AFS dDesign (draft 1) with feedback from Tim Blokdijk

AF/Tom Nowell 31/07/07

Index

1 Introduction

This document describes ... it's intended for people that ...
~~AFS/AF Lobby server, not to be confused with AFLobby Server~~, is intended to address the ~~flaws and problems in Betalords~~ original TASServer implementation. The name AFS was chosen because AF Lobby Server is both too long ~~and naming the project after its founder isn't always the best idea.~~

The general design goal of AFS is to accomplish all lobby related needs in as generic and flexible a way as is possible, in order to future proof the program and simplify its construction. To that end, the base protocol used by the server ~~will have no context, the context of which will be provided by the sender and receiver of the messages.~~

What are the lobby related needs?!!

2 Basic Design

AFS will be composed of a basic traffic manager, routing a small group of base protocol commands and managing logins and logging out. Data about ~~objects~~ will be stored in basic key,value tables with no predefined data. The data these tables store will depend on the tables name and context, and will be directly accessible to the components of the lobby system by name. Tables themselves will be keys in a 'root' table, and can store references to other table keys to save memory.

The ~~context based functional aspects of the system~~ will run in self contained modules, each acting as a plugin in the overall framework, and can be independently updated, restarted, or stopped. Channels, battles, and player properties shall be handled by such modules, with the server itself simply acting as a traffic router. The modules themselves will have addressable names just like any user name, with an 'address' being able to receive messages or send them. This way a battle manager module can present individual addresses to battles rather than complicating its own sub protocol.

The protocol will use newlines for each message like TASServer to separate messages, the separation of parameters within commands however has yet to be decided. TCP is being used, however UDP services may be offered for NAT via server modules.

3 Multiple Protocol Support

To support multiple protocols, each users traffic will be tagged as belonging to an ID representing a supported protocol, the default protocol simply passing data on to be routed to the necessary components. A module will be able to define a custom router, to route protocol specific messages to the right modules when found. If the message isn't routed then it is either dropped or an error message sent back using that protocols standards, otherwise the message is forwarded to the appropriate module.

The final version of AFS should aim to encapsulate IRC for chat purposes, TASServer for backwards compatibility, and the new protocol for future development. TASServer support could be initially implemented by taking TASServer and putting it into a module, removing the connection code, and replacing it with data ~~fed by the main server system~~, modifying it in order to support the other modules for chat etc.

4 Initial Development and Testing

Initially, with TASServer being integrated into a module, the server will have all of TAS Servers features, allowing tasclient and AFLobby to use AFS unmodified. The new modules can then start to replace the TASServer functionality while implementing the new protocol, until the entire functionality of TASServer has been replicated and it can be removed.

5 Lobby Support

Its not expected that tasclient would be modified to support the new protocol, backwards compatibility should be maintained however to ease the transition of auto hosts and bots to the new protocol. AFLobby should be expected on the new protocol, and Spring Lobby has the necessary multi protocol infrastructure to allow support.

6 Game Engines

Command Engine needs such a server because of its high flexibility demands for which TASServer does not suffice.

Spring would need such a server in order to implement several long standing requests such as integrated ranking systems and galactic warfare. The proposed design would allow a 3rd party to test and manage such an implementation within a custom module, which after ratification and testing could be introduced to the main server ~~without restarting.~~

GLEvo (Glest evolution) has need of a lobby once they have multi player support completed.

The Populous 3 community also runs on a very old and primitive VB6 matchmaker with an abysmal User interface and could benefit greatly from a spring style server lobby system.~~,~~

OTA and Supreme commander could be added through the lobby itself, however GPG Net and phoenix worx fill those niches already.

One of the TA3D developers has a lobby system that has yet to be put to good use ready for when TA3D multi player support is perfected.

7 Moderation and Access Controls

It has been requested by SwiftSpear that Linux style user access controls be possible. The ability to chmod a protocol command based on a user or user type. Thus all modules will have to register the commands they support. Users will be put into 3 categories of users, moderators and administrators and the third?, and all addresses will have a basic table specifying lists of users and user types that cannot send or receive a certain command from their address. Tables will also have such data stored in them.

8 The core protocol commands

There will be 4 basic traffic commands:

MSG – A command or message containing data

GET table.key – retrieve a value

UPD table.key=newvalue – Add/Edit a key/value pair

RMV table.key – remove a value/key

9 To-do

- Write this document so that all people involved can understand it
- Define message formatting
- Define login commands
- Devise a clean method of protocol differentiation between IRC, TASServer, and the new protocol
- Discuss the implementation of the basic set of server modules and their protocols