Review-Dokument 4. Phase

released by
Alexander Rose, Christoph Braun, Jenja Dietrich,
Marc Brunner, Michael Röding, Sven-Leonhard Weiler, Vincenz Vogel

Inhaltsverzeichnis

Kapitel 1

Gesamtübersicht der einzelnen Phasen

Wir entschieden uns für unser Softwareprojekt Carduinodroid am 04.04.2013 und arbeiteten daran bis einschließlich dem 03.07.2013 um unsere Aufgaben fertigzustellen. Das besondere an unserem Softwareprojekt war, dass wir an einem Projekt weiterarbeiten sollten, welches bereits Teil einer Reihe von verschiedenen Projekten war . Unsere Aufgabe war also “nur” eine Erweiterung von vielen, was uns aber auch gefiehl, da soetwas in der Realität oft vorkommt.
Unsere Projekt dauerte also knapp 3 Monate und wurde in vier verschiedenen Phasen eingeteilt. In jeder Phase mussten wir bestimmte Teilaufgaben erledigen und zusätzlich eine Abschlusspräsentation halten, um den Projektfortschritt aufzuzeigen und wichtige Probleme anzusprechen.

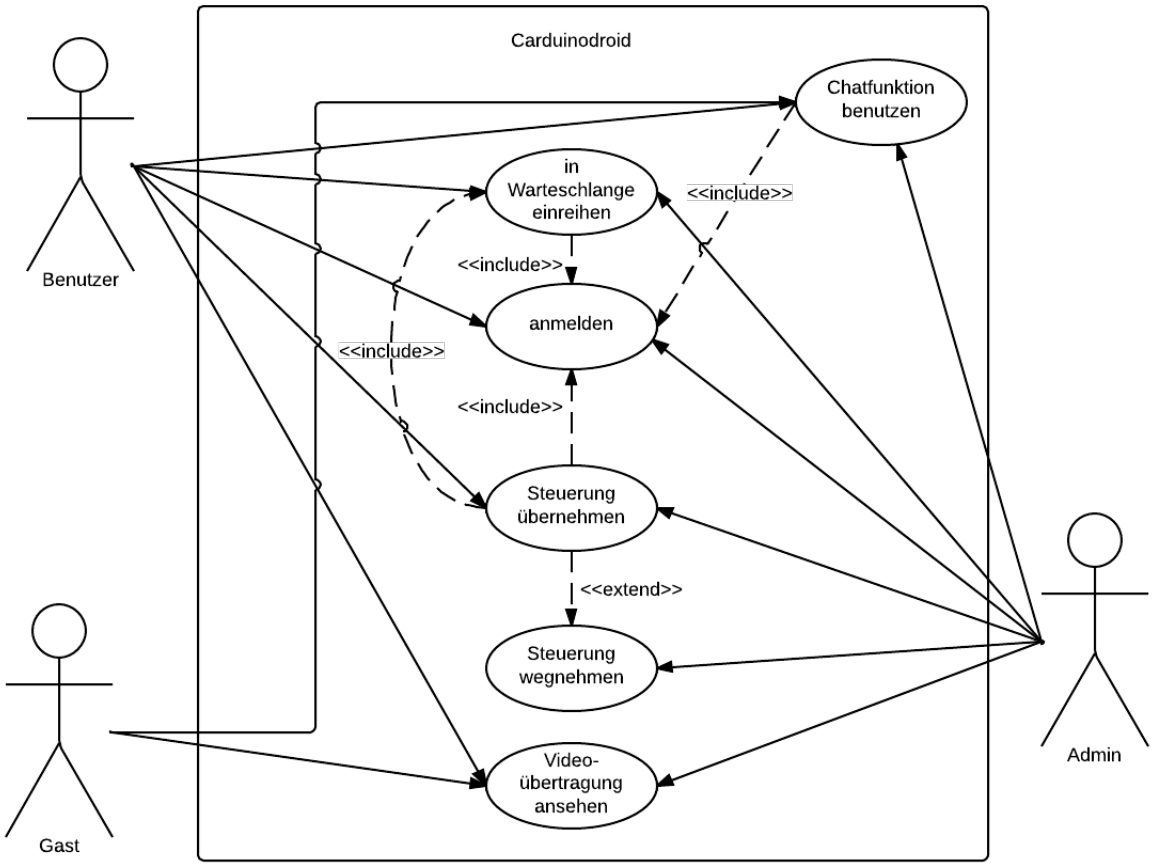
1.1 1. Phase - Planungsphase

In dieser Phase war es unser Ziel ein Pflichtenheft für unser Projekt zu erstellen. In diesem sollten alle relevanten Informationen zu unserem erstellenden Projekt zu finden sein. Somit bekamen wir einen ersten Überblick über den Umfang des Projektes und wir konnten einschätzen, was in nächster Zeit wohl auf uns zukommen werden würde. Weiterhin wählten wir das Agiles Vorgehensmodell für unsere Softwareentwicklung. Der Hauptgrund lag darin, dass wir ein bestehendes Projekt weiterentwickelten und somit eigentlich schon einen Prototypen hatten. Auch die fehlende Entwicklungserfahrung der Mitglieder veranlasste uns zu dieser Wahl. Aber auch unsere Softwareentwicklungswerkzeuge wählten wir bereits in der Planungsphase, wobei wir in der nächsten Phase einiges wieder verworfen haben.

1.2 2. Phase - Konzeptphase

In der zweiten Phase versuchten wir ein Grobkonzept sowie ein Feinkonzept zu erstellen. Wir entwickelten mehrere Diagramme um den Ablauf unserer Software darzustellen und diese so leichter nachzuvollziehen. Weiterhin erstellten wir einen Zeitplan für jedes Gruppenmitglied um eine bessere Übersicht über unsere nächste Phase zu bekommen und eine sinnvolle Arbeitsteilung zu finden. Da wir das agile Vorgehensmodell gewählt hatten, brauchten wir jetzt schon einen kleinen Prototypen, an dem wir in der dritten Phase weiterarbeiteten.

Use-Case-Diagramm



Vorläufiger Zeitplan der 2. Phase

	Woche 1	Woche 2	Woche 3	Woche 4	Woche 5
Alex	IDE / Controller	Kamera	Kamera	Admin-bereich	Testen
Christoph	GUI Javascript	GUI Javascript	Chat Bot	Chat Client	Debugging
Jenja	GUI Design	GUI Implementierung	Java Adminbereich	*Bedarf*	Debugging
Marc	Informations-sammlung	Quellcode -doku	Quellcode-doku	Doku-formatierung	Kontrolle
Michael	DB-Initialisierung	Queue	Queue	Chat Server	Codecleanup
SvenW	Kommunikation Server-Android	Komm. Server-Android	Auto-steuerung	*Bedarf*	Debugging
Vincenz	IDE / DB Verbindung	DB Verwaltung (Admin)	DB Tests / Optimierung	Frontend Finalisierung	Testen

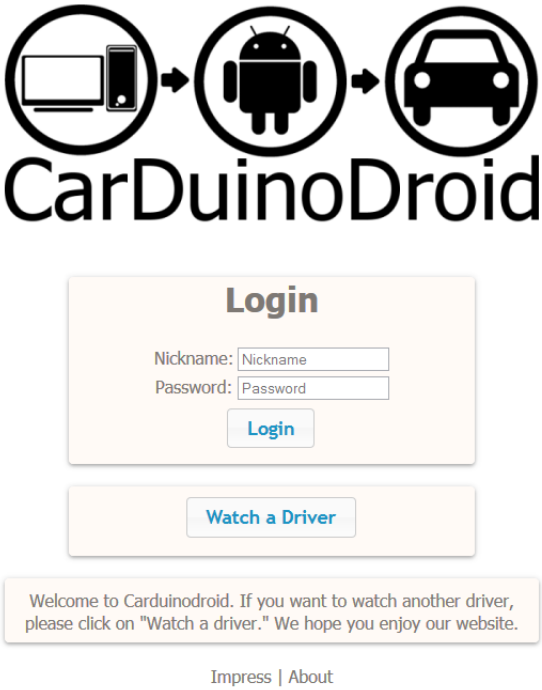
1.3 3. Phase - Implementierungsphase

Unsere Implementierungsphase war die wichtigste und gleichzeitig längste Phase in unserem Projekt gewesen. In dieser versuchten wir, aus unserem Anfangsprototyp einen voll funktionsfähigen Prototyp zu erschaffen, der es uns ermöglicht alle nötigen Funktionen ausführen zu können. Dies versuchten wir durch Einteilung des Projektes in verschiedene Teilaufgaben.
Dies führte zur Einteilung in folgende Unterpunkte:

- GUI
- Datenbank
- Server
- WebSocket
- Chat
- Dokumentation

Da in dieser Phase viel Quellcode entstand und wir diesen möglichst ordentlich und strukturiert abgeben wollten, entschieden wir uns für das Software-Dokumentationswerkzeug “Doxygen“. Mit Latex war es möglich daraus ein PDF zu erzeugen welches unseren Quellcode gut strukturiert darstellte. Weiterhin sammelten wir die bearbeiteten Aufgaben, mit allen Probleme und Lösungen und versuchten diese vorzustellen.

GUI vom Carduinodroid



1.4 4. Phase - Testphase

Besonders in der letzten Phase merkten wir, wer schon mehr Erfahrung im programmieren hatte und wer noch recht unerfahren war. Da das testen und verändern unserer Software viel Feingefühl und ganz neue Herangehensweisen nötig waren, hatten wir oft mit Rückschlägen und dem daraus folgenden Motivationsverlust zu kämpfen.

Besondere Probleme bereitete uns das Einbettung des Bildes via <iFrame>. Somit entstanden Fehler in der GUI, da das Bild zu groß wurde und die Warteschlange verdrängt wurde. Weiterhin waren die übertragenen Bilder anfangs über 2 MB groß, sodass eine verzögerunsfreie Übertragung nicht möglich war. Mit der Kodierung verringerten wir die Datengröße der Bilder um über 85 %, auf ca. 300 KB. Dadurch entstand eine brauchbare Übertragung, allerdings wird diese durch mehrere User verlangsamt und eine Verzögerung entsteht. Ein Stream, statt einer Einzelbildübertragung, kommt für uns wegen der Buffer-Verzögerung nicht in Frage, da hier die Verzögerung noch größer wäre.

Auch merkten wir, dass durch anfänglichen Optimismus Entscheidungen gefällt wurden, deren Ausmaß uns erst jetzt bewusst wurde. Auf der einen Seite lag es teilweise an der fehlenden Erfahrung, da wir nicht immer wussten, welches Programm oder Softwarewerkzeug für uns das geeignete war, auf der anderen Seite erschwerte die manchmal fehlende Kommunikation die Arbeit am Quellcode erheblich.

Kapitel 2

Produktvorstellung

In vorherigen Studentenarbeiten wurde ein handelsübliches Modellfahrzeug um ein Arduino Mikrocontrollerboard und ein Android Smartphone erweitert und somit in eine fernsteuerbare Drohne verwandelt. Dank der Wiederverwendung weniger in sich getesteter Systeme musste kein eigenes Fahrzeugsensornetz entwickelt werden. Ein Android-Smartphone bietet eine große Menge an Interaktionsmöglichkeiten und über die Verbindung mit einem Arduino Board ist auch die digitale und analoge Ein- und Ausgabe von Android aus möglich. Für das Softwareprojekt stehen ein voll funktionstüchtiger Aufbau und eine Desktop-Steuersoftware in Java zur Verfügung.

Ziel dieses Softwareprojekts ist die Entwicklung einer Webseite zur Steuerung des Carduinodroid innerhalb eines Browsers. Die Oberfläche soll vergleichbar zur Desktop-Software eine problemfreie Steuerung des Fahrzeugs ermöglichen.

Für die Darstellung des Videobilds des Android Smartphones wird im aktuellen Entwicklungsstand ein Datenstrom mit Einzelbildern übertragen. Diese Methode könnte durch eine echte Streaming-Technik ersetzt werden, eine flüssige Einzelbildübertragung wäre aber ebenso ausreichend. Diese muss möglichst verzögerungsarm und fehlerunanfällig sein. Weitere für die Webseite benötigte Funktionen sind unter anderem eine Warteschlange, ein Logging der Nutzeraktionen und ein Chat. Beobachter der Seite können sich in der Warteschlange zum Steuern des Fahrzeugs einreihen, die maximale Steuerungszeit eines Nutzers soll dabei maximal 10 Minuten betragen. Ebenso sollte es einen Administratorbereich geben, um die Accounts zu verwalten und bei Problemen handeln zu können.

Alle nötigen Aufgaben konnten wir im Rahmen unseres Softwareprojektes erfolgreich erledigen und können unsere Projektarbeit als Erfolg sehen.

Kapitel 3

Vorgehensmodell und Teamorganisation

Wir entschieden uns für das Agile Vorgehen, da es am besten von den möglichen Vorgehensmodellen zu uns passte. Auf der einen Seite, weil keiner von unserer Gruppe Erfahrungen mit einem größeren Projekt hatte und auf der anderen Seite, weil wir bereits ein funktionsfähigen Prototyp besaßen. Somit war ein schnelles und frühes programmieren möglich. Durch den einfachen Zugriff auf den bestehenden Code erarbeiteten wir unser Projekt Stück für Stück. Ein sehr wichtiger Punkt, den ich hier ausführen möchte, ist die Kommunikation des Teams. Da wir auch nicht die gleichen Studiengänge hatten, war es nicht immer einfach einen geeigneten Termin für alle zu finden, so dass Kompromisse von allen eingegangen werden mussten. Hilfreich war hier für uns das sogenannte “Doodle”. Auf dieser praktischen Website trug jeder alle möglichen, passenden Termine ein und somit konnte schnell und einfach ein Termin gefunden werden.
Zur Allgemeinen Absprache gründeten wir eine Facebookgruppe. Dies klappte anfangs wunderbar, aber spätestens als wir einige Dateien hatten, war eine Übersicht nicht mehr gegeben. Deswegen verwendeten wir zusätzlich “Google Drive” um eine gewisse Ordnung zu schaffen, aber gleichzeitig ein gemeinsames arbeiten an Dateien zu ermöglichen. Dies war besonders wichtig, für die “Review” und Präsentationsdateien, da wir dadurch ein flexibles arbeiten ermöglichten. Somit waren unsere Dateien immer online erreichbar und wurden in Echtzeit aktualisiert.
Da wir unter anderem auch diesen Vorteil mit in unserem Programm-Code verwenden wollten, entschieden wir uns für GitHub. Hier merkten wir, besonders in der letzten Phase, dass es wichtig war, anständig und genau zu arbeiten und jede Änderung oder jeden Fehler genau zu dokumentieren, damit das Team überhaupt eine Chance hatte, die Änderungen nachzuvollziehen und eventuelle Fehler zu korrigieren.

3.1 Tatsächlicher Verlauf

Wir teilten unser Projekt in verschiedene Unterpunkte und versuchten alleine oder in kleineren Teams, die nötigen Aufgaben abzuarbeiten. Da sich dann immer nur ein oder zwei Leute mit dem Thema beschäftigten und zu wenig die einzelnen Aspekte geplant wurden, gab es z.B. beim Thema Websockets einige Probleme. Dies könnte unter anderem auch an unserer mangelnden Erfahrung mit Softwareprojekten liegen. Auch die Absprachen untereinander sollte man in zukünftigen Projekten verbessern, da wir nicht immer wussten, wie weit bzw. wie viel Zeit für die einzelnen Punkte noch benötigt wurde. Zum Ende des Softwareprojektes, besonders in der vierten Phase, arbeiteten wir wieder zusammen, da die einzelnen Arbeiten nun zusammengefügt werden mussten. Genau wie das anschließende Testen und das korrigieren von Fehlern machten wir zusammen.

3.2 Bewertung

Tatsächlich eignete sich das agile Vorgehensmodell für unser Softwareprojekt am besten. Es war sehr sinnvoll einen kleinen Prototypen zu erstellen. An diesem konnten wir schon erste Tests durchführen und kontrollieren, ob alles so klappt, wie wir es geplant hatten. Allerdings hielten wir uns nicht sehr streng an dieses Vorgehen. Zum einen weil wir trotz allem eine anständige Dokumentation benötigten, zum anderen weil uns die einzelnen Phasen des vorgegebenen Projektablaufs als Orientierung dienten. Auch war das Testen und die Fehlerbehebung ein wichtiger Punkt für unser Projekt, da wir versuchten, nicht nur eine funktionsfähige, sondern auch einen verzögerungsfreie Übertragung aufzubauen. Trotz allem, würden wir Rückblickend dieses Vorgehen wieder wählen, da es einfach am besten zu unser Situation passte.

Kapitel 4

Produktanleitung

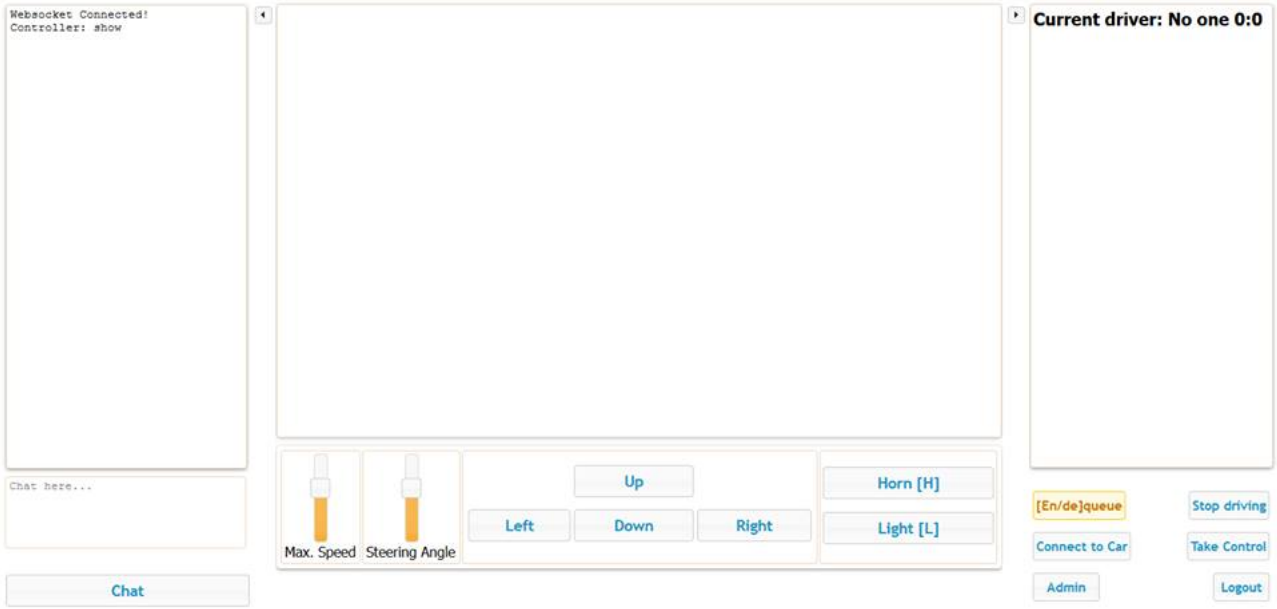
Die Bedienung unserer Software ist an sich selbsterklärend. Das einzige, was gebraucht wird, ist der Link um auf unsere Seite zu kommen. Gängige Browser sind zu empfehlen, da diese bereits getestet wurden und es bei Anderen zu Fehlern kommen könnte.
Die Grafische Benutzeroberfläche haben wir schlicht, einfach und selbsterklärend gehalten. Die UserID sowie das Password erhält man vom Administrator um sich einloggen zu können und um das Carduinodroid zu steuern. Alternativ dazu ist es möglich ohne Benutzerdaten einem Fahrer zuzuschauen und im Chat zu kommunizieren. Somit wäre es möglich, sich über das Carduinodroid zu informieren oder dem Administrator eine Nachricht zu hinterlassen, ohne sich vorher anmelden zu müssen oder sich die nötigen Informationen geben zu lassen.

Wenn man sich auf der Startseite mit einem Administratornamen und dem dazugehörigem Passwort einloggt, gelangt man zur Hauptseite. Auf dieser befindet sich dann ein zusätzlicher Button, um in den Administratorbereich zu kommen. In diesem ist es möglich einen neuen Benutzer mit verschiedenen Rechten zu erstellen. Auch das Protokoll kann angezeigt werden, darin sind unter anderem Logins / Logouts und Nachrichten enthalten.
Auf der Hauptseite ist links der Chat zu sehen, rechts die Warteschlange und in der Mitte soll das Bild der Handykamera erscheinen.

Es gibt verschiedene Einstellungen und Buttons:

- Maximalgeschwindigkeit (Regler)
- Lenkungswinkel (Regler)
- Licht ein-/ausschalten (Taste L)
- Signalton erzeugen (Taste H)
- Richtungstasten (Up, Left, Down, Right)
- Einreihung in die Warteschlange
- Steuerung beenden
- Zum Carduinodroid verbinden
- Abmeldebutton
- Steuerung übernehmen (Administrator)
- Administratorbutton zum Administratormenü (Administrator)

Hauptseite des Carduinodroid



Kapitel 5

Testläufe

Was	Ergebnis	Anmerkung
Ansehen der Videoübertragung mit einem Gast-Account	umgesetzt	
Administrator erstellt einen User-Account.	umgesetzt	
Administrator ändert Rechte eines Users	umgesetzt	
Administrator änder das PW eines Nutzers	umgesetzt	
Administrator löscht einen User-Account.	umgesetzt	auch Admin-Account möglich
Einreihen in die Steuerungswarteschlange mit einem User-Account	umgesetzt	
Der Serveradministrator verändert die maximale Fahrgeschwindigkeit.	umgesetzt	
Der Server baut die Verbindung zum Carduinodroid auf und hält die Verbindung bis zum Abbruch.	umgesetzt	
Der Serveradministrator ändert die Steuerungszeit.	umgesetzt	
Ein Benutzer meldet sich am Server an um geloggt zu werden.	umgesetzt	
Darstellung des Videomaterials mithilfe der Streaming-Technik	entfällt	
Darstellung des Videomaterials mithilfe der Einzelbild-Technik	umgesetzt	
Steuerung des Fahrzeugs mithilfe des bestehenden Protokolls zur Carduinodroid - PC-Client-Kommunikation	umgesetzt	
Steuerung des Fahrzeugs von dem Client aus mithilfe der Pfeiltasten	umgesetzt	
Wechseln der Sprache des Clients	nicht umgesetzt	
Beschleunigen und Bremsen des Fahrzeugs aus dem Client	umgesetzt	
Lenken des Fahrzeugs innerhalb des Client	umgesetzt	
Veränderung des Lenkwinkels	umgesetzt	
Betätigen der Hupe	umgesetzt	
Ein- bzw. Ausschalten des Lichtes	umgesetzt	
Verändern der Auflösung der Videoübertragung	nicht umgesetzt	

Kapitel 6

Softwarewerkzeuge

Carduinodroid Webserver
Webserver for the https://code.google.com/p/carduinodroid/ project

Carduinowebdroid uses the following softwares:

Java - http://www.java.com/de/

Tomcat - http://tomcat.apache.org/

jQuerry UI - http://jqueryui.com/

jQuery DataTable - https://datatables.net/

MariaDB - https://mariadb.org/en/

JSTL - https://jstl.java.net/

Carduinodroid - https://code.google.com/p/carduinodroid/

Kapitel 7

Lizenz

GNU LESSER GENERAL PUBLIC LICENSE Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. http://fsf.org/ Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.
This version of the GNU Lesser General Public License incorporates the terms and conditions of version 3 of the GNU General Public License, supplemented by the additional permissions listed below.

0. Additional Definitions.
As used herein, this Licenserefers to version 3 of the GNU Lesser General Public License, and the "GNU GPLrefers to version 3 of the GNU General Public License.
The Libraryrefers to a covered work governed by this License, other than an Application or a Combined Work as defined below.
An Äpplicationïs any work that makes use of an interface provided by the Library, but which is not otherwise based on the Library. Defining a subclass of a class defined by the Library is deemed a mode of using an interface provided by the Library.
A Combined Workïs a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the Linked Version".

The Minimal Corresponding Sourcefor a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version.
The Corresponding Application Codefor a Combined Work means the object code and/or source code for the Application, including any data and utility programs needed for reproducing the Combined Work from the Application, but excluding the System Libraries of the Combined Work.

1. Exception to Section 3 of the GNU GPL.

You may convey a covered work under sections 3 and 4 of this License without being bound by section 3 of the GNU GPL.

2. Conveying Modified Versions.

If you modify a copy of the Library, and, in your modifications, a facility refers to a function or data to be supplied by an Application that uses the facility (other than as an argument passed when the facility is invoked), then you may convey a copy of the modified version:

a) under this License, provided that you make a good faith effort to ensure that, in the event an Application does not supply the function or data, the facility still operates, and performs whatever part of its purpose remains meaningful, or

b) under the GNU GPL, with none of the additional permissions of this License applicable to that copy.

3. Object Code Incorporating Material from Library Header Files.

The object code form of an Application may incorporate material from a header file that is part of the Library. You may convey such object code under terms of your choice, provided that, if the incorporated material is not limited to numerical parameters, data structure layouts and accessors, or small macros, inline functions and templates (ten or fewer lines in length), you do both of the following:

a) Give prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License.

b) Accompany the object code with a copy of the GNU GPL and this license document.

4. Combined Works.

You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications, if you also do each of the following:

a) Give prominent notice with each copy of the Combined Work that the Library is used in it and that the Library and its use are covered by this License.

b) Accompany the Combined Work with a copy of the GNU GPL and this license document.

c) For a Combined Work that displays copyright notices during execution, include the copyright notice for the Library among these notices, as well as a reference directing the user to the copies of the GNU GPL and this license document.

d) Do one of the following:

0) Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version to produce a modified Combined Work, in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.

1) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (a) uses at run time a copy of the Library already present on the user's computer system, and (b) will operate properly with a modified version of the Library that is interface-compatible with the Linked Version.

e) Provide Installation Information, but only if you would otherwise be required to provide such information under section 6 of the GNU GPL, and only to the extent that such information is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version. (If you use option 4d0, the Installation Information must accompany the Minimal Corresponding Source and Corresponding Application Code. If you use option 4d1, you must provide the Installation Information in the manner specified by section 6 of the GNU GPL for conveying Corresponding Source.)

5. Combined Libraries.

You may place library facilities that are a work based on the Library side by side in a single library together with other library facilities that are not Applications and are not covered by this License, and convey such a combined library under terms of your choice, if you do both of the following:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities, conveyed under the terms of this License.

b) Give prominent notice with the combined library that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

6. Revised Versions of the GNU Lesser General Public License.

The Free Software Foundation may publish revised and/or new versions of the GNU Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library as you received it specifies that a certain numbered version of the GNU Lesser General Public License ör any later versionäpplies to it, you have the option of following the terms and conditions either of that published version or of any later version published by the Free Software Foundation. If the Library as you received it does not specify a version number of the GNU Lesser General Public License, you may choose any version of the GNU Lesser General Public License ever published by the Free Software Foundation.

If the Library as you received it specifies that a proxy can decide whether future versions of the GNU Lesser General Public License shall apply, that proxy's public statement of acceptance of any version is permanent authorization for you to choose that version for the Library.