Delta
July 5, 2018

User Requirements Document
Version 1.2

Project Team
D.P. van den Berg 0949036
R.T. van Bergen 0938857
D.J.C. Dekker 0936100
S. van den Eerenbeemt 0954445
J. Mols 0851883
B.F. Rongen 0858160
B.W.M. van Rooijen 0895073
R.P. Schellekens 0944330
A.A. Vast 0854060
G. Walravens 0904152
S. Wessel 0941508

Project Managers
S.P.O. Oostveen
A. Rajaraman

Project Supervisor
dr. N. Zannone

Customer
dr. L. Genga
Abstract

This document contains the user requirements for the APD extension developed by Delta. The extension made to the APD tool adds several features such as user management, experiment management and project sharing. The user interface will be redesigned to improve the way users work with the tool. If time constraints allow, Delta will allow for customization of the components used in the discovery of anomalous processes. This document complies with the ESA standards.
A.1.3 Logging in ............................................. 26
A.1.4 Logging out .......................................... 26
A.1.5 Access account information ......................... 27
A.1.6 Change account information ........................ 27
A.1.7 Change password .................................... 27
A.1.8 Create an account as administrator ................. 28
A.1.9 Delete an account as administrator ............... 28

A.2 User tracking ............................................ 29
A.2.1 Tracking User Activity ............................... 29
A.2.2 Accessing tracking data .............................. 29
A.2.3 Enabling tracking .................................... 30
A.2.4 Disabling tracking ................................... 30

A.3 Project management .................................. 31
A.3.1 Creating a new project ............................... 31
A.3.2 Access files of a project ............................. 32
A.3.3 Delete files of a project ............................. 32
A.3.4 Share a project ....................................... 32
A.3.5 Delete a project ...................................... 33
A.3.6 Change access rights ............................... 33
A.3.7 Remove access rights ............................... 34

A.4 Experiment management .............................. 34
A.4.1 Run anomalous subgraph discovery .................. 34
A.4.2 Run anomalous subgraph discovery with custom parameters .................. 35
A.4.3 Run partial order discovery .......................... 35
A.4.4 Stop a running experiment phase ..................... 36
A.4.5 Accessing experiment information ................. 36
A.4.6 View experiment results .............................. 37
A.4.7 Delete experiment log file ........................... 37
A.4.8 Start multiple runs of an experiment ............... 38
A.4.9 Comparing results of multiple runs of an experiment .................. 38
A.4.10 Upload custom subgraph discovery algorithm ........ 39
A.4.11 Start subgraph discovery with a custom implementation of the algorithm ........ 40

A.5 Component customization ............................ 40
A.5.1 Upload intermediate result .......................... 40
A.5.2 Select components for the experiment setup .......... 41
A.5.3 Start anomalous subgraph discovery with a custom implementation ........ 41

A.6 Visualization ............................................ 41
A.6.1 Search projects by name ............................ 41
A.6.2 Search projects by creation date ..................... 42
A.6.3 Search experiments by running date ................. 42
A.6.4 Search experiments by status ....................... 43
A.6.5 Expand a node ....................................... 43
A.6.6 Filter patterns ....................................... 44
A.6.7 Export subgraphs .................................... 44
A.6.8 Show subgraph support value ........................ 44
A.6.9 Show the number of occurrences of a subgraph ........ 45
A.6.10 Indicating children and parents ..................... 45
A.6.11 Showing parents or children up to 5 levels deep .......... 45
Document Status

General

Title of document: User Requirements Document
Document identifier: Delta.URD/1.2
Version: 1.2
Authors: D.P. van den Berg, R.T. van Bergen, D.J.C. Dekker, S. van den Eerenbeemt, J. Mols, B.F. Rongen, B.W.M. van Rooijen, R.P. Schellekens, A.A. Vast, G. Walravens, S. Wessel

History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Author(s)</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>2018-04-30</td>
<td>J. Mols, B.W.M. van Rooijen, A.A. Vast</td>
<td>First draft after first meeting with customer.</td>
</tr>
<tr>
<td>0.2</td>
<td>2018-05-01</td>
<td>J. Mols, B.W.M. van Rooijen, A.A. Vast</td>
<td>Updated draft after second meeting with supervisor and customer.</td>
</tr>
<tr>
<td>0.3</td>
<td>2018-05-04</td>
<td>J. Mols, B.W.M. van Rooijen, A.A. Vast, B.F. Rongen</td>
<td>First version, containing a more complete set of requirements and an up-to-date first and second chapter. A lot of the use cases have yet to be implemented.</td>
</tr>
<tr>
<td>0.4</td>
<td>2018-05-08</td>
<td>J. Mols, B.W.M. van Rooijen, A.A. Vast, B.F. Rongen, S. van den Eerenbeemt</td>
<td>Second version, improved most of the requirements after receiving feedback from customer and supervisor. Finished all the use cases.</td>
</tr>
<tr>
<td>0.5</td>
<td>2018-05-14</td>
<td>J. Mols, B.W.M. van Rooijen, A.A. Vast</td>
<td>Final version to be approved. Last-minute changes made to complete document.</td>
</tr>
<tr>
<td>1.0</td>
<td>2018-05-15</td>
<td>J. Mols, B.W.M. van Rooijen, A.A. Vast</td>
<td>Approved final version.</td>
</tr>
<tr>
<td>1.1</td>
<td>2018-06-20</td>
<td>B.W.M. van Rooijen, A.A. Vast</td>
<td>Some small changes to improve language.</td>
</tr>
<tr>
<td>1.2</td>
<td>2018-07-03</td>
<td>B.W.M. van Rooijen, R.T. van Bergen</td>
<td>Revised List of definitions and rewrote some use cases.</td>
</tr>
<tr>
<td>Section</td>
<td>Reason</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>---------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.3.1</td>
<td>Inconsistent definitions.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Appendix A</td>
<td>Changed use cases.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1 Introduction

1.1 Purpose

This document contains the user requirements for the Delta extension to the APD tool. It will be used to develop the extension according to the requirements defined in this document. The requirements are the result of negotiations between the customer, Laura Genga, and the Delta team. These requirements are to be implemented by the Delta team according to their corresponding priorities. Changes to the document have to be approved by both parties.

1.2 Scope

The goal of the Delta project is to extend the currently existing APD tool with several features. This tool will support multiple users. Therefore, the tool will have to have a solid workflow and proper user management controls. The user management system has to be created and several improvements have to be made to the user interface in order to realize these goals.

Reporting on the progress of the experiments done in the tool is important, as well as the extensive visualization of the output. Currently, the tool is able to discover patterns and anomalous subgraphs from a process model and an event log. These patterns and subgraphs give useful information on the business process. In the current version, however, comparing different result types is cumbersome. Delta will provide a more convenient way to manage experiments and their results as well as the projects they belong to.

The customer will be able to manage the users, projects created by users, and monitor the interactions between the users and the tool. These controls will also be provided by Delta.

1.3 List of definitions and abbreviations

1.3.1 List of definitions

**Administrator**
A registered user with the highest available access rights who manages both the tool and its users.

**Anomalous subgraph discovery**
An experiment phase that extracts recurrent subgraphs involving one or more deviations from the process model.[4]

**APD tool**
The APD tool is an extension of the Esub tool designed to extract anomalous patterns together with their correlations. These patterns are extracted from historical logging data from past process executions. Users can upload event logs and process models on which experiments can be run. After the experiments are completed, the tool supports the users in exploring the obtained results [4].

**Business process**
A set of activities performed in an organization and technical environment that are coordinated to obtain a product or service [6].

**Child subgraph**
The child $S$ of a subgraph $S'$ is a subgraph which involves $S'$ in its definition.

**Component**
A part of a phase of an experiment.

**Esub tool**
An online webtool supporting the visualization and exploration of the outcome of the frequent subgraph mining algorithm SUBDUE [2].

**Experiment**
Both the anomalous subgraph discovery and partial order discovery together.
**Experiment log** A file that tracks all activities performed within an experiment.

**Experiment phase** Anomalous subgraph discovery and partial order discovery are the two phases of one experiment.

**Event log file** A file that consists of traces [4].

**Final Result** The outcome of an experiment.

**Graph .g file** A file that collects multiple graphs, each involving a set of edges and vertices.

**Intermediate result** The outcome of either a component or an experiment phase.

**Maximal subgraph** A subgraph $s$ is maximal in a set of subgraphs if there does not exist a subgraph $s'$ such that $s'$ is a supergraph of $s$.

**Minimal subgraph** A subgraph $s$ is minimal in a set of subgraphs if there does not exist a subgraph $s'$ such that $s'$ is a subgraph of $s$.

**Parent subgraph** A subgraph $S$ is a parent of another subgraph $S'$ if $S'$ is a child of $S$.

**Partial order discovery** An experiment phase creating patterns from anomalous subgraphs and partially ordering them based on their location in the log traces. [4]

**Petri net** A mathematical model used for the specification and the analysis of parallel processes [3].

**Process model** A representation of the prescribed behavior of a business process [4].

**Project** A combination of an event log, a process model, and a unique project name. The project is stored together with any experiments run under that project name.

**Project owner** The user who created the project.

**Registered user** A user with a registered account on the APD tool.

**Responsive** A website is responsive when dynamic changes are made to the appearance of the site depending on the screen size and orientation of the device being used to view it [5].

**Result** Either an intermediate result or final result.

**Subgraph** A graph $S$ is a subgraph of $S$ if the vertices and edges of $S$ are a subset of the vertices and edges of $S$.

**Supergraph** A graph $S$ is a supergraph of graph $S'$ if the vertices and edges of $S'$ are a subset of the vertices and edges of $S$.

**Support** The support of a subgraph/pattern is equal to the fraction of graphs which involve the subgraph/pattern at least once. [4].

**Synchronous function** A task that have to be completed before a new task can be called.

**Trace** A trace in a business process model is a sequence of events generated during a process execution.

**Unregistered user** A user who does not have an account on the APD tool.
User A person who is currently using the APD tool or who has previously used the APD tool.

User activity Creating a project, viewing a project, viewing or downloading a project’s files, deleting project files, sharing a project, starting or stopping an experiment phase, viewing results and status of an experiment phase, logging in, or logging out.

User activity log A file containing information on the past user activities on the APD tool.

User tracking The act of tracking the behavior of the user on the APD tool in the form of the user activity log.

Valid email An email address of a registered user is valid when it exists and the user has access to it.

1.3.2 List of abbreviations

- **APD**: Anomalous pattern discovery
- **PNG**: Portable Network Graphics
- **TU/e**: Eindhoven University of Technology

1.4 List of references


1.5 Overview

In the next section a general description of the project is given. First the current state of the APD tool is described, followed by how the project will enhance this tool. In chapter 3 we will expand on the capabilities needed by the end product and the constraints that are put on the project. A section on the user characteristics and the environment description will elaborate on the surrounding entities concerning the product. Chapter 4 presents the list of formal capability and constraint requirements. Lastly a list of relevant use cases are presented in an the appendix.
2 General Description

2.1 Product perspective

The APD tool was designed to provide the ability to analyze their business processes. Using this tool it is possible to detect whether observed events comply with the behavior prescribed in the process model and to detect any deviations that occurred. The user can see whether their model is actually representative of the way the process is executed in reality and if a process is executed incorrectly. This type of analysis can take a long time and the APD tool handles this by running it on a server and allowing users to return to view the results at a later time. In the tool experiments are grouped in projects, which require an event log, process model, and unique name to create. These experiments are made up of two phases, anomalous subgraph discovery and partial order discovery. For the partial order discovery to run it is required to have results of the anomalous subgraph discovery, whether they are generated or uploaded to the tool. Both phases are in turn made up of multiple components, each of which serve a specific function in the experiment phase. Components generate results, which are referred to as intermediate results. The final result of a project is the partial ordering of anomalous patterns.

Delta aims to improve the APD tool in various ways. Currently, users of the tool are all treated identically, meaning it is not possible to track which user initiated which experiment phase or which results belong to whom. This will be changed by allowing users to register for an account and thus manage their own experiments and sharing the results with other users only if they choose to do so. Also users cannot see the status of any experiment phases they are running. Instead being dependent on checking back occasionally to see whether new results have appeared. In the new version users will be able to see the status of their experiment phases, seeing the progress so far, any intermediate results, and whether errors have occurred. The APD-tool with the Delta extension will most likely be used for academic and/or educational purposes only.

2.2 General capabilities

2.2.1 User management

The first addition is a system to control user access. Firstly, the system needs to keep track of users registered on the tool. These registered users will then be granted certain rights, which are explained in detail in section 2.4. To register, users need to enter the following required information:

- Username
- Password
- Email address

The information that is optional to fill in is:

- First name
- Last name
- Organization they are with
- Role in their organization
- Home country
- How the user heard about the tool

Most of this account information can be viewed and changed by the user after the registration. The user name can only be changed by an administrator. The tool will receive more functionalities to aid an administrator in managing these accounts. The manual creation and deletion of accounts will be possible. All projects created by the registered users can be accessed by an administrator.
The activity of the users can also be tracked. If user tracking is enabled for a given user, the tool will collect and log data on the activities that the user performs on the tool. The administrator can then view this information, as well as enable or disable the tracking.

To distinguish between the rights of different users, Delta will add an authentication framework to the tool so the checking of a user’s rights can be done separately from the tool’s other functions.

2.2.2 Project management

Registered users are able to create new projects when they give a project name and upload an event log and process model. These projects are stored and the creator of the project is set as its owner. Delta will give registered users a personalized page after they log in. This page then shows them their projects. Registered users can search for projects they are an owner of by name. The owner can choose to view a project, after which they are shown associated experiment states, as well as logs and results generated by these experiments. Project files can also be deleted by the owner.

The project owner can also share a project with other registered users so both can work on it together or view the results. Which access rights are given to the user the project owner shares with can be customized. Sharing a project will be done by setting the role of other registered users to either project member or project observer. For more details on sharing projects, see the user characteristics in section 2.4.

2.2.3 Experiment management

In its current state the APD tool does not give users insight into the status of running experiments, instead once an experiment phase is started the user gets no information until it completes and their results appear in the tool. Delta will provide users with an interface to monitor the status of their experiments. Users will be able to see the progress of each of their running experiments as well as to get details about any errors encountered while it is running. Experiment logs will be created such that they are understandable and note important information on the status of experiments. Registered users will also be able to stop running experiments.

Running experiments and viewing the results currently is a difficult process partly because the user interface is lacking in certain aspects. When an user is viewing their projects, experiments can be searched for by date or by status. The viewing of experiment results will also be changed to allow users to view all results in a single screen instead of forcing them to switch between multiple different views. These results include both the anomalous subgraphs and the partial orderings. When viewing the anomalous subgraphs users will also be able to view all the parents or children of subgraphs and expand a node to view its complete graph.

As the components will be modularized, phases can be run starting at a specific component. To use this functionality, a registered user has to upload their own result files so that a component can use these files as input. It is required that the phase has run completely once before this is an option to ensure that no results are missing.

To easily start a batch of experiments and view the comparative results, Delta lets users specify a range of values for the experiment phase parameters instead of a single value. The tool will then generate a set of experiments from these ranges which will then be run in sequence. When all the experiments have finished, comparative results can be shown to the user, such as the average support value and the number of anomalous patterns.

2.2.4 Subgraph discovery component customization

The final, and optional, change will be to allow users to customize the subgraph discovery phase of an experiment. Currently a standard algorithm is used to discover the subgraphs, but this is not always an optimal or accurate method for every input. To improve this, an administrator will be able to upload different algorithms which all registered users can then use.
2.3 General constraints

Delta requires the APD tool to be usable from a desktop or laptop computer running the Google Chrome version 66 browser. Secondary browsers which the tool might function on are Firefox version 59, Internet Explorer version 11, and Safari version 8, but these are not required. The current version of the tool is available in English and the extension will continue using English. The new user access system should support a minimum of 200 registered users, as the target audience is not expected to exceed this number.

This target audience also influences the usability requirements for the extension by Delta. Academics and students can be expected to have knowledge of the functionalities of the tool, so it is not necessary to design the user interface in such a way that it can be fully utilized without prior knowledge. This does not mean usability is of no concern, instead it just means that the focus needs to be on improving the experience and possibilities for more advanced users. This will be achieved by creating an interface which allows for an easier monitoring of running experiments and more in-depth analysis of results.

A low priority requirement is making the components of the experiment phases modular. Working on this requirement would be the only moment where Delta has an impact on the performance of the tool. The user access requirements should not cause any noticeable performance issues and neither should the experiment interface requirements.

2.4 User characteristics

The APD tool will be provided with user management functionality by Delta. The user management system will allow users to register and log in/log out. A distinction is made between the different types of users, where different types will also have different access rights. This offers some advantages, such as the fact that users can now create their own private projects. Projects are no longer shared between all users of the tool which makes keeping track of different experiments and projects easier.

2.4.1 Unregistered users

Users who are not registered are not able to run experiments. They are only given the right to access the demo results so that they can get a feel for what the tool offers. Unregistered users will have the option to register for an account. In order to create their own projects, they will have to be registered.

2.4.2 Registered users

Registered users have a registered account on the APD tool. These users can create new projects, run experiments, view results of experiments they have access to, and share projects they created. Registered users also get a personalized page that loads after logging in. They will be shown their projects, previously run experiments, and the status of currently running experiments. Should a user forget their password, they will be able to request a password reset. This will be sent to their email automatically.

2.4.3 Administrator

All administrators have an account, they are a type of registered user. Therefore they can access the tool and all its functions like a registered user. They can access the user activity log, as well as turn the user tracking on and off for specific users. An administrator also has the rights of a project member of every project on the APD tool. Finally user accounts can be deleted and manually created by administrators.
2.4.4 Project owner

A registered user who creates a new project is inherently that owner of that project. Only these users can share access rights of this project to other users. Sharing a project is done by setting another user as project member or project observer. These sharing rights may be set to accommodate the needs of the project owner. For example, if they want to perform some more experiments on a project and do not want others to interfere by editing or deleting files, or running their own experiments, the project owner can limit the access rights such that this is not allowed. The owner can also remove access rights for another user entirely. The project owner can also delete the whole project. The project owner has all the rights of a project member in addition to their own rights.

2.4.5 Project member

A project member is a registered user with whom a project has been shared by the owner. A project member also has all the rights of a project observer. Being a project member allows the user to start new experiments on the project or delete files of the project. A member can also upload files to a project in order to skip certain components of an experiment phase.

2.4.6 Project observer

Similar to a project member, a project observer is a registered user with whom a project has been shared by the owner. Being an observer grants the user the rights to view project files and results.

2.5 Environment description

A model of the environment is given in Figure 1. The two main components of the environment are the front-end and the back-end. The front-end will contain a web interface that will handle the displaying of the tool. This web interface will have to make requests to the back-end, where an interface will handle these requests and pass them on to either the APD tool or the authentication framework. This back-end interface enables the web interface to view the APD tool as a black box; it does not have to know the implementation of the tool.

The authentication framework receives user data from the interface with which it can check the user's access rights to the tool. It can also receive new user registration data to create new accounts that are stored on the user database.

In the front-end the main clients are desktop or laptop computers running version 66 of the Chrome web browser.

Besides the APD tool the Delta extension will not interact with any other tools/services.

2.6 Assumptions and dependencies

Delta requires the following assumptions to hold in order to function properly.

- The server supplied by the TU/e is capable of hosting the APD tool.
- The server supplied by the TU/e is online and accessible.
- The base functionality of the APD tool is stable.
Figure 1: The environment after the Delta extension
3 Specific Requirements

The requirements in this section have been constructed in agreement with the customer such that the end result complies with the requirements listed here. They indicate the needs of the customer and the capabilities of the end product. The requirements have been prioritized using the MoSCoW model [1]. The specific requirements have been split into capability—or functional—requirements and constraint requirements.

The MoSCoW model consists of the following rules:

**Must have** These requirements are fundamental and have to be implemented to the product. All must haves together constitute the minimum viable product.

**Should have** These requirement are of high importance for the product, but are not essential for the business objective.

**Could have** These requirements are not essential, but could be implemented if there is time left.

**Won't have** These requirements will not be implemented in the current delivery, but can be added to later version.

3.1 User access capability requirements

3.1.1 Registration

**URF-1**
A user can register for an account.  

**URF-2**
When registering, the user can enter their first name.  

**URF-3**
When registering, the user can enter their last name.  

**URF-4**
To register, the user has to enter a username.  

**URF-5**
Usernames have to be unique on the tool.  

**URF-6**
To register, the user has to create a password.  

**URF-7**
A password needs to be at least eight characters long.  

**URF-8**
A password needs to contain at least one number.  

**URF-9**
A password needs to contain at least one lower case letter.  

**URF-10**
A password needs to contain at least one capital letter.  

**URF-11**
A password needs to contain at least one special character.
### Rights of unregistered users

**URF-22**
An unregistered user can view a demo project.

### Rights of registered users

**URF-23**
A registered user can log in.

**URF-24**
A registered user can start experiment phases on projects they are a member of.

**URF-25**
A registered user can log out.

**URF-26**
A registered user can view their personal account information.

**URF-27**
A registered user can change the email address associated with their account.
<table>
<thead>
<tr>
<th>URF-28</th>
<th>Must have</th>
<th>A registered user can change the first name associated with their account.</th>
</tr>
</thead>
<tbody>
<tr>
<td>URF-29</td>
<td>Must have</td>
<td>A registered user can change the last name associated with their account.</td>
</tr>
<tr>
<td>URF-30</td>
<td>Must have</td>
<td>A registered user can change the organization information of their account.</td>
</tr>
<tr>
<td>URF-31</td>
<td>Must have</td>
<td>A registered user can change the organizational role information of their account.</td>
</tr>
<tr>
<td>URF-32</td>
<td>Must have</td>
<td>A registered user can change the country information of their account.</td>
</tr>
<tr>
<td>URF-33</td>
<td>Should have</td>
<td>A registered user can change their password.</td>
</tr>
<tr>
<td>URF-34</td>
<td>Should have</td>
<td>A registered user can request their password to be reset without logging in.</td>
</tr>
<tr>
<td>URF-35</td>
<td>Should have</td>
<td>Resetting a password is done by sending an email to the user.</td>
</tr>
<tr>
<td>URF-36</td>
<td>Should have</td>
<td>Sending the password reset is automatically handled.</td>
</tr>
</tbody>
</table>

### 3.1.4 Rights of administrators

<table>
<thead>
<tr>
<th>URF-37</th>
<th>Should have</th>
<th>An administrator can manually create new user accounts.</th>
</tr>
</thead>
<tbody>
<tr>
<td>URF-38</td>
<td>Must have</td>
<td>An administrator can reset the password associated with a user account.</td>
</tr>
<tr>
<td>URF-39</td>
<td>Must have</td>
<td>An administrator can delete a user account.</td>
</tr>
<tr>
<td>URF-40</td>
<td>Should have</td>
<td>An administrator can view all projects on the APD tool.</td>
</tr>
<tr>
<td>URF-41</td>
<td>Should have</td>
<td>An administrator can start experiment phases on all projects on the APD tool.</td>
</tr>
<tr>
<td>URF-42</td>
<td>Should have</td>
<td>An administrator can delete files of all projects on the APD tool.</td>
</tr>
</tbody>
</table>
### 3.1.5 User tracking

<table>
<thead>
<tr>
<th>URF-43</th>
<th>Could have</th>
</tr>
</thead>
<tbody>
<tr>
<td>User activity can be tracked.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URF-44</th>
<th>Could have</th>
</tr>
</thead>
<tbody>
<tr>
<td>An administrator can access tracking data of users.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URF-45</th>
<th>Could have</th>
</tr>
</thead>
<tbody>
<tr>
<td>An administrator can enable the tracking of a user’s data.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URF-46</th>
<th>Could have</th>
</tr>
</thead>
<tbody>
<tr>
<td>An administrator can disable the tracking of a user’s data.</td>
<td></td>
</tr>
</tbody>
</table>

### 3.2 Project management

#### 3.2.1 Project creation

<table>
<thead>
<tr>
<th>URF-47</th>
<th>Must have</th>
</tr>
</thead>
<tbody>
<tr>
<td>A registered user can create a new project.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URF-48</th>
<th>Must have</th>
</tr>
</thead>
<tbody>
<tr>
<td>When creating a new project, a registered user has to enter a project name.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URF-49</th>
<th>Must have</th>
</tr>
</thead>
<tbody>
<tr>
<td>When creating a new project, a registered user has to upload an event log file.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URF-50</th>
<th>Must have</th>
</tr>
</thead>
<tbody>
<tr>
<td>When creating a new project, a registered user has to upload a process model.</td>
<td></td>
</tr>
</tbody>
</table>

### 3.2.2 Access rights

<table>
<thead>
<tr>
<th>URF-51</th>
<th>Should have</th>
</tr>
</thead>
<tbody>
<tr>
<td>A project owner can delete that project.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URF-52</th>
<th>Must have</th>
</tr>
</thead>
<tbody>
<tr>
<td>A registered user can search for projects they are an observer of by name.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URF-53</th>
<th>Must have</th>
</tr>
</thead>
<tbody>
<tr>
<td>A registered user can search for projects they are an observer of by creation date.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URF-54</th>
<th>Could have</th>
</tr>
</thead>
<tbody>
<tr>
<td>A project observer can view the files of that project.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URF-55</th>
<th>Could have</th>
</tr>
</thead>
<tbody>
<tr>
<td>A project observer can download the files of that project.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>URF-56</th>
<th>Could have</th>
</tr>
</thead>
<tbody>
<tr>
<td>A project member can delete the files of that project.</td>
<td></td>
</tr>
</tbody>
</table>
3.2.3 Access rights allocation

**URF-57**
A project owner can make registered users an observer of their project.  
*Should have*

**URF-58**
A project owner can make registered users a member of their project.  
*Should have*

**URF-59**
A project owner can make an observer of their project a member of their project.  
*Should have*

**URF-60**
A project owner can make a member of their project an observer of their project.  
*Should have*

**URF-61**
A project owner can remove the access rights of a project member.  
*Should have*

**URF-62**
A project owner can remove the access rights of a project observer.  
*Should have*

3.3 Experiment management

3.3.1 Personalized screen of registered user

**URF-63**
A registered user can view the state of previously started experiments they are an observer of.  
*Could have*

**URF-64**
A registered user can view the set of anomalous subgraphs generated by experiments they are an observer of.  
*Must have*

**URF-65**
Text files generated by an experiment that is running can be downloaded.  
*Could have*

**URF-66**
A registered user can view final results of experiments they are an observer of.  
*Must have*

**URF-67**
A registered user can view the experiment logs generated by experiments they are an observer of.  
*Must have*

**URF-68**
A registered user can search for experiments they are an observer of by date.  
*Should have*

**URF-69**
A registered user can search for experiments they are an observer of by status.  
*Should have*
3.3.2 Experiment setup

**URF-70**  
A project member can upload result files for components.  
*Could have*

**URF-71**  
A project member can start an experiment phase from a component with an uploaded file.  
*Could have*

**URF-72**  
When starting an experiment phase, a registered user can select which components of the phase will be run.  
*Could have*

**URF-73**  
A user can set a range of parameters to run a batch of experiments.  
*Could have*

**URF-74**  
A project member has to set the frequent item threshold parameter for the partial order discovery phase to run.  
*Must have*

**URF-75**  
A project member has to set the ordering relation threshold parameter value for the partial order discovery phase to run.  
*Must have*

**URF-76**  
A project member can set the number of iterations parameter for the anomalous subgraph discovery phase.  
*Could have*

**URF-77**  
A project member can set the beam parameter for the anomalous subgraph discovery phase.  
*Could have*

**URF-78**  
A project member can set the cov parameter for the anomalous subgraph discovery phase.  
*Could have*

**URF-79**  
A project member can set the div parameter for the anomalous subgraph discovery phase.  
*Could have*

3.3.3 Experiment execution

**URF-80**  
During the execution of a synchronous function, the tool indicates that a synchronous function is running.  
*Must have*

**URF-81**  
A project member can stop a running experiment phase of that project.  
*Could have*

3.3.4 Experiment log

**URF-82**  
The progress of a running experiment is logged in the experiment log.  
*Should have*

**URF-83**  
An experiment log records the time an experiment phase is started.  
*Should have*
<table>
<thead>
<tr>
<th>URF-84</th>
<th>An experiment log records the date an experiment phase is started.</th>
<th>Should have</th>
</tr>
</thead>
<tbody>
<tr>
<td>URF-85</td>
<td>An experiment log records the time an experiment phase is finished.</td>
<td>Should have</td>
</tr>
<tr>
<td>URF-86</td>
<td>An experiment log records the date an experiment phase is finished.</td>
<td>Should have</td>
</tr>
<tr>
<td>URF-87</td>
<td>An experiment log records the components of an experiment phase that have been started.</td>
<td>Should have</td>
</tr>
<tr>
<td>URF-88</td>
<td>An experiment log records the time a component is started.</td>
<td>Should have</td>
</tr>
<tr>
<td>URF-89</td>
<td>An experiment log records the date a component is started.</td>
<td>Should have</td>
</tr>
<tr>
<td>URF-90</td>
<td>An experiment log records the components of an experiment phase that have been finished.</td>
<td>Should have</td>
</tr>
<tr>
<td>URF-91</td>
<td>An experiment log records the time a component is finished.</td>
<td>Should have</td>
</tr>
<tr>
<td>URF-92</td>
<td>An experiment log records the date a component is finished.</td>
<td>Should have</td>
</tr>
<tr>
<td>URF-93</td>
<td>An experiment log records any errors that occur in an experiment.</td>
<td>Should have</td>
</tr>
<tr>
<td>URF-94</td>
<td>An experiment log records the time an error occurs.</td>
<td>Should have</td>
</tr>
<tr>
<td>URF-95</td>
<td>An experiment log records the date an error occurs.</td>
<td>Should have</td>
</tr>
</tbody>
</table>

After an experiment phase has completed, a project member can delete the experiment log.

### 3.3.5 Visualization

<table>
<thead>
<tr>
<th>URF-97</th>
<th>Every node in a subgraph can be expanded to show the corresponding subgraph.</th>
<th>Must have</th>
</tr>
</thead>
<tbody>
<tr>
<td>URF-98</td>
<td>When viewing the patterns found by an experiment, a pattern's subgraph nodes can be expanded.</td>
<td>Must have</td>
</tr>
<tr>
<td>URF-99</td>
<td>When viewing subgraph discovery results, the support value for a subgraph can be shown.</td>
<td>Could have</td>
</tr>
<tr>
<td>URF-100</td>
<td>When viewing subgraph discovery results, the number of occurrences for a subgraph can be shown.</td>
<td>Could have</td>
</tr>
<tr>
<td>Requirement</td>
<td>Category</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>URF-101</td>
<td>Must have</td>
<td>The direct parents of a selected subgraph are indicated when viewing subgraph discovery results.</td>
</tr>
<tr>
<td>URF-102</td>
<td>Must have</td>
<td>The direct children of a selected subgraph are indicated when viewing subgraph discovery results.</td>
</tr>
<tr>
<td>URF-103</td>
<td>Should have</td>
<td>The parents of a selected subgraph can be shown up to 5 levels deep.</td>
</tr>
<tr>
<td>URF-104</td>
<td>Should have</td>
<td>The children of a selected subgraph can be shown up to 5 levels deep.</td>
</tr>
<tr>
<td>URF-105</td>
<td>Could have</td>
<td>A selection of subgraphs can be exported as a PNG file.</td>
</tr>
<tr>
<td>URF-106</td>
<td>Could have</td>
<td>A selection of subgraphs can be exported as a .g file.</td>
</tr>
<tr>
<td>URF-107</td>
<td>Could have</td>
<td>When viewing partial order discovery results, a filter can be set to only show patterns with a given support value.</td>
</tr>
<tr>
<td>URF-108</td>
<td>Could have</td>
<td>When viewing partial order discovery results, a filter can be set to only show maximal subgraphs.</td>
</tr>
<tr>
<td>URF-109</td>
<td>Could have</td>
<td>When viewing partial order discovery results, a filter can be set to only show minimal subgraphs.</td>
</tr>
<tr>
<td>URF-110</td>
<td>Could have</td>
<td>After running a batch of experiments with different parameter settings, a project observer can view the average support for the set of patterns in a single screen.</td>
</tr>
<tr>
<td>URF-111</td>
<td>Could have</td>
<td>After running a batch of experiments with different parameter settings, a project observer can view the average support for the subgraphs in a single screen.</td>
</tr>
<tr>
<td>URF-112</td>
<td>Could have</td>
<td>After running a batch of experiments with different parameter settings, a project observer can view the number of patterns for the different runs of the experiment in a single screen.</td>
</tr>
<tr>
<td>URF-113</td>
<td>Could have</td>
<td>After running a batch of experiments with different parameter settings, a project observer can sort the experiment results by support value.</td>
</tr>
</tbody>
</table>

3.4 Component customization capability requirements

3.4.1 Custom components

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URF-114</td>
<td>Could have</td>
<td>An administrator can upload a custom algorithm for the subgraph discovery phase of an experiment.</td>
</tr>
</tbody>
</table>
3.4.2 Custom component usage

**URF-115** Could have
A registered user can use every custom component uploaded by an administrator to start new experiment phases.

3.5 Constraint requirements

3.5.1 General

**URC-1** Must have
The tool is available in English.

**URC-2** Must have
The tool has to be GDPR compliant.

**URC-3** Should have
Experiment logs should be understandable to registered users who do not have experience with the tool.

3.5.2 Environment

**URC-4** Must have
The tool displays correctly on the Google Chrome version 66 browser.

**URC-5** Could have
The tool displays correctly on the Firefox version 59 browser.

**URC-6** Could have
The tool displays correctly on the Internet Explorer version 11 browser.

**URC-7** Could have
The tool displays correctly on the Safari version 8 browser.

**URC-8** Could have
The tool is responsive.

**URC-9** Won't have
The tool displays correctly on mobile browsers.

3.5.3 Performance

**URC-10** Must have
The tool can support at least 200 registered users.

**URC-11** Should have
The tool can support at least 600 registered users.

**URC-12** Could have
The tool can support at least 1000 registered users.
<table>
<thead>
<tr>
<th>URC-13</th>
<th>Must have</th>
<th>Credentials of a login request are validated within 10 seconds.</th>
</tr>
</thead>
<tbody>
<tr>
<td>URC-14</td>
<td>Should have</td>
<td>Credentials of a login request are validated within 5 seconds.</td>
</tr>
<tr>
<td>URC-15</td>
<td>Could have</td>
<td>Credentials of a login request are validated within 3 seconds.</td>
</tr>
<tr>
<td>URC-16</td>
<td>Must have</td>
<td>A request for a new account is processed within 1 minute</td>
</tr>
<tr>
<td>URC-17</td>
<td>Should have</td>
<td>A request for a new account is processed within 30 seconds</td>
</tr>
<tr>
<td>URC-18</td>
<td>Could have</td>
<td>A request for a new account is processed within 10 seconds</td>
</tr>
</tbody>
</table>
A. Use Cases

A.1 User access

A.1.1 View demo

Goal: Access a demo project by an unregistered user.

Preconditions: There exists a demo project and the user is not logged in.

Postconditions: -

Summary: An unregistered user views the demo project made available to them by the tool.

Priority: Should have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user accesses the tool.</td>
<td>3. The system displays the demo project files.</td>
</tr>
<tr>
<td>2. The user selects the demo project.</td>
<td>4. The system lists the demo project experiments.</td>
</tr>
<tr>
<td>5. The user selects an experiment.</td>
<td>6. The system displays the experiment results.</td>
</tr>
</tbody>
</table>

Alternatives:
- 5.a The user selects an experiment log file. In this case the tool displays the log file’s data.
- 5.b The user selects a single project file. In this case the tool offers the unregistered user the option to download the file.

A.1.2 Registration

Goal: Create an account for the APD tool.

Preconditions: The user is unregistered.

Postconditions: A new account is registered.

Summary: A new user creates an account for the APD tool.

Priority: Must have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user initiates the registration process.</td>
<td>2. The system displays the registration form.</td>
</tr>
<tr>
<td>3. The user fills in all required fields.</td>
<td>5. The system sends the data to the database.</td>
</tr>
<tr>
<td>4. The user confirms the data is filled in.</td>
<td>6. The system notifies the user.</td>
</tr>
<tr>
<td></td>
<td>7. The system notifies the administrators.</td>
</tr>
</tbody>
</table>

Alternatives:
• 4.a The user has not filled in all required fields appropriately. The user is informed of this and the use case goes back to step 3.
• 4.b The username entered by the user is already in use. The user is informed of this and the use case goes back to step 3.
• 4.c The e-mail address entered by the user is already in use. The user is informed of this and the use case goes back to step 3.
• 4.d The e-mail address entered by the user is not valid. The user is informed of this and the use case goes back to step 3.
• 4.e The password entered by the user does not meet all requirements, the user is informed of this. The use case goes back to step 3.
• 4.f The user fills in all optional fields. The use case goes to step 4.

A.1.3 Logging in

Goal: Log in to the APD tool.
Preconditions: The user is a registered user.
Postconditions: The user is logged in.
Summary: A registered user logs in to the APD tool.
Priority: Must have.
Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user initiates the login process.</td>
<td>2. The system displays the login screen.</td>
</tr>
<tr>
<td>3. The user fills in their username and password.</td>
<td>5. The system validates the credentials.</td>
</tr>
<tr>
<td>4. The user confirms the data is filled in.</td>
<td>6. The system displays the personalized home screen.</td>
</tr>
</tbody>
</table>

Alternatives:

• 6.a The wrong credentials were entered. A message is shown to inform the user and the use case goes back to step 3.

A.1.4 Logging out

Goal: Log out from the APD tool.
Preconditions: The user is logged in.
Postconditions: The user is logged out.
Summary: A registered user logs out from the APD tool.
Priority: Must have.
Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user initiates the logout process.</td>
<td>2. The system changes the state of the user to logged out.</td>
</tr>
<tr>
<td></td>
<td>3. The system displays the login screen.</td>
</tr>
</tbody>
</table>
A.1.5 Access account information

Goal: Access account information by a registered user.

Preconditions: The user is logged in.

Postconditions: -

Summary: A registered user accesses his account information.

Priority: Must have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to their account information.</td>
<td>2. The system displays their account information.</td>
</tr>
</tbody>
</table>

A.1.6 Change account information

Goal: Change account information by a registered user.

Preconditions: The user is logged in.

Postconditions: The account information of the registered user is changed.

Summary: A registered user changes their account information.

Priority: Must have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to their account information.</td>
<td>2. The system displays their account information.</td>
</tr>
<tr>
<td>3. The user changes the information in the appropriate field.</td>
<td></td>
</tr>
<tr>
<td>4. The user chooses to save the changes.</td>
<td>5. The system sends the data to the database.</td>
</tr>
</tbody>
</table>

A.1.7 Change password

Goal: Change the password by a registered user.

Preconditions: The user is logged in.

Postconditions: The password of the registered user is changed.

Summary: A registered user changes his password.

Priority: Should have.

Steps:
Actor actions | System actions
---|---
1. The user performs use case A.1.5: Access account information. | 3. The system displays the form to change the password.
2. The user initiates the process to change his password. | 4. The user fills in all the required fields.
5. The user confirms the data is filled in. | 6. The system sends the data to the database.
7. The system notifies the user.

Alternatives:
- 5.a The password does not meet the requirements, the user is informed about this. The use case goes back to step 4.

A.1.8 Create an account as administrator

Goal: Create a new account by an administrator.

Preconditions: The administrator is logged in.

Postconditions: A new account is registered.

Summary: An administrator creates a new account.

Priority: Should have.

Steps:

Actor actions | System actions
---|---
1. An administrator initiates the process to create a new account. | 2. The system displays a registration form.
3. An administrator fills in all required fields. | 5. The system sends the data to the database.
4. An administrator confirms the data is filled in. | 6. The system notifies the administrator.

Alternatives:
- 4.a An administrator has not filled in all required fields appropriately. The administrator is informed of this and the use case goes back to step 3.
- 4.b The username entered by the administrator is already in use. The administrator is informed of this and the use case goes back to step 3.
- 4.c The e-mail address entered by an administrator is already in use. The administrator is informed of this and the use case goes back to step 3.
- 4.d The password entered by an administrator does not meet the requirements. The administrator is informed of this and the use case goes back to step 3.

A.1.9 Delete an account as administrator

Goal: Delete a registered account by an administrator.
Preconditions: The administrator is logged in.

Postconditions: A registered account is deleted from the database.

Summary: An administrator deletes a registered account.

Priority: Must have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. An administrator navigates to the registered user accounts.</td>
<td>4. The system deletes the selected account from the database.</td>
</tr>
<tr>
<td>2. An administrator selects a registered user account.</td>
<td>5. The system notifies the administrator.</td>
</tr>
<tr>
<td>3. An administrator chooses to delete the account.</td>
<td></td>
</tr>
</tbody>
</table>

A.2 User tracking

A.2.1 Tracking User Activity

Goal: Keep track of what users do with the APD tool.

Preconditions: The user is logged in and user tracking is enabled by an administrator.

Postconditions: -

Summary: The user activity of a user is tracked.

Priority: Could have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user performs a user activity.</td>
<td>2. The system logs that the user has performed that activity in the user activity log.</td>
</tr>
</tbody>
</table>

A.2.2 Accessing tracking data

Goal: Access the user tracking data of a registered user by an administrator.

Preconditions: The administrator is logged in.

Postconditions: -

Summary: An administrator accesses the tracking data of a registered user.

Priority: Could have.

Steps:
### A.2.3 Enabling tracking

**Goal:** Enable tracking of a registered user's activity by an administrator.

**Preconditions:** The administrator is logged in and the registered user's activity tracking is disabled.

**Postconditions:** The activity of the registered user is tracked.

**Summary:** An administrator enables the tracking of a registered user's activity.

**Priority:** Could have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. An administrator navigates to the administrator page.</td>
<td>2. The system displays the administrator page.</td>
</tr>
<tr>
<td>3. An administrator chooses the registered user.</td>
<td>5. The system displays the data tracking of the registered user.</td>
</tr>
<tr>
<td>4. An administrator chooses to show user tracking data.</td>
<td></td>
</tr>
<tr>
<td>5. The system enables activity tracking of the registered user.</td>
<td></td>
</tr>
</tbody>
</table>

### A.2.4 Disabling tracking

**Goal:** Disable the tracking of a registered user's activity by an administrator.

**Preconditions:** The administrator is logged in and the registered user's activity tracking is enabled.

**Postconditions:** The activity of the registered user is not tracked.

**Summary:** An administrator disables the tracking of a registered user's activity.

**Priority:** Could have.

**Steps:**
<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. An administrator navigates to the administrator page.</td>
<td>2. The system displays the administrator page.</td>
</tr>
<tr>
<td>3. An administrator chooses the registered user.</td>
<td>5. The system disables activity tracking of the registered user.</td>
</tr>
<tr>
<td>4. An administrator chooses to disable user activity tracking.</td>
<td></td>
</tr>
</tbody>
</table>

**A.3 Project management**

**A.3.1 Creating a new project**

**Goal:** Create a new project.

**Preconditions:** The user is logged in.

**Postconditions:** A new project is created and the registered user is the project owner.

**Summary:** A registered user creates a new project.

**Priority:** Must have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user initiates the process to create a new project.</td>
<td>2. The system displays the project creation form.</td>
</tr>
<tr>
<td>3. The user enters the project name.</td>
<td>7. The system sends the data to the database.</td>
</tr>
<tr>
<td>4. The user provides the event log file.</td>
<td>8. The system creates the project.</td>
</tr>
<tr>
<td>5. The user provides the process model file.</td>
<td>9. The system sets the user as the project owner.</td>
</tr>
<tr>
<td>6. The user confirms the data is filled in.</td>
<td>10. The system notifies the user.</td>
</tr>
</tbody>
</table>

**Alternatives:**

- 6.a The project name is missing. The user is informed of this and the use case goes back to step 3.
- 6.b The event log file is missing. The user is informed of this and the use case goes back to step 4.
- 6.c The format of the event log file is incorrect. The user is informed of this and the use case goes back to step 4.
- 6.d The process model file is missing. The user is informed of this and the use case goes back to step 5.
- 6.e The format of the process model file is incorrect. The user is informed of this and the use case goes back to step 5.
A.3.2 Access files of a project

Goal: Access files of a project.

Preconditions: The user is logged in and the user is a project observer.

Postconditions: -

Summary: A registered user accesses files of a project they are an observer of.

Priority: Could have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to their projects.</td>
<td>4. The system displays the files.</td>
</tr>
<tr>
<td>2. The user selects a project.</td>
<td></td>
</tr>
<tr>
<td>3. The user chooses to access the files.</td>
<td></td>
</tr>
</tbody>
</table>

Alternatives:
- 3.a The user chooses to download the files. The use case goes to step 4.a.
- 4.a The system sends the files, the user then downloads the files.

A.3.3 Delete files of a project

Goal: Delete files of a project.

Preconditions: The user is logged in and the user is a project member.

Postconditions: The files are removed from the project.

Summary: A registered user deletes files of a project they are a member of.

Priority: Could have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to their projects.</td>
<td>4. The system displays the files.</td>
</tr>
<tr>
<td>2. The user selects a project.</td>
<td></td>
</tr>
<tr>
<td>3. The user chooses to access the files.</td>
<td></td>
</tr>
<tr>
<td>5. The user selects the files.</td>
<td>7. The system deletes the files from the project.</td>
</tr>
<tr>
<td>6. The user chooses to delete the files.</td>
<td>8. The system notifies the user.</td>
</tr>
</tbody>
</table>

A.3.4 Share a project

Goal: Give access rights for a project to another user.

Preconditions: The user is logged in and is the project owner.

Postconditions: Another registered user received access rights for the project.

Summary: A registered user shares a project with another registered user.

Priority: Should have.

Steps:
### Delete a project

**Goal:** Delete an entire project.

**Preconditions:** The user logged in and the user is project owner.

**Postconditions:** The project is deleted.

**Summary:** A project owner deletes their project.

**Priority:** Should have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to their projects.</td>
<td>4. The system notifies the user that this action is irreversible.</td>
</tr>
<tr>
<td>2. The user selects a project they are the owner of.</td>
<td></td>
</tr>
<tr>
<td>3. The user chooses to delete the project.</td>
<td>5. The user chooses to delete the project.</td>
</tr>
<tr>
<td>6. The user confirms the data is filled in.</td>
<td>7. The system deletes the project and all associated files.</td>
</tr>
<tr>
<td>7. The system sends the data to the server.</td>
<td>7. The system notifies the user.</td>
</tr>
</tbody>
</table>

**Alternatives:**

- 6.a The username is not found. The user is informed of this and the use case goes back to step 4.

---

### Change access rights

**Goal:** Change the access rights of another user.

**Preconditions:** The user is logged in, the user is the project owner and the other user is either an project observer or a project member.

**Postconditions:** The access rights of the other registered user are changed.

**Summary:** The project owner changes the access rights for another registered user.

**Priority:** Should have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to their projects.</td>
<td></td>
</tr>
<tr>
<td>2. The user selects a project they are the owner of.</td>
<td>4. The system notifies the user that this action is irreversible.</td>
</tr>
<tr>
<td>3. The user chooses to delete the project.</td>
<td>5. The user chooses to delete the project.</td>
</tr>
<tr>
<td>6. The system deletes the project and all associated files.</td>
<td>7. The system notifies the user.</td>
</tr>
</tbody>
</table>
### A.3.7 Remove access rights

**Goal:** Remove the access rights for another user.

**Preconditions:** The user is logged in, the user is the project owner and the other user is either a project observer or a project member.

**Postconditions:** The other user has no access rights for the project

**Summary:** The project owner removes the access rights of another registered user.

**Priority:** Should have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to their projects.</td>
<td></td>
</tr>
<tr>
<td>2. The user selects a project they are the owner of.</td>
<td></td>
</tr>
<tr>
<td>3. The user selects who they want to change the access rights of.</td>
<td></td>
</tr>
<tr>
<td>4. The user changes the access rights of the selected user to either project observer or project member.</td>
<td></td>
</tr>
<tr>
<td>5. The user confirms the data is filled in.</td>
<td>6. The system sends the data to the server.</td>
</tr>
<tr>
<td>5.a The username was not found. The user is informed of this and the use case goes back to step 4.</td>
<td></td>
</tr>
</tbody>
</table>

### A.4 Experiment management

#### A.4.1 Run anomalous subgraph discovery

**Goal:** Start the anomalous subgraph discovery phase.

**Preconditions:** The user is logged in and the user is a project member.

**Postconditions:** The phase is running.

**Summary:** The user starts anomalous subgraph discovery.

**Priority:** Must have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to their projects.</td>
<td></td>
</tr>
<tr>
<td>2. The user selects a project they are the owner of.</td>
<td></td>
</tr>
<tr>
<td>3. The user selects who they want to remove the access rights from.</td>
<td></td>
</tr>
<tr>
<td>4. The user removes the access rights from the selected user.</td>
<td>5. The system sends the data to the server.</td>
</tr>
</tbody>
</table>

---
A.4.2 Run anomalous subgraph discovery with custom parameters

**Goal:** Run the anomalous subgraph discovery phase with custom parameters.

**Preconditions:** The user is logged in and the user is a project member.

**Postconditions:** The parameters for anomalous subgraph discovery are set and the experiment phase is running.

**Summary:** The user sets the parameters for anomalous subgraph discovery and starts the experiment phase.

**Priority:** Could have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to their projects.</td>
<td></td>
</tr>
<tr>
<td>2. The user selects a project they are a member of.</td>
<td></td>
</tr>
<tr>
<td>3. The user initiates the process to set parameters for anomalous subgraph discovery.</td>
<td></td>
</tr>
<tr>
<td>4. The user sets the number of iterations.</td>
<td></td>
</tr>
<tr>
<td>5. The user sets the beam.</td>
<td></td>
</tr>
<tr>
<td>6. The user sets the cov.</td>
<td></td>
</tr>
<tr>
<td>7. The user sets the div.</td>
<td></td>
</tr>
<tr>
<td>8. The user confirms the data is filled in.</td>
<td></td>
</tr>
<tr>
<td>9. The system starts the anomalous subgraph discovery phase with the given parameters.</td>
<td></td>
</tr>
</tbody>
</table>

A.4.3 Run partial order discovery

**Goal:** Run the partial order discovery phase.

**Preconditions:** The user is logged in and the user is a project member.

**Postconditions:** The partial order discovery experiment phase is started.

**Summary:** The user starts partial order discovery.

**Priority:** Must have.

**Steps:**
Actor actions | System actions
---|---
1. The user navigates to their projects. | 
2. The user selects a project they are a member of. | 
3. The user initiates the process to set parameters for partial order discovery. | 
4. The user sets frequent itemset threshold. | 
5. The user sets ordering relation threshold. | 
6. The user confirms the data is filled in. | 
7. The system starts the partial order discovery phase with the given parameters. |

### A.4.4 Stop a running experiment phase

**Goal:** Stop an experiment phase while it is running.

**Preconditions:** The user is logged in, the user is a project member and an experiment phase is running on this project.

**Postconditions:** The experiment phase has stopped running.

**Summary:** A project member stops a running experiment phase.

**Priority:** Could have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to their projects.</td>
<td></td>
</tr>
<tr>
<td>2. The user selects a project they are a member of.</td>
<td></td>
</tr>
<tr>
<td>3. The user selects a running experiment phase.</td>
<td>4. The system displays the state of the running experiment.</td>
</tr>
<tr>
<td>5. The user chooses to stop the experiment phase.</td>
<td>6. The system stops the experiment phase.</td>
</tr>
<tr>
<td>7. The system notifies the user.</td>
<td></td>
</tr>
</tbody>
</table>

### A.4.5 Accessing experiment information

**Goal:** Access experiment information.

**Preconditions:** The user is logged in, the user is a project observer and an experiment of this project is in progress or finished.

**Postconditions:** -

**Summary:** A registered user accesses experiment information.

**Priority:** Must have.

**Steps:**
A.4.6 View experiment results

**Goal:** View the results of an experiment.

**Preconditions:** The user is logged in and the user is a project observer.

**Postconditions:** Experiment results are displayed.

**Summary:** A registered user views the results of an experiment.

**Priority:** Must have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to their projects.</td>
<td>4. The system displays the experiment's information.</td>
</tr>
<tr>
<td>2. The user selects a project they are an observer of.</td>
<td></td>
</tr>
<tr>
<td>3. The user selects an experiment.</td>
<td></td>
</tr>
<tr>
<td>5. The user chooses to view the results.</td>
<td>6. The system displays the results.</td>
</tr>
</tbody>
</table>

**Alternatives:**

- 5.a The user chooses to display the experiment logs. The use case goes to step 6.a.
- 5.b The user chooses to download the text files. The use case goes to step 6.b.
- 6.a The system displays the experiment logs.
- 6.b The system sends the text files. The user then downloads the text files.

A.4.7 Delete experiment log file

**Goal:** Delete the log file of an experiment.

**Preconditions:** The user is logged in, the user is a project member, an experiment phase of this project is completed and no phase of this experiment is running.

**Postconditions:** The log file of the experiment is deleted.

**Summary:** The user deletes the log file of an experiment.

**Priority:** Should have.

**Steps:**
A.4.8 Start multiple runs of an experiment

Goal: Start multiple executions of an experiment.

Preconditions: The user is logged in and the user is a project member.

Postconditions: The experiment runs are performed.

Summary: The user starts multiple executions of an experiment.

Priority: Could have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user initiates the process to start an experiment.</td>
<td>5. The system generates a set of experiments from the given parameter ranges.</td>
</tr>
<tr>
<td>2. The user specifies a range of frequent itemset threshold values.</td>
<td></td>
</tr>
<tr>
<td>3. The user specifies a range of ordering relation threshold values.</td>
<td></td>
</tr>
<tr>
<td>4. The user confirms the data is filled in.</td>
<td>6. The system starts the set of experiments.</td>
</tr>
</tbody>
</table>

A.4.9 Comparing results of multiple runs of an experiment

Goal: Display comparative results between experiments.

Preconditions: The user is logged in, the user is a project observer and an experiment in this project has been run multiple times.

Postconditions: Comparative results of the experiment are displayed.

Summary: A registered user displays comparative results between runs of an experiment.

Priority: Could have.

Steps:
### A.4.10 Upload custom subgraph discovery algorithm

**Goal:** Upload custom algorithm of the subgraph mining algorithm.

**Preconditions:** An administrator is logged in.

**Postconditions:** The custom algorithm is uploaded to the server.

**Summary:** An administrator uploads a custom algorithm of the subgraph mining algorithm.

**Priority:** Could have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>An administrator chooses to upload a custom algorithm.</td>
<td>The system sends the data to the database.</td>
</tr>
<tr>
<td>An administrator selects the custom algorithm to upload.</td>
<td></td>
</tr>
<tr>
<td>An administrator chooses which type of experiment this algorithm is associated with.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The system notifies the administrator.</td>
</tr>
</tbody>
</table>

Alternatives:

- **7.a** The user chooses to display the average support for the subgraphs. The use case goes to step 8.a.
- **7.b** The user chooses to display the number of pattern for each run of the experiment. The use case goes to step 8.b.
- **7.c** The user chooses to sort the experiment results by support value. The use case goes to step 8.c.
- **8.a** The system displays the average support for the subgraphs in the current screen.
- **8.b** The system displays the number of patterns for each run.
- **8.c** The system sorts the experiment results by support value. The system displays the sorted list of experiment results.
A.4.11 Start subgraph discovery with a custom implementation of the algorithm

Goal: Start executing subgraph discovery with a custom implementation of the algorithm.

Preconditions: The user is logged in, The user is a project member and a custom implementation of the subgraph discovery algorithm has been uploaded by an administrator.

Postconditions: The subgraph discovery phase is running, using the custom implementation.

Summary: The user starts subgraph discovery using a custom implementation of the algorithm.

Priority: Could have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to their projects.</td>
<td></td>
</tr>
<tr>
<td>2. The user selects a project they are a member of.</td>
<td></td>
</tr>
<tr>
<td>3. The user chooses to start subgraph discovery with a custom implementation.</td>
<td>4. The system displays a list of available implementations.</td>
</tr>
<tr>
<td>5. The user chooses the desired implementation.</td>
<td>6. The system starts the experiment phase, using the custom implementation.</td>
</tr>
</tbody>
</table>

A.5 Component customization

A.5.1 Upload intermediate result

Goal: Upload an intermediate result of a component of an experiment.

Preconditions: The user is logged in, the user is a project member and the user has a file representing an intermediate result of a component.

Postconditions: A file is uploaded and can be used for the project.

Summary: A user uploads a file representing an intermediate result of a component.

Priority: Could have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user initiates the process to upload an intermediate result.</td>
<td></td>
</tr>
<tr>
<td>2. The user selects a file from his local machine.</td>
<td>4. The system sends the file to the server.</td>
</tr>
<tr>
<td>3. The user confirms to upload the selected file.</td>
<td></td>
</tr>
</tbody>
</table>

Alternatives:

- 4.a The file type is not supported. The user is informed of this and the use case goes back to step 2.
- 4.b The file was not successfully uploaded. The user is informed of this and the use case goes back to step 2.
A.5.2 Select components for the experiment setup

**Goal:** Select the components that the user wants to be executed.

**Preconditions:** The user is logged in, the user is a project member and the user has uploaded intermediate result files for relevant components of the experiment.

**Postconditions:** -

**Summary:** The user selects the components that he wants to be run.

**Priority:** Could have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user initiates the process to select the component for an experiment.</td>
<td></td>
</tr>
<tr>
<td>2. The user selects the component to run.</td>
<td></td>
</tr>
<tr>
<td>3. The user selects the intermediate results for the skipped components.</td>
<td></td>
</tr>
<tr>
<td>4. The user confirms the choice for components.</td>
<td></td>
</tr>
<tr>
<td>5. The system sends the data to the server.</td>
<td></td>
</tr>
</tbody>
</table>

A.5.3 Start anomalous subgraph discovery with a custom implementation

**Goal:** Start executing an experiment with a custom implementation.

**Preconditions:** The user is logged in, the user is a project member and a custom implementation is uploaded by an administrator.

**Postconditions:** The experiment is running, using the custom implementation.

**Summary:** The user runs a custom implementation of the anomalous subgraph discovery.

**Priority:** Could have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to their projects.</td>
<td></td>
</tr>
<tr>
<td>2. The user selects a project they are a member of.</td>
<td></td>
</tr>
<tr>
<td>3. The user chooses to start anomalous subgraph discovery with a custom implementation.</td>
<td></td>
</tr>
<tr>
<td>4. The system displays a list of available implementations.</td>
<td></td>
</tr>
<tr>
<td>5. The user selects the custom implementation they wish to execute.</td>
<td></td>
</tr>
<tr>
<td>6. The system starts running the experiment phase with the custom implementation.</td>
<td></td>
</tr>
</tbody>
</table>

A.6 Visualization

A.6.1 Search projects by name

**Goal:** Search a project by name.

**Preconditions:** The user is logged in.
**Postconditions:** The projects that matches the provided name are displayed.

**Summary:** A registered user searches a project by entering its name.

**Priority:** Should have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to search.</td>
<td>4. The system searches all projects the user has access to.</td>
</tr>
<tr>
<td>2. The user enters the name of the project.</td>
<td>5. The system filters the search results by name.</td>
</tr>
<tr>
<td>3. The user chooses to search by name.</td>
<td>6. The system displays the remaining projects.</td>
</tr>
</tbody>
</table>

A.6.2 **Search projects by creation date**

**Goal:** Search a project by date.

**Preconditions:** The user is logged in.

**Postconditions:** The projects that matches the provided creation date are displayed.

**Summary:** A registered user searches a project by entering its creation date.

**Priority:** Should have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to search.</td>
<td>4. The system searches all projects the user has access to.</td>
</tr>
<tr>
<td>2. The user enters the creation date of the project.</td>
<td>5. The system filters the search results on date.</td>
</tr>
<tr>
<td>3. The user chooses to search by date.</td>
<td>6. The system displays the remaining projects.</td>
</tr>
</tbody>
</table>

A.6.3 **Search experiments by running date**

**Goal:** Search an experiment by date.

**Preconditions:** The user is logged in.

**Postconditions:** The experiments that matches the provided running date are displayed.

**Summary:** A registered user searches an experiment by entering its running date.

**Priority:** Should have.

**Steps:**
### A.6.4 Search experiments by status

**Goal:** Search an experiment by status.

**Preconditions:** The user is logged in.

**Postconditions:** The experiments that match the provided status are displayed.

**Summary:** A registered user searches an experiment by entering its status.

**Priority:** Should have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user navigates to search.</td>
<td>4. The system searches for all the experiments the user has access to.</td>
</tr>
<tr>
<td>2. The user enters the status of the experiment.</td>
<td>5. The system filters the search results on experiment status.</td>
</tr>
<tr>
<td>3. The user chooses to search by status.</td>
<td>6. The system displays the remaining experiments.</td>
</tr>
</tbody>
</table>

### A.6.5 Expand a node

**Goal:** Expand a node in the subgraph hierarchy to show the corresponding graph.

**Preconditions:** The results of an experiment are displayed.

**Postconditions:** The expanded subgraph of the selected node is shown.

**Summary:** A registered user selects a node for which they want to see the corresponding expanded subgraph.

**Priority:** Must have.

**Steps:**

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user selects a node.</td>
<td>3. The system displays the corresponding expanded subgraph of the node within the original graph.</td>
</tr>
<tr>
<td>2. The user chooses to expand the selected node.</td>
<td></td>
</tr>
</tbody>
</table>

---

Delta User Requirements Document 43
A.6.6 Filter patterns

Goal: Filter results to display patterns with a certain support value.

Preconditions: The partial order discovery results of an experiment are displayed.

Postconditions: Patterns with corresponding support value are displayed.

Summary: The registered user filters on graph data with a certain support value to only display patterns with at least that support value.

Priority: Could have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user enters a support value.</td>
<td>3. The system displays all patterns with at least the selected support value.</td>
</tr>
<tr>
<td>2. The user chooses to filter on the given support value.</td>
<td></td>
</tr>
</tbody>
</table>

Alternatives:
- 1.a The user selects the maximal subgraph option. Step 2 is executed and the use case goes to 3.a.
- 1.b The user selects the minimal subgraph option. Step 2 is executed and the use case goes to 3.b.
- 3.a The system displays all maximal subgraphs. The use case goes to step 3.
- 3.b The system displays all minimal subgraphs. The use case goes to step 3.

A.6.7 Export subgraphs

Goal: Export a subgraph to a PNG or .g file.

Preconditions: The results of an experiment are displayed.

Postconditions: A subgraph is exported.

Summary: A registered user selects subgraphs they want to export.

Priority: Could have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user selects a subgraph.</td>
<td>3. The system converts the subgraph to PNG format.</td>
</tr>
<tr>
<td>2. The user chooses to export the subgraph to a PNG file.</td>
<td>4. The system exports the file.</td>
</tr>
</tbody>
</table>

Alternatives:
- 2.a The user chooses to export the subgraph to a .g file. The use case goes to step 3.a.
- 3.a The system converts the subgraph to .g format. The use case goes to step 4.

A.6.8 Show subgraph support value

Goal: Display the support value of a subgraph.

Preconditions: Anomalous subgraph discovery results of an experiment are displayed.
Postconditions: The support value of the provided subgraph is returned.

Summary: The user chooses to display the support value of a subgraph.

Priority: Could have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user selects a subgraph.</td>
<td></td>
</tr>
<tr>
<td>2. The user chooses to display the support value.</td>
<td>3. The system displays the support value of the subgraph.</td>
</tr>
</tbody>
</table>

A.6.9 Show the number of occurrences of a subgraph

Goal: Display the number of occurrences of a subgraph.

Preconditions: Anomalous subgraph discovery results of an experiment are displayed.

Postconditions: The number of occurrences of the provided subgraph is returned.

Summary: The user chooses to display the number of occurrences of a subgraph.

Priority: Could have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user selects a subgraph.</td>
<td></td>
</tr>
<tr>
<td>2. The user chooses to display the number of</td>
<td>3. The system displays the number of occurrences of the subgraph.</td>
</tr>
<tr>
<td>occurrences of the subgraph.</td>
<td></td>
</tr>
</tbody>
</table>

A.6.10 Indicating children and parents

Goal: Indicate the direct children and parents.

Preconditions: Anomalous subgraph discovery results of an experiment are displayed.

Postconditions: The children and parents of the selected subgraph are displayed.

Summary: The user selects a subgraph. The direct parents and children are indicated.

Priority: Must have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user selects a subgraph.</td>
<td></td>
</tr>
<tr>
<td>2. The system indicates the direct parents and</td>
<td>2. The system indicates the direct parents and children of the subgraph.</td>
</tr>
<tr>
<td>children of the subgraph.</td>
<td></td>
</tr>
</tbody>
</table>

A.6.11 Showing parents or children up to 5 levels deep

Goal: Display parents or children of a subgraph up to 5 levels deep.

Preconditions: Anomalous subgraph discovery results of an experiment are displayed.

Postconditions: The children and parents of the selected subgraph are displayed up to 5 levels deep.

Summary: The user selects a subgraph. The parents or children up to 5 levels deep are displayed.
Priority: Should have.

Steps:

<table>
<thead>
<tr>
<th>Actor actions</th>
<th>System actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The user selects a subgraph.</td>
<td>4. The system displays parents with the corresponding level of depth.</td>
</tr>
<tr>
<td>2. The user enters the level of depth.</td>
<td></td>
</tr>
<tr>
<td>3. The user chooses to display the parents.</td>
<td></td>
</tr>
</tbody>
</table>

Alternatives:

- 3.a The user chooses to display the children. The use case goes to step 4.a.
- 4.a The system displays children with the corresponding level of depth.