Funky Torrents

Software Requirements Specification

Version 1.12
IT University of Göteborg
TIG029 – Software Architecture for Distributed Systems

Revision History

Name	Date	Reason For Changes	Version
M.Bergqvist, B.Eriksson	2010-11-01	Original draft	0.1
M.Bergqvist, B.Eriksson	2010-11-08	Updated non-functional requirements	0.2
M.Bergqvist	2010-11-10	Updated use case diagram	0.3
B.Eriksson M.Bergqvist	2010-12-20	Made the SRS conform to the delivered prototype	1.0

Table 1. Revision history

1. Introduction

1.1. Purpose

This SRS is for the Funky Torrents bittorrent client software. Funky Torrent is ordered by the local library to support the distribution of electronic publications to the library's customers.

1.2. Glossary

SAD	Software Architecture Description, another document within this project.
SRS	Software Requirement Specification, this document

1.3. Intended Audience and Reading Suggestions

This document is mainly intended for the development team of the Funky Torrents bittorrent client software, a.k.a. The Group, and the acquirer of the software. It is also directed to maintainers and testers as well as to the project management.

The SRS is to be used to get an understanding of the different attributes of the software as well as being a source for discussion and agreement between the acquirer and the project group when it comes to the requirements.

Reading suggestions:

- Developers
  - Functional Requirements
  - Non-Functional Requirements
  - External Interface Requirements
- Acquirer
  - Functional Requirements
  - User Interface Specifications
- Project manager
  - Functional Requirements
  - Non-Functional Requirements
- Maintainers
  - Functional Requriments
  - Non-Functional Requirements
  - External Interface Requirements

- Testers
  - Functional Requirements
  - Non-Functional Requirements
  - External Interface Requirements

1.4.Product Scope

Funky Torrents is a bittorrent client for downloading electronic publications ordered by the local library. Distribution via bittorrent technology is an efficient way to reduce the load on the library's own servers but at the same time give the library customers a fast downloading speed. Funky Torrents should be easy to use software implemented using Erlang programming language.

2. References

- The Hexa Library tracker URL: 85.228.185.132:8001/tracker/
- Bit torrent specification: http://wiki.theory.org/BitTorrentSpecification
- This SRS template: bluebird.mvsu.edu/~kudikyala/Teaching/CS455/srs_template-MVSU.docx

3. Overall Description

3.1.System Environment

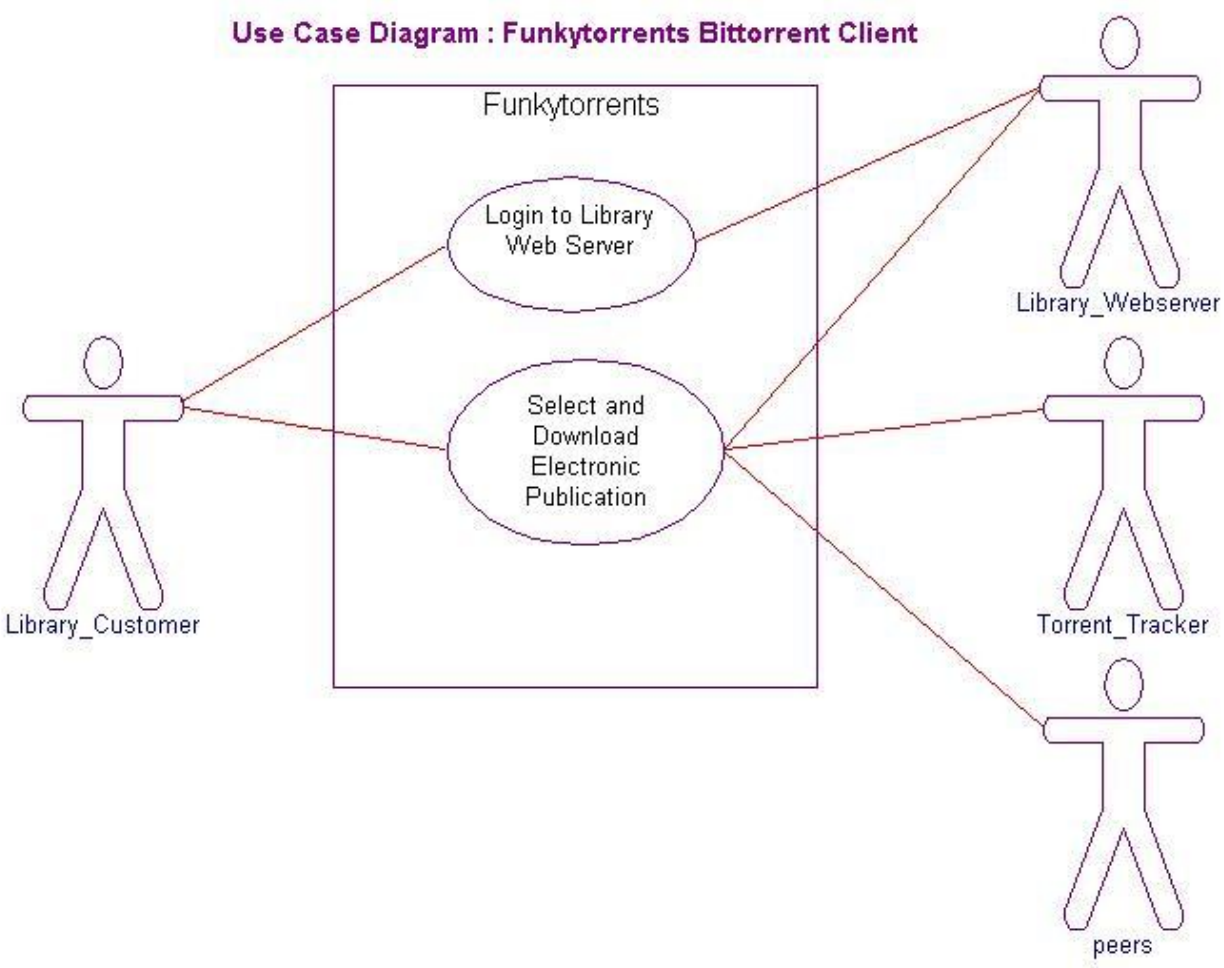

Figure 1. Use case diagram of Funky Torrents

In this Use Case diagram it is important to understand what the different actors represents in reality. At first, since Funky Torrents at the moment lacks the opportunity to upload electronic publication files to other peers, the peers in this diagram represents library customers that use other bittorrent client software to share electronic publications.

The torrent tracker used at the moment is owned by the Hexatorrent Library and can be found at 85.228.185.132:8001/tracker/. They have been kind enough to give The Group full access to the tracker.

The library web server is at this prototype stadium run locally at the same computer as Funky Torrents. In the shipping package both the web-UI and the Chicago Boss html-erlang interface are included.

The full use case description can be found in the Software Architecture Description (SAD) document for the Funky Torrents bittorrent client.

The “black box” system is the actual Funky Torrents bittorrent client.

A library customer is a customer at the local public library that ordered this software. Login details to the library web server will be gained at the local library.

4. Requirements Specifications

4.1.External Interface Requirements

Reference: http://wiki.theory.org/BitTorrentSpecification

4.1.1. Software Interface:

Funky Torrents use Chicago Boss (Framework for Erlang) to integrate the HTML code with the Erlang code for the web application.

Funky Torrents runs on top of an Erlang virtual machine, the operating system underneath should not matter.

4.1.2. Communications Protocols:

Funky Torrents following the BitTorrent protocol which includes the tracker’s HTTP/HTTPS Protocol and the Peer Wire Protocol (TCP/IP).

4.2.Functional Requirements

The functional requirements for the Funky Torrents bittorrent client can be seen in the Software Architecture Description (the SAD). See the use cases in the scenario view.

4.3. Non-Functional Requirements

- **Electronic publication restrictions:**

The distributed publication can only consist of one single file each, and maximum file size of 15MB.

- **Peer requirements:**

There must be at least one peer providing the full electronic publication.

- **Communication:**

All external communication with tracker and other peers must follow the bit torrent protocol specification; Funky Torrents only provides downloading functionality, no seeding.

- **Peer-id check:**

After receiving a list of available peers from the tracker, Funky Torrents shall check the peer-ids so that it doesn't try to connect to itself.

- **Prepared for future updates:**

Funky Torrents' source code should always be written with future updates in mind.

This includes well commented code describing each function.