Functional Requirements Specification

Organisation: `https://github.com/Hyperperform`

Developers:
Rohan Chhipa *14188377*
Avinash Singh *14043778*
Jason Gordon *14405025*
Claudio Da Silva *14205892*

Updated October 13, 2016

Client

MagnaBC
`http://www.magnabc.co.za/`

Contents

1 Introduction

Many different tools are available for measuring the quality of products made, but very few tools exist which assess the quality of the people making said products. People play a huge role in a project, and trying to monitor each and every one becomes a tedious task which diverts man power away from other more critical tasks. Whether it be for an end of year evaluation, or attempting to assess the current status of a project, generating a report on a staff member can help keep up productivity, as well as get them any help they need in order to resume quality performance. By ensuring that there is constant quality performance from each individual on a project, one can increase project quality as well as reduce project risks such as loss of an important team member during a critical stage of a project's life-cycle.

2 Vision and Objectives

2.1 Vision

The vision of this project is to create an automated performance management system, which can assess the performance and status of staff members, based on information sourced through various software systems such as card readers, version control systems and such. The system would make use of these external integrations as well as direct contact with the staff members via either web dashboard or mobile phone, to generate reports on the various staff members as well as add elements such as gamification and monitor problems that may be occurring.

2.2 Objectives

The objectives for the Hyper Perform system are:

- To source information from various integrations and treat them as events to determine performance.
- To generate real-time reports on staff members in order to evaluate performance based on these events.
- To monitor staff members in order to determine possible causes of work detriment.
- To add a form of gamification to encourage productivity, and discourage slacking and other bad behaviours.
- To make the system easily expandable to various other sub-systems.

3 User Management

3.1 Scope

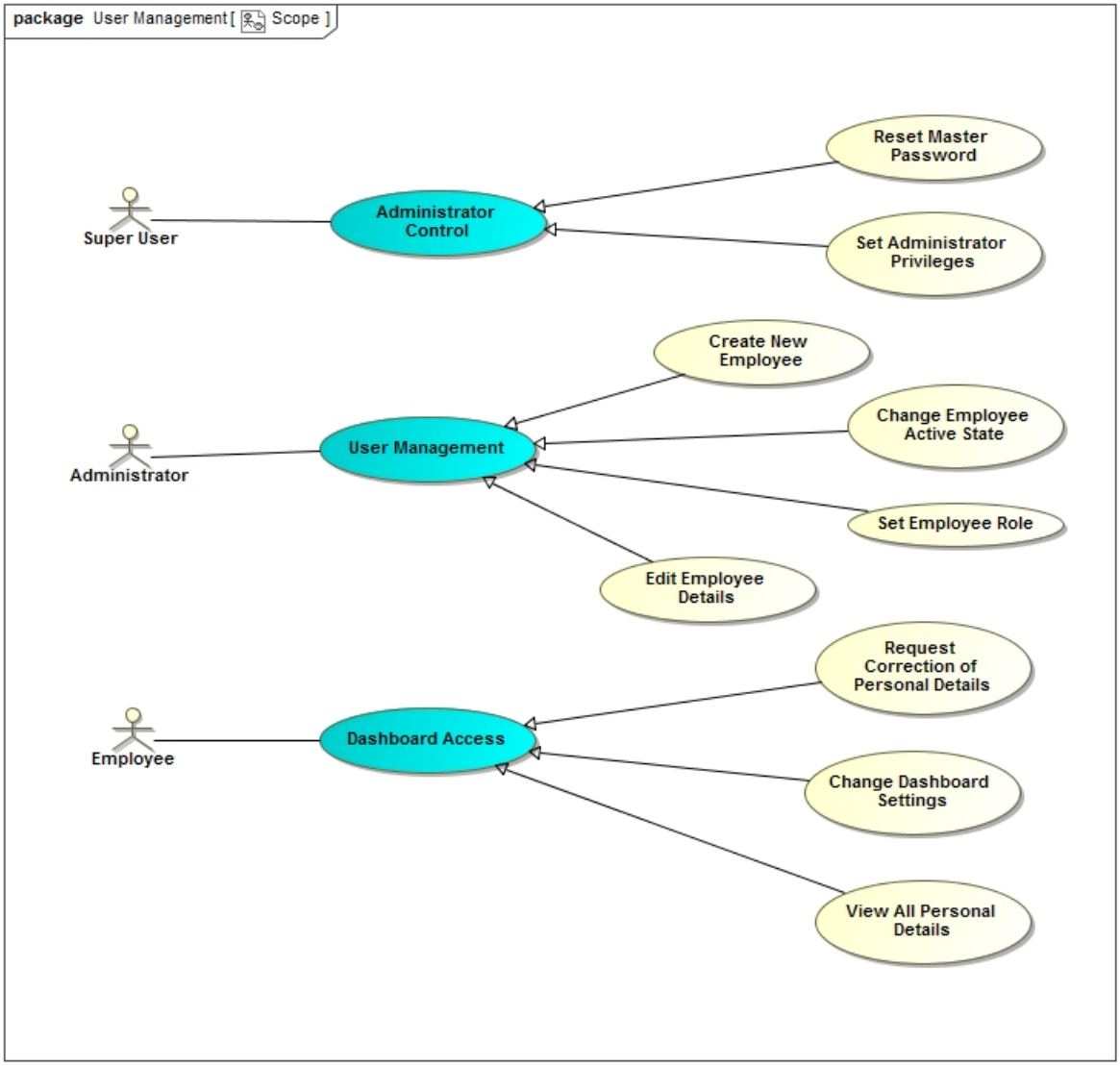

Figure 1: User Management Scope

The scope of the user management module includes:

- Super user control over administrator rights given to individuals and only one super user can exist at any moment.
- Administrators may add employees and manage their details.
- Administrators may add employee roles to employees registered, which will define which algorithms and systems will influence their performance ratings.
- A user should be able to view all their personal details, and request change immediately if something is wrong.

Note, it is assumed that a super user is not in anyway an employee within the system. It is also assumed that an Administrator is automatically an employee of the system. Administrators may not review themselves however they may change their own details.

3.2 Domain Model

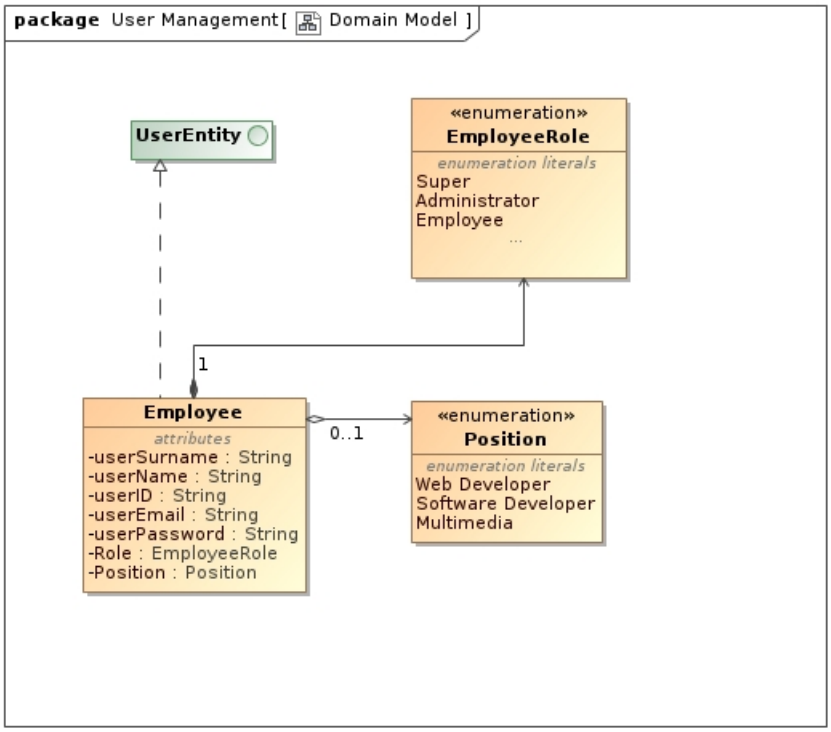

Figure 2: User Management Domain Model

4 Integration and Pre-processing

4.1 Scope

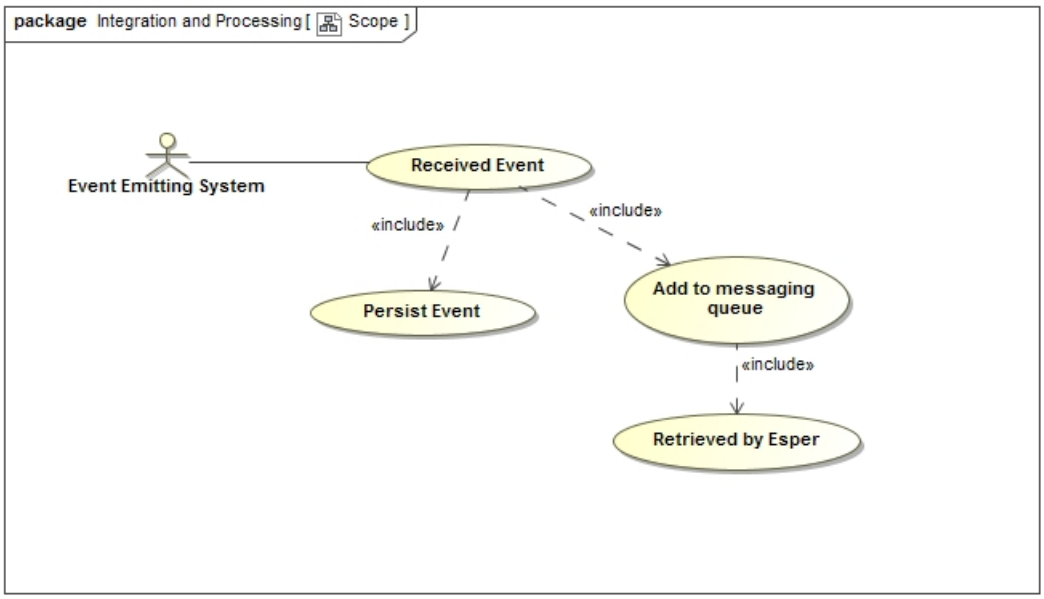

Figure 3: Integration and Pre-processing Scope

The scope of the integration module includes:

- Events are received through RESTful services that are made available to event emitting system. There is no need for polling these systems.
- Each event is mapped to a JAVA POJO and is persisted. Along with being persisted each event object is also placed onto a message queue where a CEP Engine will act as the consumer on the other end.
- Algorithms are applied to the persisted data to generate reports.

4.2 Domain Model

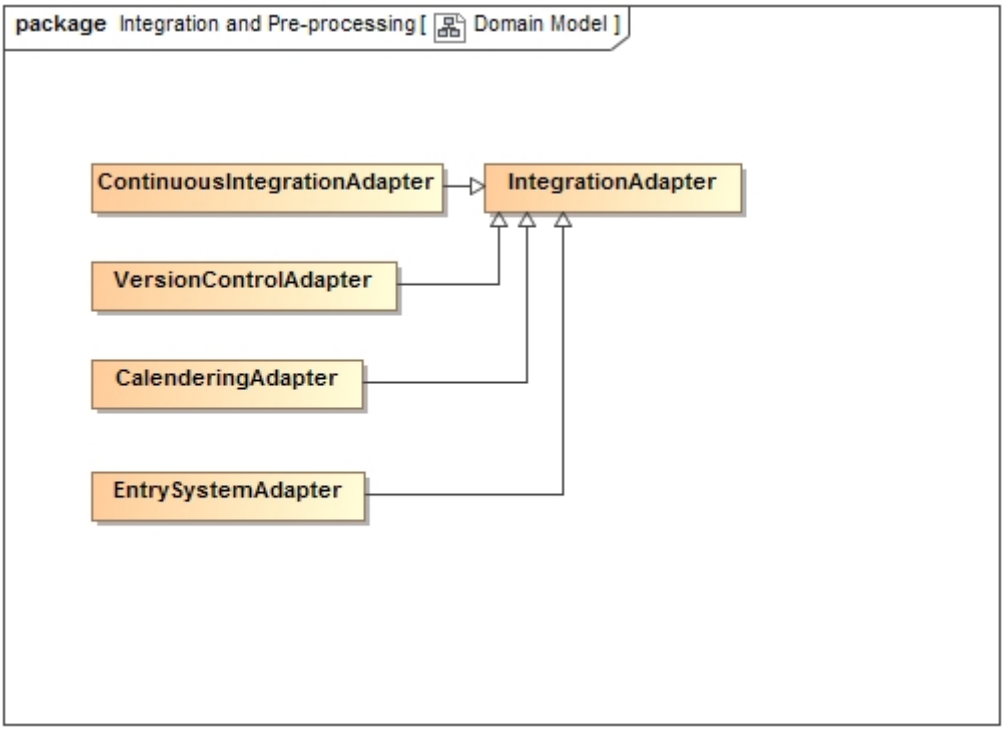

Figure 4: Integration and Pre-processing Domain Model

5 Algorithms

Algorithms are used in the calculation of performance scores. The entire Algorithms module is built using the Strategy design pattern. This allows for using different algorithms during runtime to calculate the scores. Since the Strategy design pattern is followed, new algorithms can easily be added with minimal code modification.

5.1 Scope

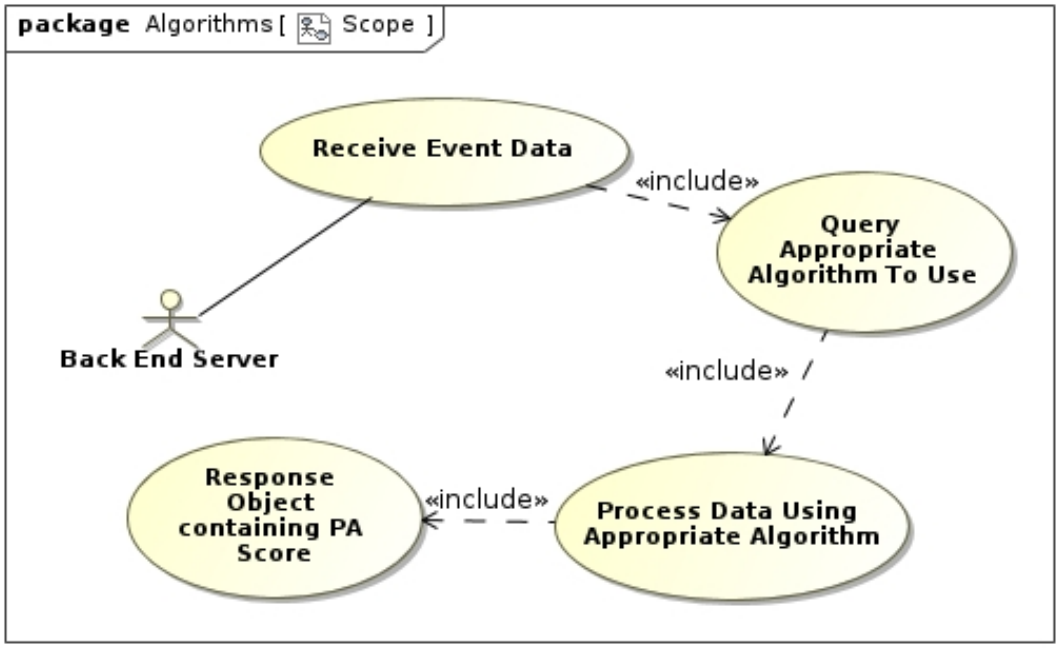

Figure 5: Algorithms Scope

The scope of the algorithms module includes:

- Determining which statistical algorithm will be appropriate with regards to the employee.
- Different algorithms can be applied to different types of employees.
- Applying this algorithm to the data provided, and then presenting employee performance scores accordingly.

5.2 Domain Model

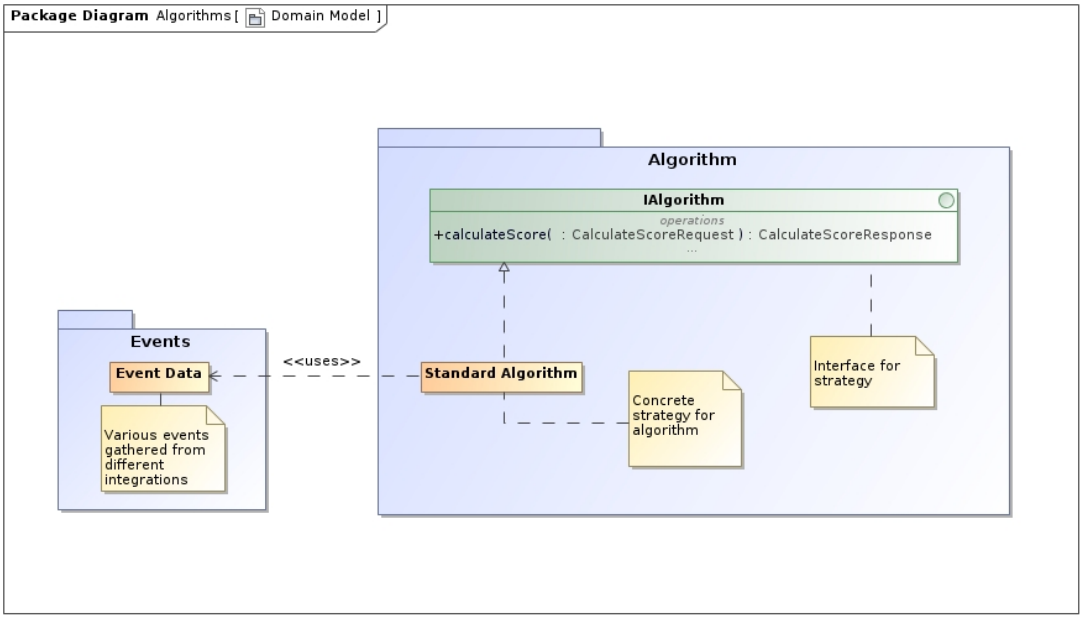

Figure 6: Algorithms Domain Model

5.3 Service Contract

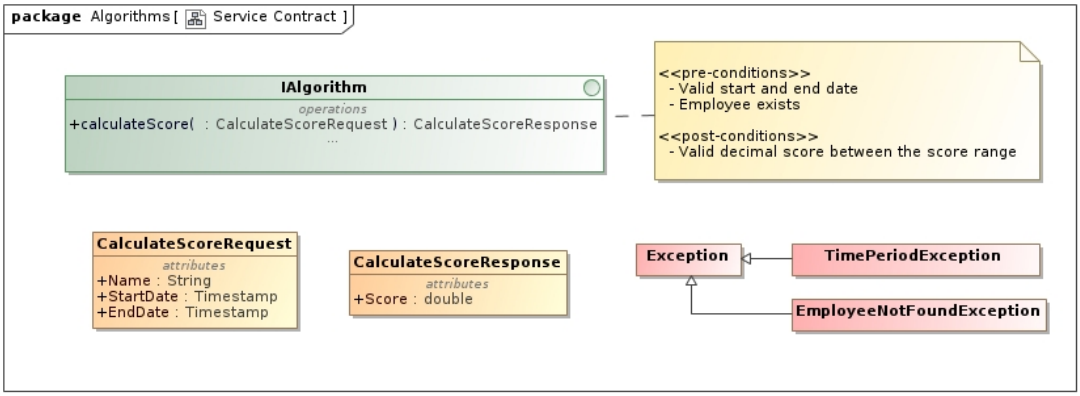

Figure 7: Algorithms Service Contract

6 Reporting

The reporting component allows for the processing and generation of data. This data allows users to see their current performance. The reporting component allows for generation of three types of reports: summarised and detailed reports as well as generation of a performance score.

6.1 Scope

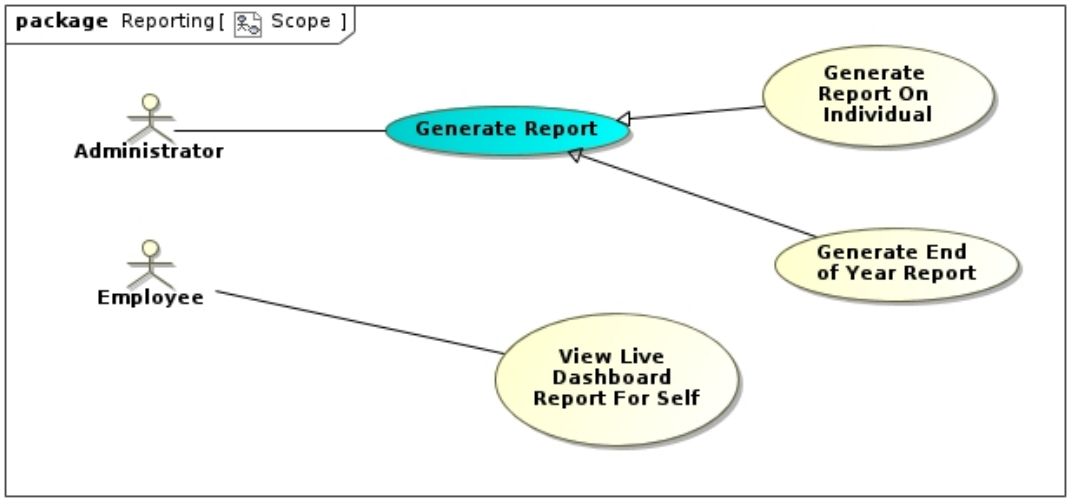

Figure 8: Reporting Scope

The scope of the reporting module includes:

- Administrators such as HR may request reports on individual employee's performance.
- All users, excluding administrators, may view their own dashboard which contains information regarding their current performance and personal details.

6.2 Domain Model

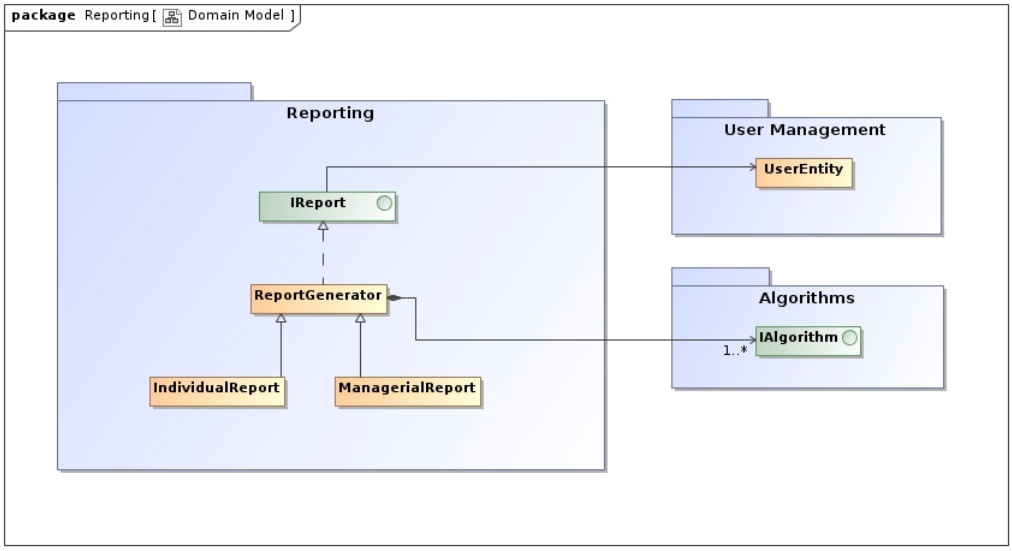

Figure 9: Reporting Domain Model

6.3 Service Contracts

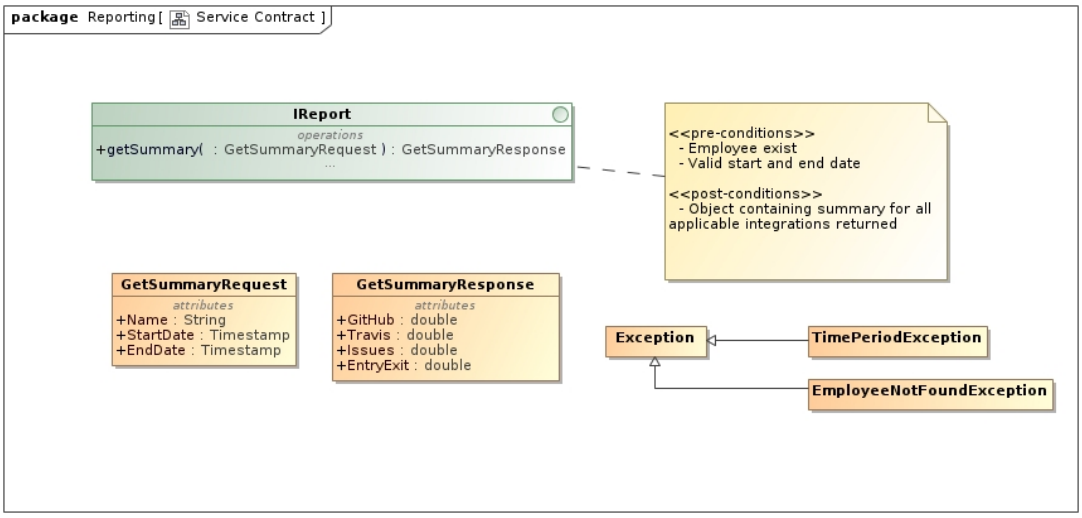

Figure 10: Get Summary Service Contract

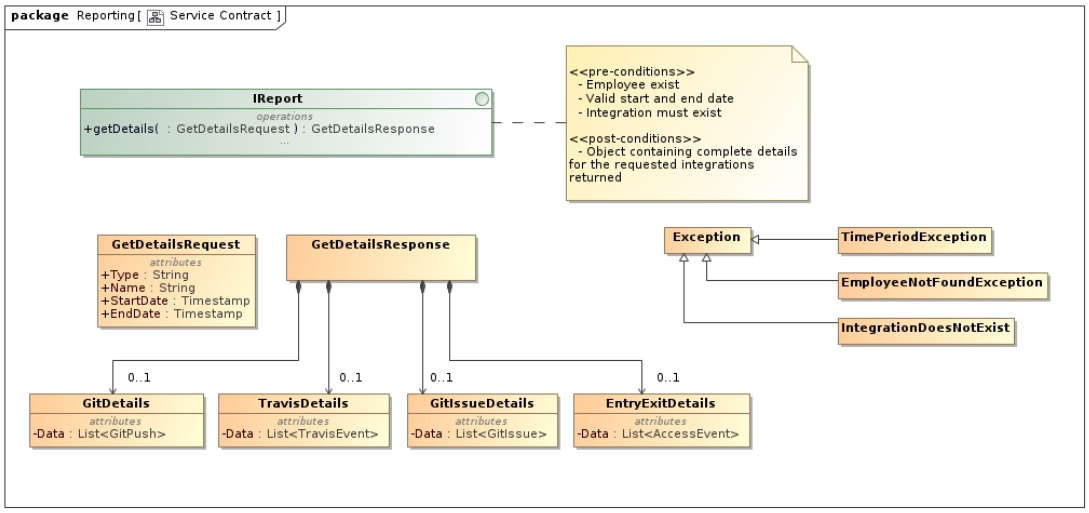

Figure 11: Get Details Service Contract

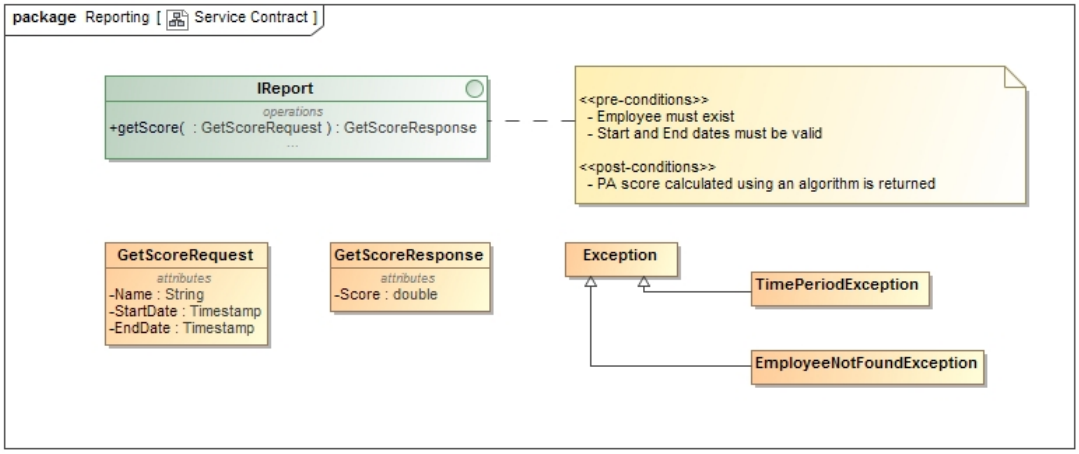

Figure 12: Get Score Service Contract

7 Notifications

7.1 Scope

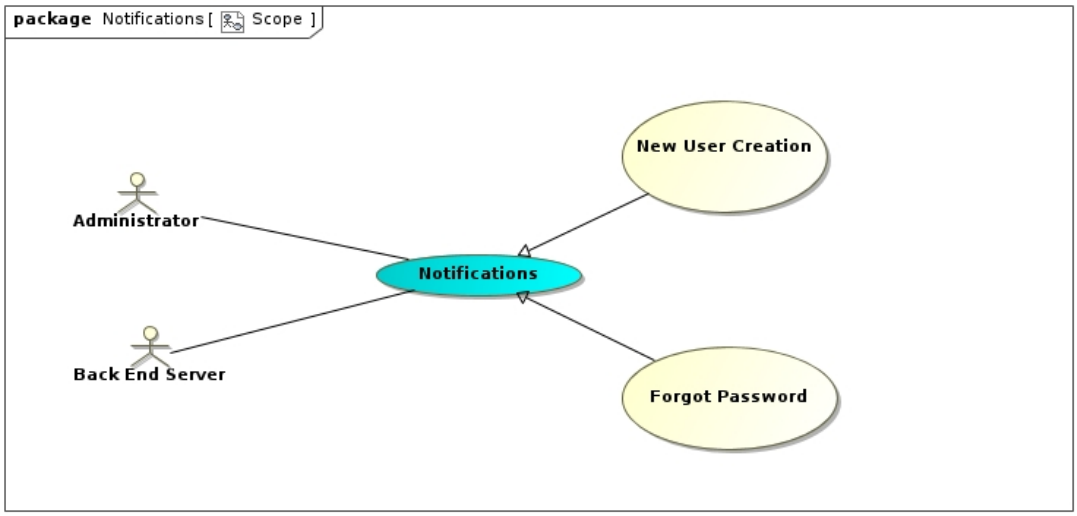

Figure 13: Notifications Scope

The scope of the notifications module includes:

- When a new User is created by an Administrator, a notification will be sent to the user with initial passwords and information.
- When a user forgets their password, a temporary password will be sent to the user and more instructions to change their password.

7.2 Domain Model

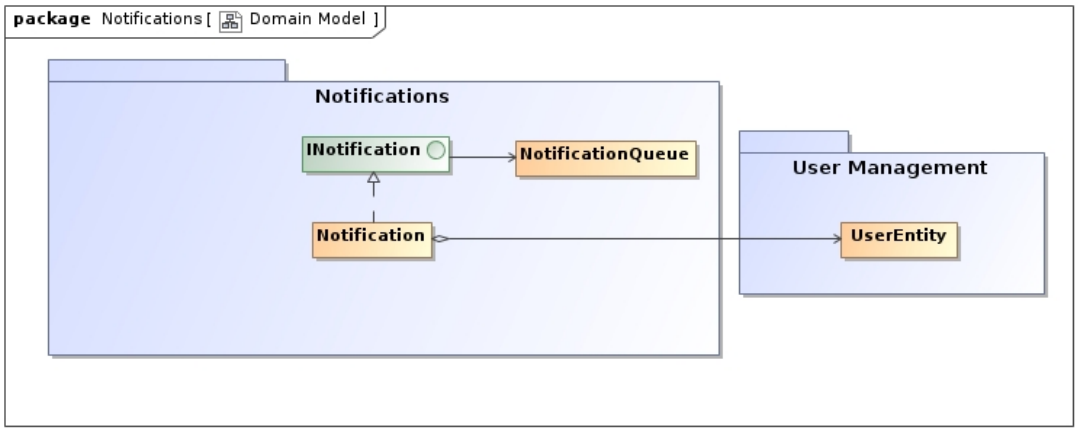

Figure 14: Notifications Domain Model

8 Forecasting

The forecasting module is responsible for allowing the addition, modification and deletion of forecasted values. These values are predictions made by management which employees must try to conform to. Forecasted values may be used in conjunction with algorithms to allow for a more accurate calculation of performance scores. Forecasted values are contained in integrations. A manager is allowed to modify and add integrations along with forecast values.

- To allow for pluggability the forecasting module consists of a contract. All realizations of the contract must adhere to the methods defined within the contract.
- Underlying representations of the forecasted values can be of any type and can be stored in any manner deemed beneficial.
- Since the forecasted values can have any representation, all necessary logic with regards to processing the data will be contained in the methods defined within the contract. This removes 'plumbing code' in the other components of the system.
- Since all forecast data processing occurs within the forecasting module only, it can be seen as a layer of abstraction over the representations of the forecasts themselves. The other components within the system will have no knowledge of the representation and will have controlled access to the data through the given contract.

- Representation of the forecast data can easily be changed without having to affecting the rest of the system. Forecast representations may be: JSON, XML, plain text etc.

8.1 Service contracts

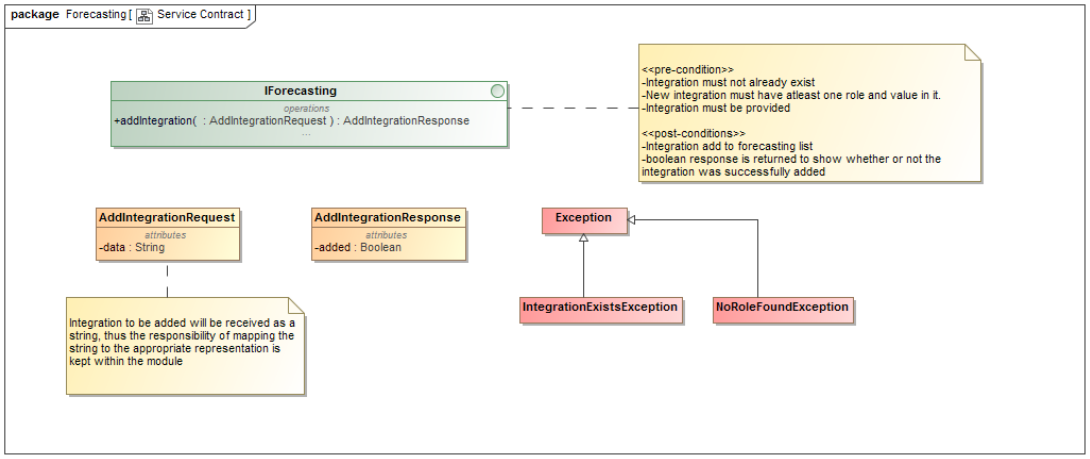

Figure 15: Service contract for adding integrations

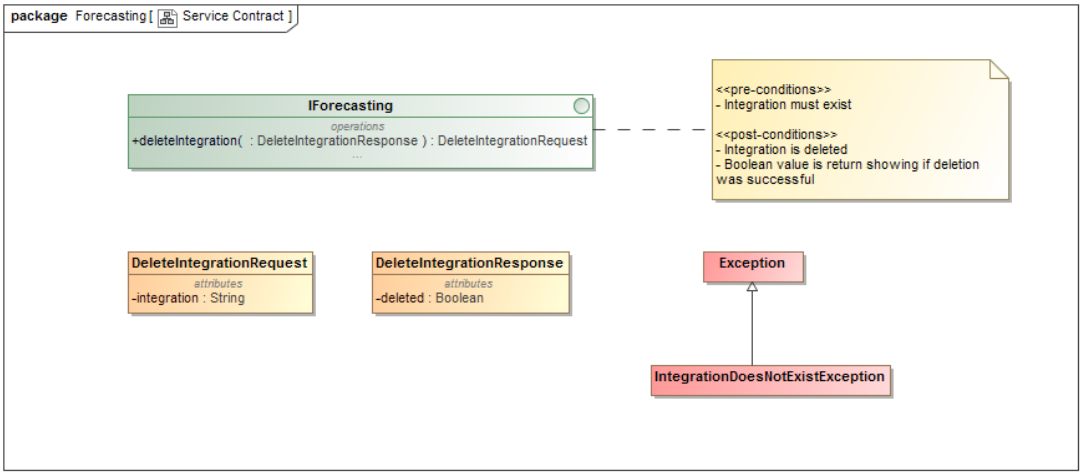

Figure 16: Service contract for deleting integrations

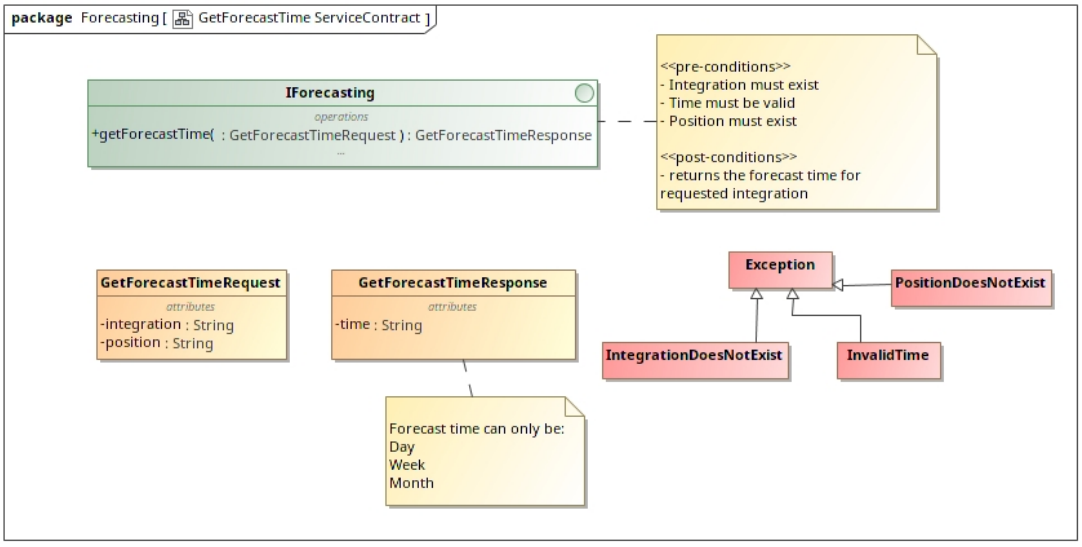

Figure 17: Service contract for getting a forecast timespan

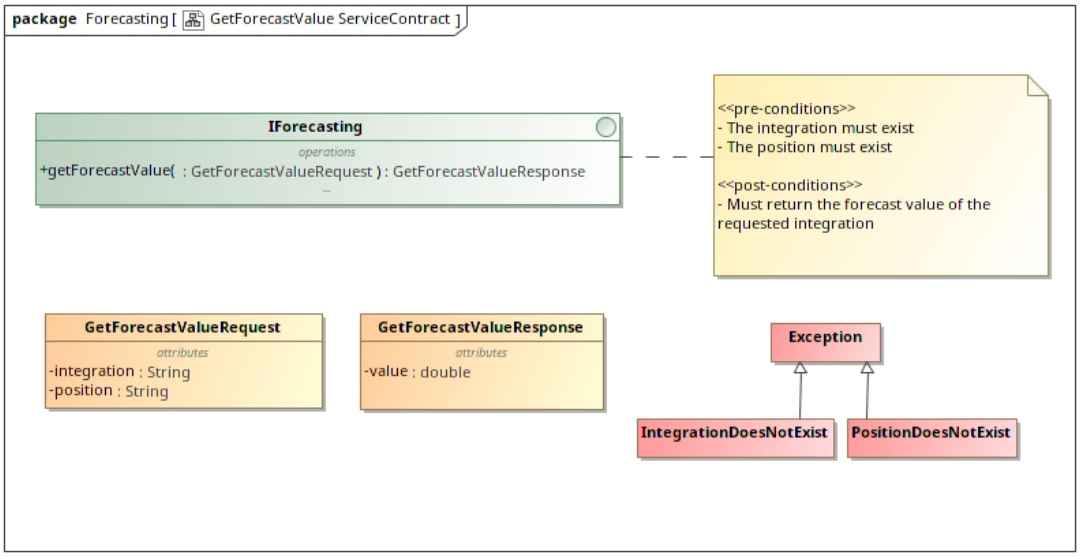

Figure 18: Service contract for getting a forecast value

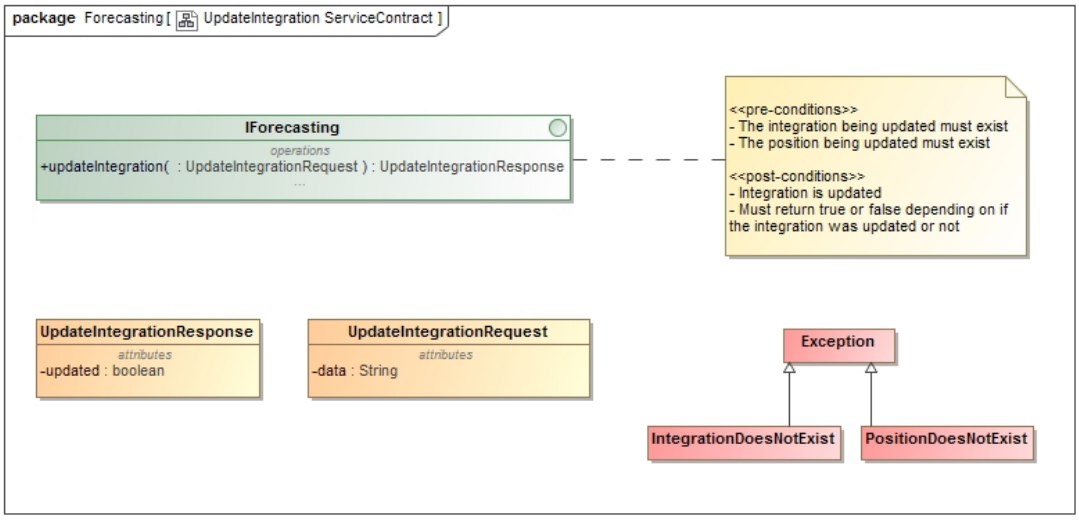

Figure 19: Service contract for updating a forecast value

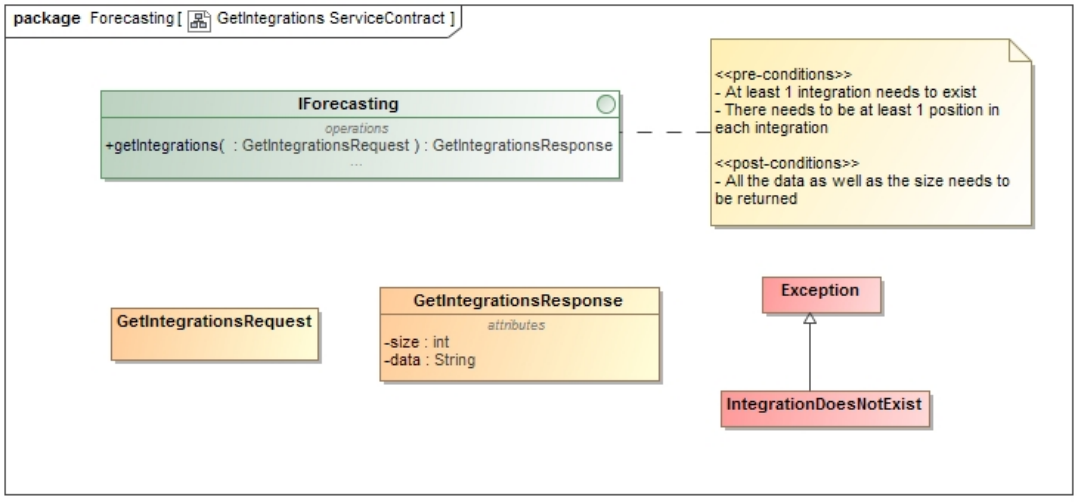

Figure 20: Service contract for getting all the forecast data