Bern University of Applied Sciences

Project 1 - ClockAlarm

Requirements (version 0.1.0)

Loïc Charrière, Samuel Gauthier

supervised by
Claude Fuhrer

March 15, 2017

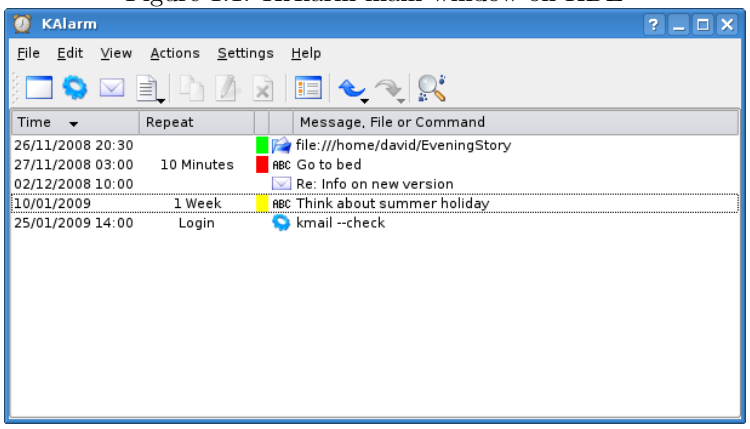
Contents

Chapter 1

Project Vision

The ClockAlarm project is an application allowing the user to manage alarms which will alert him at specified times. The user can create new alarms and assign them to categories in order to stay organised. The project is thought to replace in the long run an existing application named KAlarm.

Figure 1.1: KAlarm main window on KDE



Chapter 2

Project Goals

2.1 Goals

- Improve the overall life organisation of the user
- No more missed events

2.2 Requirements

- The application has to be cross platform (Windows, Linux, macOS)
- The database must not be stored in a binary format
- The configuration has to be easily transferable to another computer
- Customisable alerts
- Classifiable alerts
- Recurrent alerts
- Alerts can be scheduled
- Delay alerts
- Snooze alerts

Chapter 3

System and Context Boundaries

3.1 Stakeholder

1. kAlarm users

   Users who previously used kAlarm on linux, and wishing to find a similar but remastered and cross-platform software.

2. Computer users

   Those who tend to forget many things.

3. Investors

   Investors in the project. For example advertisers looking for visibility in the software.

4. The GNU Project

   The GNU Project collective could be pleased to see that an updated version of kAlarm is proposed.

3.2 Actors

1. Software User (Primary Actor)

   The peoples who will use the software. They are the first concerned by the product.

2. Administrator (Primary Actor)

   Super users with special authorizations. May be able to manipulate the data and the other users.
   It is likely that the software does not need an administrator.

3. System Time (Supporting Actor)

   The service specific to the OS and providing the exact international time.

4. Messaging Software (Supporting Actor)

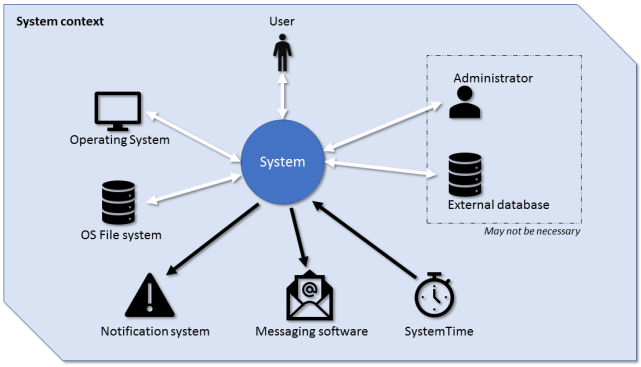
   Allows ClockAlarm to send Emails.

5. OS Notification Services (Supporting Actor)

   OS specific notification center or notification service.

3.3 System context

Figure 3.1: System context



3.4 System boundary

In addition to the elements above, special attention should be given to the following points.

- The must-have goal isn't to add functionality to the existing software kAlarm, but to create a similar and cross-platform product.
- The existence of a database and an administrator could be dropped and go out of context.
- The software isn't meant to be an alternative to an agenda. It behaves like a list of tasks and reminders.

3.5 Context boundary

In addition to the logical considerations and those stated above, the following points should be taken into account.

- The software is responsible for asking the messaging software to send to emails. This means that the way emails are sent isn't part of the environment.
- The software bases its alarms on the internal time. The accuracy of the time and of the time zone with respect to the geographical position of the machine is not part of the environment.

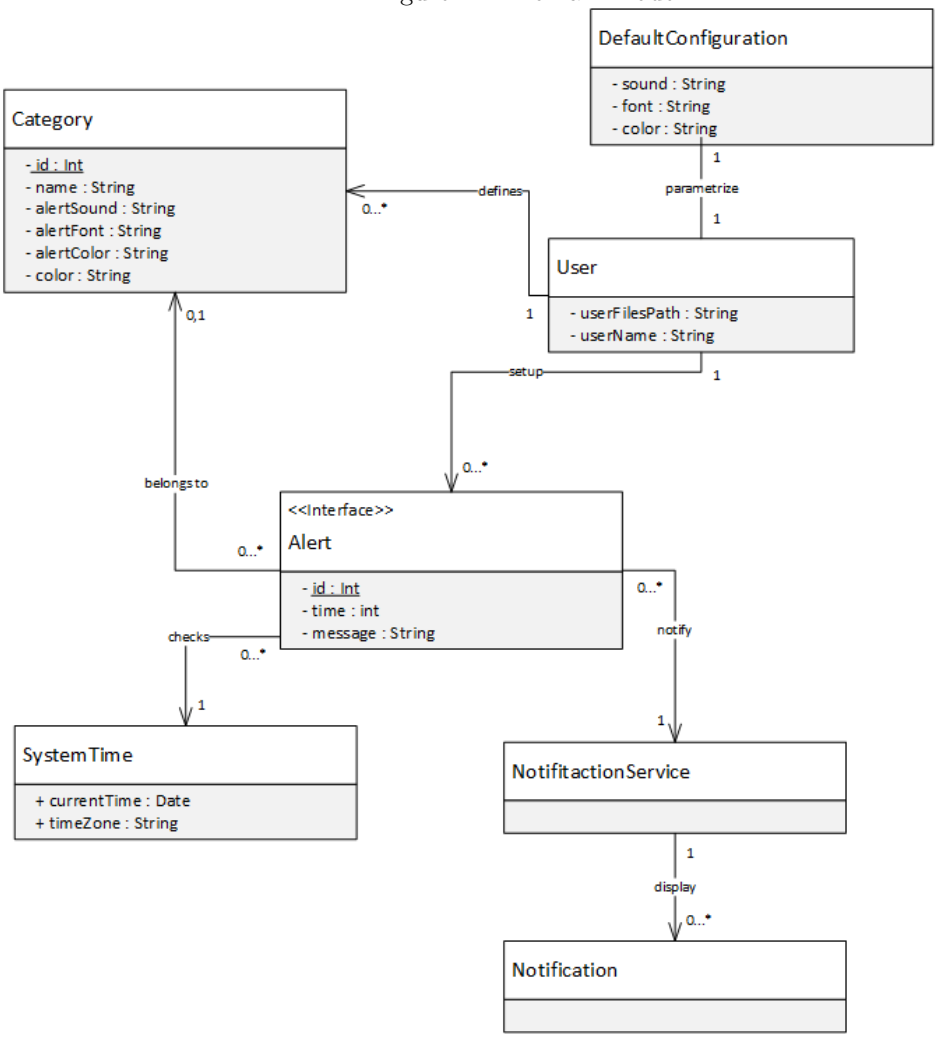
Chapter 4

System diagrams

4.1 Domain model

Conceptual model including system behavior and data.

Figure 4.1: Domain model



Chapter 5

System description

5.1 User Stories

As an [actor] I want [action] so that [achievement].

5.1.1 As a User

Alerts functionality

- i want to registrate some alerts so that the software warns me when they occure
- i want to be able to choose a diffente color, font and sound for every alert
- i want to edit the existing alerts when needed
- i want to delete the existing alerts when needed
- i want to postpone an alarm so that i'm notified again later (snooze)
- i want to mute an alert so that i'm not notified

Managing alerts

- i want to create categories of alerts, so that I can sort my alerts
- i want to edit my categories, in a way my alerts remain categorised
- i want to delete categories of alerts, so that my list of categories stay concise
- i want to assign color identity to my different categories, some that I can easily find them
- i want to personalize the default color, font and sound of my alarms in the software configurations
- i want to mute an alert catogory so that i'm not notified by any alert in this category

Persistence

- i want to retrieve the correct state of my alerts after turning my computer down and back on
- i want to be alerted at any times, as soon as I'm loged on my computer session

Exportability

- i want to export my alerts so that I can't import then on an other computer with ClockAlarm installed
- i want to import an alerts file, so that I can retrieve previously exported alerts
- i want to load configuration files, so that I can use predifined color and sound themes

Pivacy

- i want my alerts to remain private, so that nobody except me knows about them

5.2 Use Cases

5.2.1 Load Configuration

Load a ClockAlarm application configuration from an external file.

Scope

ClockAlarm application configuration phase.

Primary actor

User

Precondition

The application has to be started.

Postcondition

The application configuration is replaced with the one in the given file.

Main success scenario

1. The user selects a file to be loaded.
2. The application checks if the file is a valid configuration file.
3. The application replaces the configuration with the one specified in the file.
4. The application saves the new configuration.
5. The application notifies the user that the configuration has been loaded successfully.

Extension

2. The application detects that the file is invalid.
   - i The application notifies the user that the file is invalid.
   - ii The user chooses if he wants to select another file.

5.2.2 Launch ClockAlarm manager (no login)

In the case of an application without shared database and user accounts.

Scope

A computer running any common operating system (Windows, Mac OS, Linux).

Primary actor

User or Administrator

Precondition

The computer is on and the user is logged on his computer user session. The ClockAlarm application is correctly installed.

Postcondition

ClockAlarm is running and the manager window is open. The user is logged on his personnal ClockAlarm session.

Main success scenario

1. The user launches the ClockAlarm application.
2. The application starts and loads configurations and alarms from the user's personal files folder.
3. The manager window opens and displays the main window.

Extension

2 The application il allready running in background.

i Goto point 6.

2 Configurations or alerts can not be loaded (e.g. first use of the program).

i The application creates a new default configuration file and an empty alert file.

ii Goto point 2.

5.2.3 Launch ClockAlarm manager (with login)

In the case of an application with a shared database and user accounts.
This solution is unlikely to be kept.
For this reason, not all scenarios, especially those related to the connection with the server, will be treated here.

Scope

A computer running any common operating system (Windows, Mac OS, Linux).

Primary actor

User or Administrator

Precondition

The computer is on and the user is logged on his computer user session. The ClockAlarm application is correctly installed.
The user has a registered user account on the server.

Postcondition

ClockAlarm is running and the manager window is open. The user is logged on his personnal ClockAlarm session.

Main success scenario

1. The user launches the ClockAlarm application.
2. The application starts.
3. The user is asked to enter his credentials and the program tries to check these.
4. The application is connected to the server.
5. The application loads configurations and alarms from the server database.
6. The manager window opens and displays the main window.

Extension

2 The application il allready running in background.

i Goto point 6.

3 The credentials are incorrect.

i The connection to the server is denied.
ii The user is asked to give his credentials again.
iii Goto point 3.

3 The user isn't registered.

i The connection to the server is denied.
ii The user is redirected to a page to an account creation page.
iii The user creates a new account on the server.
iv Goto point 3.

5 Configurations or alerts can not be loaded (e.g. first use of the account).

i The application creates and upload a new default configuration file and an empty alert file.
ii Goto point 2.

Chapter 6

Development decisions

The structure of the development process of this project is based on the book 'Requirements Engineering Fundamentals'[2].

6.1 Word Processor

In order to write this document, we had to choose between several word processors. We required that the file could be used with a version control system so that the documentation and code could live in the same place. Also, we didn't want to loose time if there was a need to change the layout. After discussion, we came up with the following list:

1. Microsoft Word
2. Google Docs
3. LibreOffice
4. Scribus
5. TEX

The main problem with number 1 and 3 is that the output of these programs are binary files. A built in versioning functionality exits in Microsoft Word but it adds another version control layer to the workflow. The .docx and .odt file types are in fact ZIP archives containing XML documents which can be uncompressed. They could be stored in the uncompressed state to be compatible with the chosen version control system [3]. Or they could be converted into another format. Clearly this is not convenient. Therefore we decided that they were not compatible with our first requirement stated above.
Using a Google Doc means that the document will be stored on Google's servers and not within the folder containing the source code. I thus also violates our first requirement.
Scribus is a free alternative to InDesign and doesn't produce binary files. It allows a fine-grained control of the layout, frames and styles [4].
Finally there is TEX, a typesetting system created by Knuth. It provides a low-level language that is not directly used when writing documents. Instead there exist higher level formats such as LATEX or Plain TEX(much more lower level than LATEX) which provide a large set of macros [5].
Our supervisor strongly recommended us the use of LATEX, that is why we ended up using it.

6.2 Version Control

Out of the multitude of version control systems (VCSs) which exist, the three following are mostly used [6]:

1. Git
2. Mercurial
3. Apache Subversion

All the three offer equivalent features and are equally well suited for our project [1]. We decided to use Git with GitHub mainly because we were already familiar with them. GitHub allows us to easily manage issues and milestones (an equivalent service for mercurial is Bitbucket).

6.3 Python

As two teams are developing separately in parallel an alternative for KAlarm, we had the choice to use Java or Python. We took the latter because we wanted to learn a new programming language.

6.4 Unit Testing

6.5 Test Coverage

Chapter 7

Minutes

7.1 Meeting 1: March 1, 2017

Present at meeting

Samuel Gauthier, Loïc Charrière, Claude Fuhrer

Agenda

- Present the Use Cases and User stories
- Discussion about the usage of Python

Notes

Use Cases and User stories

- Vision&Goals: The vision and goals of the project have to be enhanced and not taken "as is".
- If a choice is made during the project, we should write down why it has been made this way. Everything that hasn't been documented is considered not been done.
- If we go for a specific product (such as DBMS, module, etc.) we should at least try out other alternatives (2-3) and explain why we made the choice of using this specific product.
- Split the user story "create/delete categories" in two.
- Sort the uses cases by topic.
- Alarms could be snoozed. (for example, if the user is at a meeting snooze all alarms so that they don't disturb the meeting)
- Alarms should be user specific, i.e., they should belong to the user connected on the computer and not be shared across all the users of the machine.
- Book recommendation (chapter about use cases) -¿ Applying UML and Patterns (Larman).

Usage of Python

- Strongly discouraged to share python code via e-mail (General life advice)
- Code coverage 70-80% is ok, 30% is not

Tasks

- Improve vision&goals
- Create use cases / user stories / actors correctly
- Read Python doc

Next Meeting

~~TBD~~(edit) March 1, 2017

Glossary

LaTeX Is a mark up language specially suited for scientific documents. 12

TeX Is a typesetting system developed by Donald Knuth. 12

Apache Subversion Is a software versioning and revision control system developed by the The Scribus Team. 13

Bitbucket Is a web-based Git and Mercurial repository and Internet hosting service owned by Atlassian. 13

Git Is a software versioning and revision control system originally created by Linus Torvalds and now developed by Junio Hamano. 13

GitHub Is a web-based Git repository and Internet hosting service. 13

Google Docs Is an online word processor created and developed by Google. 12

InDesign Is a desktop publishing software developed by Adobe Systems Incorporated. 12

KAlarm Is a personal alarm scheduler developed by David Jarvie. 3, 13

LibreOffice Is an open source office suite created and developed by The Document Foundation. 12

Mercurial Is a software versioning and revision control system developed by Matt Mackall. 13

Microsoft Word Is a word processor created and developed by Microsoft. 12

Scribus Is a is a desktop publishing application developed by The Scribus Team. 12

VCS version control system. 12

Bibliography

[1] Patrick Thomson. Git vs. Mercurial: Please Relax. Aug. 2008. URL: https://importantshock.wordpress.com/2008/08/07/git-vs-mercurial/ (visited on 03/12/2017).

[2] Klaus Pohl and Chris Rupp. Requirements Engineering Fundamentals: A Study Guide for the Certified Profes English. 1 edition. Santa Barbara, CA : Sebastopol, CA: Rocky Nook, Apr. 2011. ISBN: 978-1-933952-81-9.

[3] mpm. ZipdocExtension - Mercurial. Nov. 2012. URL: https://www.mercurial-scm.org/wiki/ZipdocExtension (visited on 03/12/2017).

[4] William von Hagen. Open source desktop publishing with Scribus. Apr. 2013. URL: http://www.ibm.com/developerworks/library/os-scribus/index.html (visited on 03/12/2017).

[5] Levels of TeX - TeX Users Group. Jan. 2017. URL: http://tug.org/levels.html (visited on 03/12/2017).

[6] Version Control Systems Popularity in 2016. URL: https://rhodecode.com/insights/version-control-systems-2016 (visited on 03/12/2017).