Project no. IST-033576

XtreemOS

Integrated Project
BUILDING AND PROMOTING A LINUX-BASED OPERATING SYSTEM TO SUPPORT VIRTUAL ORGANIZATIONS FOR NEXT GENERATION GRIDS

Requirements and Specification of Basic Services for Mobile Devices
D3.6.1

Due date of deliverable: November 30th, 2007
Actual submission date: November 30th, 2007

Start date of project: June 1^{st} 2006

Type: Deliverable
WP number: WP3.6
Task number: T3.6.1

Responsible institution: Telefónica I+D
Editor & and editor's address: Luis Pablo Prieto
Telefónica I+D
Parque Tecnológico de Boecillo
47151 Boecillo (Valladolid)
SPAIN

Version 1.0 / Last edited by Luis Pablo Prieto / November 29th, 2007

Project co-funded by the European Commission within the Sixth Framework Programme		
Dissemination Level		
PU	Public	√
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Revision history:

Version	Date	Authors	Institution	Section affected, comments
0.1	2/8/07	Luis Pablo Prieto	TID	Initial template
0.2	2/9/07	Manuel Martin	TID	Draft of current grid services
0.21	1/10/07	Luis Pablo Prieto	TID	Revision of current grid services
0.3	2/10/07	Toni Cortes	BSC	Draft of XtreemOS-MD services
0.4	2/10/07	Luis Pablo Prieto	TID	Consolidation of current and XtreemOS-MD services
0.5	4/10/07	Manuel Martin	TID	Use cases chapter
0.6	5/10/07	Luis Pablo Prieto	TID	Completion of XtreemOS-MD services chapter
0.7	9/10/07	TID Team	TID	First draft of requirements chapter
0.71	11/10/07	Toni Cortes	BSC	Contributions to requirements chapter
0.72	15/10/07	Luis Pablo Prieto	TID	Consolidation of requirements chapter
0.8	16/10/07	TID Team	TID	First draft of specifications chapter
0.9	31/10/07	TID Team	TID	Introduction, future work, glossary and minor corrections
1.0	29/11/07	TID Team	TID	Incorporated reviewer's comments and executive summary. Ready for submission

Reviewers:
Julian Gallop (STFC), Bernd Scheuermann (SAP)

Tasks related to this deliverable:

Task No.	Task description	Partners involved°
T3.6.1	Requirements and specification of basic services for mobile devices (XtreemOS-G for MD/PDA)	TID*, BSC

°This task list may not be equivalent to the list of partners contributing as authors to the deliverable
*Task leader

Executive Summary

XtreemOS not only aims at integrating grid funcionalities into GNU/Linux, but also intends to address the *hetereogeneity* of current computing, from clusters to mobile devices (MDs). The main challenge addressed in this document (and in the workpackage of which it is part) is to adapt the grid services provided by XtreemOS (grouped into what is called the XtreemOS-G layer) to the limited environment of mobile devices (PDAs, in this first phase). Taking into account that mobile devices currently outnumber PCs, and that they are continuous companions of end users, addressing this heterogeneity can be a great advantage for XtreemOS, extending the reach of grids nearer to users, and using its transparent design to open grid computing to the mass market.

This approach of extending the Grid to mobile devices, creating what has been called the Mobile Grid, is a relatively new field of research. At first, only proxy-based solutions were proposed, with dedicated nodes doing translation of grid operations to and from MDs. But more recently, other approaches try to achieve a more integrated view of MDs in the Grid, like OGSI.NET or EU-funded Akogrimo project. This concept has also reached more mainstream areas of grid research, appearing, for example, in NessiGrid's Strategic Research Agenda (SRA). Up to now, this approach has reaped mixed results, mainly because of development issues (like the difficulty of building software for embedded devices), which has been one of the main obstacles to these research initiatives.

XtreemOS for mobile devices (XtreemOS-MD) approaches the issue of mobile grids with this integrated stance, following its main design concepts of *scalability* and *transparency*. By promoting mobile devices to first-class citizens in the Grid, it will scale better than proxy-related solutions in scenarios where potentially thousands of mobile nodes can be part of a virtual organization (VO). By integrating grid mechanisms into the Linux OS (which is already one of the most popular operating systems in mobile devices), and by using Linux paradigms and interfaces, the grid functionality can be provided to users in a more transparent way.

In the basic version of XtreemOS-MD, which is specified in this document, functionality provided to end users will be mostly one of client access to grid services like XtreemOS's Application Execution Management (AEM) and XtreemFS grid filesystem. These services will be offered with the same level of security as the other flavours of XtreemOS, by using the same VO and security infrastructure as any other XtreemOS node. These services not only will provide mobile users with the ability to launch, monitor and manage grid jobs, or to access grid files from mobile devices in a secure manner, but they will also pave the way for more advanced mobile grid services that will be developed in the advanced version of XtreemOS-MD, exploiting the full potential of mobile devices for the benefit of the Grid (mostly by extending the concept of "grid resource" to other assets where mobile devices excel: mobility, context, user knowledge, etc).

Contents

List of Figures

Glossary

3G	3rd Generation Mobile Tecnology
A4C	Authentication, Authorization, Accounting, Auditing, Charging
AC	Alternating Current
API	Application Programming Interface
AEM	Application Execution Management
AMS	Account Mapping Service
AS	Attribute Service
CDA	Credential Distribution Authority
FUSE	Filesystem in Userspace
GPS	Global Positioning System
GRAM	Grid Resource Allocation Manager
GSI	Grid Security Infrastructure
GSM	Global System for Mobile communications
GridFTP	FTP extension for the Grid
IS	Identity Service
J2ME	Java Platform 2 Micro Edition
kSOAP	SOAP web service client library for constrained Java environments such as Applets or J2ME applications
kXML	little XML parser with a small footprint
LNCC	Laboratorio Nacional de Computaçao Cientifica
MANET	Mobile Ad-hoc Networks

MIPv6	Mobile Internet Protocol v6
MRC	Metadata and Replica Catalog
OGSI	Open Grid Service Infrastructure
OSD	Object Storage Device
OSS	Open-Source Software
PDA	Personal Digital Assistant
POSIX	Portable Operating System Interface
PSTN	Public Switched Telephone Network
PUX-Rio	Pontificia Universidade Catolica do Rio de Janeiro
QoS	Quality of Service
RMS	Replica Management System
RPC	Remote Procedure Call
SAGA	Simple API for Grid Applications
SLA	Service Level Agreement
SMS	Short Message Service
SSL	Secure Sockets Layer
VO	Virtual Organization
VoIP	Voice IP
VOM	Virtual Organization Manager
VOPS	Virtual Organization Policy Service
WSRF	Web Services Resource Framework
XtreemOS-F	XtreemOS Foundation Layer
XtreemFS	XtreemOS FileSystem
XtreemOS-MD	XtreemOS for Mobile Devices
XOSD	XtreemOS Daemon
X-VOMS	XtreemOS VO Membership Service

Chapter 1

Introduction

It is not very often that a new grid operating system comes into existence, and it is even rarer that the operating system addresses non-PC architectures. But that is the case of XtreemOS, which not only aims at integrating grid functionalities into the GNU/Linux OS, but also intends to address the heterogeneity of current computing, from clusters to mobile devices.

The main objective of this document is to define the requirements and specifications of the grid services as they will be offered by the mobile devices flavour of XtreemOS, the so-called XtreemOS-G for MDs[1]. Please note that this document only covers the *basic version* of XtreemOS-MD, with more features and support for mobile phones coming in the advanced version of the operating system.

For more information about the rest of the XtreemOS-MD flavour (namely, the XtreemOS-F layer), please refer to the documentation produced by WP2.3 [31, 28, 23].

1.1 Methodology

Our approach has been to first collect a set of requirements that the grid services in XtreemOS-MD should meet. This set of requiremens comes mainly from a variety grid applications selected by XtreemOS to evaluate its results (D4.2.3 [20])), and also from the state of the art in grid systems that incorporate mobile devices like PDAs and mobile phones. But another set of requirements also come from a number of grid use cases involving mobile devices, which have been devised to clarify the scope of this MD flavour and to prioritize among the different possible grid functionalities offered by XtreemOS-MD.

All these requirements have been then analysed, to make a first specification of the grid services for XtreemOS-MD, describing *how* those requirements will be met.

[1] Being XtreemOS-F (Foundation layer) the other main layer of XtreemOS software, this one more low-level, near the kernel of the operating system.

1.2 Document structure

This document is structured as follows:

In chapter 2, the current state of the art in grid services involving mobile devices is briefly described, taking a look at the different approaches taken by past research on the subject, their pros and cons, and concluding with the path that has been chosen for XtreemOS.

Next, chapter 3 provides a number of use cases of grid services involving mobile devices, prioritized in order to set the scope of the XtreemOS-MD distribution and help in the roadmapping of its functionalities.

Once the use cases have been defined, chapter 4 gives a first, high-level overview of the grid services to be included in the basic version of XtreemOS-MD.

Chapter 5 contains the concrete list of requirements for each of those services in a mobile device, which is then completed with a list of specifications to fulfill those requirements in chapter 6.

Finally, chapter 7 summarizes the results of the document, and chapter 8 anticipates the next steps to be followed in the task of constructing these services for mobile devices, namely their design and implementation.

Chapter 2

Current Mobile Grid Services

It is a well known fact that the number and types of mobile electronic devices worldwide is exploding: from laptops to personal media players, through the ever-present mobile phone, there are literally billions of electronic devices that can be connected to a network (just in 2006, more than 1 billion mobile phones were sold). Thus, it was only a matter of time that someone tries to harness this source of untapped power, be it for using its (rather limited) computing and data capabilities, or for accessing grid services from any place, at any time. In this chapter, we will analyze current approaches to the task of "mobilizing" grid services. In this field, there are two main approaches:

- Using the mobile nodes just as clients or **access points** to the Grid.
- Using the mobile devices as conventional grid nodes where jobs (or parts of them) can be executed, thus offering **mobile grid services**.

2.1 Mobile Access to the Grid

There exist several ways to access the Grid from mobile devices. Some of the most relevant ones are described below.

2.1.1 Proxy-based solutions

One of the first approaches to be proposed to access the Grid from mobile devices (MDs) was to use proxy-based solutions, as can be seen in various academic work like [9, 7] or [11], and also in other European grid projects like GridLab [8]. Figure 2.1 depicts an example of proxy-based architecture, from the GridLab project.

Proxy-based solutions use what is called a proxy server (or gateway): a special (fixed) grid node, part of the core grid infrastructure, whose role is to handle and translate requests from mobile nodes, and forward them to the Grid, adapting the responses to the limited interfaces of mobile devices. Thus, the MD's requests

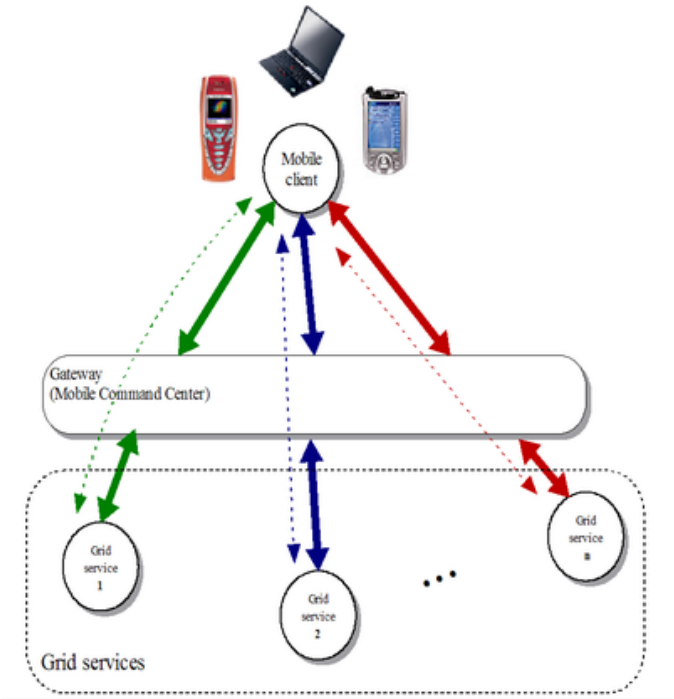

Figure 2.1: GridLab mobile access

(often using ad-hoc protocols) are handled by the proxy server, which invokes the necessary grid services and sends the response back to the MD.

Initially, all MDs connected to the same proxy are nodes of the same Grid. However, virtualisation techniques (putting several virtual machines in the proxy) allow for isolation among MDs connected to the same proxy server.

Regarding the technologies used for the implementations of these solutions, Java programming language (more concretely, J2ME) is often used to implement the clients on the MDs, although web portals are also known to be used for communicating with the proxy server.

The advantages of using this kind of solution is that one can decrease greatly the complexity of the client software to be run on the mobile devices, at the expense of increasing the complexity of the proxy's software. The increasing capabilities of mobile devices make this advantage less and less important, while its disadvantages (the proxy as a single point of failure and its scalability issues when the number of MDs grow) continue to be as important as ever.

2.1.2 Mobile access to Grid web services

Opposite to traditional grid access through a proxy server or gateway, there have been a number of works trying to probe other kind of solutions [12, 15]. Many tried to provide client (only) access to grid web services directly, on the basis of the following motivations [10]:

- Current mobile devices have sufficient resources (computational power and network connectivity) to allow direct access to grid web services.
- Compared to existing proxy-based approaches the usage of web services on the mobile device has only a small and acceptable impact on performance.
- Using standard grid web services allows for better scalability, interoperability and easier implementation of mobile grid clients.

Technologies used in this kind of approach include WSRF, GSI, J2ME, J2ME Web Services API, kSOAP/kXML...

It is important to note that several of these initiatives end up using gateway solutions, not because of their higher efficiency, but on the grounds of the difficulty of developing mobile clients and applications. This fact shows that a good developer environment/framework and a coherent API, are probably the major obstacles in the implementation of this kind of solutions.

2.2 Mobile Grid Services

The next logical step in the process of mobilization of the Grid would be considering mobile devices themselves as conventional grid nodes where grid services can be executed. This is a much less trodden path, that only a handful of researchers and projects have tried with mixed results.

2.2.1 Mobile OGSI.NET

Mobile OGSI.NET extends an implementation of Grid computing, OGSI.NET (which is a precedent of the more modern WSRF.NET), to mobile devices, on the basis of the paradigm that somehow the resource limited devices can collectively deliver the quality of service needed by the end user [2]. In fact, it allows for process execution and user-driven migration between MD's when a certain event occurs, e.g. the battery is very low or the mobile host is about to fail.

Mobile OGSI.NET was implemented on top of the Microsoft PocketPC 2003 operating system and the .NET Compact Framework and at the time only supported PDAs, not mobile phones. Its architecture consists of the following three main layers [2]:

- The Monash University Mobile Web Server, which handles endpoint to endpoint message reception and transmission.

- The Grid Services Module, which handles grid services message parsing and multiplexes messages to the appropriate grid service.
- The Grid Services, which handles application logic and processing.

In practice, just basic functionality was implemented; other important grid features like security or notifications have never been available. In fact, it seems that the interest in the project has waned, as not much beyond the work described in [2] has been done on the subject. Furthermore, the next logical step has not been attempted, which would be updating this initiative to WSRF.NET, creating a "Mobile WSRF.NET". No interoperability tests of Mobile OGSI.NET with conventional grids have been conducted either, to out knowledge.

2.2.2 MoGrid

MoGrid [4, 5], a middleware developed by researchers at the Laboratório Nacional de Computação Científica (LNCC) and the Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), in Brazil, aims to support mobile grid services, by providing means for resource discovery and task distribution in mobile ad-hoc networks.

To that end, peer-to-peer techniques are used for resource discovery, and job submission is done through standard protocols like GridFTP. It also sports a so-called "transparency layer", in order to isolate the connectivity problems typical in wireless scenarios and make them transparent to the grid applications and services. In its current form, the architecture does not provide the necessary security mechanisms for establishing privacy or trust. Also, it lacks any methods for interoperating with conventional (wired) grids, although plans for proxy-like interoperation with Globus have been mentioned.

Although this project shows some interesting concepts, its current status is uncertain. Some reference implementations were made in J2ME in 2005-2006, for publication purposes, specially in the peer-to-peer resource discovery area, and it seems that some amount of research is still going on in the subject of resource discovery and scheduling [6].

2.2.3 Akogrimo

Akogrimo (Access to Knowledge through the Grid in a mobile World) is a project funded by the EC under the FP6-IST programme that ran from July 2004 until November 2007. It aims at defining and realizing a web services-based Mobile Grid Architecture ensuring the viability of new business models developed on it. The participants of this collaborative platform can be mobile [1, 16].

Akogrimo focuses on a number of mobility aspects that can be integrated into grid environments, including:

- Context awareness (including presence information, location data, etc)
- Device mobility

- Session mobility (i.e. moving a session between two devices)
- Temporary disconnection of mobile devices
- Dynamic VOs
- Dynamic quality of service (QoS)
- Grid-controlled SIP calls

And these capabilities are shown through a series of implementations of demonstration applications, like for example eHealth applications (patient monitoring and emergency response) or disaster handling and crisis management scenarios.

Akogrimo contemplates several kinds of services, most of them using the WSRF standard:

Network services that communicate context information with higher layers and execute service needs from higher layers. This includes: (M)IPv6 components, Network QoS provisioning and Network management services.

Network middleware that glues the Grid with a commercial mobile network. Auditing is another functionality of middleware in order to keep track of potential SLA violation. Services in this category include: Service discovery, Session management, Context Management and A4C (Authentication, Authorization, Accounting, Auditing, Charging).

Grid infrastructure services: typical grid services adapted to the commercial mobile network: Data Management, Execution Management, SLA Management or Policy Management.

Application support services that support domain-independent grid applications and allow for cross layer cooperation. This services include: VO Manager, Workflow enactment, SLA Definition and Operational VO Brokers.

In general, Akogrimo is a high-level architecture with little regard to the underlying grid infrastructure. In fact, its current implementation uses Globus and WSRF.NET as its main middlewares, although using other middlewares underneath Akogrimo should be possible (although the complexity of the architecture and services implemented over Globus makes it a rather daunting task).

The Akogrimo project is probably the only one big-scale effort into making the mobile Grid a reality. It has developed a very complete architecture, paying special attention to the industrial viability of their grid models (for telecom operators, service providers, etc.). This high-level approach and their election of Globus as the middleware makes Akogrimo share some of its weaknesses (e.g. setting up a working grid is a complex task). Also, although mobility and mobile devices are at the focus of the project, the actual implementation only covers laptops, not PDAs or mobile phones (probably because of the middleware, which is not suited for mobile devices).

2.3 Conclusions

An analysis of the approaches presented in this chapter allows us to reach some conclusions:

- The two main issues (at least from an infrastructure point of view) in mobile grids are **mobility support** and **context awareness**: The first one enables ubiquitous access to the grid as well as provision of mobile nodes resources; the second allows an appropriate management of grid resources according to real time execution environments. These conclusions are also reached by recent discussions on this subject [14]. This two concepts should be addressed (most probably in that order) by the XtreemOS-F layer of the mobile device flavour, but should also be present among the grid services that are implemented over that foundation layer.
- Although the most common approach is to provide client access to grid services from mobile devices, current research is now moving into the next step, which is to provide mobile grid services in the mobile devices themselves, and its extension to wireless and ad-hoc infrastructures. Thus, the best path for XtreemOS-MD seems to provide, in a first phase, **efficient client access** to XtreemOS grid services from MDs, and later on research on extending these **services to mobile devices** and **ad-hoc infrastructures**.

Chapter 3

Mobile Grid Use Cases

In order to ascertain which grid services should be implemented in the basic version of XtreemOS flavour for mobile devices, and specially which kind of functionality is expected of each of them, we have gathered a number of mobile grid use cases. These use cases and their comparative relevance to XtreemOS, will also help us in prioritizing the implementation work, and in anticipating the functionalities that the advanced version of XtreemOS-MD should include.

These use cases have been extracted from the state of the art of the research in the area of mobile grids, including other EU grid projects like Akogrimo, NessiGrid, etc. Some of them also have raised in conversations and meetings, not only inside WP3.6, but among the whole consortium. Others are raised from the applications that XtreemOS provides in WP4.2 to assess the success of this grid operating system. And yet a number of them have been extracted by extrapolating the application of grids to known or innovative mobile services.

Moreover, a priority has been assigned to each of them, attending mainly to **dependency** criteria (i.e. if a use case is required before another one can be implemented, it is given a higher priority), but also on the grounds of the barely minimum **expected functionality** on a current mobile grid system and its adequateness to the XtreemOS philosophy and **design principles** of transparency and scalability.

When describing these use cases, we have found two broad groups of use cases, distinguished by what we could call "grid-awareness":

Grid-Aware: in the first group, grid users (e.g. people from corporations or academic institutions that usually work with grid technology) gain mobile access to the XtreemOS grid. As transparency is one of XtreemOS's main design principles, this access will be done in a more transparent way than with conventional grid middleware, but users will be conscious, to a certain extent, that there are concepts like jobs, certificates and so on, running in the background.

Grid-Transparent: in the second group of use cases, users that have no knowledge or interest in the underlying grid technology experience the benefits of

XtreemOS in a completely transparent way. These cases normally require functionalities that must be provided to fulfill the grid-aware use cases (e.g. basic grid infrastructures) and thus, they often have lower priorities than the previous ones. But they are nonetheless very interesting, for they showcase mass market application of grid technology.

So, to better understand these use cases, we have grouped them according to this grid-awareness criterium.

3.1 Grid-Aware Use Cases

3.1.1 Login to the Grid

Description

This is the prerequisite to any other Grid use case. A user enters his passphrase (either at the device's power on or using a specific "login application"), which authenticates the user with the CDA (Credential Distribution Authority) and an XOS-Cert certificate is obtained. From then on, the user is enabled to operate into the Grid according to his role and permissions.

Optionally, if the user has a home directory in XtreemFS, it is mounted.

Grid services required

- Access to a CDA.
- [Optional] Access to XtreemFS.

Importance

Basic.

3.1.2 Job management

Description

Once the user is logged in, he selects the desired job description file (which can be on the XtreemFS) and submits it to the Grid.

During execution, the user will be notified of the job's main events (changes in job status, failures, etc.), and he will be able to operate on the job (information request, pause, restart, cancel, etc.).

Grid services required

- Access to AEM: Client interface to contact an XOSD/Job Manager in a remote site.

- [Optional] Access to XtreemFS (if the job description file is in the XtreemFS).

Importance

Basic.

3.1.3 VO administration

Description

A user logs into the Grid from his MD as VO administrator and performs an administrative operation (add/delete user, create an VO subgroup, set resource policies, etc.). First, the user is authenticated by the CDA, which delivers a XOS-Cert with permissions for VO administration. Then, application(s) for VO management are executed with those credentials. These applications would contact the VO/security servers (X-VOMS, Identity...) for performing those management operations.

Grid services required

- XtreemOS VO Membership Service (X-VOMS)
- Access to the/an Identity service
- Access to the/a Credential Distribution Agency (CDA)

Importance

Advanced.

3.1.4 Resource management

Description

The user could define what resources to share to the Grid as well as monitor its usage (e.g. current status, who is using them, how are people using them, who is allowed to use them, statistics, etc.).

Grid services required

- Client interface to contact an XOSD in a remote site.

Importance

Advanced.

3.1.5 Resource searching

Description

The user should be able to get a list of available resources to him. This search could be performed according to certain parameters given by the user (e.g. more than 50% CPU free).

Grid services required

- Client interface to contact an XOSD in a remote site.

Importance

Advanced.

3.1.6 Data management

Description

The user specifies where data will be stored, the pattern of access, privacy level, places where data can be stored, required level of consistency and coherence.

Grid services required

- Access to XtreemFS in client mode.

Importance

Advanced.

3.1.7 Session mobility

Description

Like the previous use case, this is another variation to any of the other ones. The idea is that the user can transfer an ongoing session to a nearby device that belongs to the same VO without interruption.

Grid services required

- Access to the AEM.
- Automatic detection of available nearby resources.
- Grid session transfer service.

Importance

Advanced.

3.2 Grid-Transparent Use Cases

3.2.1 Messaging application

Description

Two users connected to the Grid that belong to the same VO are running a XtreemOS-enabled messaging application. By using XtreemOS authentication methods, the users can be confident that the other users are who they claim to be, and the communications are done securely. It also has the advantage of using XtreemOS Single Sign On capabilities, so that users do not have to authenticate if they have already done it for another XtreemOS operation.

Once the authentication phase is over, the application acts as any other IM application: one of them (sender) wants to send an instant message to the other (receiver), who is online and available (ready to receive messages through the application). To do so, the sender opens a message window and types the message that will immediately be received by the receiver.

Optionally, configuration and preference files could be stored in the grid, which would mean that the user has the same addressbook, configuration and "look & feel", regardless of the device he is using.

Grid services required

- Access to a Presence/Information service (this functionality can also be provided by the application itself).
- [Optional] Access to XtreemFS.

Importance

Basic.

3.2.2 File sharing/distribution

Description

Once logged into the Grid, the user can store, search for and retrieve files (e.g. multimedia content) stored and shared by users of the same VO (provided he has the adequate permissions). A specific application or a standard file browser can be used to search for files (based on certain metadata), transfer files to/from XtreemFS, etc.

Grid services required

- Access to XtreemFS.
- File search service.

Importance

Basic.

3.2.3 Advanced voice recognition

Description

Once the user is logged into the Grid, he wants to enter a text (for an SMS, instant messaging, email, word processor, etc.) without having to bother about the tiny keyboard of the device. He opens the Voice Input application (or a similar button in the desired application), and starts speaking. The resulting wave file is sent to the Grid, analyzed by a powerful recognition Grid application and the resulting text is sent back to the user's device.

Other similar uses cases could involve face recognition, image recognition, etc.

Grid services required

- Access to AEM.
- [Optional] Access to XtreemFS.

Importance

Basic.

3.2.4 People as resources

Description

Once a user is logged into the Grid, he can receive requests to participate in Grid applications as a resource (e.g. to make a decision in a complex workflow, to take a photograph of a concrete location, etc.), mainly depending on the context.

At any time, if a Grid application needs it, the AEM can pose a request to the user in order to participate in the application by performing a certain task in the way that is needed.

Grid services required

- Context management service (location, user knowledge information, device capabilities).
- Access to the AEM.

Importance

Advanced.

3.2.5 Secure MANET's

Description

The user does not have direct access to the Grid, but does have his XtreemOS certificate stored in the MD. The user also has another XOS-enabled users in range (some of them could have direct access to the Grid).

The user's MD broadcasts a MANET establishment request. Its neighbour receives the request and both perform a mutual authentication, checking that both users are valid. Once a secure channel is established between the two devices, this second user sends the first user's MD the necessary information (routing table, IP address, etc.) to join the MANET. From then on, the user's MD starts operating in MANET mode securely, with the additional feature of being able to access the Grid (through one of its peer's Internet connection).

Grid services required

- Previous access to a CDA.
- MANET services in the MD's (e.g. routing protocol).

Importance

Optional.

3.2.6 Phone communications as resources

Description

A user connected to the Grid wants to establish a voice call with an external phone number (e.g. a PSTN number), but does not have phone line, and another node in the same VO does have it. When the user types the number in his VoIP application, it contacts the AEM requesting phone line resources. Once those resources are located, the call is established using VoIP up to the phone line node and using standard telephony (PSTN, GSM, IMS...) from then on.

In addition, the information about the call is properly accounted so that the user is billed for the cost of the call.

Optionally, mobile ad-hoc networks can be integrated into this kind of scenario, allowing users that currently have no connection to the Internet or the conventional telephonic network, to make voice calls with the intermediation of other XtreemOS nodes that do have that kind of connectivity.

Grid services required

- Access to the AEM in order to start the appropriate application in the phone Grid node.
- Call routing service.
- Access to an accounting service.
- [Optional] MANET services in the MD's.

Importance

Optional.

3.2.7 Multiplayer mobile games

Description

In multiplayer games with very high demands on computing power (which the mobile device cannot provide) and/or with large numbers of users, can be offered by using massive amounts of PC or cluster nodes to power games to be visualized from mobile devices.

Moreover, since users have to authenticate themselves with the "game VO" before entering, XtreemOS security (and specially accounting) services can be used to monitor and bill the player according to the usage.

Grid services required

- Access to CDA and Accounting service
- Access to AEM in client mode.
- Visualization services

Importance

Optional.

Chapter 4

XtreemOS-MD Services

After detailing the main use cases targeted by XtreemOS-MD, now the necessary grid services for the basic version of XtreemOS-MD can be safely derived. In this chapter, a brief description of the services is given, both for their standard form (for the PC flavour of XtreemOS), and also for the changes needed in the PDA flavour. Afterwards, a list of the complete requirements and specifications is given in chapters 5 and 6.

Please note that only "basic" use cases are covered in this document; "advanced" or "optional' use cases will be addressed in a later phase in the project.

4.1 Application execution management

One of the main features of any grid system is the ability to execute and manage jobs across the Grid, using an adequate set of available resources to do so.

4.1.1 Services in the standard version

As can be extracted from the XtreemOS deliverables on application execution management [18, 24, 26], the services needed to manage jobs and resources can be divided into three groups: job management services, resource management services, and global services.

Job management services

jController holds the job information and its three main goals are first, to ensure that the scheduling agreement between the job and the resources is accomplished, second, to validate the job is executing as expected, and third, to act as a gateway for the job. The jController holds most of the information associated to the job and it is the only service that has a global vision of it. It is the service that provides self-management to XtreemOS jobs.

jScheduler schedules one job. The jScheduler receives a pre-selection of resources from the jResMatching based on job resource requirements and in a second step it performs a negotiation with pre-selected resources in order to decide the final allocation. The jScheduler service is stateless and from one job scheduling to a next one no status is stored.

jMonitor collects information from all the processes of the job and adds it in a job basis. One of the goals of the jMonitor is to provide a monitoring service as powerful and flexible as possible, allowing advanced versions of XtreemOS to add new metrics without changing either the API or the system architecture.

jExecMng is a distributed service that implements methods for managing the execution of the job.

jEvent is in charge of re-distributing job events and performing the required action to send and receive signals.

jMigrator and jCheckpointing manage migration and checkpointing functionality guided by job requirements. During the job submission the user/application will specify migration and checkpointing hint that will be used as a basis for these services.

Resource management services

Negotiator Each resource has associated one Negotiator service that is aware of the local policy associated to the resource (one resource could be in more than one VO). The Negotiator is a key service in XtreemOS since it will integrate local resource policies with VO policies, being one of the main contributions of the project. Depending on the local policies the Negotiator of a resource will offer exclusive access to the resource, different scheduling policies, etc.

rMonitor is used to get dynamic information about resources, and to report the values to other services through push and pull mechanisms.

rAllocator performs the required actions when submitting the job to a resource and when adding or releasing resources. It will contact the VO Manager for updating resources accounting associated to the job.

Global services

JobDirectory is a distributed Information Service that maintains the list of jController addresses. Since each job is created by a grid user in the context of one VO, the JobDirectory services manages the association (jobID + jController_address + user credentials). This service will be accessed each time

the jController address associated to a jobID is required. It will include job access rights validation associated to user credentials. The JobDirectory is a key component in the AEM architecture because it is the only one that has a global list of jobID's in a VO (with the security information associated), being the only way to get the list of jobID's of a user in a VO.

4.1.2 Services in the MD section

Job management services

As we have seen in the use cases, the main objective of MD users is to be able to launch jobs, check their status and get information about available resources.

To offer all these services, we have two different options. The first one, and probably the simplest one, is just to offer XATI, the interface to contact the XOSD (daemon handling resources and jobs) that can be located in any other node. With this interface we could build all needed applications to submit jobs an check status of resources and jobs.

The second option would be to have a reduced version of the XOSD daemon. In this case only the following services should be part of the MD version of the daemon: jscheduler and potentially (the benefits have still to be studied) jController. The jscheduler would allow to submit jobs and put all scheduling and negotiation overhead in the mobile device. The jController of the submitted jobs from this device could also reside in this device (this service will have fault tolerance characteristics, thus residing in a MD will not be a technical problem). On the other hand, having the jController in the MD may not be very wise if the MD gets disconnected very frequently. Although it will continue working due to the fault tolerant characteristics, the potential benefits of having the jController near the user will become less clear due to its frequent disconnections.

Resource management services

As no jobs will be executed in he mobile device it does not make sense to have any of these services in the MD flavour. The only exception, and just for information issues, is the rMonitoring. The MD flavour could have a simplified version for the rMonitoring to allow the system to query information on this device, but not to try to allocate or run jobs on it (which also means that no information on running jobs needs to be available from this service).

Global services

Again, these services are much more global and does not make sense to have them running on a MD. It would mean an extra consumption of processor, bandwidth and battery with no clear gain for the MD user.

4.2 Data management

4.2.1 Services in the standard version

As can be extracted from the deliverables on data management [19, 32], the services needed to manage data are:

MRC (Metadata and replica catalog). This service is in charge of keeping track of where files are located and how they are placed among the OSDs (stripping, replication , etc.). In order to access a file, this will be the first service to contact, and once we have the real location, the communication will be done directly with the resources keeping the real data.

OSD (Object storage device). This service runs on any resource that exports files to XtreemFS. It know how to read/write the actual data and is also in charge of the consistency of the accesses.

RMS (Replica management system). This is an optimization service that will decide when, where and how replicas are created and/or removed.

Client library (Client library). Finally, any node that is able to access XtreemFS files will have this "library" that will redirect file operation to the right service (OSD or MRC). Currently in the standard flavour it is a FUSE module.

4.2.2 Services in the MD version

Regarding XtreemFS for the MD version, we only need a mechanisms to "mount" a XtreemFS volume and be able to access files for both reading and writing. This service, in the standard version, is managed by a FUSE module that receives all file operations and redirects them to the right service in XtreemFS (MRC, OSDs, and RMS) [19, 21]. For the mobile device, a similar library to redirect all these file operation to the XtreemFS services will need to be implemented.

On the other hand, allow users in the grid to access files located in the MD is not initially planned as a feature of the MD flavour. If these files are to be accessed from out of the device, they should be copied to a XtreemFS volume.

4.3 VO management and security services

As in any other grid system, XtreemOS makes use of a number of services that deal with grid-wide VO management, authentication, authorization and other security-related tasks, in order to ensure the integrity and confidentiality of communications, and to assure that the available resources are used by authorized personnel.

4.3.1 Services in the standard version

The standard flavour of XtreemOS comprises five main services, which are further described in the security deliverables and other XtreemOS publications [25, 30, 3]:

XtreemOS VO Membership Service (X-VOMS) validates the VO membership of a user who initiates a grid request from a Linux terminal. X-VOMS looks up user information, such as identity and attributes, in a store with all the VO members information.

Identity Service (IS) generates and manages globally unique VO IDs and user IDs. Assuming that resource nodes trust the VO manager and have preinstalled the manager's root CA certificate, nodes can verify users' authenticity based on the XtreemOS certificate (XOS-Cert).

Attribute Service (AS) provides users with VO attributes. These attributes are used, for example, to allow AEM services to check against VO policies during resource selection; to perform access control to resources and XtreemFS files; to enforce system-level resource usage control; and to allow nodes to map global IDs to node-level system UIDs/GIDs.

Credential Distribution Authority (CDA) issues users with VO security credentials for accessing grid-wide services and resources. X.509 certificates are used to communicate the XOS-Cert. The CDA holds the public/private key pair of the VO manager so that it can issue signed XOS-Creds (the short-lived credentials used to access the rest of XtreemOS services).

Virtual Organization Policy Service (VOPS) provides policy related services, such as policy information and decision points to the VO Manager, so that VO level access control can not only be enforced at nodes but also at VO-level by the VO Manager. This integration of policy decisions in VO management allows for VO policies to be incorporated in job scheduling, resource negotiation, and VO-wide resource usage control.

Of all these services, only CDA and VOPS are accessible by other XtreemOS components like Application Execution Management or XtreemFS, the rest being only accessed internally.

4.3.2 Services in the MD version

To provide the kind of access needed by the use cases detailed in chapter 3, a XtreemOS-MD node will only need a way to access the Credential Distribution Authority, to obtain a XtreemOS certificate (XOS-Cert). With this XOS-Cert stored in the node (by the VO support components underneath developed in WP 2.3 [23]), the user will be able to access both the AEM and the XtreemFS services.

Also, the mutual authentication protocol described in the second specification of the security services (D3.5.4 [30]) is a good candidate to be used in the authentication of mobile devices, as it puts less load on the clients and more on the servers, something that mobile devices could certainly benefit from.

4.4 XtreemOS application programming interfaces (APIs)

Although not a grid service per se, the place that the API occupies on top of all the other services as part of the XtreemOS-G layer forces us to mention it here.

4.4.1 API of the standard flavour

The XtreemOS API will be based on the Simple API for Grid Applications (SAGA) API, as detailed in the API deliverables [17, 29, 13]. This API is aimed at high-level application developers who don't want to know anything about Grid computing or the middleware/infrastructure used, but want to make use of distributed resources through the Grid.

The structure of SAGA is modular, so as to be extended whenever needed. It is composed of:

Look & Feel API which is a common set of interfaces that must be shared by all SAGA packages, covering common tasks like error handling, monitoring or task model.

Functional packages are a series of objects and methods for accessing each area of grid functionality. Some of the most important are:

- **Namespace** provides a namespace representation as specified in POSIX standard, defining a hierarchical structure of directories and entries (i.e. SAGA objects). Thus, it is referred to by all the other packages.
- **Job** covers job handling task, including job description, job submission & control, reconnection, I/O redirection...
- **File** specifies I/O methods for files distributed in the Grid, following POSIX standards.
- **Logical File** provides file and directory replica service interfaces.

There are also other packages like RPC, Streams,... and SAGA can also extend its functionality by adding more packages that handle the appropriate operations.

Adaptors are a series of modules that bind the functionality covered by SAGA packages with the actual underlying middleware, or in this case, operating system. Thus, we can have a "File Adaptor for XtreemFS", a "Job Adaptor for XtreemOS AEM" or a "Job Adaptor for Globus GRAM", for example.

SAGA Engine is the rest of classes and libraries necessary for the SAGA infrastructure to work.

Also, extensions to SAGA will be devised where the functionality of SAGA cannot handle XtreemOS functionalities.

4.4.2 API of the MD version

From the use cases in chapter 3, we can infer that a mobile XtreemOS node will need, at least, the following components of the API:

- The Look & Feel API, as it is needed by all the other APIs, or at least the parts of it needed by the components detailed below.
- The SAGA Engine, as it is also a core element, or at least the parts of it needed by the components below.
- The Namespace package and its corresponding XtreemOS Adaptor, as it is referred by all the other packages.
- The Job package and its corresponding XtreemOS adaptor (or a subset of it, as needed by the use cases).
- The File and Logical File packages, and their corresponding XtreemOS adaptors (or the subset needed by the use cases).

4.5 Other grid services

What follows is a number of examples of more advanced and optional services that derive from the non-basic use cases. They will *not* be implemented in the basic version of XtreemOS-MD, but nevertheless they can give an idea of the kind of services that the advanced version could provide.

Visualization services

Comes from: state of the art

Their objective is to aid in the visualization of grid applications from the MDs. In its simplest form, they are already provided by the basic XtreemOS-MD, as we can access application results if they are in the form of a file in XtreemFS. For GUI applications, X forwarding, NX or other methods could be used. Moreover, for any graphic application appearing in small MD screens, transcoding/transmoding operations can be desirable, depending on the MDs capabilities, and interface peculiarities.

Advanced AEM

Comes from: state of the art, use cases

Basically, a service providing the ability to launch, monitor and manage jobs running *on MDs themselves*, and also permitting job migration to/from them.

QoS Provisioning / SLA Management

Comes from: state of the art

As the name suggests, this is a service for ensuring Quality of Service (QoS) and the management of Service Level Agreements (SLAs). This is a concept very typical of Akogrimo, along with the concept of "the network as a grid resource".

Context Management

Comes from: state of the art, use cases

In computer applications, context is defined as any data which describes the environment in which an application or system is being executed (including terminal capabilities, user preferences or knowledge, geolocalization data, etc). A context management service is one of the most likely candidates for the next version of XtreemOS-MD, as most mobile services involve some kind of context usage. Context acquisition should be handled by XtreemOS-F, but a grid service is needed for storing it, managing it and providing it to applications in a meaningful way.

Session Management

Comes from: state of the art, use cases

A service that allows to migrate sessions between two nodes in the grid. This is another Akogrimo concept, but we must look carefully at what is exactly a "grid session" in XtreemOS (in Akogrimo, sessions seem to be akin to the concept of voice/video call).

Notification service

Comes from: state of the art, use cases

Certain use cases need to receive notifications of certain events (presence information, job status changes, etc). This could be handled separately by each service/application, or unified for every notification needed in the grid.

Advanced Security Services

Comes from: use cases

Access to other security services (like VOMS or VOPS) could be needed for the "VO management from MDs" scenarios.

Call routing service

Comes from: state of the art, use cases

This component would allow to route voice/video calls to the PSTN/GSM/3G phone network, in the phone call scenarios.

Chapter 5

Requirements

Once we have derived from the use cases the grid services that are needed in the basic version of XtreemOS-MD, and the broad functionalities that they must provide, it is now necessary to provide the concrete requirements that the aforementioned services in mobile devices must fulfill.

The format we have followed to present these requirements is:

- Requirement identifier (R3.6.X).
- Short name.
- Description.
- Source of the requirement (dependency):
  - State of the Art (chapter 2)
  - Use Cases (chapter 3)
  - Deliverable D4.2.3 [20], indicated as RXX
  - Other XtreemOS deliverables, denoted with the deliverable code (DX.Y.Z) and the requirement specific code, if available.
- Level of importance: Basic|Advanced|Optional.

5.1 General service requirements

R3.6.1: Tracing System

It must be possible to obtain sufficient tracing information about application execution and resources being used from MDs. Besides, tracing level will be configurable.

Comes from: D4.2.3(R58, R59, R60, R61)
Importance: Basic

R3.6.2: Support for IPv6

XtreemOS-MD must support both IPv4 and IPv6.

Comes from: D4.2.3(R12)
Importance: Basic

R3.6.3: Degree of interoperability (middlewares)

XtreemOS-MD must be compatible with XtreemOS standard flavour as well as with other Grid middlewares (at least with regard to the key aspects e.g. security, job management, etc.). In D4.2.3(R48), it is pointed out having interoperability with GT4 WS-GRAM.

Comes from: D4.2.3(R48)
Importance: Optional

5.2 Requirements for application execution management (AEM) services

R3.6.4: Job management

XtreemOS-MD must allow full job management: launch, suspend (stop execution, leaving the job in memory), resume, cancel (kill the job) and wait. Jobs can be defined in a JSDL file either locally stored or in the XtreemFS; this file will be read by the launch mechanism before launching.

Comes from: Use Cases, D4.2.3(R18, R64, R103)
Importance: Basic

R3.6.5: Job monitor

XtreemOS-MD must provide a mechanism to monitor running jobs. Monitored information must include (but is not limited to) launch time, deemed time to end, status (running, suspended, awaiting execution...), resource consumption and special notifications.

The visibility of the jobs (either only jobs launched by the same user, or jobs launched by anyone in the same VO) will depend on the role of the user in the VO, and on the specific policies of the VO. This system would run in client mode, i.e. it would request the information to a well-known XtreemOS service.

Comes from: Use Cases, D4.2.3(R18, R53, R56)
Importance: Basic

R3.6.6: MD's as special nodes

XtreemOS-MD nodes must identify themselves with the resource management services as special nodes, which would mean a very limited storage and computation capacity.

Comes from: D4.2.3(R102)
Importance: Basic

R3.6.7: Support for interactive and batch jobs

XtreemOS-MD must allow the execution of both interactive and batch jobs. Batch jobs will be defined in an JSDL file (see R3.6.4) and potential interactions will be carried out in the MD.

Comes from: D4.2.3(R20)
Importance: Optional

R3.6.8: Checkpoint and restart job

It must be possible to checkpoint and later restart a job that is running on XtreemOS from a MD.

Comes from: State of the art
Importance: Optional

R3.6.9: Application migrations

XtreemOS-MD must allow ordering the migration of jobs (running on XtreemOS non-mobile nodes).

Comes from: Use Cases, D4.2.3(R6)
Importance: Optional

R3.6.10: Resource management

In case MD's resources are shared, this service should offer information about them to be used during external jobs execution.

Comes from: D4.2.3(R18)
Importance: Optional

R3.6.11: Software licensing

Some applications need to have locally stored a certain license file to be executed. XtreemOS-MD must provide mechanisms to distribute such license files when applications are being run in a different node and a license file is required as mentioned before. This kind of functionality would only be needed in case the MD's resources are shared with XtreemOS grid.

Comes from: D4.2.3(R9)
Importance: Optional

5.3 Requirements for data management services

R3.6.12: XtreemFS access

XtreemOS-MD must offer POSIX interface to access files from applications running on the MD, so that they are able to use files from a XtreemFS volume.

Comes from: Use Cases
Importance: Basic

R3.6.13: XtreemFS volume mounting

XtreemOS-MD must provide a mechanism to mount a XtreemFS volume and be able to access files for both reading and writing. This mounting could be executed e.g. at log on or under user request (for example, with a shell command).

Comes from: Use Cases
Importance: Basic

R3.6.14: Ensure integrity in filesystem operations

Due to MD connection unreliability, the necessary mechanisms must be put in place in order to ensure that sudden loss of connection does not affect filesystem integrity. Thus, XtreemOS-MD must provide communication and storage mechanisms to preserve data integrity.

Comes from: Use Cases, D4.2.3(R83, R84)
Importance: Basic

R3.6.15: Replication needs

XtreemOS-MD must provide a mechanism to set replication properties to files such as number of minimum replicas, synchronization policy for the replicas, and polices to avoid replication in unwanted sites (i.e no replicas out of Europe)

Comes from: State of the art
Importance: Optional

R3.6.16: File sharing from MD's

XtreemOS-MD must provide a mechanism to share local files stored in the MD through the XtreemFS.

Comes from: Use Cases
Importance: Optional

5.4 Requirements for VO management and security services

R3.6.17: Authentication

XtreemOS-MD must provide an authentication mechanism in order to authenticate users at login, or whenever a grid operation is requested. The exact form of this mechanism will depend on the mode of operation that is elected for XtreemOS-MD: either the user is logged into the Grid on login, or it is done at a later time whenever a grid functionality is needed. In any case, a way of obtaining the XOS-Cert from the VO manager is needed.

Comes from: D4.2.3(R85), Use Cases
Importance: Basic

R3.6.18: Single sign-on

XtreemOS-MD must provide authentication mechanisms that allow a user to gain authorized access to resources in a VO with a single interaction: once the user is authenticated, he can access all resources of the VO without being asked for authentication.

Comes from: D4.2.3(R85), Use Cases
Importance: Basic

R3.6.19: Independence between local and Grid user accounts

XtreemOS-MD must differentiate between local accounts and Grid accounts. Thus, a user will be able to connect to his Grid account from an MD even though he hasn't got a local account on it.

Comes from: D4.2.3(R23)
Importance: Basic

R3.6.20: An MD may belong to different VO's

XtreemOS-MD must allow the MD to belong to different VO's with occasionally different users, rules and resources. Besides, communication between those VO's must be possible.

Comes from: D4.2.3(R26, R29)
Importance: Basic

R3.6.21: Confidential communications

XtreemOS-MD must provide secure communication channels whenever confidential data are being transmitted.

Comes from: D4.2.3(R82)
Importance: Basic

R3.6.22: VO management

XtreemOS-MD must provide mechanisms or simple applications to manage VO's (users, resources, rules, etc.)

Comes from: Use Cases, D4.2.3(R22, R27, R28, R101)
Importance: Advanced

R3.6.23: Lightweight security methods

XtreemOS-MD must support lightweight security methods to improve MD's performance.

Comes from: D4.2.3(R104)
Importance: Optional

5.5 Requirements for application programming interfaces (API)

R3.6.24: Access to job management SAGA API.

XtreemOS-MD must provide applications with an API for managing and monitoring grid jobs through the AEM service, thus providing the functionalities described in section 4.1.

Comes from: Use Cases
Importance: Basic

R3.6.25: Access to data management SAGA API.

XtreemOS-MD must provide applications with an API for managing grid data available in XtreemFS, thus providing the functionalities described in section 4.2.

Comes from: Use Cases
Importance: Basic

R3.6.26: Access to VO support and security SAGA API.

XtreemOS-MD must provide applications with an API for doing security-related operations like authentication, thus providing the functionalities described in section 4.3.

Comes from: Use Cases
Importance: Basic

R3.6.27: Other API standards as basis for XtreemOS API

XtreemOS-MD must offer support for the most popular Grid APIs, which could be adapted in order to fit MD's particularities.

Comes from: Use Cases, D4.2.3(R45)
Importance: Optional

Chapter 6

Specifications

In this section, the specifics of the operations that must be performed by XtreemOS-MD grid services are detailed. Please note that only the requirements for the basic version are covered here, and advanced and optional requirements will be covered in later specifications (e.g. for the mobile phone version of XtreemOS-MD).

6.1 General service specifications

There can be several ways of logging in to a XtreemOS system, from an end user point of view. The login process can take place on the machine's startup, or be done after a user has already logged in with a pre-existing local account. In mobile devices, we can foresee at least three different mechanisms for login. XtreemOS-MD should support all three, although it is likely that integrators or administrators will want to restrict these options, so that end users of mobile devices only perceive one way of interacting with the Grid.

S3.6.1: XtreemOS login on device startup

The login process can take place when the device is powered on: in this case, the user will see the usual login screen, but will be able to state a VO identity as a valid user for login. Alternatively, if we want to make grid functionalities completely transparent to the end user (e.g. in mass market applications), the system will directly login with the user's certificate, which must already be present at the terminal (most likely, put there by the integrator, or using operator-provided certificates stored in the SIM card).

Interfaces

- *Type:* API (PAM, NSS modules and Key Retention System calls, CDA client library), GUI (login screen).
- *Input:* VO identity (e.g. VO ID, or other human-readable identifier).

- *Output:* None (access granted to the device, with grid capabilities enabled).

Internal process In the non-transparent version of this login process, a customized version of the login screen (akin to the standard Linux `login` application) will ask for the VO user identifier. With this input, the system will look for the corresponding certificate in some predefined place. If the certificate is already present, it is checked for validity by the XtreemOS PAM module, and stored into the Linux Key Retention System (LKRS) (see deliverable D2.1.2 [22] for more details). In case the certificate is not already present in the device, the application will ask for one to the Credential Distribution Agency (CDA), as per specification S3.6.10, checked for validity and stored in LKRS. In any case, if the certificate is valid, access is granted, and further grid operations will be possible from the device.

In the case of a completely transparent login, all this process is followed behind the scenes, using a user identity pre-defined (either by the integrator of the device, the operator or user-configurable). The asking of a passphrase to obtain or validate the certificate probably will be unavoidable, in a similar way as the user is asked a PIN when a mobile phone is powered on.

Optionally, if the user has a home directory in XtreemFS, it can be mounted automatically in a default location, as per specification S3.6.8.

Mobile characteristics

- *Connectivity loss:* The login process may not require network connectivity, if a valid certificate is already present in the device, so connectivity loss may not be fatal for the login process. But, if a request to the CDA is needed, the login process can be thwarted. In order not to hamper the usability of the device, other means of local login must be in place, as a fallback measure. In this case, grid login can be done at a later moment, as per the following specifications.

- *Battery shortage:* Power failures will obviously prevent the user from logging in. In any case, if the certificate is already present in the device, it should be stored in permanent storage, so that it is not lost when power failures occur.

- *User interface:* The login screen and eventual passphrase request dialogs will need a GUI, including also physical or virtual keyboards for input. Although this is technically an application (and a very likely subject of customization by the software integrator), and thus outside the development scope of XtreemOS, it is recommended that XtreemOS-MD provides this functionality “out of the box”.

Importance: Basic

S3.6.2: XtreemOS login after local login

Another way of logging in to the grid is doing it after the initial (local) login, when the user wants to use any grid functionality. The user would have an option for "activating grid access" which would invoke a login interface similar to the one mentioned in the previous specification S3.6.1. After this application is run successfully, the user will be able to access other grid functionalities of XtreemOS-MD.

Again, the user's home directory in XtreemFS can be mounted automatically in a default location, as per specification S3.6.8.

Interfaces

- *Type:* API (PAM, NSS modules and Key Retention System calls, CDA client library), GUI (login screen).
- *Input:* VO identity (e.g. VO ID, or other human-readable identifier).
- *Output:* None (grid capabilities enabled).

Internal process The internal process of login is the same as per specification S3.6.1.

Mobile characteristics See specification S3.6.1.

Importance: Basic

S3.6.3: XtreemOS 'on demand' login by applications

In this case, the user will try to start an application with grid functionalities. If the user is not logged into the grid (i.e. only local login has taken place), the login process will be *triggered automatically*, invoking the necessary dialogs for identity selection and/or passphrase input, as needed. This can be seen as a more transparent particular case of the previous specification S3.6.2.

Again, the user's home directory in XtreemFS can be mounted automatically in a default location, as per specification S3.6.8.

Interfaces

- *Type:* API (PAM, NSS modules and Key Retention System calls, CDA client library), GUI (login screen).
- *Input:* VO identity (e.g. VO ID, or other human-readable identifier).
- *Output:* None (the grid application is correctly started).

Internal process The internal process in this case is a slight modification of the previous ones. When a grid application is started, it looks into the LKRS for an existing XtreemOS certificate (i.e. checks if the login process has taken place). If it is not there, it means that the login has not taken place, and starts the login application as per specification S3.6.2. Once the login process is completed and the certificate is available, the application can continue normally.

Mobile characteristics See specification S3.6.1.

Importance: Basic

6.2 Specifications for application execution management (AEM) services

S3.6.4: Job submission

When a user of XtreemOS-MD (or a process/application acting in behalf of the user) wants to execute a job in the grid, a job description file (in JSDL language) must be submitted to the Application Execution Management, specifying the parameters (number and type of machines, memory, input and output files, etc) of the execution[1].

Interfaces

- *Type:* API (AEM client - library).
- *Input:* A JSDL file specifying the characteristics of the job. Implicitly, authentication information (e.g. a XOS-cert) must be available for the AEM client to use.
- *Output:* A confirmation that the job has been created (and, possibly, a handle or ID for identifying the job in later operations).

Internal process The call is issued to the AEM client, specifying a JSDL file with the description of the job. As authentication information for the user (in the form of a XOS-Cert) is needed for the AEM to accept the submission, the certificate must be passed in the call to the client, or it must be retrieved by the client from the user's kernel keyring (see D2.1.2 [22]), if the user had already been authenticated. With the job description and the XOS-Cert, the client contacts a well-known JobManager for the user's VO, which in turn will check the validity of the certificate and the policies concerning the user. If the user is allowed to create the job,

[1] This specification covers the submission of batch jobs only. Interactive jobs have not yet been specified in the standard flavour of XtreemOS, and will be covered in the advanced specification of XtreemOS-MD.

other entities of AEM can be contacted to that end, and a handler (likely, an ID) for the job will be returned to the client.

Mobile characteristics

- *Connectivity loss, Battery shortage:* If the submission process is interrupted by connectivity or battery shortages, the job will not be created. If the connection was lost *after* the creation of the job, the job will be created, but the client could lose the handler of the job. Thus, a way of retrieving the handlers of the jobs created by a user must be provided, at least for jobs that have not yet finalized.
- *User interface:* Application implementations are not the main goal of this WP. Fortunately, one of the applications from WP4.2 is a graphical frontend for the AEM client, allowing mobile users to submit jobs to the XtreemOS grid.

Importance: Basic

S3.6.5: Retrieve list of jobs

There are cases when a user (or, more likely, an application acting on behalf of the user) will want to know which jobs the user has created on the XtreemOS grid (e.g. if the network failed in the process of creating a job, and the job was created but no response from AEM was evere received).

Interfaces

- *Type:* API (AEM client - library).
- *Input:* User identification (including authentication information, implicit if the login already has taken place).
- *Output:* A list of the job identifiers for all the active jobs of the user in the grid (including the recently finalized jobs).

Internal process The request for the list is issued to the AEM client, specifying the identity of the user (maybe implicitly, if login already took place). As authentication information for the user (in the form of a XOS-Cert) is needed for the AEM to provide this kind of list, the certificate must be passed in the call to the client, or it must be retrieved by the client from the user's kernel keyring (see D2.1.2 [22]), if the user had already been authenticated. With the XOS-Cert, the client contacts a well-known JobManager for the user's VO, which in turn will check the validity of the certificate and the policies concerning the user. Taking into account the user's role and privacy policies of the VO, a list with all the job handlers (i.e. job IDs) of jobs that the user can "see" will be returned to the client.

Mobile characteristics

- *Connectivity loss:* If the request is interrupted by a network outage, an adequate exception will be returned to the user. In any case, the jobs themselves should not be affected by this event.
- *Battery shortage:* Since a battery shortage will render the device unusable, no special considerations have to be taken into account. When the device is powered on again, the list can be requested again, provided that the user is again logged into the grid.
- *User interface:* Application implementations are not the main goal of this WP. Fortunately, one of the applications from WP4.2 is a graphical frontend for the AEM client, allowing mobile users to get a list of the user's jobs.

Importance: Basic

S3.6.6: Retrieve data for a job (job monitoring)

Users or user applications may want to monitor the current status of a certain job, provided we know the job handler. This information should include, at least, the following:

- Job owner (the user who started it)
- Job status (running, paused, cancelled, etc)
- Date/Time of creation
- Date/Time of ending (if already finished)
- Number of nodes where it is running (plus, optionally, the IP addresses of the nodes)
- Total memory consumed by the job (plus, optionally, the memory consumed on each node)

Interfaces

- *Type:* API (AEM client - library).
- *Input:* User identification (including authentication information, implicit if the login already has taken place) and job handler (job ID).
- *Output:* Information about the job (see above).

Internal process The request for job information is issued to the AEM client, specifying the identity of the user (maybe implicitly, if login already took place) and the job handler. As authentication information for the user (in the form of a XOS-Cert) is needed for the AEM to provide this kind of information, the certificate must be passed in the call to the client, or it must be retrieved by the client from the user's kernel keyring (see D2.1.2 [22]), if the user had already been authenticated. With the job handler and the XOS-Cert, the client contacts a well-known JobManager for the user's VO, which in turn will check the validity of the certificate and the policies concerning the user. Taking into account the user's role and privacy policies of the VO, the information will be returned to the client.

Mobile characteristics

- *Connectivity loss:* If the request is interrupted by a network outage, an adequate exception will be returned to the user. In any case, the job itself should not be affected by this event.
- *Battery shortage:* Since a battery shortage will render the device unusable, no special considerations have to be taken into account. When the device is powered on again, the job information can be requested again, provided that the user is again logged into the grid.
- *User interface:* Application implementations are not the main goal of this WP. Fortunately, one of the applications from WP4.2 is a graphical frontend for the AEM client, allowing mobile users to get job information.

Importance: Basic

S3.6.7: Cancel, stop, pause and restart a job (job managing)

A user, or an application on behalf of a user, may want to manipulate jobs that are active in the grid. These operations include:

- Cancelling a job permanently
- Stopping a job (the job will be checkpointed, stopped, and deleted from memory)
- Pausing a job (the job will be interrupted, but it will remain in memory)
- Resume execution of a job

Interfaces

- *Type:* API (AEM client - library).

- *Input:* User identification (including authentication information, implicit if the login already has taken place) and job handler (job ID).
- *Output:* Confirmation of the success or failure of the operation.

Internal process The request for job operation is issued to the AEM client, specifying the identity of the user (maybe implicitly, if login already took place) and the job handler. As authentication information for the user (in the form of a XOS-Cert) is needed for the AEM to modify the job, the certificate must be passed in the call to the client, or it must be retrieved by the client from the user's kernel keyring (see D2.1.2 [22]), if the user had already been authenticated. With the job handler and the XOS-Cert, the client contacts a well-known JobManager for the user's VO, which in turn will check the validity of the certificate and the policies concerning the user. Taking into account the user's role and privacy policies of the VO, it will proceed with the operation, and return the adequate success or failure response to the client.

Mobile characteristics

- *Connectivity loss:* If the request is interrupted by a network outage, an adequate exception will be returned to the user. Depending on the moment in which the request was interrupted, the operation may or may not be completed. Thus, a request for information will be needed to verify the current status of the job (see specification S3.6.6).

  Optionally, caching of job operations can be implemented so that interrupted operations can be retried when the connection is recovered.

- *Battery shortage:* If the request was submitted to the AEM server, the operation should be completed, regardless of the state of the client. When the device is powered on again, the job information can be requested again, provided that the user is again logged into the grid, and the information should show the updated status of the job.

  Optionally, caching of job operations can be implemented so that interrupted operations can be retried when the connection is recovered.

- *User interface:* Application implementations are not the main goal of this WP. Fortunately, one of the applications from WP4.2 is a graphical frontend for the AEM client, allowing mobile users to manage jobs.

Importance: Basic

6.3 Specifications for data management services

S3.6.8: Mount volumes in XtreemFS

Grid users that need to access files stored in the XtreemFS should be able to mount XtreemFS volumes in the same way as they mount any other kind of filesystem, through standard Linux utilities like `mount`. Once mounted, the users will have access to the files in that volume in the same way as they access local files, in a completely transparent way. The permissions for accessing those files will be translated into standard Linux terms (e.g. `rwx` permissions) taking into account the VO policies regarding those files and the user identity.

Moreover, common Linux mechanisms like `/etc/fstab` will also be possible for automatic mounting of XtreemFS volumes, for example, when the user logs into the XtreemOS grid.

Interfaces

- *Type:* API (XtreemFS client - FUSE), `mount` command.
- *Input:* User identification (including authentication information, implicit if the login already has taken place) and volume identifier (e.g. path).
- *Output:* Confirmation of the success or failure of the operation.

Internal process When the mount command is invoked, the request is handled by the XtreemFS FUSE library, which will request the user authentication information (i.e. his XOS-Cert) from the LKRS (if the user has not been logged in, the process of login could be triggered at that moment, as per specification S3.6.3). With this certificate, the FUSE library will contact the XtreemFS servers, which will in turn validate the certificate and see if the user has the adequate permissions for this operation. If so, the operation is confirmed and the client will receive confirmation that the volume is mounted.

Mobile characteristics

- *Connectivity loss:* If lack of network connectivity interrupts the process of mounting an XtreemFS volume, the adequate error should be returned to the user. However, once the mounting is complete, network microcuts should not affect the user, provided that a file operation is not attempted while the device is offline.
- *Battery shortage:* The device will become unusable if power goes off, and all existing mounts will become unavailable. When the device is powered on again, and the user logs in, his home directory (and any other automatic XtreemFS mounts defined) can be mounted again, transparently.

- *User interface:* The XtreemFS client does not offer an specific user interface, apart from the command-line `mount` utility. However, it would be desirable to integrate it with GUI applications for file management already present in the mobile device (e.g. GPE's file manager), possibly automating the operation of mounting, at least, the user's home directory in XtreemFS, if available.

Importance: Basic

S3.6.9: Access files in XtreemFS

Once an XtreemFS is mounted (see specification S3.6.8), users can do standard POSIX operations with files, like opening them, reading from them or writing to them. These operations will take place using standard Linux file access calls, and thus should be completely transparent for the users. More advanced features of XtreemFS (for example, specifying the number and location of file replicas) may not be available through this transparent interface.

The operations themselves may or may not be successful depending on the identity of the user and the VO policies regarding the files, which will be translated into POSIX-like Linux permissions for use of the user applications.

Interfaces

- *Type:* API (standard file access operations: `open`, `read`, `write`...).
- *Input:* User identification (including authentication information, implicit if the login already has taken place) and file identifier (e.g. path) and any other information needed by the file operation.
- *Output:* Confirmation of success or failure, or file data in case of a `read` operation.

Internal process When the file access call is invoked, the request is handled by the XtreemFS FUSE library, which will request the user authentication information (i.e. his XOS-Cert) from the LKRS (if XtreemFS caches the certificates e.g. for the duration of a mount, this step could be unnecessary). With this certificate, the FUSE library will contact the XtreemFS servers, which will in turn perform the required operation, if the user has adequate permissions for it. If so, the operation is confirmed and the client will receive confirmation of the operation, and/or the requested file data.

Mobile characteristics

- *Connectivity loss:* If the network connection is interrupted during the performance of a file access, adequate error will be returned to the user. If,

for example, a write operation was truncated, it should be rolled back on its entirety. In any case, the integrity of the filesystem must be ensured.

Optionally, caching of file operations can be implemented so that interrupted operations can be retried when the connection is recovered.

- *Battery shortage:* If the mobile device is unexpectedly switched off, any truncated operation should be rolled back, so that filesystem integrity is ensured. When the device is powered back on, the processes of logging in and volume mounting will be needed before the operation can be attempted again.

  Optionally, caching of file operations can be implemented so that interrupted operations can be retried when the power is back on.

- *User interface:* The XtreemFS client does not offer an specific user interface, apart from the usual file access system calls. However, it would be desirable to integrate it with GUI applications for file management already present in the mobile device (e.g. GPE's file manager).

Importance: Basic

6.4 Specifications for VO management and security services

S3.6.10: Obtain a XtreemOS certificate

As a previous step to any operation involving XtreemOS grid functionalities, the users must obtain a certificate (XOS-Cert), in order to be authenticated with the various XtreemOS components. This certificate will be valid only for a certain (and probably short) time span, and is issued by the Credential Distribution Agency (CDA), as described in D3.5.4 [30].

Interfaces

- *Type:* API (CDA client - library).
- *Input:* a way of authenticating with the CDA (user/passphrase, kerberos token, etc).
- *Output:* a XOS-Cert certificate, which can be stored in a file somewhere in the MDs permanent or temporary storage.

Internal process Once the call is issued to the CDA client, with all the data needed by the CDA (token, user/passphrase, etc), the CDA client will contact a well-known CDA for a certain VO. The ensuing process will have to be conducted over a secure channel (secured by SSL), in order to preserve the confidentiality and integrity of the information passed on to the CDA, and the confidentiality of the certificate returned. The received certificate, and the locally generated private key, will be stored in a location specified in the moment of the call, or in a default location where VO support mechanisms (like PAM and others, see D2.1.2 [22]) can find it.

Mobile characteristics

- *Connectivity loss, Battery shortage:* Certificates should be checked for validity on arrival, so that truncated or manipulated certificates can be detected. Also, if the connection with the CDA is interrupted, no data should be stored, an error will be returned and the process will have to be repeated from the beginning (passphrase/token caching mechanisms are optional).
- *User interface:* Although application implementations is not the main goal of this WP, the development of a GUI application to obtain this certificates (either by the consortium or by the XtreemOS community) would be advisable. It also could include the saving of the certificate and private key in an adequate default location so that it is transparent for a grid-unaware mobile user.

Importance: Basic

S3.6.11: Single Sign On

Once the process of login has been conducted (see section 6.1), the user and the applications running in his behalf should not be bothered again with authentication prompts, until the XtreemOS certificate expires or the device is powered off (since, at power on, the login process will be repeated).

Interfaces

- *Type:* None (behind the scenes, applications use PAM and Key Retention System APIs).
- *Input:* None (implicitly, a XOS-Cred or a delegated proxy of it, stored in the LKRS).
- *Output:* None.

Internal process During the login process (see section 6.1), XtreemOS PAM modules store the limited-time user certificate into the Linux Key Retention System. From then on, any grid application that needs user authentication, like for example the AEM client or the XtreemFS, just needs to retrieve the credential from the keyring, without having to ask the user for it.

Mobile characteristics Single Sign On functionalities in mobile devices do not need special considerations.

Importance: Basic

S3.6.12: Confidentiality in communications

Communications between XtreemOS applications or components running in mobile devices and other grid nodes may require an authenticated, secure channel. This channel must provide confidentiality and integrity of the data passed, as well as non-repudiation.

Interfaces

- *Type:* API (standard SSL/TLS procedures)
- *Input:* As per SSL/TLS specifications.
- *Output:* As per SSL/TLS specifications.

Internal process The preferred method to establish a secure, mutually authenticated channel, is to use a conventional SSL/TLS implementation (with proxy certificate support). This secure channel guarantees confidentiality and integrity of data. Non repudiation is not guaranteed by a TLS channel, but it can be implemented at application level using XOS-Cred (or a delegated proxy from it) to sign the data. TLS is compatible with legacy applications, but some adaption may be necessary if using proxy certs instead of directly the XOS-Cert.

Mobile characteristics No additional considerations must be made in the case of mobile devices: Network or power failures will obviously thwart the communication channel, which will have to be renegotiated again. No user interface is present in this case.

Importance: Basic

S3.6.13: Secure, mutual authentication

Grid user applications running in XtreemOS-MD should have some method of mutually authenticating with remote XtreemOS components, so that both parties in the communication can be sure who they are talking to.

Interfaces

- *Type:* API (SSL/TLS protocol or mutual authentication protocol [27]).
- *Input:* As per SSL/TLS or mutual authentication protocols.
- *Output:* As per SSL/TLS or mutual authentication protocols.

Internal process Communications between XtreemOS applications or components may require a secure mutual authentication using XOS-Cred. It's possible to use a conventional SSL/TLS to gain a secure and authenticated channel, although other protocols (like the mutual authentication protocol described in D3.5.6 [27]) can also be used. The usage of one method or another can be predefined by the network operator, software integrator, or user defined, depending on who delivers the grid services, and under which conditions they are provided.

Mobile characteristics If the usual SSL/TLS is deemed to heavy to run in the mobile devices with acceptable performance, and the communication channel is deemed secure (e.g. if the network operator ensures this security), just using a mutual authentication protocol could be sufficient.

Importance: Basic

6.5 Specifications for application programming interfaces (API)

Although not a grid service per se, the API used by grid applications to access XtreemOS is extremely important, and thus it is included in this chapter as part of the G-layer of XtreemOS-MD. Apart from providing XtreemOS-specific interfaces to XtreemOS-aware applications, XtreemOS-MD has to offer a SAGA-compliant interface to more generic grid applications prepared to work with it:

S3.6.14: SAGA API for Java applications - Job functionalities

Java grid applications that use the Simple API for Grid Applications (SAGA) interface for accessing job management functionalities with independence of the underlying middleware and OS, should work in XtreemOS-MD, with slight or no modifications.

Interfaces

- *Type:* API (SAGA Look & Feel and Job package).
- *Input:* As per SAGA specifications.
- *Output:* As per SAGA specifications.

Internal process When a grid application makes a SAGA call for managing or monitoring grid jobs, the SAGA engine will pass it on to the XtreemOS adaptor for SAGA (or a lightweight version of it), which in turn will translate it to the adequate XtreemOS-specific calls for contacting the AEM services. These calls should make use of XtreemOS security mechanisms (e.g. XtreemOS certificates) through SAGA's Contexts.

Mobile characteristics

- *Connectivity loss, Battery shortage:* The effects of these events for the management of jobs is already explained in specifications in section 6.2. In the case of SAGA calls, unexpected errors (e.g. a network failure) should return the adequate exceptions as defined in the SAGA specifications.
- *User interface:* as an API, the SAGA engine in mobile devices will not have a recognizable user interface, as it will be provided by the applications.

Importance: Basic

S3.6.15: SAGA API for Java applications - File functionalities

Java grid applications that use the Simple API for Grid Applications (SAGA) interface for accessing files stored in the Grid, with independence of the underlying middleware and OS, should work in XtreemOS-MD, with slight or no modifications.

Interfaces

- *Type:* API (SAGA Look & Feel and Job package).
- *Input:* As per SAGA specifications.
- *Output:* As per SAGA specifications.

Internal process When a grid application makes a SAGA call for accessing or modifying grid files, the SAGA engine will pass it on to the XtreemOS adaptor for SAGA (or a lightweight version of it), which in turn will translate it to the adequate XtreemOS-specific calls for contacting the XtreemFS. These calls should make use of XtreemOS security mechanisms (e.g. XtreemOS certificates) through SAGA's Contexts.

Mobile characteristics

- *Connectivity loss, Battery shortage:* The effects of these events remote file access is already explained in specifications in section 6.3. In the case of SAGA calls, unexpected errors (e.g. a network failure) should return the adequate exceptions as defined in the SAGA specifications.

- *User interface:* as an API, the SAGA engine in mobile devices will not have a recognizable user interface, as it will be provided by the applications.

Importance: Basic

Chapter 7

Conclusions

This document has laid the first stone for the building of grid services for mobile devices in XtreemOS.

The analysis of the new, emergent field of mobile grids has shown us that using mobile devices only as clients for accessing the grid is the most common approach, due to the limits of the MD's resources. But it has also pointed out that a number of initiatives and researches are trading the paths of providing truly mobile grid services, mostly by extending the concept of "grid resource" to other assets where mobile devices excel (mobility, context, user knowledge). Moreover, we have been made aware of the fact that development issues (like the difficulty of building software for embedded devices) has been one of the main obstacles to these research initiatives, as it raises their initial costs considerably.

Because of this, in XtreemOS-MD it has been decided to provide a first, client-only version of the grid services, also because the core services in the standard flavour of XtreemOS are still maturing. Once the core infrastructure is in place, the mobile flavour workpackages will research more complex grid services that build upon them, to offer more advanced grid services. Some examples of these advanced services have been presented in chapter 3.

This basic services to be offered by the basic version of XtreemOS-G for mobile devices include the following functionalities:

- A way of obtaining XtreemOS certificates (XOS-Cert) from the security services that manage virtual organization aspects (namely, the Credential Distribution Agency). This certificate, along with the VO support features provided by the XtreemOS-F layer, will enable mobile users to access XtreemOS grid services.
- Client interface to the Application Execution Manager, in order to launch, manage and monitor grid jobs from the mobile device. This will most likely take the form of a Java implementation of the XATI interface.
- Client for the XtreemFS grid filesystem, to provide a POSIX interface to access the user's files stored in the grid, and making it possible to mount

XtreemFS volumes from a PDA. This will be achieved with a FUSE client for XtreemFS.

- A (possibly stripped-down) version of the SAGA Java API, adapted for XtreemOS and for mobile devices, allowing SAGA-aware user applications to access grid functionalities offered by XtreemOS.

With these features and use cases in mind, the document gathered a number of requirements and specifications detailing how to fulfill those requirements. This requirements not only take into account the grid services themselves, but also the environment in which they will be operating (e.g. situations with low CPU and memory, and unreliable network connectivity).

This list of requirements and specifications point to having a small number of processes and libraries providing that functionality, mostly written in Java language (with the sole exception of the XtreemFS client, which is written in C). This is partly due to the fact that the standard flavour counterparts are developed in Java, but also because the applications that XtreemOS will use to test the mobile device flavour are written in Java.

This Java dependency has to be analyzed thoroughly from a practical/development standpoint, as Java for mobile devices presents several licensing issues, being J2ME the "less open" platform for Sun's language and SDK.

Also, the performance side of this service has to be carefully looked at, as Java is known to be a big resource consumer. The evaluation of different Java Virtual Machines and the usage of tools like `gcj` will be mandatory when designing and implementing these solutions.

In any case, and in order to favor the spread and adoption of XtreemOS-MD, the implementation of C/C++ counterparts of these grid components must also be considered, as still many instances of mobile applications are written in this language, specially in the OSS community.

Chapter 8

Future Work

Now that we have laid out the concrete requirements and specifications for the grid services in mobile device version of XtreemOS (for the basic PDA version, at least), the next step will be the design of such services. In this endeavour, we foresee several steps:

- The main components and building blocks for the Grid layer of XtreemOS-MD will be detailed, including an extensive list of features and functionalities, the components that implement them and the interfaces between them and with other XtreemOS components.
- Also, a detailed list of software packages and dependencies for those components will be compiled, as a checklist to guarantee that XtreemOS-MD complies with the requirements and specifications in this document.
- In parallel with this design works, first experiments with the early prototypes of the components in a mobile device will be conducted, to detect and prevent possible issues that could arise in the implementation of the services.

All these steps will be reflected in a report, D3.6.2, entitled “Design of basic services for mobile devices”.

References

[1] Akogrimo web site.
http://www.mobilegrids.org/.

[2] David C. Chu and Marty Humphrey. Mobile OGSI.NET: Grid computing on mobile devices. In *Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing (associated with Supercomputing 2005)*, November 2004.

[3] Massimo Coppola, Yvon Jégou, Brian Matthews, Christine Morin, Luis Pablo Prieto, Oscar David Sánchez, Erica Y. Yang, and Haiyan Yu. Virtual organisation support within a grid-wide operating system. Submitted to IEEE Internet Computing, 2007.

[4] Laboratório Nacional de Computaçao Científica (LNCC). MoGrid Project web page.
http://www.lncc.br/martin/ShowProject?language=2#Mobile%20Grids.

[5] Luciana dos Santos Lima, Antônio Tadeu Azevedo Gomes, Artur Ziviani, Markus Endler, Luiz Fernando Gomes Soares, and Bruno Schulze. Peer-to-peer resource discovery in mobile grids, August 2005.

[6] Antônio Tadeu A. Gomes, Artur Ziviani, Luciana S. Lima, and Markus Endler. DICHOTOMY: A resource discovery and scheduling protocol for multihop ad hoc mobile grids. In *First Workshop on Context-awareness and Mobility in Grid Computing (WCAMG)*, 2007.

[7] Piotr Grabowski, Krzysztof Kurowski, Jarek Nabrzyski, and Michael Russell. Context sensitive mobile access to grid environments and vo workspaces. In *Seventh IEEE International Symposium on Cluster Computing and the Grid, 2007. CCGRID 2007.*, volume 00, pages 87–87, 2006.

[8] GridLab project web site.
http://www.gridlab.org.

[9] Tao Guan, Ed Zaluska, and David De Roure. A grid service infrastructure for mobile devices. In *First International Conference on Semantics, Knowledge and Grid, 2005. SKG '05.*, pages 42–42, November 2005.

[10] Torben Knerr. Mobile Access to Grid Web Services, July 2006. Presentation slides.
http://mobdev.tknerr.de/wp-content/uploads/2006/09/mobilegridaccess.pdf.

[11] Xun Luo. PACE: Augmenting personal mobile devices with scalable computing. In *Proceedings of the 7th International Conference on Mobile Data Management (MDM'06)*, pages 875–880, May 2007.

[12] David E. Millard, Arouna Woukeu, Feng (Barry) Tao, and Hugh Davis. Experiences with writing grid clients for mobile devices. In *Proceedings of 1st International ELeGI Conference on Advanced Technology for Enhanced Learning BCS Electronic Workshops in Computing (eWiC)*, March 2005.

[13] Open Grid Forum (OGF). Saga c++ reference implementation, 2007.
http://saga.cct.lsu.edu.

[14] Brazil Pontifical University of Rio de Janeiro. First workshop on context-awareness and mobility in grid computing (WCAMG), 2007.
http://wcamg2007.inf.puc-rio.br/program.html.

[15] O.M.C. Rendon, F.O.M. Pabon, M.J.G. Vargas, and J.A.H. Guaca. Architectures for web services access from mobile devices. In *Third Latin American Web Congress, 2005. LA-WEB 2005.*, November 2005.

[16] Stefan Wesner and Juergen M. Jaehnert and María Aránzazu Toro Escudero. Mobile Collaborative Business Grids - A short overview of the Akogrimo Project.
http://www.akogrimo.org/download/White_Papers_and_Publications/Akogrimo_WhitePaper_Overview.pdf.

[17] XtreemOS Consortium. First Draft Specifcation of Programming Interfaces D3.1.1. Integrated Project, December 2006.

[18] XtreemOS Consortium. Requirements and specification of XtreemOS services for job execution management D3.3.1. Integrated Project, December 2006.

[19] XtreemOS Consortium. The XtreemOS File System - Requirements and Reference Architecture D3.4.1. Integrated Project, December 2006.

[20] XtreemOS Consortium. Application References, Requirements, Use Cases and Experiments D4.2.3. Integrated Project, July 2007.

[21] XtreemOS Consortium. Basic XtreemFS object-based file system and basic Object Sharing Service D3.4.2. Integrated Project, December 2007.

[22] XtreemOS Consortium. Design and implementation of basic version of node-level VO support mechanisms D2.1.2. Integrated Project, December 2007.

[23] XtreemOS Consortium. Design of a Basic Linux Version for Mobile Devices D2.3.3. Integrated Project, December 2007.

[24] XtreemOS Consortium. Design of the architecture for application execution management in XtreemOS D3.3.2. Integrated Project, June 2007.

[25] XtreemOS Consortium. First Draft Specification of XtreemOS Security Services D3.5.3. Integrated Project, June 2007.

[26] XtreemOS Consortium. Implementation of the basic services for job submission, control and monitoring D3.3.3. Integrated Project, December 2007.

[27] XtreemOS Consortium. Report on Formal Analysis of Security Properties D3.5.6. Integrated Project, December 2007.

[28] XtreemOS Consortium. Requirements and Specifications of a Basic Linux Version for Mobile Devices D2.3.2. Integrated Project, February 2007.

[29] XtreemOS Consortium. Second draft specification of programming interfaces D3.1.1. Integrated Project, December 2007.

[30] XtreemOS Consortium. Second Draft Specification of XtreemOS Security Services D3.5.4. Integrated Project, December 2007.

[31] XtreemOS Consortium. Virtual Organization Basic Requirements and Specifications for Mobile Devices D2.3.1. Integrated Project, February 2007.

[32] XtreemOS Consortium. Design Report for Advanced XtreemFS and OSS Features D3.4.3. Integrated Project, June 2008.