Requirements and Analysis Document for ASTRo

Table of Contents:

Version: 1.0

Date: 2012-05-22

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 offer an alternative to private day trading, where someone buys and sells financial instruments to make their living. With ASTRo, anyone who knows basic programming and Java should be able to use ASTROs wrapping API to implement their own algorithms, rules of trading. Ideally, the person who previously spent time doing manual trades could write one or several trading algorithms, load them into the ASTRo environment and not having to do all the “dirty work” themselves. ASTRo aims to be 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 parsing, to be able to fetch data from vendor websites.

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.

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. ASTRo runs it’s assigned algorithms when it gets a signal from the Harvester.

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.
4. The user should be able to visually interact with the trading application through a GUI.
5. ASTRo should be able to fetch stock prices from a given data source and display it to the user.

1.5 Definitions, acronyms and abbreviations

Algorithmic trading - Using computers to make deals in the stock market
ASTRo - Automated Stock Trading Robot
GIT - Version control system
GUI - Graphical user interface
Java - Platform independent

2 Requirements

In this section we specify all requirements

ASTRo is required to work on all major computer platforms.
ASTRo is required to work with several major database providers.
ASTRo is required to be stable, in the sense that trading should not be interfered with because of erroneous behaviour. The trader should be able to rely on ASTRo with the cash he/she invests.
ASTRo's model (where the data logics are stored) will have to be tested so at least 60% of it is covered by tests.

2.1 Functional requirements

The user should be able to:

1. Create new portfolios, and select algorithm to use.
2. Run simulation of algorithms from given settings.
3. View graphs on a specific stock.
4. Watch data visualisation on trading, for instance graphs.
5. Control parsing of data.
6. User will be able to simulate stocks, to be able to test his algorithms before running them in a sharp environment
7. ASTRo should be able to run for a couple of weeks without getting facing termination.

2.2 Non-functional requirements

Usability is high priority, however to fully use the system a rather large computer skills is required to fully use this application. But we include a base to build algorithms from. The application should employ good usability practices and be appealing to use and look at. It should be easy to access all views, no more than six clicks should ever need to be clicked before arriving at the desired view.

2.2.1 Usability

Everything, GUI wise, that we provide should be quick and responsive to users. However our GUI will also depend on algorithms the user's define themselves. We can therefore not guarantee any minimum processing time frame.

2.2.2 Reliability

We use reliable systems such as Java, JPA and well known databases. Other than that no reliability is guaranteed.

2.2.3 Performance

We cannot guarantee any minimum time for processing data since users will be able to define their own trading algorithms, possibly taking lots of time.

We use thoroughly tested tools in this projects.

2.2.4 Supportability

We have designed the system in a way that supports and enforces loose coupling between the data storage and the user interface parts. It should be easy to develop additional user interface applications such as Android tablets or phones.

The robot should be easy for users to deploy to their own machines. It should include a readme file that explains how to set up the system. But no settings will be required to start the program, since we make some default decisions such as database provider.

2.2.5 Implementation

To achieve platform independence the application will use the Java environment.
The host must have the JRE installed and configured. If another database provider is needed that database must be accessible to ASTRo. Otherwise ASTRo is shipped with a standard database that works out of the box.

2.2.6 Packaging and installation

ASTRo will be deliverd as a .zip and a .tar.bz compressed file.
This file will include:

1. A jar file that contains the application.
2. A readme text file that explains how to use ASTRo.
3. Default example setting files.

2.2.7 Legal

Anyone using this program must understand that it could lose them money! We are not in any way liable for any loss of money, family, friends and/or mental health experienced by traders using ASTRo. By using ASTRo the user accepts these terms and is not able to sue us.

2.3 Application models

2.3.1 Use case model

See appendix for use cases.

2.3.2 Use cases priority

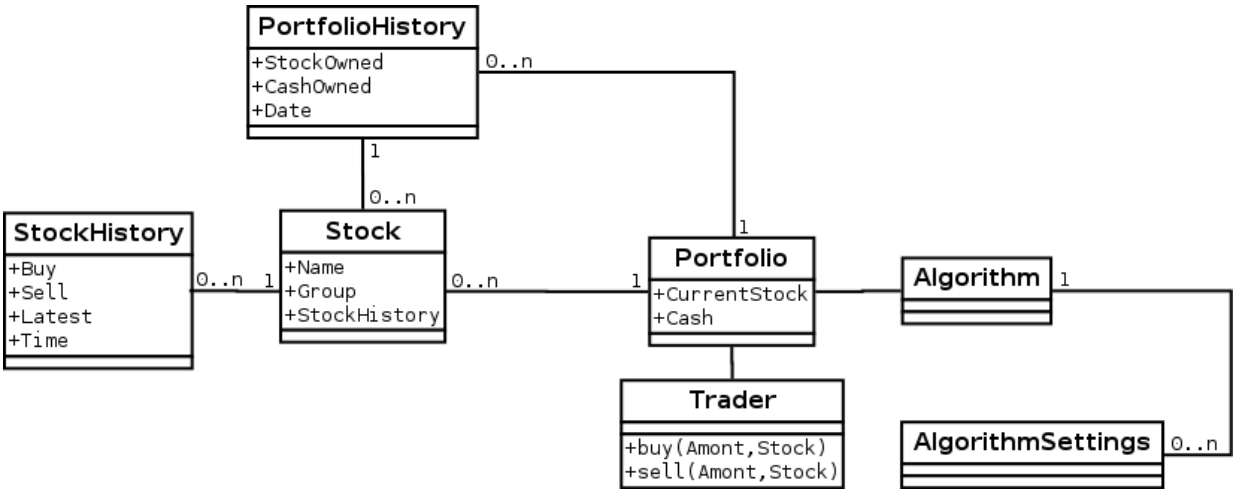
<u>High:</u>
UC_Start

UC_Astro
UC_Parser

Medium:
UC_GraphView
UC_Portfolio
UC_StockInfo

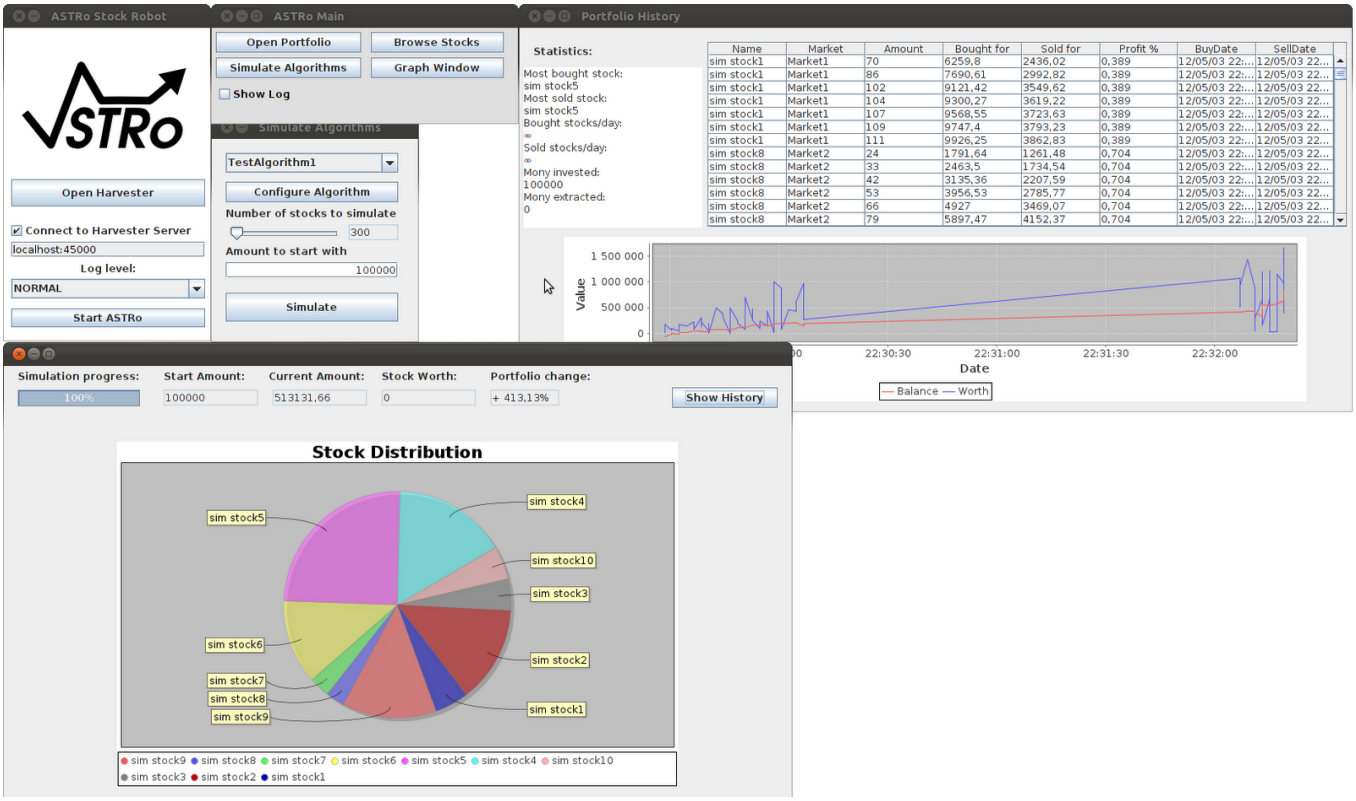
Low:
UC_AlgorithmSettings
UC_Simulation
UC_SimulationStart

2.3.3 Domain model

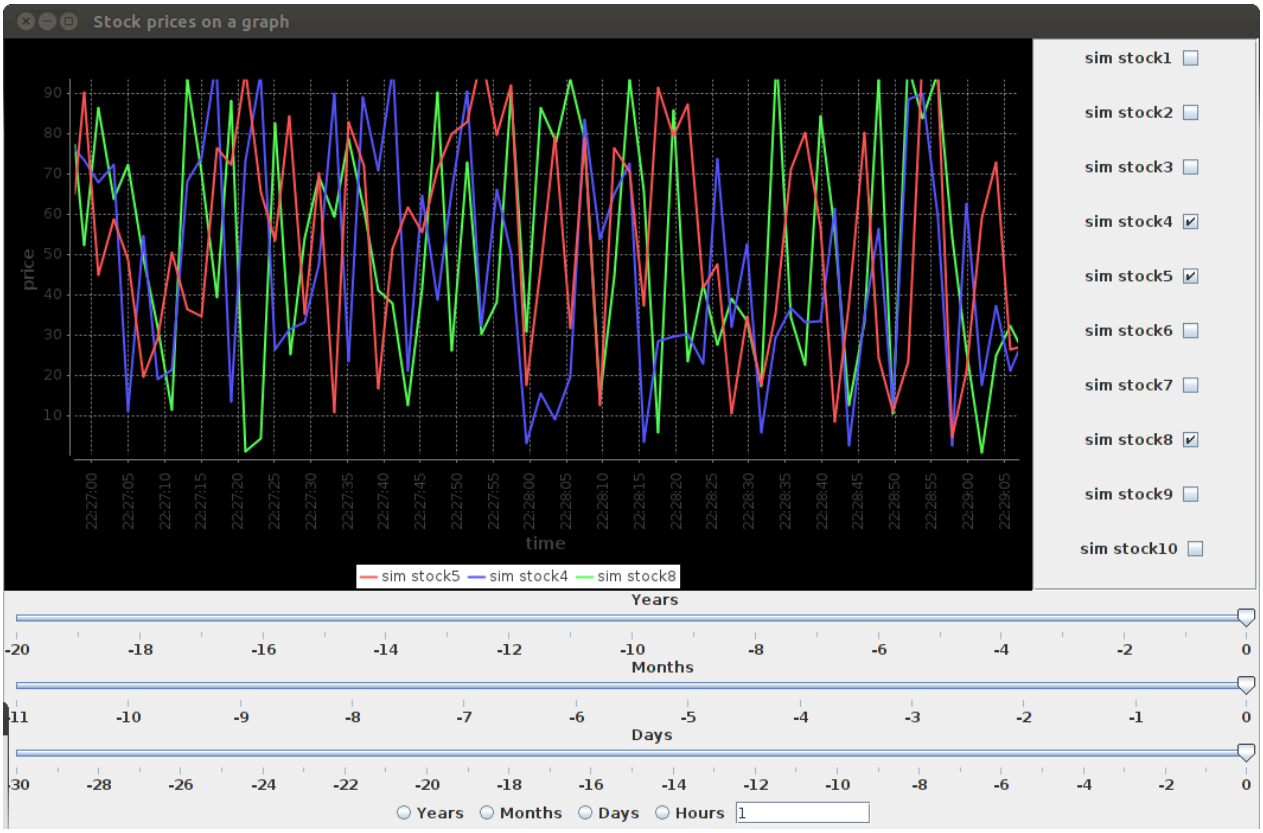


2.3.4 User interface

Here are a couple of screenshots from ASTRo in action:



Here we show an instance of a simulation.



Graph view with simulated stocks.

2.4 References

2.4.1 Appendix

Domain Model, high res: http://stock-robot.googlecode.com/files/domainmodel.png
Homepage: http://code.google.com/p/stock-robot/