KasetClock

SimpleSpecification V0.1

Uwe Gühl, Jittat Fakcharoenphol
Department of Computer Engineering
Faculty of Engineering
Kasetsart University, Bangkok, Thailand

Fall 2007/ 2008

List of Contents

1 Introduction

1.1 Purpose

Expected Result is a software, which offers the possibility to show multiple clocks.

1.2 Document Conventions

Have to be defined by the project.

1.3 Intended Audience and Reading Suggestions

This documentation is basic for the project and should be read by each participant student.

1.4 Project Scope

It should be possible to show to the user different clocks of different time zones at the same time. It should be possible to change the language easily, expected are at least English and Thai. Additionally German is requested.

1.5 References

An example is MultiClock 1.0.5, which has similar ideas realized. This and other clocks available on http://gosoftware.org/your-clocks_tag.
This document follows the Software Requirements Specification template by Karl E. Wiegers.

2 Overall Description

2.1 Product Perspective

KasetClock is a new product, which shows one to many clocks of different time zones similar.

2.2 Product Features

Important features is a comfortable easy-to-use configuration of the clock, e.g. different languages to use (at least English and Thai), the possible use of different clocks (analogue, digital) and nice signs for the time zones like flags for the countries.

2.3 User Classes and Characteristics

User: KasetClock should help e.g. students spending a semester abroad or people working in international projects to know the current time in different time zones of members of the project.
Developer: For further development it should be possible, to easy offer additional configuration items, like additional flags, or additional clocks.

2.4 Operating Environment

The software should work on PCs and Laptops using Microsoft Operation Systems like XP or Vista, and Linux, and on Workstations using UNIX-Derivatives.

2.5 Design and Implementation Constraints

The software should be easy downloadable and installable, and run as stand-alone-application.
Further requirement is to make it accessible via Internet on a specific web page.
It is recommended to use design pattern, e.g. the Observer Pattern.

2.6 User Documentation

A user manual and help should be available.

2.7 Assumptions and Dependencies

The application should work as a stand-alone-application without any side effects, which could disturb other applications. There should be no dependency to other programs.
The time displayed should depend on the internal clock or by a timer accessible via Internet. It is necessary that the most current time gets displayed. It must not be possible that no time is displayed , e.g. because of problems with an internet connection.

3 System Features

3.1 Installable / Uninstallable

3.2 Basic configuration

3.3 Edit clock

3.4 Add clock

3.5 Delete clock

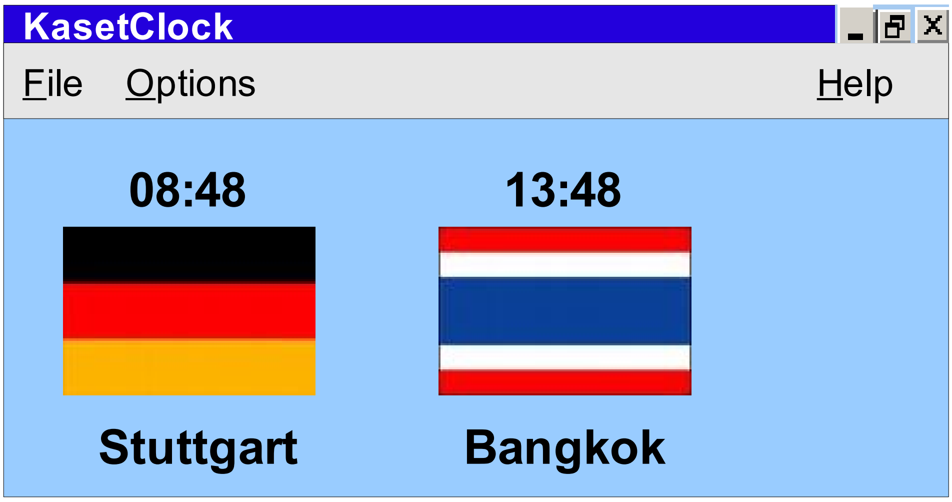
3.6 Help

3.7 (Information) request / Bug report

4 External Interface Requirements

4.1 User Interfaces

Suggestion:



4.2 Hardware Interfaces

Not defined.

4.3 Software Interfaces

Connection to system clock or a timer via internet. It must be asured, that the accurate time gets displayed.

4.4 Communications Interfaces

Http-Connection for

- automated or manual Software Updates
- statistic (number of users, location of users)
- requests, requirements of users
- bug reports

5 Nonfunctional Requirements

5.1 Performance Requirements

Maximum 1 second processing time of an user request.
No significant performance difference because of number of clocks used in the application.

5.2 Safety Requirements

No side effects, should not damage anything on a computer.

5.3 Security Requirements

Respect of the privacy of an user

5.4 Software Quality Attributes

Adaptability	it should be usable for other software projects
availability	
correctness	The time shown should be as exact as possible. Winter / Summertime (daylight saving time) should be considered
flexibility	New requirements should be easily realized
interoperability	
maintainability	Configuration should be done easily and repeatable, even after shut-down of the application
portability	
reliability	
reusability	The code should be programmed in the way, components could be reused by other applications
robustness	The application should not generate a core dump.
Testability	The application must be testable.
Usability	Easy to use instead of easy to learn.

6 Other Requirements

The software should be written as open source.

Appendix A: Glossary

KasetClock = An application showing different clocks at the same time as specified by a user.

Appendix B: Analysis Models

to be defined

Appendix C: Issues List

to be defined