski, G. V., Foster, I., Gawor, J., and Lane, P. A java commodity grid kit. *Concurrency and Computation: Practice and Experience*, 13(8-9):643–662, 2001.

[19] Lee, S., Wang, T. D., Hashmi, N., and Cummings, M. P. Bio-steer: A semantic web workflow tool for grid computing in the life sciences. *FGCS*, 23(3):497–509, 2007.

[20] Lemos, M. Workflow para bioinformática. Master's thesis, (PUC-Rio), 2004.

[21] McGuiness, D. and Harmelen, F. V. Owl web ontology language overview. *W3C*, February 2004.

[22] McKnight, L. W., Howison, J., and Bradner, S. Guest editors' introduction: Wireless grids–distributed resource sharing by mobile, nomadic, and fixed devices. *IEEE Internet Computing*, 8(4):24–31, 2004.

[23] Mohapatra, S., Cornea, R., Oh, H., Lee, K., Kim, M., Dutt, N. D., Gupta, R., Nicolau, A., Shukla, S. K., and Venkatasubramanian, N. A cross-layer approach for power-performance optimization in distributed mobile systems. In *IPDPS*, 2005.

[24] Montgomery, D. *Design and Analysis of Experiments*. John Wiley and Sons Ltd, 2005.

[25] Noy, N. F. Semantic integration: a survey of ontology-based approaches. *ACM SIGMOD Record, Special Issue on Semantic Integration*, 33(4):65 – 70, 2004.

[26] Pernas, A. and Dantas, M. A. R. Grid computing environment using ontology based service. *5th (ICCS'05),LNCS 3516 Springer 2005*, 3516:858–861, May 2005.

[27] Ramos, T. G. and Melo, A. C. M. A. An extensible resource discovery mechanism for grid computing environments. *6th IEEE CCGRID*, 1:115–122, 2006.

[28] Rong, P. and Pedram, M. Extending the lifetime of a network of battery-powered mobile devices by remote processing: a markovian decision-based approach. In *proceedings of DAC '03*, pages 906–911. ACM Press, 2003.

[29] Santo, M. D., Ranaldo, N., and Zimeo, E. Metacomputing through the enactment of a bpel4ws workflow in a grid environment. *ITCC*, 1:316–321, 2005.

[30] Shi, W., Li, S., and Lin, X. Towards merging pervasive computing into grid - lightweight portal, dynamic collaborating and semantic supporting. *IMSCCS, IEEE Computer Society*, 1:560–563, 2006.

[31] Silva, A. P. C. and Dantas, M. A. R. An efficient approach for resource set-matching in grid computing configurations. *20th HPCS*, 0:5, 2006.

[32] Studer, R., Benjamins, R., and Fensel, D. Knowledge engineering: Principles and methods. *IEEE TDKE*, 12(25):161–197, 1998.

[33] Xing, W., Dikaiakos, M. D., and Sakellariou, R. A core grid ontology for the semantic grid. *6th IEEE CCGRID*, 0:178–184, 2006.