

Extending Pervasive Devices with the Semantic Grid: A Service Infrastructure Approach

Tao Guan, Ed Zaluska, David De Roure
School of Electronics and Computer Science
University of Southampton
Highfield, Southampton, UK
{tg04r, ejz, dder}@ecs.soton.ac.uk

Abstract

Pervasive computing devices are gradually becoming increasingly prevalent in our everyday life, enabling the users in the physical world to interact with the digital world. Grid computing provides a vision of accessing computational resources automatically on demand to deliver the services required with the appropriate quality. Semantic Web technologies enable automation and interoperability. It is the interaction of Semantic Grid and the physical world (pervasive devices) that will enable us to a new concept of “ambient intelligence”, the notion of intelligence in the surrounding environment supporting the activities and interactions of the users. This paper presents a service approach that extends pervasive devices with semantic Grid services by using a device-proxy-Grid system infrastructure. OWL-S is used to describe different proxy devices’ capability in order to realize automatic proxy service discovery. A context model is also discussed to enhance the intelligence of the system.

Key words: Grid Computing, Pervasive Devices, Semantic Web, System Infrastructure, Automatic Behaviour

1. Introduction

One of the visions of Grid computing is to access computational resources automatically on demand in order to deliver the services required with the appropriate quality. As a service-oriented approach to Grid computing is increasingly adopted, many systems which can discover Grid Resources on-the-fly and access them dynamically come into existence. Instead of being limited to local computing resources, high-performance computing equipments around the world can be used conveniently, forming a virtual Grid that aims at effectively sharing, using and managing versatile resources [1].

At the same time, there are two significant trends in modern computing technology with rapid improvements in the area of ubiquitous and mobile computing. More pervasive devices are deployed, and more integration and computation power is required behind the scenes. While these new general purpose pervasive devices (e.g. smart phone, PDA) improve the absolute capability of enabling users to access computing resources conveniently [2], there is no doubt that creating complicated applications executing on various pervasive devices is a challenging objective because such ubiquitous devices are typically resource-constrained, relative to the static counterparts (e.g. desktops, workstation), with limited processing memory, storage, energy, and network resources.

A possible solution is that pervasive devices make use of Grid services to enable users to access computational resources automatically on demand with appropriate quality of service delivery. The various Grid services enhance the ability of ubiquitous devices to execute complex applications which cannot be run directly on the devices themselves. For example, the traveller’s smart phone with a build-in camera can produce a large volume of data which needs to be processed if any special tasks are required. Large volumes of data demand significant computational power (such as image compression or place recognition) which can be best supplied by Grid services. The traveller can use his smart phone to discover the local available Grid services and submit the request. The Grid will achieve this task based on the client’s requirement and return the results to the user’s smart phone through the distributed resources anywhere in the world. In addition to providing new application opportunities for these devices, integrating pervasive devices with the Grid can save energy, storage space, memory capability, and processing cycles, hence further reducing their size, weight, and cost.

The concept of Grid services on pervasive devices can also benefit from another significant movement in comput-

ing, the move toward machine-processable explicit knowledge as exemplified by the Semantic Web. Semantic Web technologies are now being adopted within the practice of Grid computing, a field becoming known as the Semantic Grid [3]; Semantic Web technologies are also being used within pervasive computing, such as designing ontologies of various devices and representing context information. They provide a very considerable degree of automatic processing, interoperation and integration, which is the central requirement of the system infrastructure allowing pervasive devices to use Grid services effectively.

Pervasive devices form the intersection between the physical world and the digital world [4]. The digital world of the Grid meets the physical world through a variety of sensors, instruments and interfaces. Pervasive devices need the Grid for computation and integration [5], while the Grid needs the pervasive devices to interface with the physical world. The Semantic Web enables the automation and interoperability, which helps building ambient intelligence infrastructure. However, in order to realize such a vision of integrating pervasive devices into the Grid services with the support of Semantic Web technologies, we have to meet the following challenges [6]:

- Establishing the use of Grid services on-the-fly, and the negotiation mechanisms to select appropriate grid services when many are available.
- A framework is required to bring devices into the Grid in a flexible, open and interoperable way.
- The description mechanisms have yet to be agreed, and the techniques for service discovery and composition are not very sophisticated at this time. Semantic Web Services proposals exist but are still some time away from standardisation.
- Realising the autonomic vision, such as self-configuration, self-optimization and self-healing properties.

In this paper, we will present our current and future research work to solve several key issues about bringing pervasive devices into semantic Grid. Section two introduces the related research work. Section three describes a basic device-proxy-Grid system infrastructure which allows pervasive devices to access typical Grid services. Using OWL-S to describe service's capabilities which implements automatic service discovery and a context model with the initial ontology approach to enable the system infrastructure to present the right information to the right user at the right time in the right place are discussed in section four. In section five the current implementation and future work are discussed, with the conclusions in section six.

2. Related Work

In 2001, M.Satyanarayanan [7] referred to "Cyber Foraging", a new concept of dynamically augmenting pervasive device's computing capability by offloading the complex tasks to the local high-performance desktops or workstations. The key of "Cyber Foraging" is that pervasive devices could exploit various nearby high-performance computing infrastructure on demand through ubiquitous networking. From then on, several task-offloading systems have been demonstrated, such as a lightweight secure cyber foraging infrastructure [8] and Slingshot [9]. Although the purpose of "Cyber Foraging" is to enhance the capability of the pervasive device, and the related task-offloading systems realize the vision of transferring complicated tasks from pervasive devices to not resource-intensive surrogate devices, the notion of Grid services is not mentioned. All the pervasive device can access is a couple of local computer resources, which restricts the integration and computation power of pervasive devices deployed ubiquitously around the world.

Pervasive devices form the intersection between the physical world and the digital world. It enables Grid services available and accessible anytime anywhere. At present, a number of research projects and papers are dealing with the investigation of these possibilities. In [10], the authors describe their attempts to write Grid clients for mobile devices, and implementing a web-based proxy that uses Globus toolkit 3 to talk to distributed Grid services. However, their experience are based on an implementation of a mobile Grid client for an existing web-based e-learning system, and the function of the proxy device is only transmit the request from mobile devices to Grid, almost no processing at all. The Grid can be considered as a flexible, self-configuring dynamic network of independent, mobile, intelligent agents using each other's resources [11]. Based on this distributed mobile agent-based architecture in cellular networks, a problem-solved environment for wireless mobile devices is built using the computational Grid paradigm [12]. Each mobile device in this Grid-based environment runs a client application implementing a lightweight communication protocol for information exchange with the Grid Brokering Service. Three control variables are allowed to be modified under different scenarios: initial Grid population size, device mobility and frequency of task initiation. However, this Grid-based architecture needs to add a new agent component into the existing Grid system, which is not a suitable way of enabling pervasive devices to use any existing Grid services anytime anywhere.

IITRC program support a series of work of designing a middleware approach that aims at assisting handheld devices in accessing Grid services by wrapping the computational and resource intensive tasks in a surrogate and

shifting them to a capable machine for execution [13] [14] [15]. Besides presenting a three-tier architecture (device-gateway-Grid), the communication mechanisms and the proposed optimizations in the middleware are emphasized in [13]. It provides the multiple discovery techniques because of efficiency and fault tolerance. In order to measure the impact of the executing system and highlight the benefits and shortcomings of the approach, a bare-bones implementation of the proposed architecture is realized [14]. The surrogate has been coded for the handheld device with the functionality of monitoring the remote server and the results are to be displayed at the handheld devices. Implementing a detailed application system is not the only way of estimating the performance of the existing infrastructure. In [15], the authors model the interaction between the mobile client and the mobile Grid middleware service with non-Markovian Stochastic Petri nets. Their goal is to evaluate the delay caused by the communication between the client and the middleware service, which should be within acceptable limits so that this three-tier architecture does not produce plethoric overheads for mobile client users compared to the normal Grid users.

3. System Architecture

At present, there is no OSGI Java API for pervasive devices [10], and the existing client middleware technologies make assumptions about capabilities of their host environments so that the resource-constrained devices are very difficult to be configured and supplemented with software-based techniques to make them Grid-interaction capable. Our solution is to implement a proxy that communicates between pervasive devices and distributed Grid services.

3.1. System Overview

Three kinds of components are included in our system architecture, portable devices, proxy devices, and Grid services. Portable devices refer to general-purpose pervasive mobile devices, such as smart phones [16], or PDAs. While there is no commonly-accepted universal device platform, mobile phones have the greatest potential to replace all the different devices we currently carry in our pockets [17] because of emerging mobile phone technologies which provide various styles of networking capability through which the Grid can be accessed. Proxy devices might be a desktop computer or a small server available for nearby portable devices via the local wireless network, providing a relatively resource-rich execution environment and network connectivity compared to portable devices. The responsibility of the proxy device is the transferring media between pervasive devices and Grid services, or the intensive tasks

executing environment depending on the specific applications. Grid services are registered at the proxy device, or one of the Grid nodes which can be easily accessed by the proxy, providing an indirect but convenient mechanism for portable devices to find the required Grid service. The resource-intensive tasks and the requests of accessing Grid from the portable device will be submitted to the appropriate surrogate device based on the ability of what portable device can find a suitable “proxy service”, leaving the less resource-intensive tasks to the device, such as displaying the final results of the request or managing a variety of application requests of the device. The rich external resource in the Grid can be used by the proxy device, enhancing its capability of processing unexpected complicated tasks. The following figure shows a diagram of the system architecture. Figure 1 shows a diagram of the system architecture.

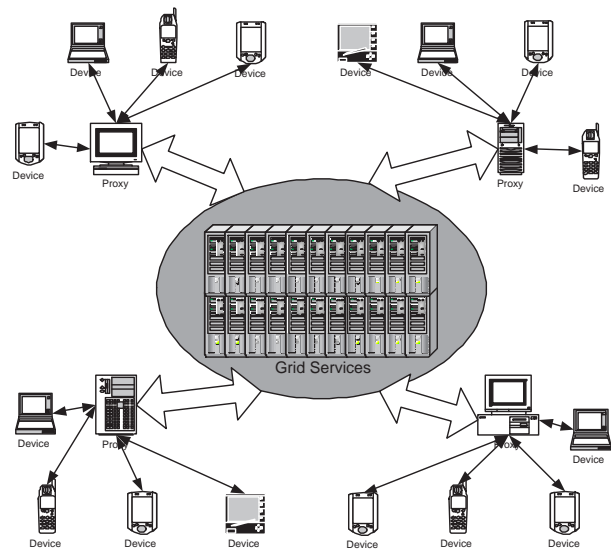


Figure 1. Outline System Architecture.

3.2. Device

Portable devices refer to general-purpose pervasive mobile device, such as mobile phones, or small laptops. All of these devices have the capability of communicating with other devices through a variety of wireless network protocols, including IEEE802.11 families and Bluetooth, depending on the resource of different pervasive devices (802.11 consumes more energy, which exhausts a cell phone’s batteries quickly, but may be well suitable for laptop devices). When a portable device starts to connect a proxy, proxy locating, discovering, and communication authorization will be processed at first. After establishing the connecting contract between the device and the proxy, the device will send a command request to the proxy, consisting

of enough information for the proxy to download the real application code or invoke the appropriate Grid services. Once the proxy is ready for the the request from the device, the device will send the input data to the proxy executing portion, collect the response, and display the results on the portable device's screen.

The user personal information or some security data, which are required during the authentication process with the proxy, are stored on the portable device. It will also be used before invoking specialized Grid services. For example, the scientist's personal information must be send to the Grid every time he or she requests to modify the important lab parameter, to ensure that there is enough power to invoke the modification service. The used interface code and un-finished service status can also be stored on the portable device, to avoid requesting the same information every time.

3.3. Proxy

Originally, there are two candidate mechanisms for transferring the device's request to the proxy. One is to require devices and proxies to share a common file system and demand the client to run a relatively heavyweight middleware system. This mechanism requires the application-specific knowledge regarding how to partition an application between a client and the nearby proxy. The other mechanism does not share a common file system. It only relies on the proxy being connected to the Internet to locate and download client application code. Although the former one has a long research history, however, for most applications, the best partitioning may not be immediately clear. The latter mechanism is easy to realize because it only relies on application writers to partition application at development time and assume that applications are split in a way that mitigates client-proxy communication. What is more, with the increasing on-line devices and the deployment of a variety of Grid services in the smart environment, it is more convenient for proxies to locate and download the executing code after receiving the "command word" from the clients. The idea of running most of application on the proxy or Grid and using client for input-output would be the ideal solution in the system infrastructure.

The proxy offers an interface for pervasive devices to access it. The proxy manager is responsible for initializing the communication between the pervasive device and the proxy. It accepts the request from any pervasive device which desires to request a virtual server. If the requesting pervasive device is authorized to use the proxy, the proxy manager tries to allocate the resources to that pervasive device in term of the application to be executed on the proxy. If allocating the resource is successful, there will then be a contract established between the proxy and that pervasive device.

The proxy provides an interface for accessing to the Grid services. For Grid services, proxy is the typical Grid service user. Hence, the existing Grid middleware technologies can be used on it. Although Globus Toolkit 3 [18] is a popular integrated set of tools and software that facilitates the creation of grids and applications, we use OMII [19] to create the communication between the less resource-intensive proxy and the resource-rich Grid to achieve the tasks assigned by the pervasive devices. OMII is an open source, robust and secure Web services platform for building Grid applications, which supports the most stable elements of the WSRF standard, a refactoring of the OGSF [20] that is totally based on Web Services. OMII provides a Client/Server infrastructure to make clients and service providers collaborate. Its server component, which is used by service providers, enables secure authorised access to their resources, while its client component, used by a client, enables access to a Service Provider's resources and applications. OMII Client middleware will be installed on the proxy device, enabling it to access the services and resources of the Grid conveniently.

There may be several proxy devices available to a pervasive device. How the pervasive device locates the appropriate proxy is about the question of proxy description and proxy discovery. Proxy devices must be able to advertise their attributes to allow pervasive clients to locate them based on their location, their performance, and other features. In our proxy design, the relevant attributes of the proxy includes:

- Grid services available on and through the proxy device
- Available computing resources on the proxy device, such as CPU cycle, memory capability, storage space, and network bandwidth
- Software executing environment, such as the version of operating system, the current status of different Vservers.
- Authentication mechanism, e.g. public key scheme using RSA algorithm or ECC (Elliptic Curve Cryptography)
- Proxy discovery protocol, such as UPnP, Jini, Bluetooth and so on

4. System Intelligence

Pervasive computing exhibits the vision that a variety of computer systems are integrated into the users' everyday life seamlessly, providing services and information "anytime, anywhere". Our system infrastructure, enabling the

pervasive device to access Grid services, tries to incorporate physical computing entities into a digital world. The purpose of this open and dynamic digital world is to present the right information to the right user at the right time in the right place. Hence, various computing entities (e.g. service consumers, and service providers) in such a context-aware environment require to be able to share knowledge with each other, reason contexts, and support automatic behavior. Applications of this digital world must be context-aware so that they can adapt rapidly changing conditions [21] [22]. In order to achieve this purpose, the system infrastructure must understand its working environment comprehensively. It should know all the computing entities within the environment, the relationship between these computing entities, and the events and assignments being undertaken by these entities.

4.1. Automatic Service Discovery

A context-aware space has the potential to integrate a great number of pervasive devices, from tiny sensors to extremely dynamic and powerful devices. It enables great access not only to content but also to services. However, it is not easy for new-coming users to locate, find, and invoke their required services. Service discovery protocols simplify the interaction among users, devices, and services. It minimizes administrative overhead and save the system developers from coding all possible interactions at the designing time [23].

OWL-S [24] is a language for describing services, and it provides a standard vocabulary that can be used together with other aspects of the OWL [25] description language to create service descriptions. The types of knowledge about describing a service are the profile of the service, the using method of the service, and the interacting way of the service. The structure of OWL-S upper-level ontology is just based on these three perspectives. The “Service Profile” provides the information for the service requester to discover a service, while the “Service Model” and “Service Grounding”, provide enough information of how the service is used and how to interact with the service.

The proxy can be considered as a set of services for the pervasive device. In order to enable the new service requester to locate and find the required services accurately and efficiently, each service’s profile is required to be described exactly, especially the functionality description of the service, including the information transformation (represented by the inputs and outputs) and the state change produced by the execution of the service (represented by the preconditions and effects). After the required service is found, the service profile is useless, and the service requester will use “Service Model” and “Service Grounding” to invoke and monitor that service.

4.2. Ontology Approach for Context Model

Context is any information that can be used to characterize the status of various entities in the space [26]. In our system infrastructure and preferred smart space, context can be defined as the information that characterizes the identity and attributes of people, pervasive devices, proxy devices, and Grid services. The information includes the performance and status of proxy devices, the Grid services accessible through the proxy device, the function of various pervasive devices, the relationship between pervasive devices and proxy devices, and so on. It may also involve the surrounding environment of devices, such as networking capability, because these conditions are able to change the interaction between devices. A well defined model is a key integrator of the context for the context-aware system.

Ontology refers to the specification a conceptualization of a knowledge domain [27]. It is a group of controlled vocabularies that describe objects and the relations between them to express something meaningful within a specified interest domain. Historically, there are several context modeling approaches [28], such as key-value model, markup scheme models, graphical models, object-oriented models, and logic-based models. However, using ontology to model contexts for an ubiquitous computing environment offers more advantages.

We have developed a style of ontology for modeling the context of the system infrastructure with the OWL language. In order to provide the universality of the ontology, we divide the ontologies into two distinctive but related parts: the initial ontology and the extended ontology. The initial ontology defines a set of fundamental concepts and general vocabularies, while the extended ontology, inherited from the initial ontology, provides additional concepts and vocabularies to support the specific application. Figure 2 shows the diagram of the ontologies and their relations.

The initial ontology consists of the vocabularies for the core concept that are associated with the entities in the system infrastructure. Five classes of objects are identified, “User”, “Device”, “Location”, “Event”, and “Restriction”, which appear to cover most of the basic elements for various applications. The “User” class defines the most general attributes of a person. In the device-proxy-Grid system infrastructure, the devices (the service consumers) belong to the “User” class, because actually it is the person behind the device who requires to access various services, and those portal devices can be considered as the agents of real users. We adopt the mechanism of binding properties between the person and his/her portable device, so that each portable device requiring to access services has the individual personal information, such as the name, the email address, the accessing password and so on. The status attributes such as the location of the user or current activity can also be ac-

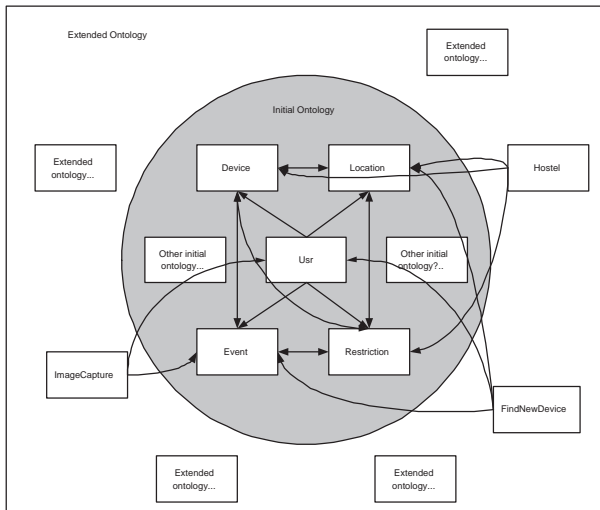


Figure 2. Outline Ontology.

quired through querying with other ontology classes.

The “Device” class characterizes the attributes of various types computing entities. It defines standard vocabularies for describing the information and profile of the device, such as the device ID, the location of the proxy device, the network address, listening port for the client request, the accessible Grid service, the software executing environment and so on. The “Location” class is designed to support context querying and context reasoning about the spatial relation among various types of pervasive devices. Generally, it has two subclass, “indoorLocation” and “outdoorLocation”. For a given application, the location is a realization of the abstract “Location” class. It might be an office, a lab room, a demo room, or a meeting room. Each device has the property about its location, and the context reasoner may reason that a new device should be integrated into the relevant space according to its location property.

The above three classes, “User”, “Device”, and “Location” are the classes defining real-world objects. The remaining two classes, “Event” and “Restriction”, specify the conceptual objects in the digital world. The “Event” class provides the vocabularies about the user’s request, which has a set of property values, including the requester, the receiver, and the location of the event. The “Restriction” defines a set of rules for permitting and guiding the users’ specific action. Combined with the “Event” class, it provides a feasible approach for security and privacy issue, enabling authenticated users to access the required service and hiding services from unauthenticated users.

The “Restriction” class represents all restrictions for an appointed user, with the use of the “restrictionOf” property. Each “Restriction” class have “permit” and “forbid” property, describing which style of events are permitted to the

user, and which styles of event are forbidden to the user. The system administrator defines the security level for every event (service) provided in this system, which is transferred to the security and privacy protection of the system. After registering, every event’s security level will be informed to the context reasoner. Hence, when new users enter the service space, their personal informations decide their secure accessing level during the authentication process. The context reasoner then is able to expose the right services to the authenticated users by comparing to the event’s security level and users’ accessing level.

4.3. Context Infrastructure

Based on the context model discussed in the above section, a context infrastructure can be constructed under the basic device-proxy-Grid system architecture. The context infrastructure enables applications to retrieve any contexts within the system and supports the inference of high-level contexts. The prerequisite of realizing such a infrastructure is the presence of a context reasoner, which could infer abstract high-level contexts from basic low-level contexts or raw sensor information. The context reasoner can be developed from the proxy device by integrating general purpose reasoning engines and application-based heuristic rules into the proxy, because the context reasoner requires a resource-rich computing platform. The responsibility of the context reasoner includes acquiring the context information from other computing entities in the space, answering the context querying, reasoning high-level context information that cannot obtain from the space, and determining the right service to be accessed by the authenticated user.

5. Current Implementation and Future Work

In this stage, we have implemented a basic device-proxy-Grid system infrastructure, and demonstrated that it is valuable for offloading portable device’s complex tasks to Grid services. For our experiment, the proxy platform is a desktop with a 3.00GHz P4 processor and dual-channel 1GB RAM. The portable device is a HP iPAQ hx2790 running Windows CE OS. The iPAQ hx2790 has Intel PXA270 624 MHz high-performance processor, and 256 MB total memory (192 MB ROM and 64 MB SDRAM). It connects to the LAN via its integrated 802.11b high speed wireless network card. We use another small server to simulate a simple Grid, by installing OMII-server and deploying required applications on it. For all the demonstration, the portable device, the proxy, and the “mini Grid” are connected directly to the ECS department LAN.

We consider the proxy service to be a typical web service and use Tomcat as the selected service platform. Tomcat has been installed on the Linux OS of the proxy as the service

container. Java, Jsp, HTML, and other mark-up languages are used to develop user interface. We assume each portable device has a web explorer so that it could view, access, invoke, and monitor proxy services conveniently. A proxy device is able to be accessed by a number of pervasive devices at the same time. In order to enable these pervasive devices not to affect each other, we adopt virtual machine technology. Each pervasive device requesting complex tasks will be assigned a virtual server, which provides considerable isolation of different applications under a single physical platform.

Some of simple applications have been deployed on the device-proxy-Grid system, such as data mining, which could be the portable device asking proxy to download a couple of files from Grid, performing an MD5 message digest operation on each file and outputting the resulting to the portable device. The OMITestApp application is another system testing example. Portable devices send single or multiple input files to the proxy, which are transferred to the Grid. The Grid will sort the words within the input files, and produces an equivalent number of sorted output files to the proxy. Except these simple demo application, we have implemented a complex computing-intensive application on the system. Calculating Cauchy Horizons is a complicated physical problem. The application takes three coordinates representing a point in this Cauchy space and then calculates the positive z-position, where the horizon is based, using a mathematical method. This result is output into a file named as the fourth parameter. In the system, we input several points' coordinates and through portable device randomly, and transfer them to the proxy. Because calculating z-position is a time-consuming and resource-consuming process, the proxy has to ask the Grid for help. After getting the results from the Grid, the proxy will build a 3D surface graph to display the Cauchy Horizon from the portable device.

The initial ontologies and parts of extended ontologies have been designed. In the future, new elements require to be added into the current ontology version based on new possible applications. For example, to represent the goals, plans, and beliefs of various computing entities, a new class "Desire" is required. We will also prototype different applications based on our system infrastructure to demonstrate the usefulness of the designed ontologies because it is difficult to develop methodologies to evaluate the success of a designed ontology. Just as we have discussed in the above section, the key of realizing a context-aware system infrastructure is the context reasoner. At present, we are adding general-purpose reasoning engines and some pre-defined rules to the proxy device so that it could be simple context reasoner. Raw information wrapper, context aggregator, and context query engine, which are also the required components of a system infrastructure, will be developed in

the future.

6. Conclusion

Pervasive computing devices are gradually becoming increasingly prevalent in our everyday life, which enables the users in the physical world to interact with the digital world. Grid computing provides a vision of accessing computational resources automatically on demand to deliver the services required with the appropriate quality. Semantic Web technologies enable automation and interoperability, which are now being adopted within the practice of Grid computing. It is the meeting of Semantic Grid and the pervasive devices that will enable us to step into a new concept "ambient intelligence", the notion of intelligence in the surrounding environment supporting the activities and interactions of the users [6].

Realizing such a vision of integrating pervasive devices into the Grid services with the support of Semantic Web technologies requires to meet several problems. In this paper, we have presented a system infrastructure that enables general-purpose pervasive devices to interact with various Grid services. This system infrastructure not only provides new opportunities that allow the complicated resource-intensive applications to be executed on the resource-constrained pervasive device, but also saves energy, storage space, memory capability, and processing cycles of the pervasive device, further reducing their size, weight, and cost. We have also designed a context model with the initial ontology approach to address the issue of realising the autonomic behaviour and knowledge sharing. Although it is no doubt that there are a number of challenges required to be overcome before the long term vision of bringing Grid systems into the world of Ambient Intelligence comes into being, we believe the system infrastructure that we are engaged in building must be the first steps to realize a variety of computing models (e.g. Grid computing) in the physical world.

References

- [1] Hai Zhuge. Resource space grid: Model, method, and platform. *Concurrency and Computation: Practice and Experience of Wiley InterScience* (www.interscience.wiley.com), pages 1385–1413, September 2004.
- [2] M.Satyanaayanan. Swiss army knife or wallet. *IEEE Pervasive Computing*, 4(2):2–3, April 2005.
- [3] David De Roure and James A.Hendler. E-science: The grid and the semantic web. *IEEE Intelligent System*, pages 65–71, Jan/Feb 2004.

- [4] David De Roure, Nicholas R. Jennings, and Nigel R. Shadbolt. The semantic grid: Past, present and future. *Proceedings of IEEE*, 93(3):669–681, March 2005.
- [5] N. Davies, A. Friday, and O. Storz. Exploring the grid’s potential for ubiquitous computing. *Exploring the Grid’s Potential for Ubiquitous Computing*, 3(2):74–75, March 2004.
- [6] David De Roure, Tony Hey, and Anne E. Trefethen. A global e-infrastructure for e-science - a step on the road to ambient intelligence. *True Visions: Tales on the Realization of Ambient Intelligence*, 2006.
- [7] M.Satynanayanan. Pervasive computing: Vision and challenges. *IEEE Personal Communication*, 8(4):10–17, 2001.
- [8] Sachin Goyal and John Carter. A lightweight secure cyber foraging infrastructure for resource-constrained devices. In *Sixth IEEE Workshop on Mobile Computing Systems and Applications*, pages 186–195, 2004.
- [9] Ya-Yunn Suu and Jason Flinn. Slingshot: Deploying stateful services in wireless hotspots. In *the 3rd Annual Conference on Mobile Systems, Applications, and Services*, pages 79–92, June 2005.
- [10] Feng Tao David E. Millard, Arouna Woukeu. Experience with writing grid clients for mobile devices. In *1st International ELEGI Conference*, 2005.
- [11] Bhagyavati and S.Kurkovsky. Wireless grid enables ubiquitous computing. In *the International Conference on Parallel and Distributed Computing Systems*, August 2003.
- [12] Bhagyvatim Stan Kurkovsky and Mei Yang. Medeling a grid-based problem solving environment for mobile devices. In *the International Conference on Information Technology: Coding and Computing*, pages 05–07, April 2004.
- [13] Anjum Shehzad Maria Riaz, Sadd Liaquat Kiani and Sungyuong Lee. Bringing handhelds to the grid resourcefully: A surrogate middleware approach. In *ICCSA 2005*, pages LNCS 3481 1096–1105, 2005.
- [14] Sungyoung Lee Taewoong Jeon Saad Liaquat Kiani, Maria Riza and Hagbae Kim. Grid access middleware for handheld devices. In *Europe Grid Computing 2005*, pages LNCS 3470 1002–1011, 2005.
- [15] Ali Sajjad Sungyoung Lee Hassan Jameel, Umar Kalim and Taewoong Jeon. Mobile-to-grid middleware: Bridging the gap between mobile and grid environments. In *Europe Grid Computing 2005*, pages LNCS 3470 932–941, 2005.
- [16] George Roussos, Andy J.Marsh, and Stavroula Maglavera. Enabling pervasive computing with smart phones. *IEEE Pervasive Computing*, 4(2):20–27, 2005.
- [17] Liviu Iftode, Cristian borcea, Nishkam Ravi, Porlin Kang, and Peng Zhou. Smart phone: An embedded system for universal interactions. In *Tenth International Workshop on Future Trends in Distributed Computing Systems*, pages 88–94, 2004.
- [18] Jarek Gawor Thomas Sandholm. Globus toolkit3 core - a grid service container framework, last accessed Feb. 2005.
- [19] Omii-2, last accessed Feb. 2005.
- [20] GGF. Open grid service infrastructure, version 1.0, last accessed Feb. 2005.
- [21] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. The anatomy of a context-aware application. In *Proceeding of the ACM/IEEE MobiCom*, August 1999.
- [22] A. K. Dey and G. D. Abowd. Towards a better understanding of context and context-awareness. In *Workshop on The What, Who, Where, When, and How of Context-Awareness, part of Conference on Human Factors in Computing Systems*, Chicago 2000.
- [23] Feng Zhu and Matt W. Mutka. Service discovery in pervasive computing environments. *IEEE Pervasive Computer*, 4(4):81–90, 2005.
- [24] Jerry Hobbs etc. David Martin, Mark Burstein. Owl-s: Semantic markup for web services, 22 November 2004.
- [25] C.Welty M.Smith and D.McGuinness. Web ontology language guide version 1, 2003.
- [26] Anind K. Dey. Understanding and using context. *Personal And Ubiquitous Computing*, 5(1), 2001.
- [27] Mike Uschold and Michael Gruniger. Ontologies: Principles, methods, and applications. *Knowledge Engineering Review*, 11(2):93–155, 1996.
- [28] Thomas Strang and Claudia Linnhoff-Popien. A context modeling survey. In *First International Workshop on Advanced Context Modelling, Reasoning and Management*, pages 34–39, September 2004.