Opeka - System Design

Design Documentation for Opeka - An Aggregative Questionnaire and Reporting System

Contents

Version history

1.0	4.2.2014	Mikko Vuorinen	First public release

Introduction

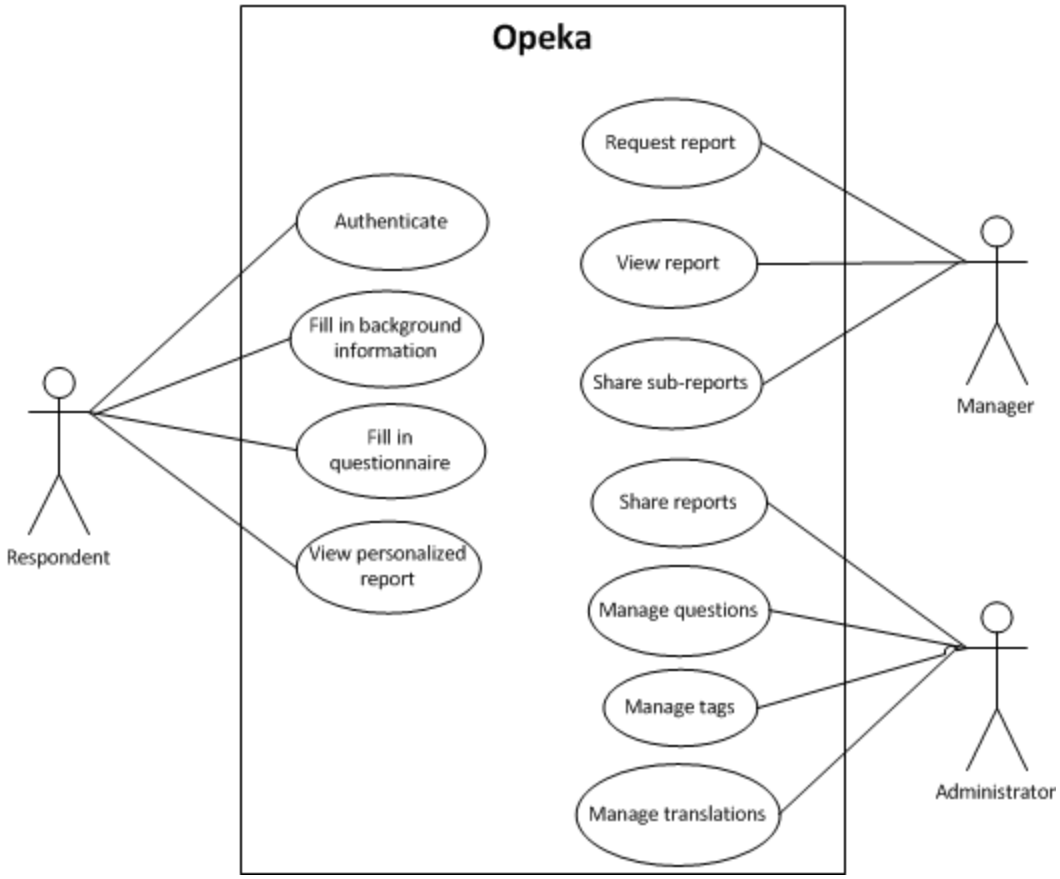
Opeka is an online questionnaire and reporting tool for improving ICT environment in schools. It is run by a web service that collects data through various questionnaires filled by teachers, students, rectors and other relevant parties. The system then, automatically and in real time, aggregates, filters and analyzes the data and provides comprehensive reports for schools and administration. Also the effort of filling in the questionnaire is rewarded with a personal report that contains personalized aggregations and analyses of the whole dataset available.

This document describes how the web service is designed and implemented in a general level. It provides overall picture of how the whole system is structured and how the parts have been designed.

The first chapter describes the usage of the system and how all the different aspects of the system are related. The second chapter shows the data structure and database design with relevant level of detail. The software design with some implementation details is described in the third chapter.

Service description

The functionalities of the web service running Opeka can be divided into three different segments based on the users' point of view. In the following illustration the basic functionalities of Opeka are shown in an use case diagram. The three actors (representing the three user aspects respectively) are **respondent**, **manager** and **administrator**.



Respondent is an actor that is required to authenticate to the site by email address and password. After authentication he can fill in the background information and the available questionnaires. After completing at least one questionnaire the personalized report becomes available. The respondent can leave the site at any time and use the email address and password to sign in and continue where he left, modify his answers (within the current year) and review his report with the most recent comparison data.

Manager, i.e. school rector, ICT development team, municipality or other administration personnel, uses Opeka to view reports of his province. These reports are distributed using an obfuscated url address to provide both the required security and the ease of access and re-distribution. Reports include many different graphs, visualizations and analyzes of the data.

They also include sub-report links that the manager can share to concerned parties.

Administrator shares the links to the top-level managers and uses the administrator tools to maintain the system and keep it up-to-date.

The following sections describe how the data is collected and processed in and between the functionalities of the three actors introduced above.

Questionnaire modules

The primary purpose of the respondent aspect is to collect structured and quantitative data using questionnaires. In Opeka, a questionnaire is a type of module used to combine a group of questions into a consistent entity. Currently the question sets of Opeka are divided into three questionnaires: *digital environment* (digitaalinen toimintaympäristö), *devices and software* (laitteet ja ohjelmistot) and *ICT skills* (TVT-osaaminen).

Questionnaires are shown to respondents as separate links in the front page of the service after signing in. Questionnaires can be filled in any desired order. If the respondent has already completed the questionnaire, a link to modify answers is displayed. If the questionnaire has been partially completed, a link to continue where the used previously left is displayed.

A new questionnaire module can also be created and provided for respondents to fill in. Respondent background information can be used to filter which questionnaires are provided. The structure of the questionnaire module and the questions it contains are described in more detail in following chapters.

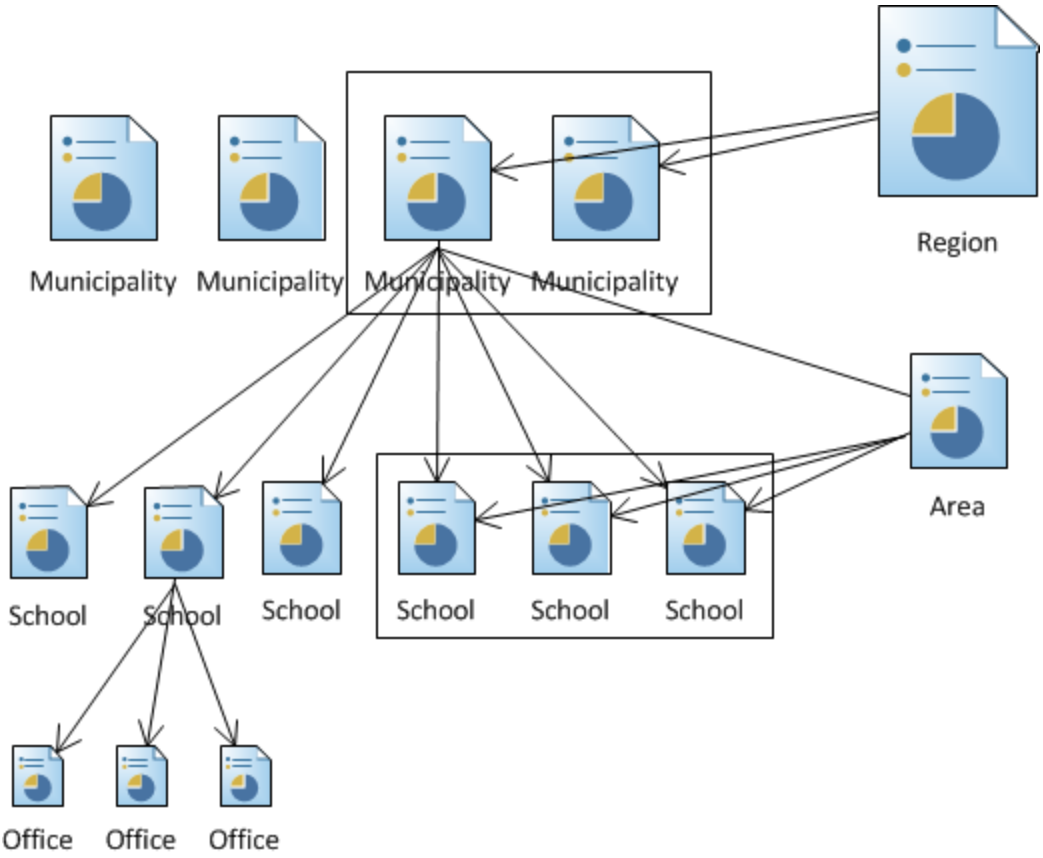
Report structure

A basic division in all the reports in Opeka is a **semester**. In most cases this means that a report represents data from one full calendar year. All the reports are also divided by either **education level** or **school degree**. This means that the reports described below always contain data only from a specific education level or school degree. Education levels are combinations of a selected list of school degrees, which are part of the background information given by the respondents. All the reports can also be filtered by the teaching language, so that there is e.g. a specific school degree report for the finnish-speaking teachers. This filter also applies to all the reports below.

Opeka reports to local level are structured hierarchically according to respondents' background information, in specific their municipality, school and office. The hierarchy is shown in the following illustration. From top to bottom: **municipality report** contains all the data collected from respondents that have selected the particular municipality as their primary place of work; **school report** contains the data with the particular municipality and school; **office report** contains data with the particular municipality, school and office. In addition, **region report** contains data from a group of municipalities and **area report** contains data from a group of

schools.

Although the hierarchical structure is the primary structure of Opeka reports, there are also other rules by which a report can be compiled. **Public report** is a special type of report that any interested individual can freely browse on the Opeka web site. Public report includes answers from all the respondents within a specific semester. It also contains links to other reports such as **subject report** and **personnel category report**.



Each type of report contains links to sub-reports shown in the illustration as arrows. Municipality report contains links to school reports of all the schools of the municipality, i.e. schools that have respondents from the municipality. School report contains links to office reports in corresponding fashion. Region reports contains a link to each municipality inside the region and area reports contains a link to each school in the area.

There are two types of links that can be shared to view reports. Normal reports contain all the links to sub-reports described above. **Restricted reports** contain all the information of the report, but there are no links to sub-reports. Thus, restricted reports can be used to share e.g. the information of a municipality report while preserving the confidentiality of each individual school report.

Report contents

The contents of each report is similar in every type of report. The data is always shown in comparison to corresponding data in upper levels in the report hierarchy. That is, in office report you can compare data with the whole school, the area, the municipality, the region or all the available data. These comparisons are shown in all charts and can be used as a compare group in statistical analysis. Comparison is always made to the last 12 months, so that e.g. a school report of the year 2013 viewed in march can be compared with the data gathered from the whole municipality between march 2012 and march 2013. This is because schools can use Opeka at any time of the year and there must always be a possibility to compare data with a relevant respondent group.

The front page of a report provides basic status information: how many respondents have currently completed the questionnaires, what is mean answer time and which and how big are the compare groups. It also contains links to all the sub-reports and their response counts (if the report is not restricted).

Differences to other respondents, a chart representation of the data in a report shows average levels of competency in several categories. Currently these competencies are *technology*, *community*, *attitudes*, *usage* and *skills*, but these can be changed or adjusted in the future development of Opeka. Levels are calculated using a selection of questions from all the questionnaires. The chart can be navigated to show exactly which questions are used to calculate each level, and in the deepest level of detail, to show the distribution of answers in each question individually.

There is also a chart with a list of **devices and software** selected by the respondents and distributions of a set of questions to provide more detailed information about the usage of the device or software.

Differences to other respondents is an automated analysis page, a more sophisticated way to view the data in a report. It uses statistical methods to extract meaningful information by comparing answers. This is the most powerful tool in Opeka to use in ICT development process in schools and administration. The page highlights questions that have statistically significant differences when comparing with a selected compare group and divides these into potentially beneficial and potentially hindering factors. The page also highlights devices and software with significantly different number of users when comparing with the selected compare group.

The **ICT-skills** report page is a more detailed compilation of the skills questionnaire. This page can be used to plan personnel training. It lists the skill levels of respondents in comparison to the general skill levels of all the respondents in the system. It also includes a list of themes on which the respondents are interested to have training. Each skill level has a link to a page that shows the questions that are used to calculate that skill level and how the respondents have answered to those questions. This page shows the comparison of answer distribution between all the respondents and those interested in training of that skill.

There is also a more purpose built report page for examining the **digital learning culture** of the respondent group. This report page contains the data divided under several headlines such as

ICT planning, Pedagogical application and pedagogical support, ICT skills and support services. It gives the reader a compact view to the responses in one page.

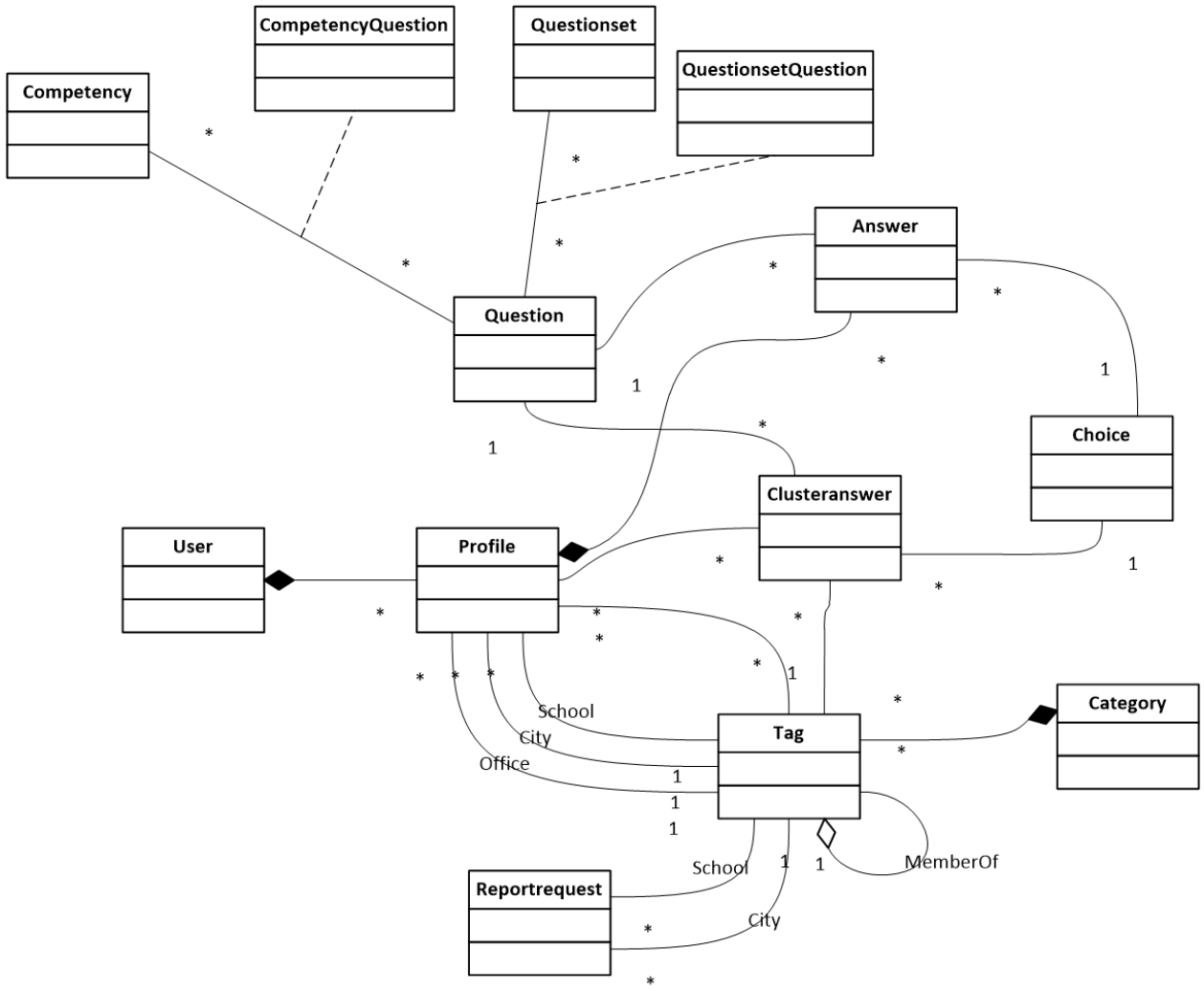
Administrator tools

The main tasks of the administrator is to distribute top-level reports and manage tag system and questions. To accomplish these tasks Opeka has an administration toolbar containing **report administration** tools, **tag administration** tools **questions administration** tools and **translation administration** tools. The administration tools are described in detail in the administration guide for Opeka.

Data structure

This chapter contains a detailed description of data structures that are implemented in the Opeka system.

The diagram below shows a slight simplification the database structure of the data storage used by the system. Below the diagram there is a table describing all the main concepts shown in the diagram. The diagram and the table together give a clear picture of the data structure of the system. This includes both the structure of the answer profiles and reporting data and structures that are derived from the software design of the Opeka web service.



User	User details are used to identify a person. User can have multiple answer profiles. One person should only use one email address to log in to the system so that he/she can be identified.
Profile	All the answers are linked to a specific answer profile. Profile contains all the information required to filter answers for all the reports.
Tag	Opeka has an evolving tag system that is used to store different devices, software, schools, cities, subjects etc. Most of the tags can be added by users, resulting in constantly changing tag system.
Category	Tags can (but doesn't need to) be categorized. Currently categories are only used for software.
Reportrequest	School or municipality management can create report request for a specific report. These requests can be used to ease report distribution.
Question	Questions contain the question label and the question type that defines the choices that can be used as the answer for the question. Questions without question type are considered open-ended questions.
Choice	Each question type has a number of choices. Choice contains a label, order number, numerical value and color that represents the choice.
Questionset	Question sets are used as a building parts of questionnaires. Questionset can contain any number of questions.
QuestionsetQuestion	Questions can be associated with one or more questionsets. This table contains the information of the placement of the question in a specific questionset.
Competency	Competencies are all the levels that are used in reports to display results automatically calculated by the system.
CompetencyQuestion	Competencies are calculated using the numerical value of the answers. CompetencyQuestion-table stores the multipliers that define which questions affect which competency levels and how.
Answer	Answer is the answer to a single question of a single profile.
Clusteranswer	Clusteranswer is the answer to a single question of a single profile considering a spesific tag (i.e. software or device).

In addition to structures described in the diagram and the table, the database contains some

tables that are required by the system. These are e.g. translation texts of the user interface and implicit join tables between structures.

Translation

Translations are implemented in Opeka without adding any unnecessary complexity to the database. Each translatable field in a table has translated versions for each language in the table structure throughout the database. This is a design decision that clearly adds some inflexibility to the database as each new language requires changes to a number of tables in the database structure. Behind this decision are the following factors when choosing this design over a separate translation table:

- Translations are not equal. That is, it is much more important that a question label is translated precisely than a single user interface text or a school tag. Thus keeping different translations separate allows easier and safer management of the key translations.
- Adding a separate translation table for each translatable table or field would considerably increase complexity of the database
- Applying a translation is faster when the translation is available from the same table as the data. This is essential especially when a list of tags is filtered according to user input and displayed in the interface through background queries.
- Having to modify the database structure in each additional language is a minor task compared to other tasks in the translation process, so the benefit of easier translation implementation in the code would not make the translation process as a whole much easier.

The decision has following main disadvantages:

- All the translated tables require structural modification when adding a new language.
- Translation texts are separated into different tables throughout the database and need to be updated separately.

In the tables shown in the previous figure there are following fields that have a corresponding translated field in the database:

- Questionset: name
- Question: text
- Choice: label
- Competency: name
- Category: name
- Tag: name

The translated fields are named by the original field name with underscore and the language identifier (2-letter code) added to the end of the name, e.g. *fieldname_xx*. Currently there are

three languages in the database: Finnish (fi), English (en) and Swedish (sv).

User interface texts have their own translation table structure that follows the requirements of the Yii framework database message source class CDbMessageSource.

Software design

This chapter describes the design of the software that runs the service. It includes an explanation of chosen background technologies and shows in detail the structure of the software. This is the most important chapter for understanding the service and how to modify its functionalities.

Background technologies

Opeka web service is implemented using PHP as a language for the backend functionality. PHP has been chosen mainly because the previous experience of the implementers, the ease of prototyping and the availability of a suitable framework. It is also easy to find hosting for a PHP-based web application.

PostgreSQL is used as the database management system (DBMS) of the service. This is mainly because of the decision to host the service in-house (in School of Information Sciences in University of Tampere), as it was the only available option at the time. The service can be transferred to another DBMS, but some implementation details might have to be revised in case of different implementations of SQL interfaces.

The service is implemented on Yii framework (http://www.yiiframework.com/) that has proven to be a solid choice for a PHP-based web service that is expected to require a lot of tailoring and specific functionality and interface. The prior experience of the implementers was also a major factor in the decision, as well as comprehensive documentation of the framework.

In addition to typical front-end technologies (e.g. HTML, CSS, JavaScript) the service leans heavily on jQuery (http://jquery.com/). Other libraries currently used are Highcharts JS (http://www.highcharts.com/) for various charts in the reports and Google Charts (https://developers.google.com/chart/) for immediate-response charts.

Structure

The chosen framework defines the key structures of the software. With Yii, the most important structures are **MVC** design pattern for separation of data, business logic and front-end and **Active Records** for data abstraction. These structures are well described in the documentation of the framework (http://www.yiiframework.com/doc/guide/1.1/en/basics.mvc and http://www.yiiframework.com/doc/guide/1.1/en/database.ar). The framework provides also other important structural elements that are used in Opeka (such as widgets, layout and access control).

Model layer

All the database structures shown in the database diagram are implemented as active record models in Opeka. These models are inherited from common *ActiveRecord* class that implements the common functionalities of all Opeka-related models. Besides the common base class, the class hierarchy of the model layer is flat. Relations between models are implemented as active record relationships defined by the Yii framework.

Controller layer

All the controllers of Opeka are derived from a common abstract base class *Controller*. It implements some basic tasks that are common to all or most functionalities of the service, such as user profile handling, access control and language selection. Actual controllers can be divided into four categories: questionnaire controllers, report controllers, administration controllers and common controllers.

Questionnaire controllers have a common base class *QuestionnaireController* that handles common tasks such as questionnaire workflow, form rendering and saving answer times to the database. Each questionnaire of Opeka is implemented as a sub class of *QuestionnaireController*. Currently these controllers are:

- *DigitaalinenToimintakulttuuriController*
- *LaitteetJaOhjelmistotController*
- *OsaamiskartoitusController*.

Report controllers implement all the different kind of reports that are available in Opeka. A report controller implements a set of filters to select which answer profiles are shown in the report. A common base class *ReportController* loads the profiles using the filters and uses them to display all the pages of the report. Typically a report controller does not implement any specific report pages itself but uses generic functionalities of *ReportController* and its views to display them. Report controllers are named according to the filters they use to select the answer profiles. Currently implemented report controllers are:

- *PublicController* - Public report that can be viewed by any visitor.
- *EducationlevelController* - Specific education level.
- *StateController* -Selection of cities. Contains also links to the reports of all the cities in the specified state.
- *CityController* - Single city and links to the reports of all the schools inside the city.
- *AreaController* - Selection of schools and links to the reports of those schools.
- *SchoolController* - Single school and links to the reports of all the offices of the school.
- *OfficeController* - Single office of a school.
- *DegreeController* - Specific degree (used in vocational reports)
- *FieldController* - Specific field of education (used in vocational reports)
- *ProfileReportController* - Can contain any desired filters.
- *UserReprotController* - Filters answer profiles by selecting specific users by email address.

All the report controllers also have a semester filter, an education level filter or a school category filter, and a teaching language filter by default. These and all the report-specific filters are stored in the URL-address that is used to share and distribute the report as a report-id. The report-id is encrypted to ensure that non-authorized visitors cannot view the reports.

Each report controller has following key tasks that are implemented as methods and class methods for the controller class:

- *createParam* and *parseParamStr* - creates and parses the parameter string that contains all the information required to specify the report
- *getSelfFilter* - specifies filters for the respondent group of the report
- *getCompareFilters* - specifies all the compare groups and their filters (except for global education level, school category and teaching language filters)
- *getSubReports* - creates a list of sub-reports (grouped by their report type) visible in this report

Administration controllers contain all the administrative functionalities in Opeka. Common base class *BaseAdminController* implements access control and some common tasks for administration. Administration controllers contain:

- *AdminController* - Logging in and out from the administration tools
- *ReportAdminController* - Report administration
- *QuestionAdminController* - Question administration
- *TagAdminController* - Tag administration
- *TranslationAdminController* - Translation administration
- *SysAdminController* - System administration tools such as change log and data-dump

Administration controllers have a number of specific functionalities which can change over the development of the system. There should always be a limited number of people using the administration tools, especially the system administration tools.

View layer

All the views of the service are implemented following the conventions of the framework. Single page views are structured under the same structure as the controllers. Each controller has a directory containing all the views used by that controller. Base controllers may also have views, which can be used in any derived controller class. This is the case especially in report views and questionnaire views, which are completely implemented in the views of *ReportController* and *QuestionnaireController*, respectively. Individual report controllers can also implement any single view by simply including a view file in the view folder of that controller, but the basic functionality is available without implementing any report-specific views. Currently only public report has specially implemented views. This design enables adding a new report or questionnaire without having to implement any views.

Following elements are implemented as widgets that can be easily used in multiple views.

- *QuestionInput* - Input element to be used in forms. Displays a question and choices according to the question type.
- *TagElement* - Input element to be used in forms. Displays a list of tags from a specified collection and shows an element to allow user to select and add tags.
- *QuestionFeedback* - Feedback buttons used by e.g. QuestionInput and TagEleme.

In addition, there are few form input widgets (*TextWithUnit*, *TextareaInput*) and other widgets (*JuiProgressBarWithContent*) used in Opeka to help simplify some basic tasks.