Project no. IST-033576

# XtreemOS

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

## D3.6.5
## Design of advanced services for mobile devices

Due date of deliverable: 30th November 2009
Actual submission date: 21st December 2009

*Start date of project: June 1st 2006*

*Type: Deliverable*
*WP number: WP3.6*
*Task number: T3.6.5*

*Responsible institution: Telefónica I+D*
Editor & editor's address: Santiago Prieto
Telefónica I+D, Parque Tecnológico de Boecillo
Boecillo (Valladolid)
SPAIN

Version 1.0/ Last edited by Telefónica I+D/ Date 21-12-2009

| 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) | |

Keyword List: design, linux, mobile device, smartphone

Revision history:

| Version | Date | Authors | Institution | Sections Affected / Comments |
|---|---|---|---|---|
| 0.1 | 16-Oct-09 | Telefónica I+D | Telefónica I+D | Document created |
| 0.2 | 10-Nov-2009 | Telefónica I+D | Telefónica I+D | Initial sections drafted |
| 0.3 | 12-Nov-2009 | Toni Cortes | BSC | AEM + XtreemFS |
| 0.4 | 17-Nov-2009 | Telefónica I+D | Telefónica I+D | Future work defined |
| 0.5 | 25-Nov-2009 | Telefónica I+D | Telefónica I+D | Resource sharing & general review |
| 0.6 | 27-Nov-2009 | Telefónica I+D | Telefónica I+D | Security services and intro completed |
| 0.7 | 30-Nov-2009 | Telefónica I+D | Telefónica I+D | Version ready for review. |
| 1.0 | 21-Dec-2009 | Telefónica I+D | Telefónica I+D | Final version after review |
| | | | | |
| | | | | |

Reviewers
Guillaume Pierre (VUA), Marjan Sterk (XLAB)

Tasks related to this deliverable

| Task No. | Task description | Partners involved ° |
|---|---|---|
| T3.6.5 | Design of advanced services for mobile devices (XtreemOS-G for MD/MP) | TID*, BSC |
| | | |
| | | |
| | | |

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

* Task leader

# Executive Summary

The XtreemOS project aims at integrating native Grid functionalities into Linux operating systems and addressing the heterogeneity of current Grid computing technology, from clusters to mobile devices (MDs). The objective of this deliverable is the design of the advanced G-layer for the XtreemOS Mobile Device flavour. This design will be the basis for the implementation of the final release that is targeted at selected mobile devices. The particular characteristics of these devices and the identified use cases and scenarios are carefully taken into account. Addressing this heterogeneity and providing advanced Grid services will be a great advantage for XtreemOS as a whole. This will bring Grids nearer to users, and the XtreemOS' transparent design will be a driver to open Grid computing to the mass market while expanding the mobile broadband marketplace.

The advanced version presented in this document, compared to the first version previously released, includes support for smartphones, and also many additional features on the three main G-layer services: Security services, Application Execution and Management and Data management services. Furthermore, this deliverable deals with the design of a new Resource Sharing service for mobile devices, which turns the mobile device from a Grid client into a special Grid resource.

The design work starts by reviewing the previous work done about specifications and requirements in the G-layer from WP3.6, and also considering the advanced F-layer implementation from WP2.3. The final general architecture is then derived, and it is used during the rest of the document. Then, for each main service the specific new features and their design are detailed.

Finally, this deliverable presents the last milestone towards the implementation of the advanced version of XtreemOS-MD flavor. This final version will be integrated with the rest of XtreemOS components.

# Table of contents

# 1 Introduction

This document presents the design of advanced version of XtreemOS-MD services layer (G-layer).
This advanced version, compared to the first version previously released, includes support for *smartphones*, and also many additional features on the three main services: Security, AEM and XtreemFS. In addition, it also includes new G-layer features capable of providing a complete Resource Sharing Service in mobile devices, thus fully converting the mobile device from a Grid client into a special Grid resource.

First, the new security services are focused on providing the necessary adaptation to the final XtreemOS architecture as designed in D3.5.13. In addition, the *CDAProxy* component is improved with PAM modules to ease the integration with third-party corporate authentication systems and enable more transparent operation for mobile device users in those environments.

Next, the design of the new features of Application Execution Management for mobile devices is addressed. Mobile device G-layer follows the architecture of AEM in XtreemOS and completes the design with two specific features, monitoring call-backs and interactive jobs. These features are particularly useful for upper-layer applications such as JobMA that will be able to monitor the progress of jobs from mobile devices.

Then, the new features for data management services are covered. In this case, the improvements come from including the Vivaldi algorithm in mobile devices to support the selection of replicas. Also, the design of offline mode support is covered, particularly useful for mobile devices that are not constantly connected to the Internet or the Grid.

Finally, resource sharing services are covered in detail. As it was identified in D2.3.6 and D3.6.4 resource sharing service is completely new in the mobile device flavor. The specific user needs and terminal requirements obtained in those two previous deliverables are carefully considered. The design covers data sharing from mobile devices by means of an OSD proxy element. We also cover the sharing of input/output devices of the mobile terminal, in a generic approach, and also how network sharing on the mobile device may be achieved.

## 1.1 Document structure

- Chapter 2 provides an introduction to the advanced services for mobile devices that will be designed in the following chapters.
- Then, Chapter 3 deals with the specific design of the new features concerning the security services. Chapter 4 focuses on the new Application Execution Management services, including design principles for monitoring call-backs and interactive jobs in mobile devices.
- Chapter 5 provides the design of the new features for Data Management; replica selection based on Vivaldi algorithm and offline mode support.
- Chapter 6 covers the design of all the new features related to Resource Sharing in mobile devices, including data sharing and I/O device plus network sharing.
- We finalize with Chapters 7 and 8 by giving the conclusions and describing the future work.

# 2 Advanced services for mobile devices overview

This chapter introduces a high level vision of the advanced services for mobile devices. We first present a comprehensive overview of the requirements and specifications already identified in previous deliverables and then we will show the high-level architecture of the final XtreemOS-MD advanced version. Later chapters will address the specific design issues, service by service.

## 2.1 Requirements and specifications review

The requirements and specifications related to advanced services for MD were defined and described in the D3.6.4 [1]. Now, these requirements must be carried out by the advanced services on the G-layer. To present a clear vision of all of them, we proceed to review the main requirements and specifications ordered by Grid services.

### 2.1.1 AEM service

Regarding the Application Execution Management (AEM), the requirements and specifications are first targeted at improving the monitoring system for mobile devices. In order to include this feature, an active monitoring functionality is needed first, as specified in requirement R3.6.29 and specification S3.6.20. Then, the monitoring can be performed using either local buffers to store job related events (req. R3.6.30, spec. S3.6.21) or asynchronous notification of events using call-backs (req. R3.6.31, spec. S3.6.22).

The AEM has been undergoing improvement since the XtreemOS 1.0 was released. Two additional important features of AEM were designated to be accessible from XtreemOS-MD advanced version: resource reservations (req. R3.6.33, specs. S3.6.16 and S3.6.17) and dependency trees (req. R3.6.34 and specs. S3.6.18, S3.6.23 and S3.6.24). Basically, mobile devices will be able to create resource reservation and use them to launch jobs over previously reserved Grid resources. In addition, users will be able to navigate through jobs that are linked to each other by dependency trees.

Finally, XtreemOS-MD should provide support for interactive applications (req. R3.6.28). Part of this support should be provided in the context of AEM by the interactive jobs feature (following req. R3.6.28 and spec. S3.6.19).

### 2.1.2 Data management service

In the context of XtreemFS, the first advanced feature deals with including support for the *Vivaldi algorithm* into the mobile device (req. req. R3.6.36). Basically, the new functionalities are: being able to place the MD clients into the Vivaldi space (spec. S3.6.25) - that is, assigning the Vivaldi coordinates to the MD-, and using the Vivaldi coordinates to choose the closest replica to the MD whenever an XtreemFS replica is needed (spec. S3.6.26).

Given that mobile device users are not necessarily always connected (as PC users may be), being able to *work in offline mode* is a desired feature (req. R3.6.38). Hence, XtreemOS-MD, using XtreemFS mechanisms, will provide an offline way to work with data without a permanent connection to the Grid (spec. S3.6.28).

Furthermore, in order to fulfil the requirements R3.6.39 and R3.6.40 reflect that XtreemOS-MD should offer two new features related to XtreemFS specially designed for mobile devices and its

particularities: on demand file uploading mechanism and transparent file sharing behaviour respectively.

The auto-mounting mechanism for the XtreemFS volume (R3.6.41) and the management of IP address changes (R3.6.42) will be available and very useful for XtreemOS-MD users.

### 2.1.3 VO management and security

In terms of VO management and security services, the most important feature of XtreemOS-MD is to integrate the VO authentication process within the legacy SSO systems of user's organizations (req. R3.6.43). That feature requires modifications in the CDAProxy component (S3.6.32) and expanding the list of authentication methods supported by the system (S3.6.33).

### 2.1.4 Resource Sharing

Finally, G-layer requires a design of the specific resource sharing features that were identified in D2.3.6 and D3.6.4. The resources that mobile devices can share are data, I/O devices, and network connections.

As a rule of thumb, resource sharing should be easy to use for mobile device users and at the same time robust and reliable. Hence, some common features like volume sharing (spec. R3.6.35), auto-mounting (R3.6.41 & S3.6.31) and managing potential IP address changes (req. R3.6.42, and spec. S3.6.30) should be addressed in all the types of resource sharing.

The bases for resource sharing are developed in the F-layer. Basically, a resource sharing daemon is in charge of providing the necessary APIs to upper-layer applications and also to communicate with proxy components that are executed in trusted nodes of the Grid. For more details about the resource sharing module in F-layer, see D2.3.7.

The resource sharing module shall be completed with specific G-layer features. First we design a mechanism to publish and discover which resources a mobile device is willing to share. Then, we design the access mechanism to those resources, which will depend on the type of resource. For data sharing, several requirements and specifications were covered in D3.6.4: on demand file uploading and transparent file sharing requirements and specifications (req. R3.6.39, R3.6.40 and spec. S3.6.29). In contrast the requirements for I/O device sharing and network sharing come from WP2.3 but the G-layer design shall address them as well so as to provide access from upper-layer applications.

## 2.2 Architecture

Figure 1 shows the architecture of the advanced version of Xtreemos-MD. In comparison with the basic version the main architecture changes are the addition of the context-awareness and resource sharing modules. Context-awareness is fully developed in D2.3.7 whereas resource sharing is developed partially in D2.3.7 and completed in this deliverable.

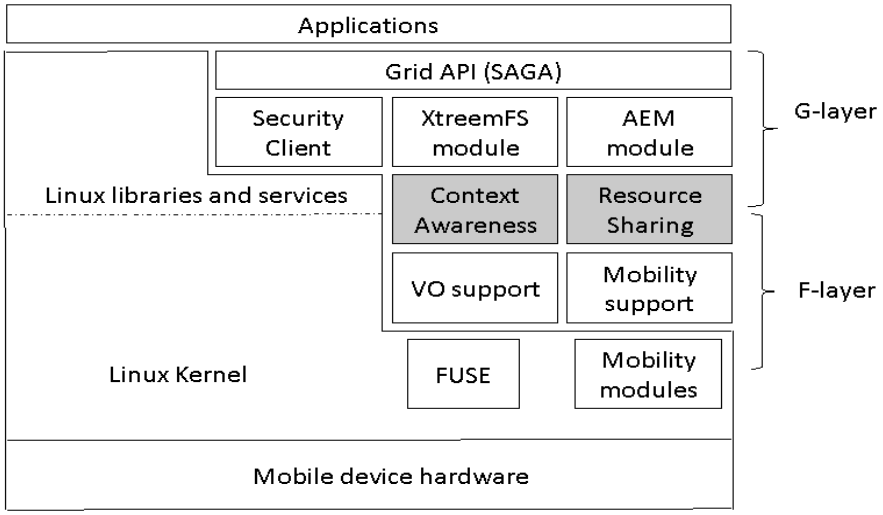


*Figure 1. XtreemOS-MD advanced version architecture*

The following chapters provide the specific design grouped by the main services: Security, AEM, XtreemFS and Resource Sharing.

# 3 Security services

In the context of security services, the advanced version of XtreemOS-MD G-layer will be adapted to the new CDA server of XtreemOS in order to guarantee the compatibility with the security services architecture of XtreemOS as a whole as detailed in D3.5.13 [10]. In this section we will focus on the design of the new security features provided. Unless otherwise stated, the security features of XOS-MD basic version will also be available on XOS-MD advanced version.

## 3.1 Features and functionalities

The first step in our design is adapting the *CDAProxy* module and the *libcdaclient* library to support the changes introduced in the last version of the CDA server. These changes do not imply major architecture modifications and the design just follows the general XtreemOS security architecture.

Besides, XtreemOS-MD advanced version will provide a better integration with the SSO implementation of the user's enterprise system (or with the SSO of the user's ISP), according to requirement R3.6.43 of deliverable D3.6.4. This will be enforced through the *CDAProxy*. The design is based on adapting the *CDAProxy* to support PAM modules using the Linux *libpam* library. Thanks to PAM modules, the *CDAProxy* will support the use of an enterprise LDAP server to authenticate users. Using CDAProxy with PAM support offers these advantages:

- The enterprise has full control of security and accountability aspects, because users need to authenticate directly against the enterprise *CDAProxy*, not against an external entity that is out of the scope of the enterprise administrator, the *CDAServer*. This is important, because one of the possible fears to externalize services to a grid-based infrastructure is a certain lack of control. Another reason is that network security policy of the organization may differ from security policy of the VO, because a VO involves several organizations.
- Users enjoy SSO advantages and it avoids "password fatigue": there are no separate authentication scheme to access local enterprise resources and another one to access grid resources. This security unification simplifies the migration of services to a Grid platform based on XtreemOS, because local services and grid services may coexist with a higher level of transparency to users.

We should highlight that, although this feature is specific to XtreemOS-MD flavour, the solution is portable and usable from the XtreemOS PC flavour, because no mobile specific components are required for it. The changes done in mobile device consist on including *LibPAM* support, but *LibPAM* is already used in XtreemOS PC flavour.
Finally, a SSO system is currently under consideration in XtreemOS. The main expected are delegation and multiplexing connections over a unique master SSH authenticated connection. When this new SSO system is completely defined, XtreemOS-MD will port it to the supported platforms. Besides, the code will be integrated with XtreemOS-MD SSO components. For example, the new SSO client code will be modified to use *libxos_getcred* to get the credential using the modular implementation of XtreemOS-MD SSO, instead of reading the credential from file system and requesting a passphrase.

# 4 Application Execution Management service

In the context of AEM, the advanced version of XtreemOS-MD provides new mechanisms for reservations, jobs with multiple resources, dependencies, advanced monitoring and interactive execution. In this section we will only focus on the design of the new features provided. The AEM features of XOS-MD basic version will be also available on XOS-MD advanced version.

## 4.1 Features and functionalities

As it is derived from D3.6.4, the new functionalities that will be included in the advanced version of XtreemOS-MD (on top of the already existing in the basic version) are:

- Reservations: this will allow a user to book resources in advance. The booked resources can then be used for one or several jobs.
- Job execution in multiple resources: so far, jobs submitted from an MD could only span processes over a single resource, now this limitation will be removed.
- Dependencies: in order to perform some monitoring or management tasks, it is often helpful to have dependency graphs between jobs. In the advanced version of MD, this feature will become accessible to mobile users.
- Monitoring: advanced features such as buffering, user events, and call backs will be added to the basic ones already available.
- Interactive jobs: finally, interactive jobs will also be able to interact from a mobile device.

## 4.2 Architecture and design

The architecture needed to implement all these new functionality is exactly the same as presented in D3.6.2 because most of the new functionalities imply changes on the resources but not on the client. They mainly need some new interfaces that have already been added to the C version of the AEM interface (C-XATI).

Monitoring call-backs and interactive jobs, however, require significant changes on the client as well.

### 4.2.1 Monitoring call-backs

The main problem with call backs is that a persistent connection is needed, but, on the other hand, it is not reasonable to assume that mobile devices will be always connected. For this reason, we need to implement a proxy that will receive all call-back events when the mobile device is not available. Once the device reconnects, all missing events will be delivered. On the other hand, if call-back events are received while the mobile device is connected, then they will be delivered immediately (as expected in a traditional client).

### 4.2.2 Interactive jobs

The support of interactive jobs [2] in XtreemOS is based on SSH-XOS, which adds to SSH the possibility to authenticate users with their XtreemOS X.509 certificates. The management of interactive jobs from client side is implemented as a command of a classical UNIX API: xrun command, which allows submitting interactive jobs and running new commands in an existing job context. In order to extend this functionality to XtreemOS–MD flavour, it will be necessary to port the SSH-XOS system plus the implementation of the xrun command to mobile device flavour.

The rest of the elements and interactions related to interactive jobs are located in the Grid side (AEM and resources nodes) so no other changes in the mobile device implementation are necessary.

The capabilities and limitations of interactive jobs for Mobile devices will be the same l as for the PC implementation of *xrun* (for more information see section 6 of [2]).

# 5 Data management service

## 5.1 Features and functionalities

As it is derived from D3.6.4 [6], the new functionalities that will be included in the advanced version of XtreemOS-MD (on top of those in the basic version) are:

- Using Vivaldi to select replicas: this feature will allow MD to decide which of the available replicas is better given the current location of the device
- Offline mode: This feature will allow the user to explicitly cache a file to be accessible while offline.

In D3.6.4, we also mentioned that in-kernel caching would be part of the XtreemOS MD version, but this feature has been removed from the XtreemFS roadmap, and without this support it is impossible to offer it from the MD (and would make no sense anyway).

## 5.2 Architecture and design

### 5.2.1 Using Vivaldi to select replicas

To summarize the idea of Vivaldi, all nodes in XtreemFS (OSDs and clients) place themselves in a 2D space. The distance between two nodes in this 2D space is related to the communication latency between these two nodes (bandwidth will be added in the future), thus it also reflects the expected speed of accessing file objects. In order to compute this coordinates, OSDs and clients randomly contact other OSDs every few minutes and with the obtained RTT they try to update their position in the 2D space.

The architecture and design needed to implement replica selection using Vivaldi does not need any architectural change compared to what regular clients do. On the other hand, the Vivaldi algorithm was proposed for environments where the latency does not change significantly over short periods of time. As this assumption does not hold for mobile devices, we will first need to see how important breaking these assumptions is in the overall behavior of the algorithm and, if needed, propose mechanisms to overcome this extra “mobility” of the nodes. For instance, we may develop some kind of triangulation mechanisms to be able to find, in a fast way, the new coordinates of the MD that has just re-entered the system. Of course, this will only work if the whole system is already in a “stable” state. Another solution would be to add some extra communication from the MD when changing

position to update its position (though it would mean more network and energy consumption). In addition, it is important to clarify that MD (actually all clients) will never affect the position of servers (OSDs). They will only compute their position against OSDs, but OSDS will never contact a client to update their position (they are too untrusty).

### 5.2.2 Offline mode

The idea of offline mode is to allow MD users to access to files while they are not connected. We have to clarify that this is not an automatic feature, but something the user requests explicitly (i.e. using some kind of application, he marks the file as offline file). In addition, it is the responsibility of the user, and not the file system, to keep the file consistent (though the system will mark the file as read only to avoid modifications by error). Nevertheless, users should know that nobody else will modify these files. If after all, users modify the file, the modifications done from the MD will be the ones taken into account.

In order to implement this feature we have to take care of two different issues. First, we need to download the file to the MD. This can be easily solved by fetching it from XtreemFS once marked as offline and once the file is marked again as on-line, it is written back into XtreemFS as if modified at that time. Another option could be to update it into the general file system as soon as a connection is established. Nevertheless we prefer to give more control on the user.

The second issue is how to allow transparent access to these off-line files while the MD is not connected. The solution we propose is to build a new FUSE file system that only manages offline files. This module should be built on top of the XtreemFS client module, thus users first access the off-line file system, and if the file is not there, the request is automatically forwarded to the regular XtreemFS module. Of course, this is completely transparent to the user/application and is managed internally by the OS.

# 6 Resource sharing service

Resource sharing for mobile devices was addressed in D2.3.5 [3] and D2.3.6 [4] in the context of the Foundation layer support. For the G-layer design, we shall take into account the same considerations: *"Resource sharing for mobile devices must be adapted to the intrinsic mobile device limitations: battery life, reduced computing capacity and small local storage capacity."*

In this chapter we propose a design for the different resource sharing options specifically adapted to the mobile device flavor in XtreemOS. We will consider the three different options for resource sharing previously identified in D2.3.6: data sharing, input/output device sharing and network access sharing.

## 6.1 Features and functionalities

As specified in D2.3.6, data sharing for mobile devices provides a mechanism for mobile users to share their files with other Grid users (mobile or fixed). However, in order to save valuable and limited mobile device resources, the data will be shared on demand: *"XtreemOS-MD advanced version may provide a mechanism to upload files from the MD only when other nodes request the file contents. […] In other words, when a MD user wants to share a file with other users, he will mark the file to be shared but it will not be actually shared until some conditions are met. The first condition is related to network access: users might want their files to be shared only when WiFi access is available and not when 3G access is available. The second condition is related to the moment when other users request the shared file. At that moment, the first time that the file is requested by another user, the file*

*would be replicated to XtreemFS; later on, other users will get the file from the XtreemFS replica. Whenever the original file is changed by the owner it may be updated in XtreemFS."*

Providing this feature requires a new functional entity in XtreemFS that should act as a data provider to the Grid. The first approach could be to run an adapted OSD in mobile device, but this is not possible as mobile devices are not considered as trusted nodes by XtreemOS security framework. In fact, in the XtreemOS context the trusted nodes should only be accessible to the Grid administrators but any mobile devices would always be physically accessible by their owners. Being a trusted node is a necessary condition to be an OSD. Thus, instead we propose to create a proxy entity that will be used by mobile devices for resource sharing. This new entity will be developed and it will run in the Grid not in the mobile. For data sharing, this new entity will behave as an OSD proxy and it will provide a specific interface to mobile devices willing to share data. The next section shows the details and how this proxy OSD is designed.

In the context of I/O devices and network sharing, our design will add new features. Up to now XtreemOS has just considered the sharing of storage and computational power. In order to share additional elements (such as the camera, micro, etc.), we should follow the same ideas; first we need a mechanism to publish mobile device resources and second we need a mechanism to provide access to that resource from remote client nodes. It is noted that although this feature is designed for mobile devices, it could be extended to PC flavor as well, so that the PC devices could also share their IO devices for example.

## 6.2 Architecture and design

As it was shown in Figure 1, there is a specific "resource sharing module" in charge of the different types of resource sharing that could be implemented in the MD. But this is a simple black-box representation of the different elements involved in the resource sharing for MDs, so let's analyze it with more detail.

### 6.2.1 Data sharing

In order to become a "data server", the mobile device should act somehow as an OSD in XtreemFS. The specific security policies of XtreemFS, including being a trusted node to act as an OSD make the implementation of an OSD inside the terminal not possible. Our solution is a "Proxy OSD" that will be in direct communication with the mobile devices and that will form part of the XtreemFS architecture as a special OSD with the lowest priority.

As seen in Figure 2, when the user marks a file as "shared" in his MD (let's say, when the file is moved to a specific folder for file-sharing), a request is sent to the Proxy OSD element, inside a persistent connection (SSH tunnel) that will be kept between the proxy and the MD. The Proxy in turn will contact the MRC in order to request the creation of a replica in another OSD. When a different user requests the shared file, the OSD with the replica will request the content of the file from the Proxy OSD (as the replica is still empty) and then the Proxy OSD will contact the Mobile Device to upload the content of the file. This way, the upload of the data from the Mobile Device would be done just if the content is requested by someone.

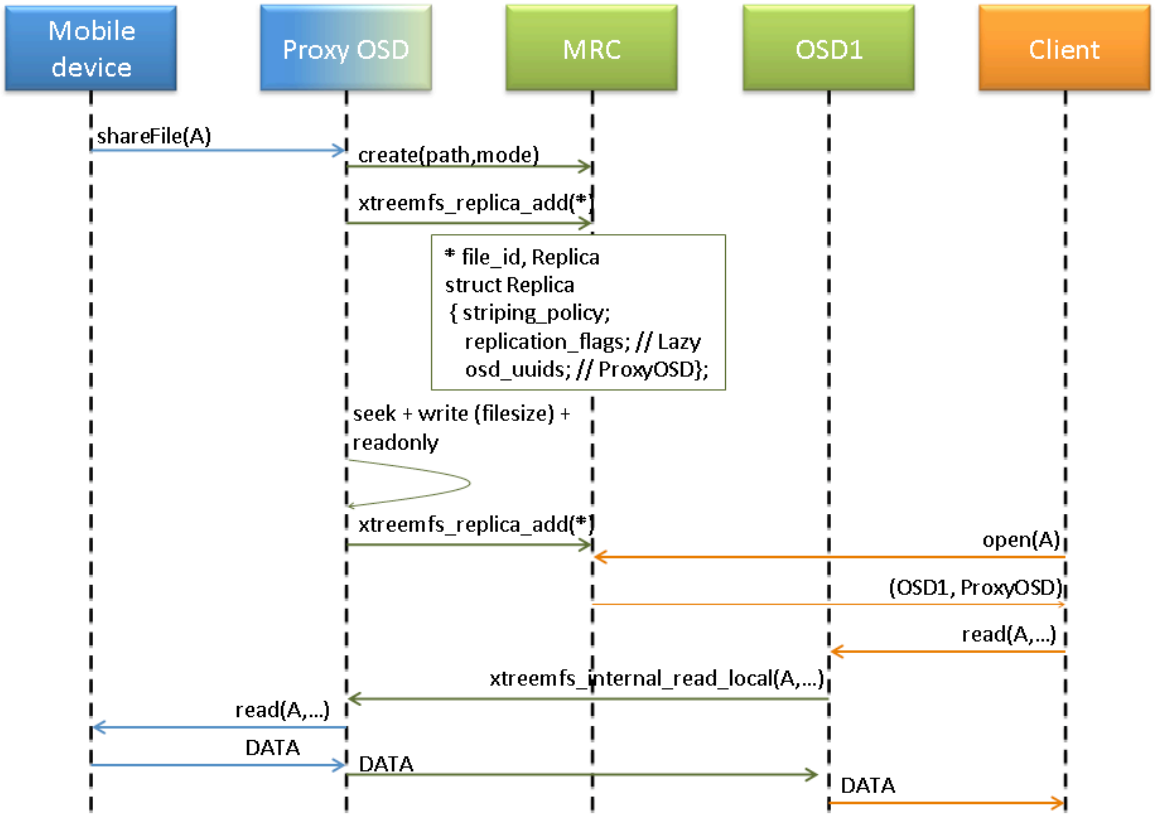


*Figure 2. Data sharing*

### 6.2.2 I/O devices and network sharing

For this kind of sharing we need two different mechanisms: publish/discovery mechanism and an access mechanism.

#### *6.2.2.1 Resource Discovery mechanism*

A publish/discovery mechanism should be used to notify what I/O devices a mobile phone is sharing, their characteristics and how to reach them. Basically we need a generic *rendezvous* service. This service could be implemented in a centralized way or following a distributed architecture. If we wish to follow a distributed approach suitable for Grids, then a rendezvous distributed service may be accomplished in several ways. XtreemOS itself provides the SRDS (*Service/Resource Discovery System*) [7][8] that offers to applications and other components of XtreemOS the capability of searching for and selecting services and resources. The SRDS is composed of two services, the ADS (*Application Directory Service*) and the RSS (*Resource Selection Service*). Both of them have a distributed implementation, based on P2P overlay networks.

Thus, this design is based on using this overlay infrastructure from the mobile device but this time to publish/discover I/O device information instead of general device characteristics (CPU, memory, etc). Using this approach the mobile device will communicate with an XtreemOS node where the SRDS service resides in order to publish or discover I/O shared devices. Figure 3 shows the architecture for resource discovery mechanism using a trusted node running SRDS and AEM

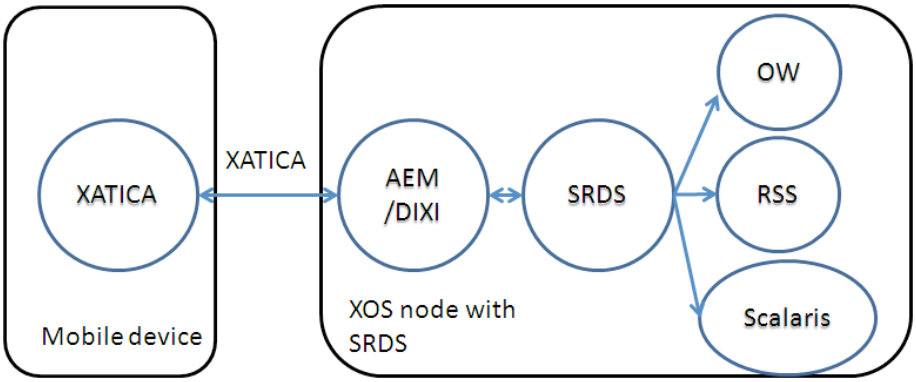


*Figure 3. Resource Discovery architecture*

#### *6.2.2.2 Access mechanism*

The second mechanism we need is related to giving access to the shared resource. For I/O device sharing the access mechanism is provided by the F-layer (*Resource Sharing Daemon*) that was already detailed in D2.3.7 [5]. Once the access is granted to an I/O device, in general the way a resource is used will depend on the type of resource. For instance, the way of sharing a GPS will not be the same as the way a multimedia device (voice, camera) may be used, etc. For instance, a remote client may request to start capturing GPS traces in a shared GPS and save the results into an XtreemFS file. In contrast, the required command to capture a picture with a shared camera and save the picture will be different. Hence, I/O device sharing is based on F-layer support and upper-layer applications will make use of the remote resource by specific means that will depend on the shared resource.

On the other hand, for network sharing, F-layer does not provide specific support, so this feature will provided by G-layer. The access mechanism we propose for network sharing is based on connecting both ends of the communication with a virtual private network that is created dynamically. The sharing device will act as the server side of the VPN and the consumer as the client side. There are many different technologies to create VPNs (layer 2 and layer 3 VPNs). Given that VOs will typically be connected at IP layer level we propose to use IP-layer VPN. In this context, OpenVPN [9] is a widely used technology and it is available for many operating systems. OpenVPN is also available in some Linux-based mobile devices (Maemo, for instance). In addition, OpenVPN allows the server side to provide access to more than one client (star topology) which is particularly useful for the Grid scenario. Using OpenVPN a sharing device may provide one network connection (for instance, 3G, Bluetooth, etc.) to be shared while the consumer (a PC or an MD) will have a mechanism to connect to the sharing device (using WiFi or LAN for instance) and then getting access to a foreign network.

## 6.3 Specific modifications for Mobile Devices

As it's derived from previous sections, there are significant modifications on the design for mobile device:

- The data sharing requires implementing a protocol between the proxy OSD and mobile G-layer in order to share a file and upload it upon request.
- The I/O device and network sharing requires first an adaptation of mobile modules to publish resources and to discover remote resources using SRDS service. In addition, mobile devices will include the *openvpn* stack in order to act as a servers or clients on the VPN connection.

## 6.4 Examples of use

Resource sharing on mobile device provides new scenarios in XtreemOS. Some of them are:

- A MD user will use the native file manager in his mobile device to move the file that he wants to share (for example a photo) to the folder "OnDemandFileSharing". The content of the file will not be uploaded to the Grid at that moment, but just when another user will try to access the shared file.
- MD user A has got an XtreemOS smarthphone with WiFi and 3G interfaces. The mobile device is connected to the Grid using its WiFi interface. User A activates the 3G connection to be shared. Other users of the same VO request a direct connection to the Internet and query the system to find out available access points. The shared 3G connection is found on mobile device of user A and a VPN connection is created where mobile device A is acting as the server. This is useful in a meeting, for example, in order to share a single 3G connection to access the Internet with every MD user, while they will use their free WiFi connections to access to the "3G access point".

# 7 Conclusions

In this document we have reviewed the design of the advanced version of XtreemOS-MD G-layer. The design comes from the requirements compiled in deliverable D3.6.4 [6], and the design principles of the layer F have also been taken into account, as some architectural modules are part of both F and G layers. The main new features, compared to the basic version, covered by this design are:

- A SSO system based on the "control master" feature provided by *openssh*.
- Data management features: using Vivaldi to select replicas in XtreemFS and file caching for offline mode support.
- AEM improvements related to call-back monitoring and interactive jobs.
- Resource sharing capabilities allowing the mobile device to share data on demand (acting like a fake OSD), and to share the access to the I/O devices and network access provided by the mobile terminal.

It is especially relevant the addition of resource sharing capabilities to XtreemOS-MD, as is makes possible the inclusion of the mobile devices as special resource nodes of the Grid, and not just Grid clients like with the XtreemOS-MD basic version. As commented in deliverable D2.3.6, where part of the resource sharing design was tackled, the mobile device won't be a conventional resource node of the Grid, as taking into account the XtreemOS security model, it cannot be considered a trusted node. That is the reason to include a special trusted proxy in the Grid, which will be in charge of the direct communication with the resources-sharing mobile devices.

# 8 Future work

The next phase towards the completion of the advanced version of XtreemOS-G for mobile devices is the implementation itself as a first internal release. Then, after a period for testing and integration with the existing XtreemOS components (core node, resource node, etc), the final release of the XtreemOS-MD advanced version code should be ready. All the software (F layer and G layer) will be packaged in cooperation with WP4.1, for both smartphones and PDAs, in order to have the final packaged version of XtreemOS-MD by the end of the project.

A more detailed future work is as follows:

1. The XtreemOS-MD version 1.0 (G-layer), available for PDAs, shall be ported to mobile phones.
2. At the same time the G-layer services should be adapted to the new XtreemOS release, which at least means recompiling XtreemFS and XATICA client modules and modifications on security modules.
3. Then, we will work on the development of the new designed features as described in this deliverable, working in parallel in all the existing modules and the new resource sharing module.
4. The very last step will be porting XtremOS-MD to netbooks once all the software pieces will be stable enough. This porting is expected to be available by the end of the project.

All these WP3.6 developments will be part of the last deliverable of this work package: D3.5.6 “Implementation and optimization of advanced services in mobile devices (XtreemOS-G for MD/MP)”.

# 9 Bibliography


[1] XtreemOS Consortium. *Deliverable 3.6.4. Requirements and specification of advanced services for mobile devices*. June 2009.

[2] Xtreemos Consortium. *XtreemOS Technical Report: Supporting interactive jobs in XtreemOS.* August 2009.

[3] XtreemOS Consortium. *Deliverable D2.3.5. Requirements and Specifications for Advanced VO Support in Mobile Devices.* November 2008.

[4] XtreemOS Consortium. *Deliverable D2.3.6. Design of an advanced Linux version for mobile devices. October 2009.*

[5] XtreemOS Consortium. *Deliverable D2.3.7. Linux-XOS for MD/MP.* 2009.

[6] XtreemOS Consortium. *Deliverable D3.6.4. Requirements and specification of advanced services for mobile devices.* June 2009.

[7] XtreemOS Consortium. *Deliverable D3.2.4. Design and Specification of a Prototype Service/Resource Discovery System.* November 2007.

[8] XtreemOS Consortium. *Deliverable D3.2.8. Reproducible evaluation of a service/resource discovery system.* January 2009.

[9] OpenVPN technologies. *OpenVPN documentation.* http://openvpn.net/index.php/open-source/documentation.html. Last visited November 2009.

[10] XtreemOS Consortium. *D3.5.13. Fourth Specification, Design and Architecture of the Security and VO Management Services.* 2009.

# A. Appendix: Acronyms and abbreviations

| | |
|---|---|
| AEM | Application Execution Management |
| FUSE | Filesystem in Userspace |
| JSDL | Job Submission Description Language |
| LDAP | Lightweight Directory Access Protocol |
| MD | Mobile device |
| OSD | Object Storage Devices |
| SSH | Secure SHell |
| SSO | Single Sign-on |
| VO | Virtual Organization |
| XATICA | XOS Application Toolkit Interface C Adaptation |
| XtreemFS | XtreemOS File System |