# Project plan

DaCoPAn2

Helsinki, 22th February 2005

Software Engineering Project

UNIVERSITY OF HELSINKI
Department of Computer Science

**Course**

581260-4 Software Engineering Project (6 cr)

**Project Group**

Mikko Airaksinen
Tomi Korkki
Pauli Miettinen
Timo Tuominen
Mikko Väänänen

**Customer**

Markku Kojo

**Project Masters**

Juha Taina (Supervisor)
Marianne Korpela (Instructor)

**Homepage**

`http://www.cs.helsinki.fi/group/dacopan2`

**Change Log**

| Version | Date | Modifications |
|---|---|---|
| 1.00 | 9.02.2005 | The first published version. |
| 1.10 | 22.02.2005 | Final and corrected version. |

# Contents

# 1 Introduction

DaCoPAn2 is a software engingeering project at the University of Helsinki Department of Computer Science. It is a follow-up of the DaCoPAn project between the Computer Science Departments of the Universities of Helsinki and Petrozavodsk, which took place from January to May 2004.

DaCoPAn2 extends version 1.0 of the DaCoPAn system. DaCoPAn is a software developed for analyzing packet trace information captured from tcpdump log files at either end point participating in network communication, and animating the analyzed data for educational and research purposes. DaCoPAn is divided into two subsystems: the Analyzer produces XML based protocol event files from raw tcpdump data, and the Animator visualises the information found in these files using various methods.

DaCoPAn2 is an extension of the Animator subsystem. The project has two main goals:

1. Creating a new visualisation view showing a Time Sequence Chart of transmitted, received, re-transmitted, and lost packages.
2. Adding new features and making improvements to the existing user interface according to the wishes of the customer.

This project plan describes the organisation and schedule of the project, as well as states the working procedures and tools, resource requirements, and risks involved in the project. It also includes a rough size estimate of produced code.

The DaCoPAn2 project starts 27.1.2005 and ends 6.5.2005.

# 2 Organization

General contact information:

Home page:
`http://www.cs.helsinki.fi/group/dacopan2`

TWiki website:
`http://db.cs.helsinki.fi/~tkt_dac2/twiki`

CVS repository:
`cs.helsinki.fi: /home/group/dacopan2/`

Mailing list of the global team:
`ohtuk05-dacopan2-list@cs.helsinki.fi`

## 2.1 Participants

### 2.1.1 Supervisors

**Juha Taina** University Lecturer, PhD. Project supervisor.

**Marianne Korpela** Project instructor.

### 2.1.2 Customers

**Markku Kojo** Senior Researcher. Project customer.

### 2.1.3 Technical advisors

**Ilpo Järvinen** Advisor.

**DaCoPAn team**
Members of the original DaCoPAn Animator project team can be called upon for help and advise and will be informed when the project has successfully finished.

### 2.1.4 Project team

**Mikko Airaksinen** Webmaster.

**Tomi Korkki** Documentation manager.

**Pauli Miettinen** Customer contact person.

**Timo Tuominen** Project manager.

**Mikko Väänänen** Environment manager.

## 2.2 Explanation of the roles

The roles of the project team members are described below. Each role has a corresponding area of responsibility, which the holder is to supervise and to manage, although every task will be completed as a combined effort and every decision made together in a democratic fashion.

### 2.2.1 Project manager

This person will take a higher level responsibility for the flow and schedule of the project. He will strive to act as a chairman at any meetings where it is not otherwise predefined who will lead the discussion. He will try to keep meetings on track and act as an outside spokesperson wherever the project group needs one. Reporting project progress should be orchestrated by this person.

### 2.2.2 Customer contact person

The customer contact person has the main responsibility for fluid communication between the team and the customer. He will take care that the customer receives any questions the team might have in due time, and acknowledges the restrictions that the customer's strict schedule imposes on the communication.

### 2.2.3 Webmaster

This person is responsible for setting up the TWiki page and the team home page and updating the home page with e.g. all the latest documents.

### 2.2.4 Environment manager

This person installs and manages the development environment, making sure that CVS tree and Apache Ant configuration files are properly structured and up-to-date.

### 2.2.5 Documentation manager

The documentation manager is resposible for combining the final documents from different authors' contributed parts, and seeing that the documents conform to the LaTeX templates. His task is also to make sure that the documents are finished in time.

# 3 Working procedures, monitoring and reporting

The procedures used in the day-to-day operation of the project are described in this section to make it as easy and straightforward as possible for the group members to contribute to the project.

## 3.1 Communication

There are five different communication channels used during the process: email, the group mailing list, IRC, the TWiki web system, and meetings. In special time critical cases, telephone can also be used.

### 3.1.1 Group mailing list

The address of the group mailing list is `ohtuk05-dacopan2-list@cs.helsinki.fi`. Mail sent there is delivered to each team member and the instructor. The mailing list is most convenient for quickly informing other group members, but it should not be used for longer discussions.

In all email messages it should be clearly stated in the first text line how soon and from whom a reply is expected. If this is not stated, it is supposed that the message is not time critical and everyone can use their common sense to decide whether to reply to the mail or not.

The group mailing list is also the primary communication medium between the customer and the team, and for reaching the technical advisors. All email sent to them has to be echoed to the group mailing list.

### 3.1.2 Email

Individual team members can send each other email messages when the subject in question is of such a specific nature that the attention of the whole group is not needed. Email should not be used for long discussion threads, for which the discussion sections of TWiki are intended.

### 3.1.3 TWiki

In addition to collaborative team work on different products of the project, the group TWiki pages are used for lenghtier discussions. The TWiki site contains subsections that can be used as discussion forums.

### 3.1.4 IRC

The group has an Internet Relay Chat channel located in IRCnet. The channel name is `#dacopan2`. IRC should be regarded only as a complementary means of communication in such cases where two or more team members are simultaneously working on the same problem and need advice from each other or otherwise require a quick, short time span communication method.

### 3.1.5 Meetings

The whole group has agreed upon two regular meeting times per week, each meeting lasting 2 hours. The meetings take place every Wednesday and Friday at 8.30, Wednesdays in room CK109 and Fridays in room CK108 of the Exactum building. These times and places apply only to the first period.

The role of these formal meetings vary between design meetings, project status monitoring, and reviews. A secretary is assigned for each meeting, and his main task is to write a minute which is later published in the team home page.

There are several types of reviews. Of these, a formal technical review is most strictly moderated, and one will be held for the finished design document according to the guidelines described in section 3.3.4. Additionally, an informal technical review will

be held for the requirements document. For the other documents, reviews will not follow any strict predefined procedures.

The function of monitoring meetings is to verify the status of the project and make sure everything is on schedule. The project manager will send invitations to the monitoring meetings the day before each meeting at the latest. Every member of the group has to make sure he has updated his progress report according to the instructions set out in section 3.3.6 in time for the meeting.

Additional informal design meetings between group members should also be used for more detailed work on the project. Although no minutes are required from these meetings, important design decisions emerging there should always be documented and/or brought to the attention of the entire group by means of some other communication method.

### 3.1.6 Communication with the customer

Communication with customer is handled using email and is echoed to the group mailing list. It is the prime responsibility of the customer contact person. When e-mail is used, it is important that the information from the customer reaches the whole group. Notifying other group members is the responsibility of the person communicating with Mr. Kojo. If no reply is received from the customer after two or more days, the author of the message can poll by sending another message asking for reply.

The most important thing in communicating with the customer is that he does not have to discuss the same things many times with different people. Thus, it is important that the whole group is informed on discussion that has taken place between Mr. Kojo and some group members.

These procedures also apply to communication with Mr. Järvinen and the DaCoPAn team. Mr. Järvinen is possibly also available in his office at room C210 depending on the time of day.

## 3.2 Cooperative work using CVS

The collaborative work on documents and program code is achieved using a CVS repository, in which documentation and programming language code is stored in text format. Each member of the project should check the repository every time before starting working on the project and update their work to it as often as it is reasonably possible.

Any project member can make changes to the documents created by other members but if the changes are somewhat bigger than just changing a word, and it is not clear that the original author would agree on the changes, he should be contacted by e-mail prior to making the change.

The original version of the DaCoPAn software is stored in the repository in the

beginning of the project and changes are made to it by the team during the process.

The repository is organized in the following way:

- documents: The documents created in the course of the DaCoPAn2 project
- animator/main/fi/helsinki/dacopan: The project source code of the Animator subsystem.
- Other directories also exist.

## 3.3 Monitoring and reporting mechanisms

An additional demand for the working procedures is that the supervisors and the customer should be kept up to date with the state of the project, and the status of the project should be kept track of internally to the group. This is achieved through different means of monitoring and reporting discussed in this section.

### 3.3.1 Monitoring

The communication between DaCoPAn2 group members should be conducted in a transparent way using the methods presented in the previous section. The instructor follows the group mailing list, TWiki, and web page, so monitoring the communication between group members is quite easy. Everything other than communication can be monitored through the web page by any person interested in the project.

Beside that, following actions have to be taken to make it possible for the supervisors and customer to get a complete picture of the project.

The project manager will also announce special project status monitoring meetings where the current status of the project is compared to the schedule and the realisation of risks is evaluated.

- Minutes of the meetings must be taken for every formal meeting using the template available on the home page.
- During the course of the project several documents have to be published on the project web page. In addition to that the instructor can request the latest versions of the documents directly from the documentation manager. The documents are to be published in both PS and PDF formats. The documents published during the project are:
  - Project plan (this document)
  - Requirements specification
  - Design document
  - Test plan

- Implementation document
- Well commented Java source code
- Test execution document
- Conclusion
- User manual

### 3.3.2 The group home page

The group home page is used mainly for storing finished versions of the documents created in the project and to present the project to visitors from outside. The address of the home page is `http://www.cs.helsinki.fi/group/dacopan2`.

### 3.3.3 The TWiki web system

The TWiki web system `http://db.cs.helsinki.fi/~tkt_dac2/twiki` is used for collaborative team work and lengthier discussions which should not be made via email.

In addition to the main purpose described above the TWiki also serves as a means of sharing useful project related information between the group members by adding links or small documents to the system. Every section in TWiki can be edited and updated by any group member.

### 3.3.4 Formal technical reviews

Formal technical reviews are an important part of well-organized quality assurance as they provide a means to spot errors in an early phase of development. It is well known that a good review meeting is a much more effective tool for finding errors than even the best testing can ever be.

A formal technical review concentrates on one piece of programming code or documentation that is discussed based on preparations done in advance by attendants. Each member attending a review should thoroughly read through the product to be reviewed and write down their comments. A checklist can be used to help finding errors. Careful preparation assures that the meeting will not be longer than two hours, and the meeting stays effective.

In a formal technical review, the person responsible for writing the documents to be reviewed presents them and then the review leader leads the conversation by walking through the documents. Each participant then presents his notes at the appropriate time.

It is important to note that in a review the point is not to find solutions for problems but just to spot errors. The producer probably will be able to fix them later by himself. Another important thing is that debate should be kept to the minimum

and if such issue raises that the attendants cannot agree on, it should be left open for further discussion or the producer to decide.

In a formal technical review notes containing the following information should be taken:

- What was reviewed?
- Who reviewed it?
- What were the findings and conclusions?

At the end of the review the reviewers decide whether the reviewed product should be (1) accepted without further modifications, (2) rejected due to severe errors — once the errors have been fixed a new review must be performed — or (3) accepted provisionally, so that when the errors have been fixed no new review is needed.

### 3.3.5 Other reviews

In addition to formal technical reviews, there will be informal reviews and reviews.

An informal review will be held for the finished requirements document. The main difference to a formal technical review is that solutions to problems or errors can be brought up in the review itself.

All other documents will also be checked in reviews. These reviews are less moderated, and will follow no strict predefined procedures.

### 3.3.6 Reporting

The project group should report to its instructor by sending the following reports:

- **Progress report** Every member collects a report of his working hours, and stores it in the CVS repository, under the directory `documents/time_reports`, in a text file following the format declared at `http://www.cs.helsinki.fi/group/ohtu/resurssit/report/`. The project manager collects summaries of the individual reports and reports them to the instructor and the whole group.

  The task codes used in the reports are described below:

  ME - Official MEeting of the whole group

  LC - Local Conversation other than meeting

  PP - Project Planning

  DE - DEsign, writing of design document

  DO - DOcumentation of code or writing of user manual

CO - COding, including debugging

TE - TEsting (unit testings, functional testing, final testing)

AD - ADministrative tasks, e.g. updating progress reports

TM - Technical maintenance, e.g. maintaining web page, CVS

RE - REquirements engineering

CE - Code Exploring, studying the original software

GE - GEneral, other small tasks not mentioned above

- **Conclusion** An analysis of the work completed and a self-evaluation document must be written at end of the project.

# 4 Resource requirements

This section describes the general requirements of the project.

## 4.1 File system

A group directory on the Computer Science department's file system is required as well as an account on the db server for the TWiki system.

## 4.2 PEF files

DaCoPAn2 uses the PEF files provided by the original DaCoPAn project. In case the Time Sequence Chart visualisation mode requires information not provided by the existing Analyzer substem and the current version of the PEF file format, individual elements will be added to the XML- based PEF files by hand.

## 4.3 Development and testing tools

Common tools for modelling must also be available, as well as appropriate tools for software development and testing.

## 4.4 Working environment

Individual group members can set up their own working environments at home, or use the Computer Science Linux or Windows environments. As the animator is written in Java, it is portable to different systems.

# 5 Size estimate

To estimate the size of the produced software in Lines Of Code, the software is decomposed into parts and the sizes of individual parts are estimated.

In addition, the actual size of the DaCoPAn Animator 1.0 has been measured to help estimate the size of new features. The total size of the Animator is approximately 8600 LOC, and the new features to be added are estimated to require well under half that amount of written code.

Since the major part of the project consists of updating and maintaining existing program code, the LOC measure in this estimate refers to modified and added lines of code.

The relative effort percentages listed in Table 1 takes into account that DaCoPAn2 is a follow-up project. Although the new Time Sequence Chart animation type will have to be designed from scratch, all the other activities require a significant amount of code exploring. This lowers the relative effort of the TSC part.

- **Animation type: TSC** means the Time Sequence Chart animation. This includes all the code that is needed to visually present animation data as a time sequence chart animation. The animation needs to be configurable for presenting different header fields and variables using e.g. pop-up labels or windows. The animation code also needs to provide accessors for controlling the animation (play, pause, step back, step forward etc.).
- **XML I/O** Some minor additions might be required to the XML I/O module in order to enable the TSC animation type to display relevant information regarding package re-send time-outs.
- **Data structures** In order to accommodate the aforementioned new package information fields, some small additions to the data structures might have to be made.
- **Animation type: MSC** means the Message Sequence Chart animation. There are several fixes and UI improvements requested by the client for this animation type.
- **UI fixes and new features** means all the user interface improvements not applicable to any other component.
- **Managing settings** means internal representation of animator settings, including writing them on file in XML format and reading them from file. The internal animator settings including the header fields chosen for display should be handled in a more systematic way and a default settings file will have to be implemented.
- **Animation type: Enc** means the animation for packet encapsulation. Some minor presentation and layout improvements might have to be applied.

### 5.1 Integral estimate

Table 1 contains results of size estimates for the animator, and an integral estimate in LOC.

| Component | Programming language | LOC | Effort |
|---|---|---|---|
| XML I/O | Java, XML | $< 100$ | 5% |
| Data structures | Java | $< 100$ | 5% |
| Animation type: TSC | Java | $< 2000$ | 40% |
| Animation type: MSC | Java | $< 500$ | 20% |
| Animation type: Enc | Java | $< 100$ | 5% |
| User interface | Java | $< 400$ | 15% |
| Managing settings | Java | $< 300$ | 10% |
| **Total size estimate** | | $< 3500$ | |

Table 1: Estimated size of code and relative effort estimates for DaCoPAn system

## 6 Schedule

This section contains the project schedule.

During the project the following artifacts are produced and delivered to the customer: Project plan, Requirements document, User manual and DaCoPAn2 system (source code and executable program installed and ready to use).

The schedule is shown in Table 2. The corresponded GANTT diagram is shown in Figure 1.

## 7 Risk analysis

This section contains a breakdown of unforseen difficulties. Each risk that the project might face is documented, so it can be recognized at review meetings. There is an action plan of what to do if the risk materializes, and an estimate of the probability of the actualization of the risk. These probabilities can be graded as high, medium and low.

| | |
|---|---|
| **Risk** | Short description of the risk |
| **Probability** | Low - Medium - High |
| **Severity** | Low - Medium - High |
| **Minimizing risk** | Steps to minimize the probability of the risk |
| **Recognition** | What are the signs of the risk materializing? |
| **If materializes** | What to do if the risk materializes, ”damage control” |

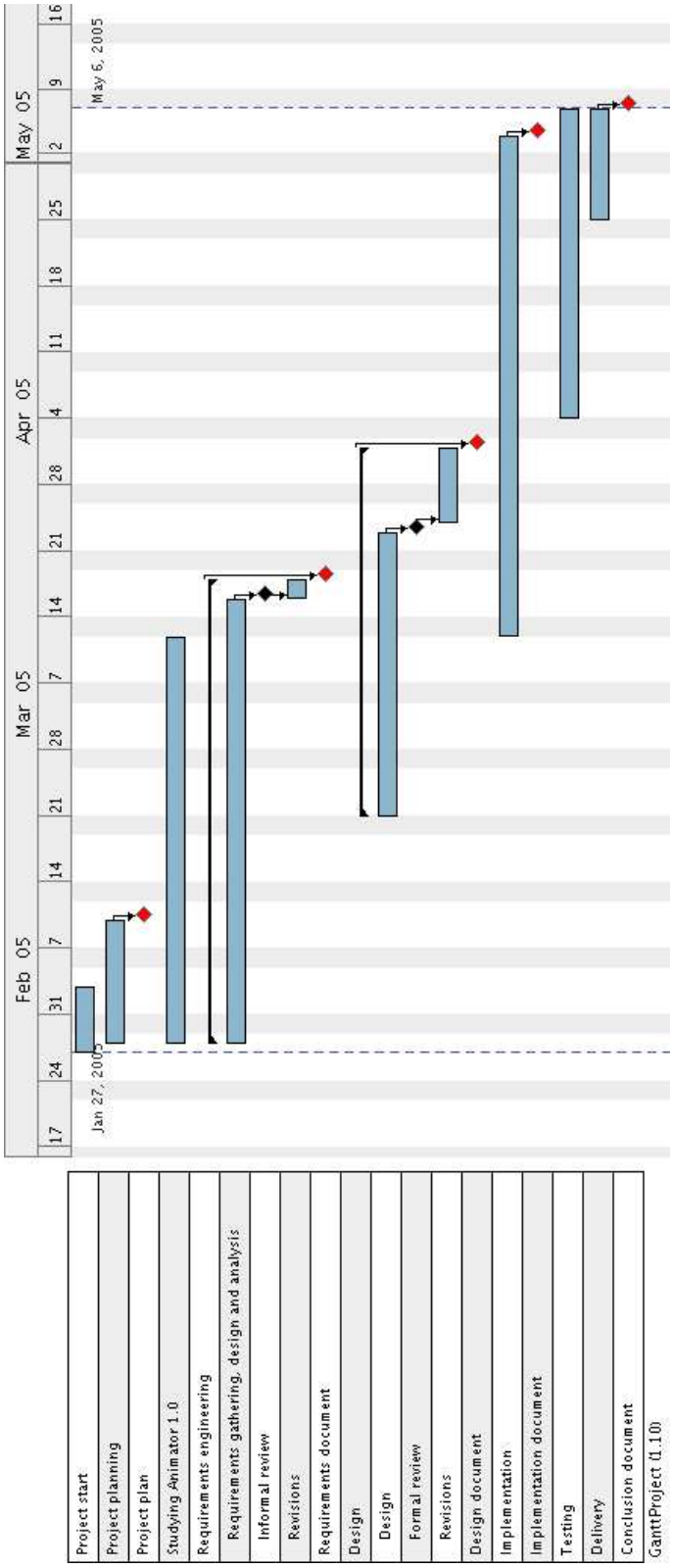


Figure 1: GANTT diagram

| Phase | Task | Date |
|---|---|---|
| Start | Project starts | 27.1.2005 |
| | Project planning begins | 28.1.2005 |
| | Studying of Animator 1.0 begins | 28.1.2005 |
| Requirements engineering | Requirements engineering begins | 28.1.2005 |
| | Project plan finished | 10.2.2005 |
| | Project plan review | 16.2.2005 |
| | First prototype review | 16.2.2005 |
| | Monitoring meeting | 18.2.2005 |
| | Second prototype review | 2.3.2005 |
| | Requirements document finished | 7.3.2005 |
| | Informal review of requirements document | 16.3.2005 |
| | Requirements document frozen | 18.3.2005 |
| Design | Design documentation begins | 21.2.2005 |
| | Monitoring meeting | 18.3.2005 |
| | Design document finished | 21.3.2005 |
| | Formal technical review of design document | 23.3.2005 |
| | Monitoring meeting | 1.4.2005 |
| | Design document frozen | 1.4.2005 |
| Implementation | Programming begins | 11.3.2005 |
| | Monitoring meeting | 18.3.2005 |
| | User manual documentation begins | 28.3.2005 |
| | Implementation documentation begins | 28.3.2005 |
| | Monitoring meeting | 1.4.2005 |
| | Monitoring meeting | 13.4.2005 |
| | Program code review | 15.4.2005 |
| | Monitoring meeting | 29.4.2005 |
| | Implementation document finished | 2.5.2005 |
| | User manual finished | 2.5.2005 |
| | User manual review | 4.5.2005 |
| | Implementation document review | 4.5.2005 |
| | Coding finished | 4.5.2005 |
| Testing | Testing begins | 4.4.2005 |
| | Test planning begins | 4.4.2005 |
| | Monitoring meeting | 13.4.2005 |
| | Test plan finished | 25.4.2005 |
| | Test plan review | 27.4.2005 |
| | Monitoring meeting | 29.4.2005 |
| | Test plan frozen | 29.4.2005 |
| | Test execution documentation begins | 29.4.2005 |
| | Test execution document finished | 2.5.2005 |
| | Test execution document review | 4.5.2005 |
| | Test execution document frozen | 6.5.2005 |
| Delivery of work | Writing of conclusion document begins | 29.4.2005 |
| | User manual frozen | 4.5.2005 |
| | Conclusion document frozen | 6.5.2005 |
| | Delivery of project material | 6.5.2005 |
| | Project ends | 6.5.2005 |

Table 2: Project schedule

| | |
|---|---|
| **Risk** | The schedule of the project isn't met |
| **Probability** | Medium |
| **Severity** | Medium |
| **Minimizing risk** | The project manager should monitor the progress of the project group and react if the project is lagging behind. Each member of the group should keep track of his own progress, and report to the project manager should his task(s) prove to be more time-consuming than was expected. Work should be distributed so that approximately the same amount of time is spent on the project per week. |
| **Recognition** | Milestones / checkpoints are delayed |
| **If materializes** | Rework the schedule so that the project can be finished on time, for example some parts of the project could be left out in order to deliver the product on time. |

| | |
|---|---|
| **Risk** | The original software proves hard to extend and maintain |
| **Probability** | Medium |
| **Severity** | High |
| **Minimizing risk** | Communication with members of the original DaCoPAn group could help to make clear hard to understand design decisions in the original software. |
| **Recognition** | The studying phase of Animator 1.0 proves difficult or is delayed. |
| **If materializes** | Concentrate work effort on those extensions and improvements which are easily implemented on top of the existing software. |

| | |
|---|---|
| **Risk** | Incorrect distribution of development work |
| **Probability** | Low |
| **Severity** | Medium |
| **Minimizing risk** | Careful planning in requirement phase and in design phase |
| **Recognition** | If work load is not balanced |
| **If materializes** | Some tasks can be reassigned and communication increased |

| | |
|---|---|
| **Risk** | Members get sick or suffer other temporary incapacitation |
| **Probability** | Medium |
| **Severity** | Low |
| **Minimizing risk** | - |
| **Recognition** | The group has to be aware of each member's work, so that the redistribution of tasks becomes straightforward. Individual members are to inform the group if they get sick or otherwise are unable to carry on with their tasks. |
| **If materializes** | Some tasks can be reassigned and workload redistributed |

| | |
|---|---|
| **Risk** | Finished product doesn't meet customer's requirements |
| **Probability** | Low |
| **Severity** | High |
| **Minimizing risk** | Communicating with the customer efficiently during the requirements phase, also keeping close contact during the subsequent phases. Prototypes of the software could be shown to the customer before the product is finished in order to notice any inconsistencies with the requirements before the product is finished |
| **Recognition** | Customer is not satisfied with the final product |
| **If materializes** | If the risk materializes after the product has been finished, the chances of making any changes to it are minimal, since this project has a tight schedule. However, if prototypes of the program are shown to the customer during development and it is clear that the program will not meet the original requirements, some adjustments to the design / implementation of the program could be made. |

# 8 SE techniques and CASE tools

## 8.1 SE techniques

The project will be carried out using the waterfall development model with an additional iterational model during the requirements engineering phase. The iterations are carried out with help of sketches and prototypes of the required additional animation types and UI improvements. This model is sufficiently simple, yet efficient for smaller projects. Another reason for this model is that it limits the customer's presence mainly to the beginning of the process. His future availability for intense communication remains uncertain.

## 8.2 CASE tools

Table 3 shows a summary of chosen CASE tools for the DaCoPAn2 project.

### 8.2.1 Documentation

Documentation is done using the LaTeX typesetting system. The group members can use any text editors they like. A script for compiling the documents to their final format is also available.

### 8.2.2 Design

In designing the product some UML and other diagram tools are needed.

### 8.2.3 Programming

A Java compiler and various development environments are used.

### 8.2.4 Testing

In unit testing JUnit and Cppunit can be used.

### 8.2.5 Support systems

Other tools used during the whole course of the project are CVS, TWiki web system, E-mail and other communication means.

| **CASE tool** | Communication | Planning | Documentation | Reporting | Modeling | Integration | Version management | Building | Prototyping | Method-support | Language-processing | Program analysis | Testing | Debugging |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Email | x | | | | | | | | | | | | | |
| IRC | x | | | | | | | | | | | | | |
| LaTeX | | | x | | | | | | | | | | | |
| CVS | | | | | | x | x | | | | | | | |
| Team Wiki | x | | | x | | | | | | | | | | |
| XFig, Dia | | | | | x | | | | | | | | | |
| Java | | | | | | | | | | | x | | | |
| Ant | | | x | | | | | x | | | | | x | |
| Progress report script | | | | x | | | | | | | | | | |
| Gantt project | | x | | | | | | | | | | | | |

Table 3: CASE tools selection and distribution