

Eiffel SDL multimedia library (ESDL)
DIPLOMA THESIS PROJECT PLAN

Till G. Bay

28th May 2003

Project Period: Monday, 2003-05-26 - Friday, 2003-09-26

Student: Till G. Bay (tillbay@student.ethz.ch)

Student-No: 98-803-000

Supervising Assistant: Karine Arnout

Supervising Professor: Bertrand Meyer

1 Project description

1.1 Overview

Support for building multimedia applications is mandatory for libraries accompanying modern programming languages. Therefore many programming languages provide not only standard libraries that allow a programmer to perform simple tasks like I/O, memory access or string handling, but they also ship with libraries that provide support for video, audio, CD-ROM, multi threading, networking, access to OpenGL[15] and access to input devices such as mice, keyboards and joysticks. Naturally a multimedia library is bigger than one of the mentioned standard libraries, since it tries to harbor all APIs for the various applications in one comprehensive package (more on this in section 3.1 on page 4).

The history of OpenGL[15] shows that the success and the widespread acceptance of a multimedia library depend on the portability of the applications created using it. Eiffel[1, 13] is lacking a platform independent multimedia library. This diploma thesis aims to fill this gap by building an object oriented multimedia library in Eiffel based on SDL[16] which itself constitutes a platform independent multimedia library written in C[9].

Summary of the multimedia library options available for Eiffel today and corresponding drawbacks:

- EiffelFURY[6]: a 2D and multimedia library
 - only for Windows
- EiffelOpenGL[5]: an Eiffel wrapper for OpenGL, GLU[7] and GLUT[8]
 - not multimedia, neither sound nor video support
- jegl[10]: a set of wrapper classes around SDL written in SmartEiffel[17]
 - is incomplete and not maintained anymore
- Eiffel_SDL[19]: multimedia library for Eiffel
 - under development

This listing shows that there have been and that there are attempts to providing a multimedia library for Eiffel but the limitations of those attempts also become clearly visible.

1.2 Scope of the work

The ultimate goal of the diploma thesis is the creation of a cross platform multimedia library for Eiffel. There are two key forms of multimedia library usability - interface and organizational¹. The interface usability dimension comprises the following four aspects:

¹The criteria mentioned here are partly inspired by the judging criteria mentioned on <http://www.eiffel-nice.org/eiffelstruggle/2003/judging.html>[2].

- **Learn-ability** indicates how fast a programmer can start using the library productively.
- **Memorability** indicates how easily a programmer can continue using the library after not using it for some period of time.
- **Efficiency** indicates the ability to use the library with a high level of productivity.
- **Errors** produced using the library should not be catastrophic, the number of user errors producible should be low, they should be documented and the system should be able to easily recover from them. The library would benefit from a concise logging facility for errors.

On the other hand the organizational dimension can be interpreted as the indicator of how effectively the library can be integrated into working practices of people and organizations using it. The three factors describing organizational usability are:

- **Portability** measures the ability of using the library productively in various environments transparently. In the case of a library for Eiffel portability among compilers is a second portability requirement which will not be respected in the scope of this thesis. I will focus on ISE EiffelStudio's compiler.
- **Compatibility** with other systems or libraries that interact with the library.
- **Extendibility** indicates the ability of the library being able to be extended to perform new functionality.
- **Installation** indicates how easy it is to get up and running with the library.

Even though both forms of library usability are equally important the diploma thesis will focus on the first form and provide solutions of the aspects of the second form where it is easily achievable. In other words the interface usability is the main goal of the diploma thesis and the audience in mind are in the first place programmers using the library to write multimedia applications, and not entire organisations embedding the library into their working practices. Organisational usability is in my opinion the second step on the way to a good library and it would require extensions that surpass the scope of this diploma thesis. To illustrate these thoughts one can for example take SDL's audio capabilities: To ensure organizational usability, more precisely to grant the second aspect of it - compatibility - it would be necessary to extend ESDL so it could handle different audio formats - e.g. the ever so popular ISO 11172-3 (mp3).

1.3 Intended results

The intended results of the diploma thesis are:

- A cross platform multimedia library for Eiffel

The multimedia library enabling programmers to build multimedia applications with Eiffel on any platform. The multimedia library (hereafter referred to as ESDL) aims at ultimately replacing all other alternatives available for Eiffel today. Therefore it has to be designed very carefully and its building process needs to include adjusting and reviewing.

- Ports of demos available on www.libsdl.org

The graphical demonstration programs (hereafter referred to as demos) illustrate the possibilities of the multimedia library. There are numerous demos available for SDL. Since ESDL is meant to wrap SDL in an object oriented way, it makes sense to port some interesting demos to ESDL to provide future library users with a starting point for their development and to illustrate the possibilities of ESDL.

2 Background material

2.1 Reading list

[14]Bertrand Meyer, "Object-oriented Software Construction", 2nd Edition, Prentice Hall, 1997.

3 Project management

3.1 Objectives and priorities

There are as we have seen two categories of objectives, the ones from section 1.2 on page 2 and the concrete ones from section 1.3. Therefore I will provide two separate priority listings here. Where necessary I am going to give explanations for my choice of priorities. The priorities range from one to three, where one is the highest and three the lowest. Objectives of priority three are beyond the scope of this diploma thesis.

3.1.1 Design objectives

Objective	Priority
Learn-ability	1
Memorability	1
Efficiency	1
Errors	1
Portability	3 ^a
Compatibility	1
Extendibility	1
Installation	2

^aPortability is inherent because we will be using SDL

Table 1: Design objectives and their priorities

3.1.2 Objectives for the intended results

Objective	Priority
API design	1 ^a
API completeness	2
Demos	1
API Documentation	1
Thesis Documentation	1
API Extensions	3

^aThis can also be regarded as the sum of the priorities from table 1

Table 2: Priorities of intended results

3.2 Criteria for success

While the ultimate goal of the project is the creation of a complete cross-platform multimedia library for Eiffel, the criteria for success is the quality of the software and the documentation handed in at the end. The result may be a partial implementation of the above objectives without implying any penalty on the success of the project. In this respect the quality of the software is measured according to the following points:

- use of Design by Contract
 - pre- and postconditions
 - class invariants
 - loop variants and invariants

- careful design
 - design patterns
 - extendibility
 - reuse-ability
 - careful abstraction
- core principles of OOSC2 [14]
 - command/query separation
 - simple interfaces
 - uniform access
 - information hiding
- style guidelines (from OOSC2 [14], and from Gobo[3])
- correct and robust code
 - test-suite available and passed
- readability of the source code

And the quality of the documentation is measured according to the following points:

- completeness
- understandability
- usefulness
- accessibility and structure

3.3 Method of work

The technologies involved are:

- Gobo Eiffel Test[4], used for implementing test cases.
- EWG[11] , used to wrap SDL[16]
- Eiffel[1, 13]

ESDL is developed according to the development plan in section 4 on page 8. The library will be developed on Linux and the developed test-suites are run on Linux and Windows to be able to detect possible portability problems. The documentation is written using the L_AT_EX[12] text-setting software and bibliographic references are handled by Bib-T_EX[18].

3.4 Quality management

The quality will be ensured by (in descending order of importance) :

- Close contact with the supervisor, recurring discussions about the design of ESDL
- The creation of relevant test-suites
- The documentation, see subsection [3.4.1](#)

3.4.1 Documentation

The documentation will be written throughout the whole project. Each of the project steps will be reflected as a chapter in the thesis report. ESDL itself will be documented on one hand in the source code and on the other hand in a chapter "ESDL Tutorial". The whole documentation will be comprised in the thesis report.

The thesis report will be exposed to constant review by the supervisor.

3.4.2 Validation steps

The validation steps are guidelines for the recurrent events that allow me to validate my progress along the project plan. If it is not possible to have a meeting with my supervisor on a weekly basis, there should be a meeting at least at the end of each project step (see below). The ending dates of the project steps form at the same time the partition of the project into milestones. Since the dates of those milestones are apparent in [figure 1](#) on [page 9](#), I omit listing the milestones separately.

- Weekly status report to supervising assistant, either via e-mail or personally
- Intermediate presentations of the work during group meetings (dates yet to be confirmed)
- Final presentation

4 Plan with milestones

4.1 Project steps

Below is a figure listing the different project steps that will lead to the development of ESDL. In the following I will present a short description and a grouping of the project steps:

- Evaluation

- Evaluate existing code wrappers for Eiffel (EWG[11], jegl[10])

The goal is to be able to determine the fitness of the existing wrapping tools in order to decide whether they will be used in the implementation of ESDL, or whether the wrapping of the underlying C library will be done manually.

- Evaluate existing multimedia libraries (EiffelFURY[6], Eiffel.SDL [19])

The existing multimedia libraries need to be examined, so the decision whether the design of ESDL should be influenced by their design can be made.

- Usability discussion

- Creation of a demo with the SDL library generated by the C wrapping tools for each subsection of the library

This will allow us to spot eventual flaws in the API.

- Discuss the found usability flaws

Determine what should be done better

- Redesign

- Design of the object oriented library
- Implementation of the object oriented library, in parallel with the implementation of the test-suites
- Porting of the demos to the object oriented version of the library

- Usability discussion II

- Second review of the usability of ESDL
- Second adjustment cycle of the usability flaws of ESDL
- Port of the demos to this final version of ESDL

- Application

- Creation of an example application using various capabilities of ESDL

Possibly create the final presentation of the diploma thesis as a multimedia application based on ESDL (self-hosting)

- Extension
 - Extend ESDL to support existing multimedia standards²

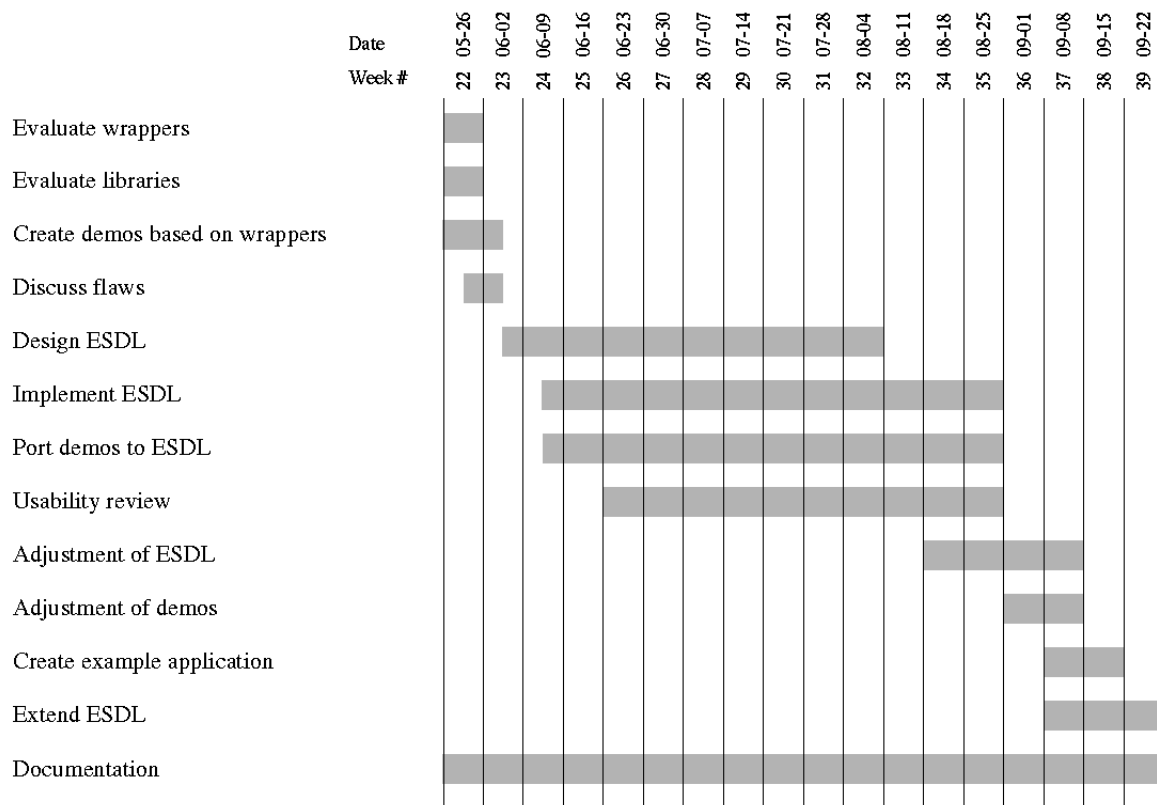


Figure 1: Project steps

4.2 Deadline

Project start: Monday, 2003-05-26

Project end: Friday, 2003-09-26

Total work time: 17 weeks = 85 days

4.3 Tentative schedule

See dates in subsection [4.1](#)

²The multimedia standards to be implemented are subject of discussion: A possible candidate would be the scalable vector graphics[20].

References

- [1] Eiffel, retrieved May 2003.
<http://www.eiffel.com>.
- [2] EiffelStruggle, retrieved May 2003.
eiffel-nice.org/eiffelstruggle/2003/judging.html.
- [3] Eric Bezault. GOBO, retrieved May 2003.
www.gobosoft.com.
- [4] Eric Bezault. Gobo Eiffel Test, retrieved May 2003.
<http://www.gobosoft.com/eiffel/gobo/getest>.
- [5] EiffelOpenGL, retrieved May 2003.
<http://eifogl.sourceforge.net>.
- [6] EiffelZone. EiffelFURY, retrieved May 2003.
<http://eiffelzone.sourceforge.net/eiffelfury/eiffelfury.html>.
- [7] GLU. OpenGL Utility Library, retrieved May 2003.
<ftp://ftp.sgi.com/opengl/doc/opengl1.2/glu1.3.pdf>.
- [8] GLUT. OpenGL Utility Toolkit, retrieved May 2003.
192.48.159.181/developers/documentation/glut/spec3/spec3.html.
- [9] American National Standards Institute. American National Standard for Information Systems#173;Programming Language C, 1989. X3.159-1989.
- [10] jegl. jan's eiffel game library, retrieved May 2003.
<http://jegl.sourceforge.net>.
- [11] Andreas Leitner. EWG - Eiffel Wrapper Generator, retrieved May 2003.
<http://ewg.sourceforge.net>.
- [12] LyX. The Document Processor, retrieved May 2003.
www.lyx.org.
- [13] Bertrand Meyer. *Eiffel The Language*. Prentice Hall PTR, 1992.
- [14] Bertrand Meyer. *Object-Oriented Software Construction, SECOND EDITION*. Prentice Hall PTR, 1997.
- [15] OpenGL, retrieved May 2003.
<http://www.opengl.org>.
- [16] SDL. Simple DirectMedia Layer, retrieved May 2003.
<http://www.libsdl.org>.
- [17] SmartEiffel, retrieved May 2003.
<http://smarteiffel.loria.fr>.
- [18] Bib TeX, retrieved May 2003.
www.ecst.csuchico.edu/~jacobsd/bib/formats/bibtex.html.
- [19] Steve Thompson. Eiffel.SDL (Colorado Eiffel User Group).
- [20] W3C. Scalable Vector Graphics (SVG), retrieved May 2003.
<http://www.w3.org/TR/SVG>.