IT University of Copenhagen

BDSA 2014

Calendar System Requirement Analysis Document

Assignment 37

Anders Wind Steffensen - awis@itu.dk
Christopher Blundell - mail@itu.dk
Pierre Mandas - mail@itu.dk

September 18, 2014

Contents

CHAPTER 1

Introduction

1.1 Purpose of the system

1.2 Scope of the system

1.3 Objectives and succes criteria of the project

1.4 Definitions, acronyms and abbrevations

1.5 References

1.6 Overview

CHAPTER 2

Current System

CHAPTER 3

Proposed System

3.1 Overview

3.2 Functional requirements

3.3 Nonfunctional requirements

Beneath this sentence is the first draft of the non functional requirements written. It does not include, operation, interface or legal, since these subjects were irrelevant.

Usability
User must know how to operate a computer and being familiar with the windows interface standard will help, but should not need any training to use the system. Documentation of how to use the program should be included an available in the program.

Reliability
Should the system crash, all userdata should still be available on next launch. The system should at worst loose the data concerning an event the user was creating during the crash. Force restarting should not be acceptable and most exceptions must be caught and handled runtime.

Performance
The system should be able to hold an arbitrary number of events, without slowing down. Furthermore most user actions must be near instant, and heawy work should run in the background and thereby not disturbing the user.

Supportability
Updating by reinstalling is acceptable in this project, but the system should be easy to extent for a programmer.

Implementation
The testing will probably be a little scarce due to the small amount of time available to develop the system, but Key components must be tested thoroughly. The Calendar is a lightweight system, and stores data on the cloud, so there will not be any restraints on the hardware. Additionally, in it's current form the CalendarSystem is only runnable on Windows OS.

3.4 System model

Scenarios

Scenario: createCalendarEntry

---

actors: USER, DATASTORAGE

Flow:
1. USER gets invited to the most rad party of the year in the hall of his high school by his friend Alfred.
2. USER wishes to be reminded of this rad party on Friday at 3 pm., and decides to enter the EVENT into a CALENDARSYSTEM.
3. USER opens CALENDARSYSTEM, opens his Party-CALENDAR, selects Friday the 2nd. in the CALENDAR-OVERVIEW, and is prompted to enter information about TIME, DESCRIPTION (3pm., rad highschool party with alfred.
4. CALENDARSYSTEM stores the information to the PAC-ClOUD.

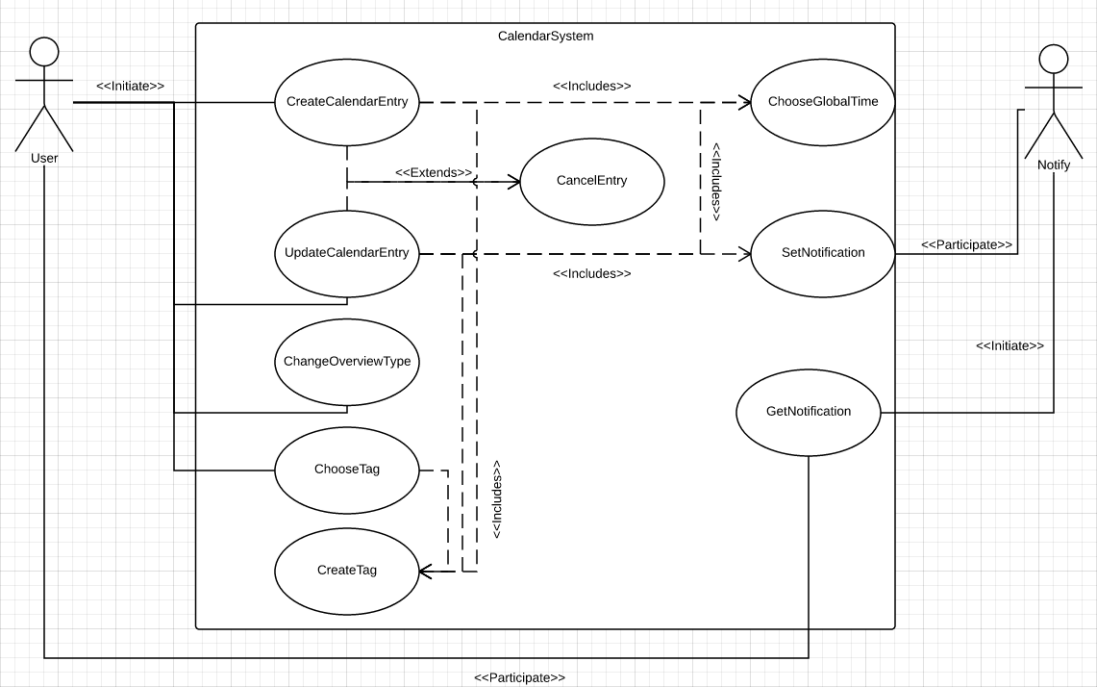
Scenario: updateCalendarEntry

Actors: USER, DATASTORAGE

Flow:
1. USER discovers that his rad party event was moved from Friday the 2nd. at 3pm to Saturday the 3rd at 7pm. due the host Alexis' parent being home on that Friday, because their cruise on the titanic had reported possible delay.
2. USER is currently logged on to the CALENDARSYSTEM. USER locates the event, and opens the EVENT on Friday the 2nd.
3. USER then changes the TIME and DATE to Saturday the 3rd. at 7pm, proceeds to save the EVENT, and the EVENT is saved to the PAC-CLOUD.

Use Case Models

User case diagram:



Here you can see a short description of the three user cases CreateTag, ChooseTag and GetNotifikation

User Case name: CreateTag

Participating Actors:
Initiated by User
Communicates with PAC-CLOUD data storage

Flow of events
User initiates the CreateTag
User chooses a name for the tag
- Exceptional case: User cancels the creation.
User accepts and finish

Entry Condition
The user has logged into the CALENDARSYSTEM

The user is in the tag drop down menu OR creating an Event or Updating an Event

---

EXIT Condition
The user has created a tag or cancelled the creation

User Case name: ChooseTag

Participating Actors:
Initiated by User
Communicates with PAC-CLOUD data storage

Flow of events
User opens the TAG drop down menu
- Exceptional case: CreateTag use case - user wants to create a new tag
- Exceptional case: user closes the dropdown menu
User chooses between his tags or <All>
The Calendar overview is updated with all events with the tag.

Entry Condition
The user has logged into the CALENDARSYSTEM

EXIT Condition
The user has picked a tag or choosen to keep the current tag

User Case name: GetNotifikation

Participating Actors:
Initiated by CALENDARSYSTEM
Communicates with PAC-CLOUD data storage
Prompted to the user

Flow of events

CALENDARSYSTEM creates a popup with a notifikation message of the event
User reads the notification

Entry Condition
The user has logged into the CALENDARSYSTEM
CALENDARSYSTEM has a notifikation from an event set at a specific time.

EXIT Condition
User closes the notification

Initial object analysis

The table describes the predicted classes and their purpose and relations. This will be elaborated on when the class diagram will be added in assignment 38.

Event
A collection of information such as DATE, TIME, DESCRIPTION, NOTIFICATION, TAG.

Calendar
Holds an abitrary number of EVENTS

User
A person who uses the system, who wants to get an overview of the events he/she has.

PAC-CLOUD
A storage unit who can send and retrieve CALENDARs for USERs

Notification
A message prompting the USER with information about an upcoming EVENT

Tag
An one-worded description of an EVENT - used to group simular kinds of events together.

Notifier
A collection of notifications that will be popped and shown to the user when a time threshold has been exceeded.

View
A visual representation of the CALENDAR to the USER

Dynamic Model

User Interface

CHAPTER 4

Glossary