Group 8 - Noin Seitsemän Veljestä

Biodata Dependency Manager

User Requirements Document

23.10.2006

NOIN 7 VELJESTÄ

Document Version Control

Version	**Date**	**Author**	**Changes**
1.0	12.9.2006	Kauppinen, Lehtola	Template
1.1	7.10.2006	Tom Wik	First Version
1.2	9.10.2006	Tom Wik, Timo Nieminen, Topias Uotila	Added first use cases and requirements
1.3	17.10.2006	Tom Wik	Use case changes and improvements
1.4	19.10.2006	Timo Nieminen	Typo fixing, added details in various parts, changed some priorities
1.5	20.10.2005	Timo Nieminen	Added use case diagram and high level system description
1.6	20.10.2006	Tom Wik	Refined chapter 3 and 6.4, added chapter 8. Modified use cases and requirements lists.
1.7	22.10.2006	Tom Wik	Document refined based on customer feedback
1.8	22.10.2006	Timo Nieminen	Updated use case diagram
2.0	22.10.2006	Tom Wik	PP iteration version

Document owner: Tom Wik

Table of contents

Index of tables

Index of figures

1. Introduction

The purpose of this document is to present the requirements of the Biodata Dependency Manager system from the user's point of view. The system is to be implemented as a part of the course T-76.4115 at Helsinki University of Technology. This documents acts as a means of communication between the project group and the customer in order to determine a mutual understanding of the outcome and functionality of the final product. Furthermore, this document acts as the basis for determining what functionality to implement during a given iteration. This document is constantly updated as the project progresses, and all changes associated with requirements management are done in this document. This document is intended for the parties presented in Table 1

Table 1 - Intended audience of this document

Group or party	Reasons for reading
Customer	
Customer manager	Feedback to be used for requirements validation and prioritizing
Technical advisor	Technical feedback on more detailed requirements as well as feasibility for project
End user	Feedback on the level of usability of the program and on the inclusion of all necessary functionality from a less technical perspective
Project group	
Management group	Requirements management and project status follow-up
Developers	Source for information about system functionality to be used for knowing what to implement
Testers	Source for information about how the system should be working during the test-phase
Documentation group	Material source for user manual
Course staff	
Mentor	Project progress follow-up and iteration grading and feedback

2. Business goals

This chapter describes the business situation and current problem that this project is intended to address. It describes the current situation, the role of the project as well as potential benefits for both the user and the customer company.

2.1 Current situation

In the field of biotechnology research there exists a great deal of opportunities for using software-based solutions as means of assistance. For this purpose, Medicel has developed a solution called Medicel Integrator. In essence, Medicel Integrator is “a professional heavy duty IT platform to support biological applications from a sophisticated literature mining application to a systems biology research platform”. Associated with this solution is a great deal of data in the form of various databases and information packages, such as KEGG, UniProt, and Gene Ontology. At present, databases are set up in many separate instances within one company to suit their individual needs. Movement of data from one instance to the other is cumbersome and requires a manual transfer on a case-by-case basis.

2.2 Role of project deliverable

To address the situation described in 2.1, the objective is to develop an application capable of handling the installation and management of individual data packages. These packages can be installed onto the client system as part of the Medicel Integrator platform. The package source can be other Integrator platform instances or a local install media. The application automatically handles the physical installation to the local system; more exactly the data is imported into the local database instance. In addition to this, the application also performs the mandatory checks associated with package dependencies and versions.

2.3 Project benefits

The project deliverable would benefit both Medicel as well as Medicel’s customers using the Medicel Integrator platform. For Medicel, the project deliverable means a more appealing and complete software solution, efficiently improving the procurement of new customers. This of course has direct economical benefits. For the end user, the deliverable allows a smoother means of transferring data between different Integrator instances and in general reduces administrative efforts needed. The set-up times of system installation is reduced and system update can now be performed with minimal effort. This would benefit current as well as future users.

3. Main domain concepts

This chapter defines the most important concepts of the problem domain. A glossary has also been provided in the project plan. Table 2 presents the main concepts below.

Table 2 - Main domain concepts

Group or party	
User	A Medicel customer who wants to use the domain application to manage his Medicel Integrator platform
Medicel Integrator Platform	A biological applications support platform
Biodata Dependency Manager client	The domain application used for the process of managing the content of the Medicel Integrator platform

Installation source database	A database for storing the information available to the Medicel Integrator Platform
Local database	A local database for storing received information
Package	A set of data to be installed into the Medicel Integrator Platform

In order to visualize the domain, Figure 1 shows the connection between the different domain concepts. The user, often a biological researcher or otherwise affiliated with biological data utilizes and accesses the data stored in the Medicel Integrator Platform. If need arises to complement the data in the platform, the user utilizes the Biodata Dependency Manager domain application to search and manage datasets, or packages. These packages are available for install from remote locations or other media through some form of data repository, and by using the application the user can install the packages into the local Medicel Integrator Platform. In essence, the project deliverable, the Biodata Dependecy Manager, is a data installer.

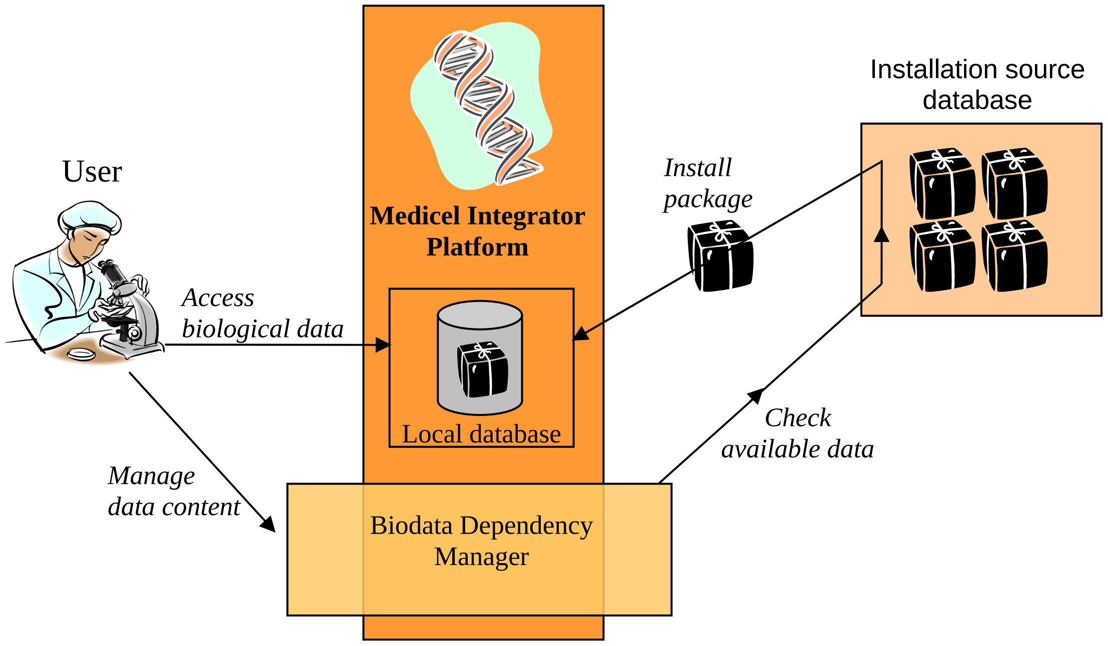

Figure 1 - Main domain concepts

4. System overview

This chapter gives a high level overview of the system from the user’s point of view. The main use cases are presented in the following use case diagram:

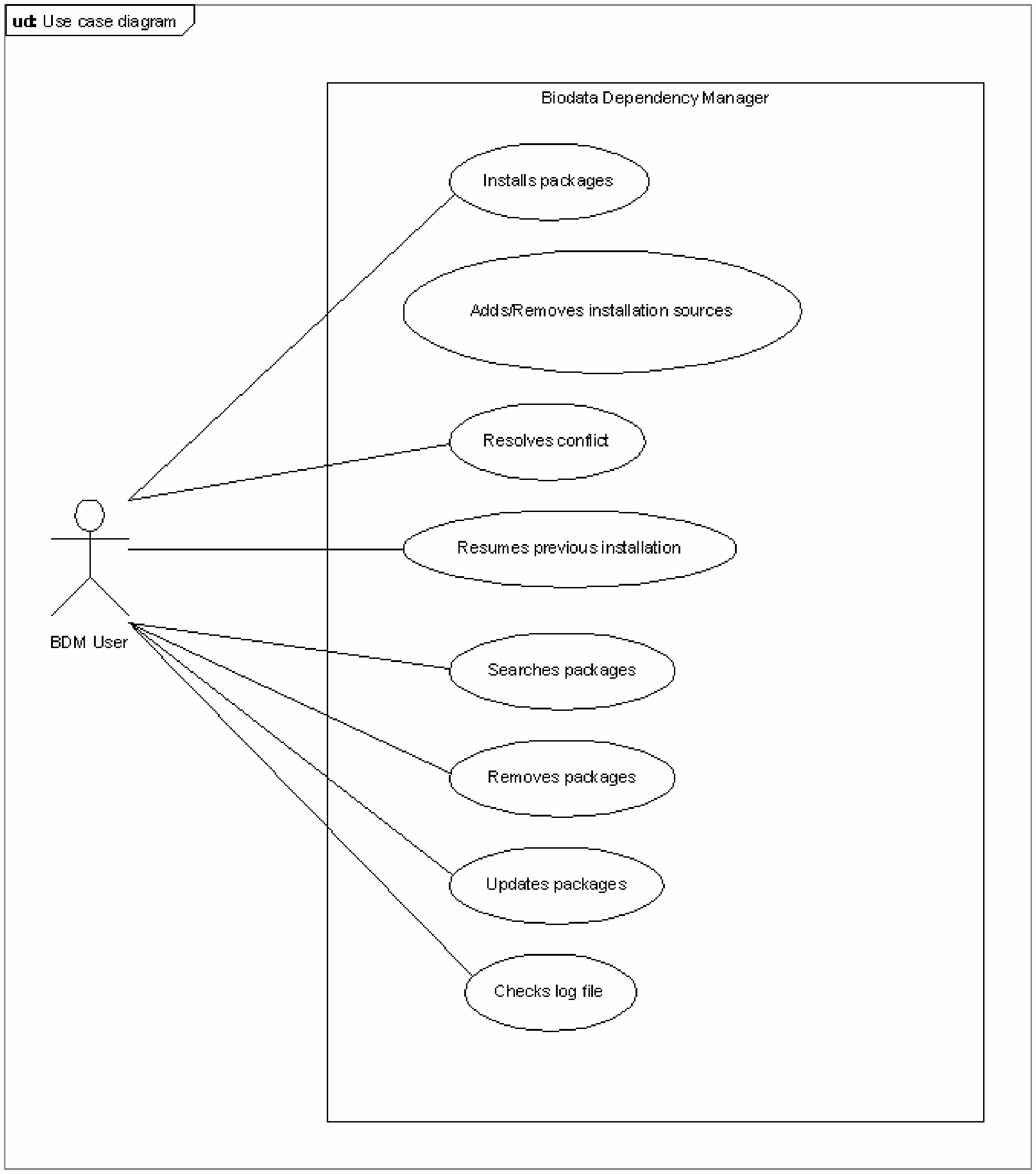

Figure 2 - Use Case Diagram

The main task of the user of the Biodata Dependency Manager is to install packages. In order to install packages, the user has to add a source for the installation. Before installing packages, the user might want to search from among all available packages. It is also essential that the user can resolve a possible conflict which has arisen from the package dependencies. If the installation takes a very long time, it’s essential that the user can close the Biodata Dependency Manager and resume the installation in progress later on. User also wants to inspect the previous installations from log file. Removing and updating packages are considered low priority but they are presented as possible use case because they have extra value for customer. More specific details of the use cases are found in chapter 6.3.

5. User groups

This chapter describes the intended users of the system. The number of users is depicted as per system/installation basis

Table 3 – User groups of the system

User group	Background	Role	Number of users	Impor-tance
Integrator Data Warehouse Administrator	Technical background, trained to use and administer the data contained within an Integrator instance. Knowledgeable of biological databases and data integration issues.	Responsible with administering the Data Warehouse Contents. Administers (Installs, updates, removes) big data packages comprising typically of whole databases such as UniProt, KEGG etc. These data packages contain all types of data: proteins, compounds, pathways, workflows etc.	1-2	High
Bioinformatics Researcher	Typically biology or bioinformatics background. Knowledgeable of the biological domain. Also trained in basic skills related to Integrator usage.	Responsible for performing research (data analysis etc.) using the Integrator instance. Needs to install smallish data packages for his/her own use, typically consisting of workflows and pathways etc.	>10	High
Application Engineer	Technical background; software development and/or bioinformatics.	Creates workflows for data analysis and pathways for modeling biological systems. Mostly needs to create and test data packages for publication. Needs to install packages mostly for testing purposes.	1-10	Medium

Integrator Data Warehouse Administrator is the main user group but the other user groups are important as well.

6. Functional requirements

This chapter defines the functionality of the system with the help of a requirements list and use cases.

6.1 Requirements elicitation

The requirements and use cases where elicited during the first three weeks of the planning iteration. The process involved two workshop afternoons as well as documentation feedback after the workshops. During the first workshop, a user story was created with the help of the customer, technical advisor, a customer GUI expert and a customer developer. Based on this story, basic use cases where constructed along with a preliminary list of requirements. During the second workshop, the use cases and requirements where scrutinized, developed and prioritized from the customer’s point of view. Final adjustments where made and the final outcome during the planning iteration passed through customer acceptance before handed in.

In this phase, focus was on both the requirements list and the use cases. The combination of both allows for a smooth transition into the next iteration since iteration and GUI design can begin simultaneously based on the contents of this document.

6.2 BDM functional requirements list

The requirements are listed in tables below, collected based on the use cases, the user story and the GUI prototype. The requirements at this stage can change during the course of the project. The priority is at this stage preliminary but has been discussed and set to the values seen in the table. Before the requirements can be taken into the development process, they have to be approved by the customer.

The priority of a requirement can be mandatory, essential or conditional. Impact and effort is based on a high, medium, low scale. Requirements with mandatory priority must be implemented in order to the project be successful. Requirements with essential priority are very important for the project. Conditional requirements create extra value for the customer. The status of a requirement can be proposed, approved, implemented, verified or deleted. The requirements are grouped according to their functionality, and sorted according to the flow in use case 1.

Table 4 - Functional requirements

ID	**Requirement**	**Prioritization viewpoints**			**Status**	**Related test case**
		Customer importance	**Architect-ure impact**	**Develop-ment effort**		
Installation source functionality						
FR01	User has to be able to add a local installation sources	Mandatory			Proposed	UC2
FR02	User has to be able to add remote installation sources	Essential			Proposed	UC2
FR03	User has to be able to remove installation sources	Mandatory			Proposed	UC3
FR04	User has to be able to browse the source of installation from the local file system	Essential			Proposed	UC2
FR05	When browsing, user has to be able to see only the valid source file types (for example .xml)	Conditional			Proposed	UC2
Package management, information						
FR06	User has to be able to install packages from many different sources, such as local sources or other Integrator systems.	Conditional			Proposed	UC1

FR07	User has to be able to see the installed packages and specific information of them, such as name, description, version and dependencies	Mandatory			Proposed	UC1
FR08	The user has to be able to check the status of a package (installed/installed partially/not installed)	Essential			Proposed	UC1
FR09	User has to be able to see the available packages and specific information of them such as name, description, version and dependencies	Mandatory			Proposed	UC1
Search functionality						
FR10	User has to be able to search packages by name	Conditional			Proposed	UC6
Package installation						
FR11	User has to be able to edit the set of packages to be installed	Mandatory			Proposed	UC1, UC5
FR12	User has to be able to differentiate the packages the system has chosen automatically and the packages that cannot be installed from all packages	Mandatory			Proposed	UC1, UC5
Package conflict presentation and resolution						
FR13	User has to be able to see the conflicts born from installation/dependencies	Mandatory			Proposed	UC5
FR14	User has to be able to solve the dependency conflicts arisen from the proposed installation.	Mandatory			Proposed	UC5
FR15	User has to be able to edit the set of packages to be installed also when a conflict has occurred	Mandatory			Proposed	UC5
FR16	The user has to be able to install packages partially when a dependency is not mandatory although suggested by the system	Essential			Proposed	UC5
Installation confirmation and progress reporting						
FR17	User has to be able to see the confirmation of the installation as dependency tree	Mandatory			Proposed	UC1, UC5
FR18	User has to be able to see the progress of installation	Essential			Proposed	UC1
FR19	User has to be informed when installation is done	Essential			Proposed	UC1
FR20	User has to be able to see information of installation afterwards from log files	Conditional			Proposed	UC7
Installation session handling						
FR21	The user has to be able to close the installation client while the installation continues since some installations take a long time	Essential			Proposed	UC4
FR22	The user has to be able to restore a previous installation session progress	Essential			Proposed	UC4
Error reporting and navigation						
FR23	Necessary error messages have to be shown to the user during the use of the	Conditional			Proposed	UC1

	system. These are presented in Table 14					
FR24	User has to be able to exit (go back) from every situation	Conditional			Proposed	UC1
Functionality not yet included in the scope						
FR25	User has to be able to remove packages	Conditional			Proposed	UC8
FR26	User has to be able to update packages.	Conditional			Proposed	UC9

6.3 BDM Use cases

The use cases are presented below in tables. Per request of the customer, the amount of use cases is kept at a minimum and instead the individual use cases are of a longer nature. The main use case, UC1, acts as the central hub for all other use cases and effectively binds them together to a logical sequence. UC1 is also the main and most typical user scenario of the system.
The use cases are derived based on the original user story created in collaboration with the customer, as well as on the GUI prototype created during the planning iteration. Since the project has a rather high dependency on user input and user interaction is of great importance, the detail of the use cases is high already at the planning stage. Some use cases use “hard-coded” steps from UC1 for their preconditions. This is done to decrease overhead, but at the same time increases maintainability problems.

Table 5 - UC1 - Install Packages

ID	UC1
Name	Install Packages
Summary	The main user sequence, where the user installs a package without problems
Actors	Integrator Data Warehouse Administrator (IDWA), Bioinformatics Researcher, Application Engineer
Preconditions	User has opened the Integrator system's start page and has the correct credentials for using the system. The Biodata Dependency Manager (BDM) application is deployed and functional.
Basic Sequence	1. User selects the BDM link from the Integrator system start page. 2. User logs in with the Integrator system's authentication system. 3. BDM presents its main screen, which shows a list of currently installed packages and installation sources. 4. User selects "Install Packages...". 5. BDM presents the installation screen, which shows a list of packages available from the installation sources. 6. User examines the packages, selects the ones to be installed and selects "Install".

		7. BDM presents the confirmation screen, which shows the packages to be installed. 8. User selects "Confirm". 9. BDM presents the installation progress and the installation completed notification. 10. User selects "Ok". 11. BDM returns to the main screen, which shows the updated list of installed packages.
Post Conditions		The package(s) is installed successfully and can be accessed from the Integrator system.
Exceptions and extensions		*2a. Login fails* 1. The Integrator system responds by presenting its standard module for user authentication again. *4a. The selection can not be made, because no installation sources are found.* 1. BDM informs user of the error. 2. User selects "Ok". 3. BDM returns to the main screen. *4b. User adds installation source* 1. Link to **Use Case UC2: Add Installation Source**. *4c. User removes installation source* 1. Link to **Use Case UC3: Remove Installation Source** *4d. User resumes previous installation* 1. Link to **Use Case UC4: Resume previous installation** *4e. User Checks Log files* 1. Link to **Use Case UC7: Check Log file** *6a. "Install" is "grayed out", if no packages are selected.* *6b. "Install" is "grayed out", if installation has conflicts* 1. Link to **Use Case UC5: Resolve Conflicts**. *6c. IDWA uses the search function.* 1. Link to **Use Case UC6: Search Packages**. *9a. Installation Fails* 1. BDM informs user of the errors and available options.
Priority	**Importance**	Must
	Architecture	High
	Effort	High
Status		Proposed
Test Cases		

Table 6 - UC2 - Add Installation Sources

ID	UC2
Name	Add Installation Sources

Summary		The user adds an installation source to the application
Actors		Integrator Data Warehouse Administrator (IDWA), Application Engineer
Preconditions		UC1 steps 1-3
Basic Sequence		1. User types the location of the installation source into the search location 2. User selects "Add". 3. BDM adds the source to the list of installation sources.
Post Conditions		The new sources are available for the BDM to use. The source is persistent and is shown the next time the user logs in.
Exceptions and extensions		*1a. User selects "Browse" instead of typing the location.* 1. BDM presents the standard file system view, which shows only .xml files. 2. User browses to the file and selects it. *2a. The indicated source can not be accessed.* 1. BDM informs the user of the error and available options.
Priority	**Importance**	Essential
	Architecture	Low
	Effort	Medium
Status		Proposed
Test Cases		

Table 7 - UC3 - Remove Installation Sources

ID		UC3
Name		Remove Installation Sources
Summary		The user removes an installation source from the application
Actors		Integrator Data Warehouse Administrator (IDWA), Application Engineer
Preconditions		UC1 steps 1-3
Basic Sequence		1. User selects an installation source from the list in the main screen. 2. User selects "Remove". 3. BDM removes the source from the list of installation sources.
Post Conditions		The new sources are available for the BDM to use.
Exceptions and extensions		*1a. The list of installation sources is empty* 1. The use case ends
Priority	**Importance**	Conditional
	Architecture	Low
	Effort	Low
Status		Proposed
Test Cases		

Table 8 - UC4 - Resume Previous Installation

ID		UC4
Name		Resume Previous Installation
Summary		The user resumes a previous installation that is still in progress
Actors		Integrator Data Warehouse Administrator (IDWA), Application Engineer
Preconditions		UC1 steps 1-3. There exists a still running installation indicated by a progress bar in the main window
Basic Sequence		1. User selects "Details" to bring back the previous installation 2. User is presented with the installation progress screen. 3. BDM shows the current status of the installation in progress. 4. Installation progresses according to step 9 in UC1
Post Conditions		The user can manage and monitor the installation in progress
Exceptions and extensions		
Priority	**Importance**	Essential
	Architecture	High
	Effort	High
Status		Proposed
Test Cases		

Table 9 – UC5 - Resolve Conflict

ID	UC5
Name	Resolve Conflict
Summary	The user wants to edit the list of packages so that they can be installed without errors
Actors	Integrator Data Warehouse Administrator (IDWA), Bioinformatics Researcher, Application Engineer
Preconditions	UC1 steps 1-7. BDM has detected that the selected packages can not be installed, because they depend on not selected packages.
Basic Sequence	1. BDM marks the conflict in the "to be installed" screen. 2. BDM suggests other packages that need to be installed 3. User accepts changes 4. The user continues the installation
Post Conditions	UC1 can proceed with a coherent set of packages.
Exceptions and extensions	*2a. All the needed packages are not available from the sources.* 1. User removes the packages depending on the unavailable

		ones and selects "Confirm". *2b. Some packages are really not needed despite the system requirement* 1. The user allows the conflict to happen and forces the installation to continue
Priority	**Importance**	Must
	Architecture	High
	Effort	High
Status		Proposed
Test Cases		

Table 10 – UC6 - Search Packages

ID		UC6
Name		Search Packages
Summary		The user wants to search for a specific packages or set of packages using a search string
Actors		Integrator Data Warehouse Administrator (IDWA), Bioinformatics Researcher, Application Engineer
Preconditions		UC1 steps 1-5. The list of available packages is shown in full
Basic Sequence		1. The user inputs his search text into the search field 2. BDM changes the list to show only packages where the packet name includes the search string in a “type-ahead” fashion
Post Conditions		The user gets a list of packages consistent with his search terms
Exceptions and extensions		*3a. The list is empty if no packages are found*
Priority	**Importance**	Conditional
	Architecture	Low
	Effort	Medium
Status		Proposed
Test Cases		

Table 11 - UC7 - Check Log file

ID	UC7
Name	Check Log file
Summary	The user wants to check the log file for data from previous installations
Actors	Integrator Data Warehouse Administrator (IDWA), Bioinformatics Researcher, Application Engineer
Preconditions	The user is logged in and has been presented with the BDM

		main screen
Basic Sequence		1. The user selects the ‘view’ drop-down menu and selects ‘View Log file’. 2. The user is presented with the log file 3. The user closes the log file
Post Conditions		The user has checked the log file and returned to the main screen
Exceptions and extensions		*1a. The log file can not be found* 1. The user is presented with an ‘not found’ error
Priority	**Importance**	Essential
	Architecture	Medium
	Effort	Medium
Status		Proposed
Test Cases		

Table 12 – UC8 - Remove Packages

ID		UC8
Name		Remove Packages
Summary		The user wants to remove a specific packages or set of packages from the Integrator system
Actors		Integrator Data Warehouse Administrator (IDWA), Bioinformatics Researcher, Application Engineer
Preconditions		
Basic Sequence		
Post Conditions		
Exceptions and extensions		
Priority	**Importance**	Conditional
	Architecture	High
	Effort	High
Status		Proposed, scope inclusion still open
Test Cases		
Open Issues		*Should this Use Case be included in the scope of the system?*

Table 13 – UC9 - Update Packages

ID	UC9
Name	Update Packages
Summary	The user wants to update a specific packages or set of packages
Actors	Integrator Data Warehouse Administrator (IDWA), Bioinformatics Researcher, Application Engineer
Preconditions	

Basic Sequence		
Post Conditions		
Exceptions and extensions		
Priority	**Importance**	Conditional
	Architecture	High
	Effort	High
Status		Proposed, scope inclusion still open
Test Cases		
Open Issues		*Should this Use Case be included in the scope of the system?*

6.4 System functionality and error situations

This chapter presents the most important functionality from the point of view of the system in the form of an extensive list of error situations and problems that might arise in combination with this. It also acts as a means of documenting what kind of errors to show the user, and is a natural extension to the use cases to be used for determining what error to present. The data is presented below in Table 14.

Table 14 - System failure and error situations

ID	Error	Description and user presentation
E1	Unable to connect to installation source provided by installer.	An installation source that has been added earlier is no longer available. The installer prompts: *"Installation source connection failed"*
E2	Interrupted connection to installation source	An installation source that has been added earlier fails during a transaction. The installer prompts: *"Installation source connection failed"*
E3	Missing data package	The installer has read the supplied metadata from the source and expected a data package to be found, but it is missing. It prompts: *"Error in package information: Package not found from installation source"*
E4	Corrupted data package	A corrupted data package can have serious consequences. If the corruption is to be detected by the installer, a sort of checksum would have to be implemented. In the case of an error like this, the source package will have to be disabled and the system prompts: *"There was an error during the installation of package X"* If the corruption takes place during the installation phase of the package, so that the corruption resides in the actual database data, the error will be caught by Medicels DB

		parser. There are a couple of reasons for this invalid content, being DB key constraints not valid or DB dump record datatype not consistent with database datatype. These errors come from the "populate framework" provided by Medicel, and the installer needs to present these errors to the user. The transaction rollback is also designed by Medicel. Packages are installed in dependency order, meaning that the package without dependency (leaf in a tree) is installed first. They are installed a package at a time and if it fails the installation is aborted, but the successfully installed packages remain in the system. In the case of a corruption error in the data, the system prompts: *"There was an error during the installation of package X"*
E5	Missing dependency package specification	The identification of this error will be very hard, the problem being that the system can no know if a dependency is missing or there simply doesn't exist one. This remains an open issue.
E6	Incorrect metadata	Incorrect metadata, except for incorrect dependencies, can lead to eternal loops etc. The installer needs to catch these kinds of inconsistencies and report it to the user. The system prompts: *"Corrupted data in installation source, packages disabled"* Furthermore, a means of communication this error to the administrator of the data source needs to be done.
E7	Missing database (local Integrator/remote installation source)	A database connection cannot be made, the system prompts: *"Local/Remote database connection failure"*
E8	Database failure (local Integrator/remote installation source)	A database connection fails, the system prompts: *"Local/Remote database connection failure"*

7. Non-functional requirements

The non-functional requirements are listed in Table 15 below. These requirements are collected based on the user interviews, and upon the main guidelines of usability, performance, reliability, security, and safety. The requirements at this stage can change during the course of the project. Before the requirements can be taken into the development process, they have to be approved by the customer. An appropriate indicator also has to be applied during the next iteration.

The priority of a requirement can be mandatory, essential or conditional. Impact and effort is based on a high, medium, low scale. Requirements with mandatory priority must be implemented in order to the project be successful. Requirements with essential priority are very important for the project. Conditional requirements create extra value for the customer. The status of a requirement can be proposed, approved, implemented, verified or deleted.

Table 15 - Non-functional requirements

ID	**Requirement**	**Prioritization viewpoints**			**Status**	**Related test case**
		Customer importance	**Architect-ure impact**	**Develop-ment effort**		
Usability						
NFR01	GUI has to be implemented according to Medicels Look & Feel guidelines	Mandatory			Proposed	
NFR02	GUI has to be easy to use (measured with usability test with the real user)	Mandatory			Proposed	
NFR03	The required help has to be provided to user in every situation	Mandatory			Proposed	
Performance						
NFR04	The system has to have a conflict resolve time of less than 10 seconds to add one package	Essential			Proposed	
NFR05	The system must handle a package amount of at least 1000	Essential			Proposed	
NFR06	The system must handle a package depth of at least 10	Essential			Proposed	
Scalability						
NFR07	The system has to have a low degree of dependency between the client and remote installation source	Conditional			Proposed	
Security						
NFR08	The system needs to use the authentication methods from the Integrator system	Mandatory			Proposed	
NFR09	The system needs to use a secure communication on all traffic	Mandatory			Proposed	
NFR10	The system needs to log all traffic both on server and client side	Essential			Proposed	
Reliability						
NFR11	The system needs to be fault tolerant by providing transaction rollback	Mandatory			Proposed	
NFR12	The system needs to provide data integrity when copying files	Essential			Proposed	

NFR13	The system needs to handle system failures without risk of data corruption	Mandatory			Proposed	
Maintainability						
NFR14	The system needs to take into account the inclusion of many different installation sessions in a later development stage	Essential			Proposed	
NFR15	The software needs to following the given coding conventions and documentation guidelines	Essential			Proposed	

8. Constraints

In this chapter, a list of hardware and software constraints, as well as standards used, is presented. It allows for a quick check of what constraints the program has on its users and environment.

Table 16 – Constraints

ID	Constraint	Related test cases
C1	The user needs to have configured the Medicel Integrator system	
C2	The user needs the Java Runtime Edition 1.5	
C3	The user needs to have an internet connection or a local media (DVD) drive	
C4	The user needs to have a database connection on the local machine to the local Integrator database	