Requirements and Analysis Document for ASTRO

Table of Contents

Version: 0.1

Date: 2012-03-14

Author: Daniel Malmqvist, Kristian Sällberg, Mattias Markehed, Erik Ramqvist

This version overrides all previous versions.

1 Introduction

This section gives a brief overview of the project.

1.1 Purpose of application

We want to offer an alternative to private day trading, where someone who's financially and programing savvy buys and sells financial instruments at desired intervals and times. Ideally, the same day trader could write one or several trading algorithms, load them into the ASTRO environment and not having to do all the "dirty work" themselves. ASTRO is a level of abstraction for traders.

1.2 General characteristics of application

Most of this application's functionality will not be visible to the end user, it will make heavy use of databases and user created algorithms. It will include HTML and possibly json parsing, depending on what data sources end users want to use.

The GUI will display real-time stock quotes and price graphs. Traders will be able to track their own portfolios and algorithm performance as they run. Optionally we will add support for intervening and placing buy or sell orders that are not done by the robot.

1.3 Scope of application

Our focus is not to design algorithms. It is to have a framework for traders, where the users can design algorithms themselves. We will not focus on defending against security threats.

1.4 Objectives and success criteria of the project

1. Private traders should be able to use our framework and extend it for their own purpose.
2. The program should be able to run by itself and not by commands from the user.
3. Standalone working algorithmic stock trading framework with database, for private people to use for free.

1.5 Definitions, acronyms and abbreviations

ASTRO - Automated Stock Trading RObot
GIT - Version control system
GUI - Graphical user interface
Java - Platform independent
JDBC - Java database connection
JRE - Java runtime environment
JUnit - Testing framework for java
MVC - Model View Controller

2 Requirements

In this section we specify all requirements

The framework will function on most platforms since it will be written in java.
However it will also depend upon a database, which is not included.

2.1 Functional requirements

Create a list of high level funtions here (from the use cases).

2.2 Non-functional requirements

Possible NA (not applicable).

2.2.1 Usability

x

2.2.2 Reliability

x

2.2.3 Performance

x

2.2.4 Supportability

x

2.2.5 Implementation

x

2.2.6 Packaging and installation

x

2.2.7 Legal

Anyone using this program “IRL” must understand that it could lose them money!

2.3 Application models

x

2.3.1 Use case model

UML and a list of UC names (text for all in appendix)

2.3.2 Use cases priority

A list

2.3.3 Domain model

UML, possible some text.

2.3.4 User interface

Text to motivate a picture.

2.4 References

APPENDIX

GUI

Domain model

Use case texts