Software Requirements Specification

for
AzureusHistory Plugin

Version 1.2

Bhardwaj Sandeep
Giannikis Georgios
Jianyuan Li

ETH Zürich

April 24, 2008

Revision History

Name	Date	Reason For Changes	Version
Georgios Giannikis	18.04.2008	Initial (Short) Version	1.0
Sandeep Bhardwaj	24.04.2008	SRS version	1.1
Jianyuan Li	24.04.2008	introduction part	1.1.1
Georgios Giannikis	24.04.2008	Tests	1.1.2
Sandeep Bhardwaj	24.04.2008	Requirements	1.2

Table of Contents

1. Introduction 5
    1.1 Purpose 5
    1.2 Scope 5
    1.3 Definitions, acronyms and abbreviations 5
    1.4 Glossary 6
    1.5 References 7
    1.6 Open Issues 7
2. Overall Description.....7
    2.1 Product Perspective.....7
    2.3 Design and Implementation Constraints 9
    2.4 Assumptions and Dependencies 9
    2.4.1 Development Environment 9
3. Specific Requirements 10
    3.1 Functional Requirements 11
    3.2 Non-Functional Requirements.....12
4. Appendix.....12
    4.1 Tests 12

1. Introduction

Azureus is one of the most popular BitTorrent clients written in Java. It allows users to download multiple files in a single graphical user interface (GUI). The program also features detailed statistics and a large number of user-configurable settings. Azureus is free software. At present, many useful plugins are available. The main goal of developing our plugin is to enhance the user experience by giving him/her the possibility to manage the torrents more effectively.

We choose the Gantt Chart as a visualization tool since its usage is well understood and it's highly effective in the project management area. A Gantt chart is a popular type of bar chart that illustrates a project schedule.

We choose HSQLDB as the persistent data storage of the plugin. HSQLDB is a relational database management system written in Java. It offers a fast, small database engine which offers both in-memory and disk-based tables.

1.1 Purpose

This document specifies the Software Requirements Specification (SRS) for the plug-in for Azureus Application to show the download History and a Gantt Chart. It describes the scope of the system, functional, non-functional, quality and development requirements for the software, design constraints and system interfaces.

1.2 Scope

The name of our software product to be produced is *Azureus History & Scheduling Plugin*.

The goal of this project is to create a plug-in for the Azureus Java Application that will be used to Manage the history of the downloaded files and provide priority scheduling mechanism for downloads. The visualization of the history will be done using a dynamic Gantt Chart which will give the user the ability to graphically modify the priorities of the current downloads. The plug-in shall be used by the open source community.

1.3 Definitions, acronyms and abbreviations

DBMS	Database Management System
HSQLDB	Hypersonic SQL Database
GUI	Graphical User Interface
SRS	Software Requirement Specification
SWT	Standard Widget Toolkit, a graphical widget toolkit for use with the Java platform.
P2P	Peer-to-peer

1.4 Glossary

The glossary defines the key terms and concepts mentioned and used in this SRS.

BitTorrent BitTorrent is a peer-to-peer file sharing (P2P) communications protocol. BitTorrent is a method of distributing large amounts of data widely without the original distributor incurring the entire costs of hardware, hosting, and bandwidth resources. Instead, when data is distributed using the BitTorrent protocol, each recipient supplies pieces of the data to newer recipients, reducing the cost and burden on any given individual source, providing redundancy against system problems, and reducing dependence on the original distributor.

Azureus A BitTorrent client. Like other BitTorrent clients, it is used to transfer files via the BitTorrent protocol. Azureus is written in Java programming language using the SWT library.

Plugin A plugin is a computer program that interacts with a host application (a web browser or an email client, for example) to provide a certain, usually very specific, function "on demand".

Peer A client downloading or uploading a set of files using the bittorrent protocol

Peer-to-peer A type of ad-hoc computer networks.A peer-to-peer computer network uses diverse connectivity between participants in a network and the cumulative bandwidth of network participants.

Tracker A server which assists in the communication between peers using the BitTorrent protocol.

Torrent File A small file (typically a couple of kB) containing metadata about the shared files and about the tracker. The torrent file is shared by the tracker and is necessary to start downloading/sharing a file. The term "torrent" is also ambiguously used by users to describe the whole procedure of downloading a file.

Gantt Chart A Gantt chart is a popular type of bar chart that illustrates schedule.

1.5 References

- Azureus Website

Official website of Azureus: http://azureus.sourceforge.net
AzureusWiki: http://azureuswiki.com/index.php/Main_Page

- Gannt Charts

Wikipedia about Gantt chart: http://en.wikipedia.org/wiki/Gantt_chart

- HSQLDB

Official website of HSQLDB: http://www.hsqldb.org

1.6 Open Issues

2. Overall Description

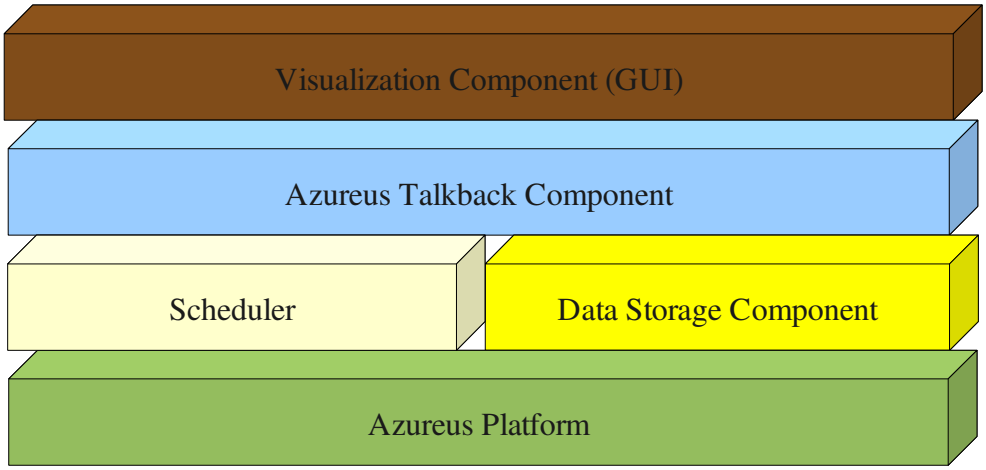
Azureus implements the BitTorrent protocol and its written in Java, it has many invaluable features for both beginners and advanced users. The tool has a very large user base. Our main goal is to enhance the user experience by giving him/her the possibility to manage the torrents more effectively.

We choose the Gantt Chart as a visualization tool since it's usage is well understood and it highly effective in the project management area.

2.1 Product Perspective

The project will be divided in four discrete components. Each participant on this project will be responsible for one component and the last one will be developed by all three participants. Each participant will also be responsible for maintaining the javadoc and a clean and reusable interface for the component he/she is responsible for.

The inner architecture of this plugin is illustrated in the following figure:



- **Azureus talkback component:**

This component is responsible for providing a simpler API of the Azureus interface. It should be able to read the status the current downloaded torrents and provide some basic info regarding them, i.e. the download/upload limits for each torrent, the different priorities they currently have and so on. It should provide some status of the Azureus platform in general, such as the current download speed and the max download speed.

- **Data Storage component:**

This component serves as the persistent data storage of the plugin. We are planning to use HSQLDB[3] as a storage mechanism. HSQLDB runs in the same process with the JVM and is able to be started and stopped by the JVM itself. In this case it needs no TCP/IP configuration, since it doesn't operate as a normal network database. This component should also be able to pull data from the Azureus talkback component. This requires implementation of a listener interface that has to be notified by the previous component. The API of this component should provides methods for data extraction without using SQL commands.

- **Scheduler component:**

This component is responsible for scheduling the queued downloads taking into consideration priorities provided by the user. The scheduling algorithm should consider the download rate and the download fluctuations. These data should be provided by the Data Storage Component.

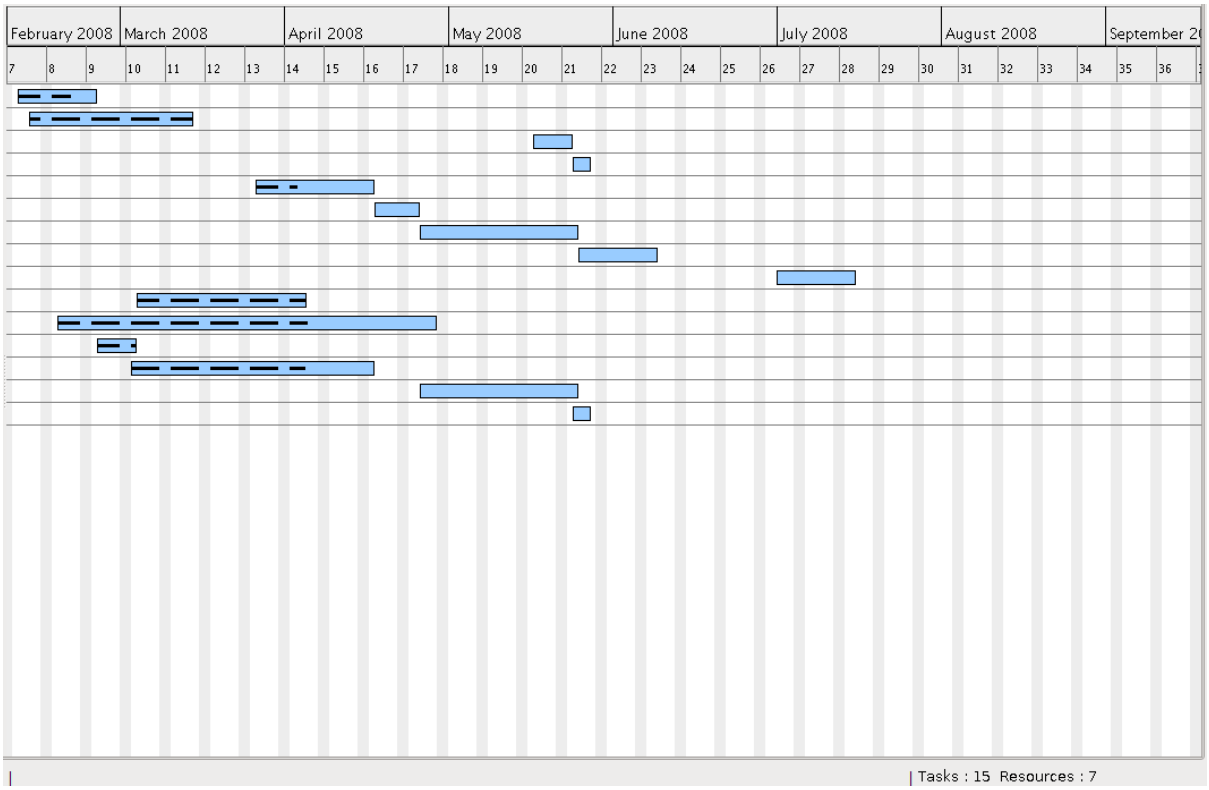
- **Visualization component:**

This component's role is to display the download history and provide the user with a rich user interface which will enable assigning priorities to torrents and/or

discrete files within torrents with a higher granularity than the existing azureus scheduler (“Do not download”, “Normal”, “High”). This will be implemented using a Gantt Chart visualization. The component will be build using the swing widget toolkit. The whole plugin should appear as an extra Tab on the Azureus GUI.

2.2 Product Functions

The plugin should provide a new view in the Azureus application. This view will resemble a gannt chart used in monitoring and planning a set of predefined tasks.



The provided GUI should add functionality to easily manage priorities of current downloads.

2.3 Design and Implementation Constraints

The plugin must fit the Azureus application. It must use as much as possible of the Azureus API, while minimizing external dependencies. As a consequence using an external RDBMS as persistent storage is not allowed. For the GUI we have

to use the SWT graphical toolkit since it is also used by Azureus.

2.4 Assumptions and Dependencies

2.4.1 Development Environment

The development environment chosen is the Eclipse IDE 3.3. For the GUI design we will use the Visual Editor Plugin (VEP) for Eclipse. Since VEP supports only Eclipse 3.2, we are going to use eclipse 3.3 for writing the non-GUI code and 3.2 for composing the view of the GUI part.

3. Specific Requirements

All requirements are defined using the following template:

Identifier	**Title**	**Priority**
Input		
Prerequisite		
Description		
Postcondition		

Identifier (ID) The identifier is a unique identification given to every requirement which shall be used throughout the project for cross-referencing and traceability.

Title The title is the name given for each requirement.

Priority The systems priorities are divided into 3 main categories:

1. *Essential:* All the requirements that must be implemented for the first release.
2. *Conditional:* Requirements that provide an extra functionality but may need further negotiation.
3. *Optional:* Requirements that are desirable but may be out of

scope.

Input Describe what information the functionality needs in order to function.

Prerequisite The prerequisite define the minimum requirement that shall be met in order to call that functionality.

Description The description provides a brief description of what the functionality shall do.

Postcondition Provide a description of the outcome of the execution of that functionality.

3.1 Functional Requirements

FN-01	**Polling Service**	**High**
Input	Azureus plugin interface	
Prerequisite		
Description	Poll the azureus plugin interface and retrieve statistics about the current downloaded torrents.	
Postcondition	The persistent storage component should record the average speed of each torrent. The visualization component should display the data.	

FN-02	**Attach Torrent Listeners**	**Essential**
Input	Append Listeners to the Azureus Platform to notify the plugin	
Prerequisite		
Description	Add listeners to the plugin interface to notify the plugin for certain events. Such events are torrent additions, deletions and pauses. Additionally listeners should notify the plugin of shutdown process	
Postcondition	The persistent storage component should record the average speed for each torrent The visualization component should provide a way to clearly	

	present the download speed fluctuation over time

FN-03	Schedule torrents	Essential
Input	Scheduling Component	
Prerequisite	Data collection by the persistent storage component.	
Description	Reschedule all incomplete torrents according to user input and average download speed.	
Postcondition	Download of torrents is rescheduled as displayed in the Gantt Chart	

3.2 Non-Functional Requirements

NF-01	Data Quantization	Essential
Input	Data from the Azureus Talkback Component	
Prerequisite	The user has started downloading some torrents	
Description	Statistics have to be quantized over time. We choose a time quantum of 1 minute.	
Postcondition		

NF-02	Data Recovery	Optional
Input	The history is stored on the Local Machine using the HSQLDB	
Prerequisite	The user shall add the plug-in in the Application	
Description	The download history is saved to the user's personal folder in a plain text file (or a gzip file).	
Postcondition		

4. Appendix

4.1 Tests

TST-01	**Data Capturing & Storing**	**Essential**
Input	Azureus Platform	
Prerequisite	FN-01, FN-02	
Description	Check if data is correctly captured by Azureus History plugin. Check will be performed by querying HSQLDB.	
Postcondition	HSQLDB should contain the data for all current torrents.	

TST-02	**Data Presentation**	**Essential**
Input	Azureus Platform	
Prerequisite	TST-01, FN-01, FN-02	
Description	Check if the visualization component correctly presents data stored in HSQLDB.	
Postcondition	Visualization component should display a flawless gannt chart.	