Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Cliff Parnitzky

Design, Realisierung und Bewertung eines Informationssystems unter dem Aspekt der Software-Ergonomie

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Cliff Parnitzky

Design, Realisierung und Bewertung eines Informationssystems unter dem Aspekt der Software-Ergonomie

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang Angewandte Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Olaf Zukunft
Zweitgutachter: Prof. Dr. rer. nat. Jörg Raasch

Abgegeben am 22. November 2007

Cliff Parnitzky

Thema der Bachelorarbeit

Design, Realisierung und Bewertung eines Informationssystems unter dem Aspekt der Software-Ergonomie

Stichworte

Software-Ergonomie, Ergonomie, Zeiterfassung, Usability-Analyse

Kurzzusammenfassung

Die zunehmende Verbreitung interaktiver Softwaresysteme zur Erleichterung der täglichen Arbeit erfordert ein hohes Maß an Disziplin bei der Gestaltung der Benutzungsoberfläche. Um diese auf die menschlichen Bedürfnisse abzustimmen, existieren Normen, Empfehlungen und Richtlinien. Sie unterstützen bei der Konzeption und Umsetzung ergonomischer Anwendungen. Ziel dieser Arbeit ist die Entwicklung und Bewertung eines Informationssystems insbesondere unter dem Aspekt der Software-Ergonomie. Dazu wird exemplarisch eine webbasierte Applikation zur Erfassung von Arbeitszeiten entworfen, implementiert und auf seine Benutzbarkeit untersucht.

Cliff Parnitzky

Title of the paper

Design, realization and evaluation of an information system in terms of software ergonomics

Keywords

software ergonomics, time recording, workload management, usability analysis

Abstract

The increasing availability of interactive software systems, which are to facilitate the daily work, requires a high degree of discipline when designing the user interface. In order to harmonize this user interface with human needs, there are standards, recommendations and guidelines available, supporting the design and implementation of ergonomic applications. The aim of this study is the development and evaluation of an information system considering software ergonomics in particular. This is exemplified in designing, implementing and examining the usability of a web-based time recording system.

Meiner Lebensgefährtin,
meiner Tochter
und meinen Eltern

Danksagung

Diese Bachelorarbeit entstand im Rahmen meines Studiums an der Hochschule für Angewandte Wissenschaften in Hamburg. Hiermit danke ich allen, die mich während meines Studiums begleitet haben.

Speziell danke ich meiner Lebensgefährtin Sabine Ebisch und meiner Tochter Safiya Shanice, die mir immer zur Seite standen und mich jederzeit unterstützten.

Weiterhin möchte ich meinen Eltern Dank aussprechen, denn ohne ihre jahrelange Unterstützung wäre mein Studium nicht möglich gewesen.

Ich bedanke mich bei Prof. Dr. Olaf Zukunft und Prof. Dr. rer. nat. Jörg Raasch für die hervorragende Betreuung und Unterstützung während der Erstellung dieser Arbeit.

Außerdem danke ich Boris Klengel, der mich bei den Arbeiten im Usability-Labor unterstützt hat und mir mit Rat und Tat zur Seite stand.

Weiterer Dank gilt der Werum Software & Systems AG in Lüneburg, die es mir ermöglichte, während meines Studiums als Werkstudent tätig zu sein, um so viele praktische Erfahrungen sammeln zu können.

Zu guter Letzt danke ich Martin Sadowski, Dennis Dedaj, Oliver Neumann, Dennis Hollatz, Manual Sewzyk, Gunnar Tabel und Marco Sass, die mir als Testpersonen für die Usability-Analyse zur Verfügung standen.

Cliff Parnitzky, November 2007

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Quelltextverzeichnis

1 Einleitung

Die Anpassung von Arbeitsbedingungen und Werkzeuge an die individuellen Bedürfnisse und die gleichzeitige Verbesserung der gesundheitlichen Situation ist schon seit je her ein Ziel, welches der Mensch verfolgt. Der Neandertaler entwickelte sich Hilfsmittel zum Jagen und Schneiden. Die Erfindung des Rades ermöglicht es, schwere Dinge einfach und schnell zu transportieren ohne den Körper zu sehr zu belasten. Mit Hilfe von Ferngläsern kann in weite Ferne geschaut werden, ohne dabei die Augen zu schädigen. Da das Thema Ergonomie auch in der Softwareentwicklung relevant ist und es dort noch große Defizite gibt, muss die Weiterentwicklung und Verbreitung stets vorangetrieben werden.

1.1 Motivation

traditionelles Werkzeug	**computerbasiertes Werkzeug**
vorgegeben, starr	gestaltbar, flexibel
passiv	aktiv
nicht kommunikationsfähig	kommunikationsfähig
Gebrauchsanweisung	selbsterklärungsfähig
optimiert in der Bedienung	eingeschränkt in der Bedienung
eingeschränkt in der Verwendung	universell in der Verwendung

Tabelle 1.1: traditionelles Werkzeug vs. computerbasiertes Werkzeug [Her94, S.2]

Seit Jahrzehnten verbreitet sich der Computer im Berufs- und Privatleben. Verfügten 1993 erst 21,2% der deutschen Privathaushalte über einen Personalcomputer, stieg die Zahl innerhalb von 10 Jahren um mehr als 40 Prozentpunkte auf 61,4% [Sta06]. Hinzu kommt, dass mittlerweile ca. 58% der Beschäftigten in Unternehmen einen Computer in ihrem täglichen Arbeitsleben nutzen. Es stellt sich allerdings die Frage, was einen PC so viel interessanter macht, als die traditionellen Werkzeuge. Wie in Tabelle 1.1 zu sehen, zeichnen sie sich durch Optimierung für ihren Nutzungskontext aus, was ihrer Bedienbarkeit zugute kommt, allerdings die Verwendung stark einschränkt. Bei den computerbasierten Werkzeugen ist das genau umgekehrt. Sie sind sehr vielseitig einsetzbar, dadurch allerdings oftmals nicht sehr gut bedienbar.

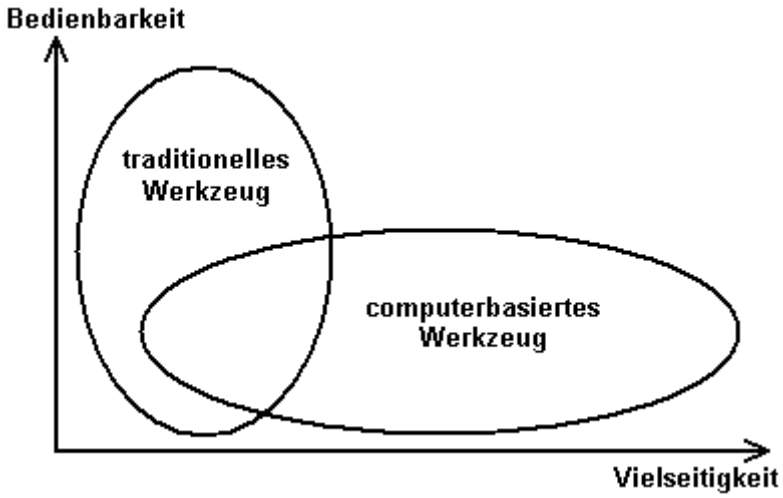


Abbildung 1.1: Bedienbarkeit vs. Vielseitigkeit von Werkzeugen [Her94, S.3]

Abbildung 1.1 verdeutlicht diese Problematik. Mit hohem Entwicklungsaufwand über lange Zeiträume wurde die Bedienbarkeit der Werkzeuge an die Anatomie des Menschen und seine Fertigkeiten angepasst. Dies brachte eine Vielzahl von ergonomischen Erfahrungen, Entwurfsregeln, Vorschriften und Standards hervor. In der Computerwelt fehlen diese evolutionären Entwicklungsprozesse meist jedoch. Mit Hilfe der Software-Ergonomie wird nun das Verhältnis von Mensch und Arbeitswelt (siehe Abbildung 1.2) bezüglich dem Einsatz interaktiver Computersysteme neu betrachtet.

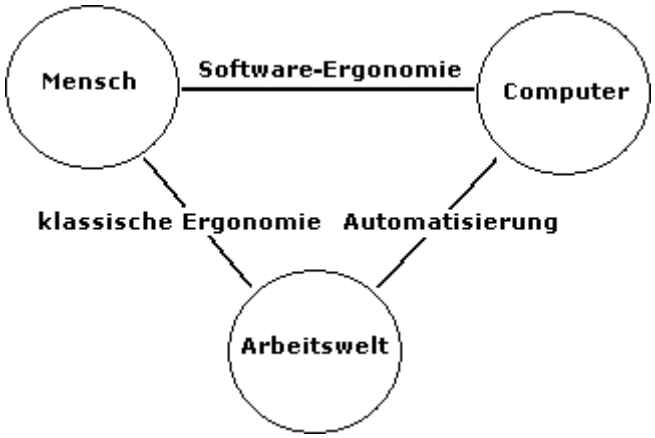


Abbildung 1.2: Software-Ergonomie im neuen Problemkreis [Her94, S.4]

Als allerdings 1983 die erste Fachtagung zum Thema „Software-Ergonomie“ stattfand, war noch völlig unklar, was unter diesem Begriff überhaupt zu verstehen ist [EOO94, S.1]. Dies hat sich im Laufe der Jahre jedoch geändert und so hat sich ein allgemeines Verständnis

herausgebildet. Das Problem ist, dass die Grundsätze der Software-Ergonomie oft verletzt oder ignoriert werden, wobei meist unzureichende Kenntnis, Verfügbarkeit, Verständlichkeit oder Akzeptanz der Grund ist. Daraus resultieren dann negative Auswirkungen auf den Benutzer des interaktiven Computersystems. Unnötige körperliche und geistige Belastungen, hoher Einarbeitungsaufwand und die Herausbildung „unentbehrlicher“ Experten sind nur einige der Folgen für den Anwender.

Da der Stellenwert interaktiver Computersysteme stetig steigt, ist es zwingend notwendig, den Softwareentwicklern diese Sammlung von Normen, Empfehlungen und Richtlinien nahe zu bringen. Nur mit einem vielseitigen software-ergonomischen Grundwissen wird es künftig möglich sein, menschengerechte und nützliche Softwaresysteme zu erstellen, die auf die körperlichen und geistigen Belastungen ihrer Anwender Rücksicht nehmen [Her94, S.2ff].

Diese Arbeit wird exemplarisch die Notwendigkeit von Software-Ergonomie an einem webbasierten Zeiterfassungssystem zeigen. Da die Auswertung von Projektarbeitszeiten ein wichtiger Bestandteil zur Leitung eines Projektes ist, sind Projektmanager von der Vollständigkeit dieser Daten abhängig. Zur Kommunikation mit dem Kunden aber auch zur Kontrolle des Projektplans ist es unbedingt notwendig, exakt den aktuellen Arbeitsaufwand festzustellen. Um so mehr ist der Projektleiter auf ein gut bedienbares Zeiterfassungssystem angewiesen, welches von seinen Mitarbeitern genutzt wird. Da die Personalarbeitszeiterfassung einen zusätzlichen Aufwand neben der normalen Arbeit für jeden Beschäftigten bedeutet, ist es umso wichtiger, dass diese Systeme einen hohen Grad an Benutzbarkeit aufweisen. Nur wenn der Anwender ohne Probleme schnell und einfach seine täglichen Arbeitsstunden notieren kann, wird das System eine Entlastung für alle Beteiligten bringen.

1.2 Aufgabenstellung

Zuerst werden die Anforderungen an ein solches webbasiertes Zeiterfassungssystem analysiert. Dazu werden 3 Referenzsysteme untersucht, um die Grundbestandteile für ein solches System zu identifizieren. Außerdem wird auf die Verwendung software-ergonomischer Grundsätze achtgegeben. Die gewonnenen Ergebnisse fließen dann in die Anforderungsanalyse des zu entwerfenden Systems ein.

Nach diesem Analyseschritt wird ein Entwurf des Systems erstellt. Dort finden sich sowohl die allgemein erforderlichen Bestandteile, als auch spezielle Merkmale, die sich positiv auf die Software-Ergonomie auswirken sollen, wieder.

Als nächstes wird der Entwurf realisiert. Hier wird ein Prototyp erstellt, weil ein fertiges, marktreifes Werkzeug nicht das Ziel dieser Arbeit ist.

Der entstandene Prototyp wird dann mittels einer Usability-Analyse untersucht. Dieser Test wird im Usability-Labor der Hochschule für Angewandte Wissenschaften Hamburg durchgeführt. Nach Abschluss dieser Untersuchung wird das System bezüglich seiner Software-Ergonomie bewertet.

1.3 Ziel der Arbeit

Diese Arbeit hat zum Ziel, das Konzept für ein ergonomisches webbasiertes Zeiterfassungssystem zu entwerfen und dieses prototypisch umzusetzen. Hierbei sollen die Grundsätze und Richtlinien der Software-Ergonomie besonders beachtet werden. Die anschließende Usability-Analyse soll Auskunft darüber geben, ob das entwickelte System an die Bedürfnisse seiner Anwender angepasst wurde.

Weiterhin soll gezeigt werden, dass Usability-Analysen heute zwingend notwendige Hilfsmittel bei der Entwicklung interaktiver Softwaresysteme sind und dass sie ein Scheitern eines Projektes vermeiden können.

2 Grundlagen

Um ein allgemeines Verständnis für den Inhalt dieser Arbeit zu bekommen, werden im folgenden Kapitel die grundlegenden Begriffe geklärt. Dazu wird als erstes der Fachausdruck Software-Ergonomie definiert. Danach folgen die Bewertungskriterien sowie eine Zusammenfassung zur Usability-Analyse und zum Vorgehensmodell bei der Entwicklung ergonomischer Software. Anschließend folgt eine Definition von Zeiterfassung und ein Überblick über Kommunikationsmodelle im Internet.

2.1 Software-Ergonomie

„Software-Ergonomie ist heute zu einem der wesentlichen Akzeptanzfaktoren der Informationstechnik geworden. Bei ständig steigender Verbreitung der Rechner am Arbeitsplatz und im Privatleben kommt der Anpassung der Benutzungsoberfläche an den Menschen eine immer größere Bedeutung zu.“ [Lau87]

Die Benutzergruppen, welche Computer verwenden, um mit interaktiven Softwaresystemen zu arbeiten, reichen vom Laien bis zum Experten. Umso wichtiger ist es deshalb, die Anwendungen nach menschlichen Bedürfnissen zu gestalten. Dabei sind Effektivität, Effizienz und Zufriedenstellung eines Produktes Maßstab für die Bewertung der Brauchbarkeit dieses Softwaresystems [Ins02]. Abbildung 2.1 zeigt den Zusammenhang zwischen dem Nutzungskontext eines Produktes und seiner Gebrauchstauglichkeit. Diese ergibt sich aus dem Grad der Nutzung des Produktes. Die Software-Ergonomie befasst sich mit der menschengerechten Gestaltung und Bewertung von Softwaresystemen. Das dabei verfolgte Ziel ist ein möglichst hoher Tauglichkeitsgrad bzgl. des jeweiligen Nutzungskontextes.

2.1.1 Definition

Die Erledigung vieler Arbeiten ist ohne den Einsatz Interaktiver Softwaresysteme heutzutage kaum mehr vorstellbar. Laut einer Mitteilung des Statischen Bundesamtes nutzen mehr als jeder zweiter Beschäftigte im täglichen Arbeitsleben einen Computer [Sta07]. Dies war allerdings nicht immer so. Am Anfang der Softwareentwicklung stand der funktionale Aspekt

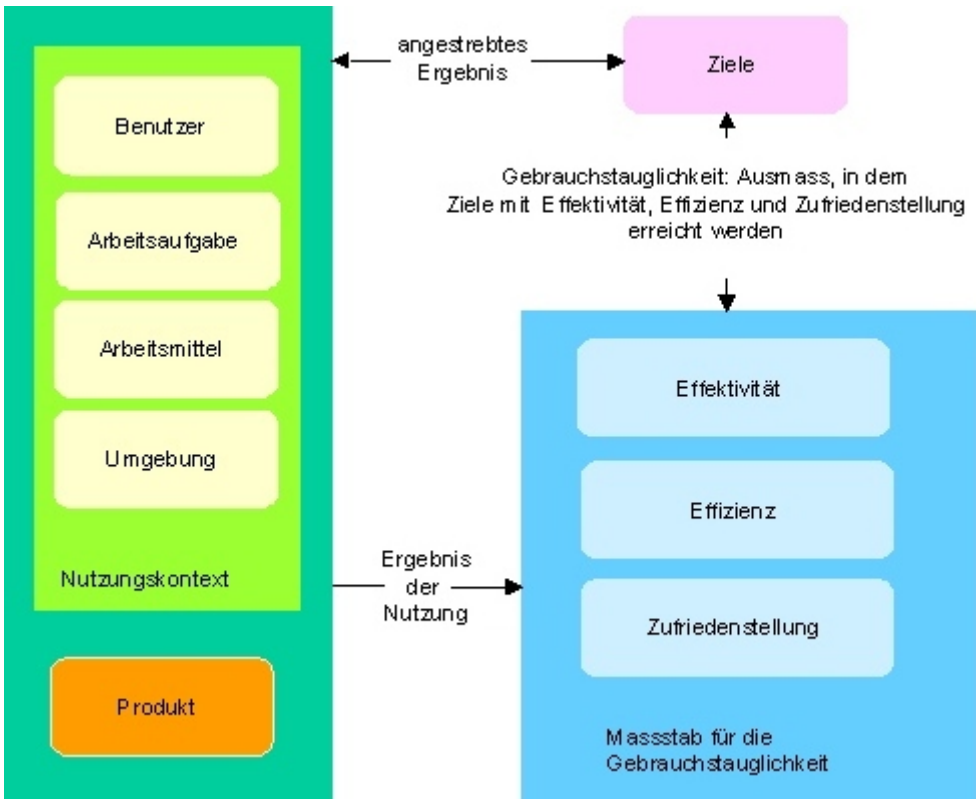


Abbildung 2.1: Anwendungsrahmen der Gebrauchstauglichkeit [Ins02]

eines Systems im Vordergrund. Auf Ergonomie musste wenig geachtet werden, denn meist waren die Entwickler dieser softwarebasierten Werkzeuge auch die Benutzer. Allerdings wurden in den letzten Jahrzehnten vermehrt neue Systeme zur Aufgabenorganisation und -abarbeitung eingeführt. Die starke Verbreitung dieser Systeme und die zunehmende Anzahl von Benutzern rückt dessen Bedürfnisse deutlich in den Vordergrund. So gibt es nicht nur viele Konstruktionsprinzipien, Normen, Regeln und Erkenntnisse sondern auch ein Gesetz, welches die korrekte Gestaltung von Bildschirmarbeitsplätzen vorschreibt.

Um nun den Begriff Software-Ergonomie zu klären, müssen erstmal die beiden Bestandteile erläutert werden. Bei Software handelt es sich um jede Art von Programmen die auf einem Computer ausführbar sind. Meist werden auch die von Computerprogrammen verwendeten Daten als Software bezeichnet. Das Wort Ergonomie lässt sich in das griechische Wort ergon (Arbeit) und nomos (Lehre, Gesetz) zerlegen. Ergonomie ist also die Wissenschaft von der Arbeit. Der in der englischen Übersetzung Usability Engineering benutzte Begriff Usability wird dabei oft für Gebrauchstauglichkeit verwendet. Die Zerlegung dieses Begriffs in die zwei Worte use (benutzen) und ability (die Fähigkeit) ergibt die Definition: Fähigkeit, etwas zu benutzen oder die Nutzbarkeit einer Sache [Ins02].

Software-Ergonomie ist die Wissenschaft von der Benutzbarkeit von Softwaresystemen. Diese Wissenschaft beschäftigt sich mit der Gestaltung und Bewertung von Bildschirmarbeitsplätzen, unter dem Gesichtspunkt der mentalen, psychischen und emotionalen Leistungsmöglichkeit und Belastbarkeit der Softwarebenutzer. Als Teilgebiet der Mensch-Computer-Interaktion, welche sich mit der benutzergerechten Gestaltung von interaktiven Systemen und ihren Mensch-Maschine-Schnittstellen beschäftigt, konzentriert sich Software-Ergonomie im Wesentlichen auf den arbeitenden Menschen und auf die nach dessen Bedürfnissen gestalteten Softwaresysteme.

Formal wurden die Erkenntnisse der Software-Ergonomie in der DIN EN ISO 9241-10, im Arbeitsschutzgesetz und in der Bildschirmarbeitsverordnung festgehalten. In Teil 10 der europäischen Norm EN ISO 9241 sind die allgemeingültigen Kriterien für das Design ergonomischer Benutzungsschnittstellen definiert [Wir05]. Das Arbeitsschutzgesetz und die Bildschirmarbeitsverordnung legen fest, wie menschen- und arbeitsgerechte Prinzipien und Merkmale zu gestalten und zu bewerten sind [Fra05]. Am 20.12.1996 trat die Verordnung zur Gestaltung von Bildschirmarbeitsplätzen in Kraft. Bei Verstößen begehen die Unternehmen laut §7 eine Ordnungswidrigkeit, welche hohe Bußgelder zur Folge haben [Bun06; Gre01].

2.1.2 Bewertungskriterien

Software-Ergonomie zeichnet sich durch viele verschiedene Merkmale aus. Die in der Folge genannten Prinzipien beruhen auf den Grundsätzen der Arbeitsgestaltung und zielen darauf ab, die Tätigkeiten an Bildschirmarbeitsplätzen menschengerecht zu gestalten. Gleichzeitig soll ein hohes Maß an Arbeitszufriedenheit und Wohlbefinden unter Berücksichtigung der geistigen Fähigkeiten und körperlichen Gegebenheiten der Beschäftigten erzielt werden. Durch Heranziehen der Prinzipien ist dies zu erreichen. Sie sind in der Umsetzung direkt (z. B. Aufgabenangemessenheit) oder indirekt (z. B. Erwartungskonformität) zu operationalisieren und beziehen sich auf die Software (Funktionalität, Effektivität, Effizienz), den Menschen (Wahrnehmung, Gedächtnisleistung, etc.) und die Arbeitsorganisation (Informationsfluss, Synchronisation, etc).

2.1.2.1 Aufgabenangemessenheit

„Ein Dialog ist aufgabenangemessen, wenn er den Benutzer unterstützt, seine Arbeitsaufgabe effektiv und effizient zu erledigen.“ [Wir05]

Ein Softwaresystem wird als aufgabenangemessen bezeichnet, wenn es den Benutzer in der Durchführung seiner Aufgaben effektiv und effizient unterstützt. Es muss also bei der

Erledigung der Arbeit entsprechende Hilfestellung leisten, wodurch der Abarbeitungsprozess für den Benutzer erleichtert wird. So ist es z. B. wichtig Schaltflächen so zu positionieren, dass sie bequem und schnell erreicht werden können. Jegliche Behinderung oder Verzögerung der Arbeit beeinträchtigt die Aufgabenangemessenheit einer Software negativ [Fra05; SRC99, S.12f].

> *Beispiel:*
> *Der Cursor muss automatisch in dem Formularfeld gesetzt sein, welches im Handlungsablauf an erster Stelle steht.*

2.1.2.2 Selbstbeschreibungsfähigkeit

„Ein Dialog ist selbstbeschreibungsfähig, wenn jeder einzelne Dialogschritt durch Rückmeldung des Dialogsystems unmittelbar verständlich ist oder dem Benutzer auf Anfrage erklärt wird.“ [Wir05]

Die Selbstbeschreibungsfähigkeit eines Softwaresystems bezeichnet die Fähigkeit des Systems, dem Benutzer den Einsatzzweck sowie den Leistungsumfang der Software zu erläutern. Weiterhin gilt ein System als selbstbeschreibungsfähig, wenn es jeden Interaktionsschritt durch eine verständliche Rückmeldung bestätigt, oder wenn dem Benutzer bei jedem Interaktionsschritt entsprechende Erläuterungen zur Verfügung gestellt werden. Ein System muss also zur Kommunikation die Sprache des Benutzers verwenden und möglichst genau durch die einzelnen Interaktionsschritte führen, um Kommunikationsfehler und daraus folgende weitere Probleme zu vermeiden [Fra05; SRC99, S.13].

> *Beispiel:*
> *Nach der Durchführung eines Arbeitsschrittes, z. B. dem Schreiben in eine Datenbank, wird dem Benutzer eine Meldung angezeigt, ob die Aktion erfolgreich war, oder nicht.*

2.1.2.3 Steuerbarkeit

„Ein Dialog ist steuerbar, wenn der Benutzer in der Lage ist, den Dialogablauf zu starten sowie seine Richtung und Geschwindigkeit zu beeinflussen, bis das Ziel erreicht ist.“ [Wir05]

Mit der Steuerbarkeit wird bezeichnet, ob ein Softwaresystem bedienbar ist. Hierzu ist es wichtig, dass der Benutzer die Geschwindigkeit des Ablaufs einer Interaktion beeinflussen kann. Weiterhin muss es jederzeit möglich sein, die Auswahl und Reihenfolge von Arbeitsgegenständen und Interaktionshilfsmitteln sowie die Art und den Umfang von Ein- und Ausgaben, zu steuern. Um das zu gewährleisten, muss der Benutzer die Freiheit besitzen, sich

seine persönlich beste Softwarekonfiguration zu erstellen. Somit übernimmt er die Steuerung der Software, was dessen Effektivität und Effizienz maximiert [Fra05; SRC99, S.13].

> *Beispiel:*
> *Um Übersichtlichkeit zu schaffen, können Fensterinhalte eines Systems jederzeit deaktiviert und wieder aktiviert werden. Angefangene Arbeiten in einem Fenster können unterbrochen werden, um Arbeiten in anderen Fenstern durchzuführen und dann wieder ins Ausgangsfenster zurückzukehren.*

2.1.2.4 Erwartungskonformität

„Ein Dialog ist erwartungskonform, wenn er konsistent ist und den Merkmalen des Benutzers entspricht, z. B. seinen Kenntnissen aus dem Arbeitsgebiet, seiner Ausbildung und seiner Erfahrung sowie den allgemein anerkannten Konventionen.“ [Wir05]

Die Erwartungskonformität eines Softwaresystems beschreibt dessen benutzerbezogene Zuverlässigkeit. Es müssen also die Erwartungen der Anwender, welche auf bisherigen Arbeitsabläufen, Ausbildungen und Erfahrungen beruhen, erfüllt werden. Die Benutzer haben aber auch Erwartungen an das System, welche aus dem bisherigen Umgang mit der Software resultieren. Das ist zum Einen die Systemtransparenz, welche die menschlich durchschaubare Konzeption von Abläufen, Aufgaben und Interaktionsmitteln betrifft, und zum Anderen die Systemkonsistenz, die sich sowohl auf den regelmäßigen Aufbau von Benutzungsschnittstellen als auch auf den gleichartigen Ablauf von Interaktionen von identischen oder ähnlichen Situationen bezieht. Werden also Eingaben in ähnlichen Aktionen getätigt, muss auch das Verhalten der Software ähnlich sein [Fra05; SRC99, S.13f].

> *Beispiel:*
> *Die Erfassung und Auswertung von Daten für ähnliche Aktionen weisen immer eine gleichartige visuelle Darstellung auf.*

2.1.2.5 Fehlertoleranz

„Ein Dialog ist fehlertolerant, wenn das beabsichtigte Arbeitsergebnis trotz erkennbar fehlerhafter Eingaben entweder mit keinem oder mit minimalem Korrekturaufwand seitens des Benutzers erreicht werden kann.“ [Wir05]

Ein fehlertolerantes Softwaresystem erzeugt, trotz erkennbarer fehlerhafter Eingaben durch den Benutzer, das beabsichtigte Arbeitsergebnis. Dies wird ohne oder nur mit wenig Korrekturaufwand erreicht, wobei dem Benutzer die Fehler zum Zweck der Verbesserung verständlich gemacht werden müssen. Wichtig dabei ist, dass die Fehler entsprechend spezifisch

behandelt werden, ohne den kompletten Arbeitsablauf neu zu beginnen. Diese Robustheit gegen Fehler betrifft unter anderem die Erkennung von Eingabefehlern (Plausibilisierung), die automatisierte Korrektur von Fehlern und die Stabilität des System ohne unkontrollierte Abbrüche oder undefinierte Zustände [Fra05; SRC99, S.14].

> *Beispiel:*
> *Bei der Überschreitung eines Zahlenbereichs in der Benutzereingabe wird automatisch die höchste oder die niedrigste Zahl eingetragen. Zusätzlich wird dem Benutzer eine Fehlermeldung, mit dem Hinweis auf diese Falscheingabe, angezeigt.*

2.1.2.6 Individualisierbarkeit

„Ein Dialog ist individualisierbar, wenn das Dialogsystem Anpassungen an die Erfordernisse der Arbeitsaufgabe sowie an die individuellen Fähigkeiten und Vorlieben des Benutzers zuläßt.“ [Wir05]

Die Fähigkeit einer Software, sie individuell an die Benutzeransprüche anzupassen ist ein sehr kontroverses Kriterium. So widerspricht die Individualisierbarkeit dem Einsatz von Standard Software, welche mit vielen gleichen Funktionen einer großen Gruppe von Benutzern zugänglich sein soll. Der Konflikt, individuelle Lösungen gegen Rücksicht auf persönliche Kenntnisse und Erfahrungen, tritt häufig beim Design ergonomischer Software auf. Trotz allem wird mit der Individualisierbarkeit auf eine Erhöhung der Effizienz abgezielt [Fra05; SRC99, S.14f].

> *Beispiel:*
> *Ein Benutzer kann sich die Symbolleisten nach seinem persönlichen Bedarf, durch Hinzufügen oder Entfernen von Schaltflächen, zusammenstellen.*

2.1.2.7 Lernförderlichkeit

„Ein Dialog ist lernförderlich, wenn er den Benutzer beim Erlernen des Dialogsystems unterstützt und anleitet.“ [Wir05]

Ein Softwaresystem ist lernförderlich, wenn es den Anwender bei der Bewältigung seiner Aufgaben unterstützt und auch Anleitungen gibt. Da jeder Benutzer die Bedienung einer Software erstmal lernen muss, ist ihm auch eine angemessene Lernzeit zu gewähren, in der er sich mit der Software vertraut machen kann. So muss er sich nicht nur auf die Anordnung und Funktionalität der Schaltflächen einstellen, sondern auch auf die Art und Weise der Bearbeitung seiner Aufgaben. Um ein hohes Maß an Lernförderlichkeit zu erreichen, muss die

Lerndauer minimal sein. Ist die Einarbeitungsphase abgeschlossen, muss es dem Benutzer möglich sein, die Software umfassend zu benutzen, ohne lange nach benötigten Funktionen zu suchen. Deshalb ist es sehr wichtig, die Benutzungsoberfläche gut zu organisieren und häufig verwendete Funktionen direkt sichtbar zu machen [Fra05; SRC99, S.15].

> *Beispiel:*
> *Beim Start des Programms wird dem Benutzer eine geführte Tour (Guided Tour) durch die wichtigsten Funktionen der Software angeboten.*

2.1.3 Usability-Analyse

Um die Ergonomie eines Softwaresystems festzustellen und etwaige Defizite aufzudecken werden Usability-Analysen durchgeführt. Dazu gibt es unterschiedliche Vorgehensweisen und eine Vielzahl von Testmethoden. Diese sind verschieden komplex und damit verschieden aufwendig und kostenintensiv. Eines haben sie aber alle gemeinsam, sie sollen den Grad der Benutzbarkeit einer Software bestimmen.

2.1.3.1 Vorgehensweisen

Die Testformen für die Durchführung von Usability-Analysen lassen sich in 3 grundlegende Vorgehensweisen unterscheiden.

Explorativer Test

Bei dieser Vorgehensweise testen verschiedene Probanden ein Softwareprodukt auf seine Gebrauchstauglichkeit. Im Idealfall haben sie keinerlei Vorkenntnisse über das Produkt, wobei Kenntnisse über das Arbeitsumfeld erwünscht sind. Die durch die Tester identifizierten Schwachstellen werden im Anschluss weiteren Analysen und Priorisierungen unterzogen [Mek03].

Schwellentest

Mittels dieser Testform wird der Grad des Erreichens vorher festgelegter Leistungsgrößen bestimmt. Dazu werden zuerst die Ziele, welche die Software erreichen soll, festgelegt. Dann wird ermittelt, wie gut diese Ziele erreicht werden. So muss das zu untersuchende Produkt z. B. bestimmte Antwortzeiten einhalten [Mek03].

Vergleichstest

Bei einem Vergleichstest werden zwei oder mehrere ähnliche Produkte hinsichtlich ihrer Usability-Kriterien miteinander verglichen. Die verschiedenen Alternativen können sowohl Entwicklungsvarianten als auch Konkurrenzprodukte sein. Es kann sich aber auch um Prototypen des gleichen Produktes in unterschiedlichen Entwicklungsstadien handeln [Mek03].

2.1.3.2 Prüftechniken

Zur Untersuchung der Software-Ergonomie steht eine Vielzahl verschiedener Prüftechniken zur Verfügung, von denen hier einige erklärt werden. Eine Klassifizierung der Techniken ist auf unterschiedliche Art und Weise möglich. So gibt es unter anderem Benutzer- und Expertentests.

Die Untersuchung der Usability durch Benutzer dient sowohl dem Identifizieren fehlender Funktionen, als auch dem Aufdecken von Fehlern, die durch erfahrene Benutzer kaum auftreten würden. Dazu werden verschiedene Probanden bei der Benutzung einer Software beobachtet. Die Ergebnisse dieser Beobachtungen eröffnet den Entwicklern meist ganz neue Perspektiven und schult sie, Fehlerquellen besser zu erkennen.

Die Expertentests werden von Spezialisten einer Softwareart durchgeführt. Ihre Kompetenzen sind die langjährige Erfahrung mit einer speziellen Softwaregattung, fundierte Kenntnisse wissenschaftlicher Grundlagen der Lern- und Wahrnehmungspsychologie, eine gute Menschenkenntnis und das Vermögen, sich in einen anderen Menschen hineinversetzen zu können.

Videodokumentationen

Bei dieser Technik werden die Probanden während der Durchführung des Tests von Kameras gefilmt. Dazu ist ein Usability-Labor (siehe Kapitel 2.1.3.3) besonders geeignet. Dort sind mehrere Kameras installiert, welche den Benutzer aus verschiedenen Perspektiven aufnehmen. Außerdem wird der Desktop des Benutzers aufgezeichnet. Im Anschluss an den Test werden die Videodaten und der Film des Desktops zusammengeschnitten, um alle Aufnahmen in einem Bild zu haben. Das resultierende Video dient als multimediales Protokoll. Durch die Aufzeichnung der Beobachtungen können Aktivitäten mehrfach angesehen und ausgewertet werden.

Think Aloud

Diese Methode ist eine der populärsten Techniken im Bereich der Benutzertests. Hierbei werden die Probanden zuerst in das zu testende Produkt und in die auszuführenden Aufgaben eingewiesen. Während der Benutzung des Produktes sollen die Tester dann verbal

ihre Gedanken, Gefühle und Meinungen äußern. Dazu kann sie der Testleiter auch durch Fragen, wie z. B. „Entspricht dies ihren Erwartungen?“, animieren. Auf diese Weise kann ein besseres Verständnis vom mentalen Modell des Benutzers und dessen Umgang mit dem Produkt erlangt werden [Ram07; Fra05].

Fragebogen

Der Einsatz von Fragebögen gehört zur Klasse der Benutzertests. Er wird sowohl von Anwendern zur Überprüfung der Nutzungsqualität einer Software genutzt als auch von Softwareeinkäufern zur Ermittlung des besten Produktes für deren Unternehmen. Außerdem wird er von Entwicklern verwendet, da damit die Nutzungsqualität eines Softwareproduktes erarbeitet und optimiert werden kann.

Der SUMI-Fragenkatalog[1] stellt 50 Fragen zur Verfügung, die mit Hilfe von umfangreichen wissenschaftlichen Methoden zur Fragebogenkonstruktion entwickelt wurden. Die resultierende emotionale Beurteilung der Software ist zwar nur eine subjektive Bewertung, stellt allerdings ein verlässliches Instrument zur Messung der Benutzbarkeit des Produktes dar [Fra05].

Cognitive Walkthrough

Bei dieser aufgabenorientierten Evaluationsmethode untersuchen Usability-Experten die Software anhand bestimmter Aufgaben. Dazu werden konkrete Handlungsabläufe des Anwendungskontextes gesichtet und bewertet. So ist es möglich zu überprüfen, ob die Software ihre Aufgaben erfüllt, welche Probleme ggf. auftreten und ob die Erwartungen des Benutzers erfüllt werden. Da diese Methode rasch durchführbar ist, kann sie schon in einer frühen Phase des Entwicklungsprozesses zum Einsatz kommen [Med07].

Heuristische Evaluation

Bei der Heuristischen Evaluation wird ein Softwareprodukt von Usability-Experten anhand anerkannter Normen, Standards und Gestaltungsrichtlinien überprüft und bewertet. Das resultierende Expertengutachten enthält Abweichungen und Lösungsvorschläge. Diese Methode liefert relativ schnell qualitativ hochwertige Ergebnisse, welche eine gute Basis für die Behebung von Usability-Problemen darstellen. Dazu werden die identifizierten Problembereiche nach ihrem Schweregrad klassifiziert, was anschließend eine Prioritätenliste für die Behebung hervorbringt [Med07; Fra05; Ram07].

[1] Software Usability Measurement Inventory

Checkliste

Checklisten werden eingesetzt, um Probleme in Anwendungen zu identifizieren. Sie haben einen sehr praxisbezogenen Charakter, was nicht sehr viele Interpretationen zulässt. Eine individuelle Anpassung der Listen ist oft notwendig, da sie allgemeingültig für Klassen von Software konzipiert sind. Sie werden vorwiegend für Laien entwickelt, können aber von Experten meist besser interpretiert werden. Deshalb gehören sie zur Klasse der Expertentests. [Ram07].

Eyetracking

Beim Eyetracking misst eine hochauflösende Kamera die Augenbewegungen von Versuchspersonen, was Aufschluss darüber gibt, welche Elemente der Benutzungsoberfläche für den Anwender im Vordergrund stehen und Aufmerksamkeit erzeugen. Die zwei wichtigsten Messgrößen für die Interpretation sind dabei der Fokus und die Zeit. Auf Grundlage dieser objektiven Ergebnisse können der Blickverlauf während der Orientierungsphase und die unterschiedliche Verweildauer in einzelnen Bereichen ausgewertet werden. So ist es möglich, schlecht strukturierte Benutzungsoberflächen zu identifizieren [Med07; Ram07].

Logfileanalyse

Eine Logfileanalyse bietet die Möglichkeit nach der Fertigstellung des finalen Produktes die tatsächliche Benutzung des Systems aufzuzeichnen und auszuwerten. Dazu werden alle Benutzeraktionen und die darauf folgenden Systemreaktionen in einer speziellen Datei in einer zuvor festgelegten Art und Weise dokumentiert und gespeichert. Mittels geeigneter Auswertungstechniken können die gesammelten Daten dann größtenteils automatisiert ausgewertet werden. So können Rückschlüsse über die tatsächliche Nutzung des Systems und über eventuell auftretende Nutzungsprobleme gezogen werden. Bei der Verwendung dieser Methoden muss allerdings der Datenschutz beachtet werden, da so auch ein komplettes Benutzerprofil erstellt werden kann [Fra05; Ram07].

2.1.3.3 Usability-Labor

Zur Durchführung von Usability-Analysen wird eine spezielle Laborumgebung benötigt. In einem solchen Usability-Labor (siehe Abbildung 2.2) ist es möglich, die Tests ohne störende Einflüsse zu protokollieren. Dazu gibt es einen Testraum und einen Beobachtungsraum, die durch einen halbdurchsichtigen Spiegel voneinander getrennt sind. Im Testraum werden alle Aktionen des Benutzers von Kameras und Mikrofonen aufgezeichnet. Die gesammelten Daten laufen im Beobachtungsraum zusammen und werden dort zu einem multimedialen Protokoll zusammengefasst [Usa].

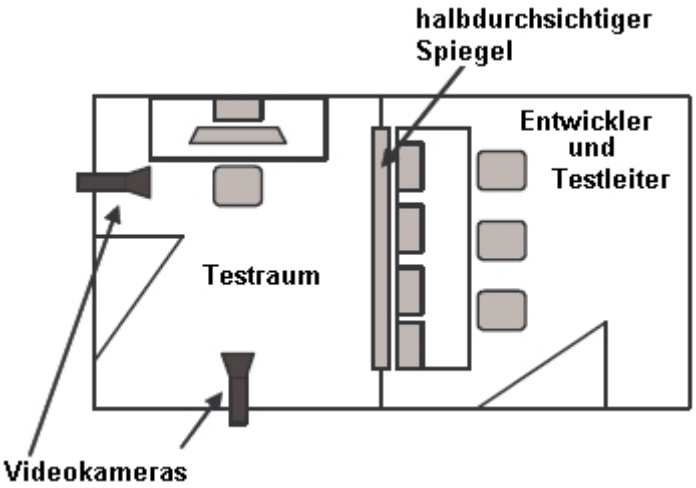


Abbildung 2.2: Grundriss eines Usability-Labors (angelehnt an [Szw06, S.1])

2.1.4 Vorgehensmodell

Der Entwicklungsprozess von Softwaresytemen orientiert sich an Vorgehensmodellen. Dabei gliedern die Modelle den Prozess in strukturierte Phasen. Somit werden die allgemeinen Aufgaben und Aktivitäten in einer logischen Ordnung dargestellt. Abhängig von ihrem Einsatzgebiet variiert die Anzahl und Bedeutung der Phasen in den verschiedenen Modellen.

Zur Entwicklung ergonomischer Software ist das Evolutionäre Vorgehensmodell (siehe Abbildung 2.3) besonders gut geeignet. Dieser Autor-Kritiker-Zyklus zeichnet sich durch eine iterative Abfolge von Analysieren, Modellieren und Bewerten aus. Nach dem identifizieren der Anforderungen wird hier ein Prototyp erstellt. Danach folgt die Untersuchung und Bewertung der Benutzbarkeit. Die Auswahl der Autoren und Kritiker muss besonders geeignet sein, da ständige Rückkopplung zwischen den Parteien eine Soll- / Ist-Gleichheit gewährleisten soll. Am Schluss jeder Iteration wird entschieden, ob die aktuelle Softwareversion verwendet werden kann oder weiter verbessert werden muss [KRZ05, S.111].

Der Vorteil dieses Vorgehensmodells ist, dass es mehrere Zwischenergebnisse statt einem fertigen Ergebnis gibt. So ist es möglich in kurzen Zeitabständen einen Prototypen zu erstellen und diesen als einsatzfähiges Produkt an den Kunden auszuliefern. Die dann folgende Interaktion mit dem Kunden trägt dazu bei, die Produktspezifikation zu verfeinern und weitere Anforderungen zu erheben. Besonders gut eignet sich dieses Modell wenn die Anforderungen anfangs noch recht unklar sind und die Rückkopplung mit dem Kunden sehr wichtig ist. Die während des Projekts gewonnenen Erkenntnisse oder geänderten Anforderungen lassen sich dann sehr leicht integrieren, da mit jedem Zyklus sowohl der Analyse- als auch der Gestaltungsschritt durchlaufen wird [Bar06].

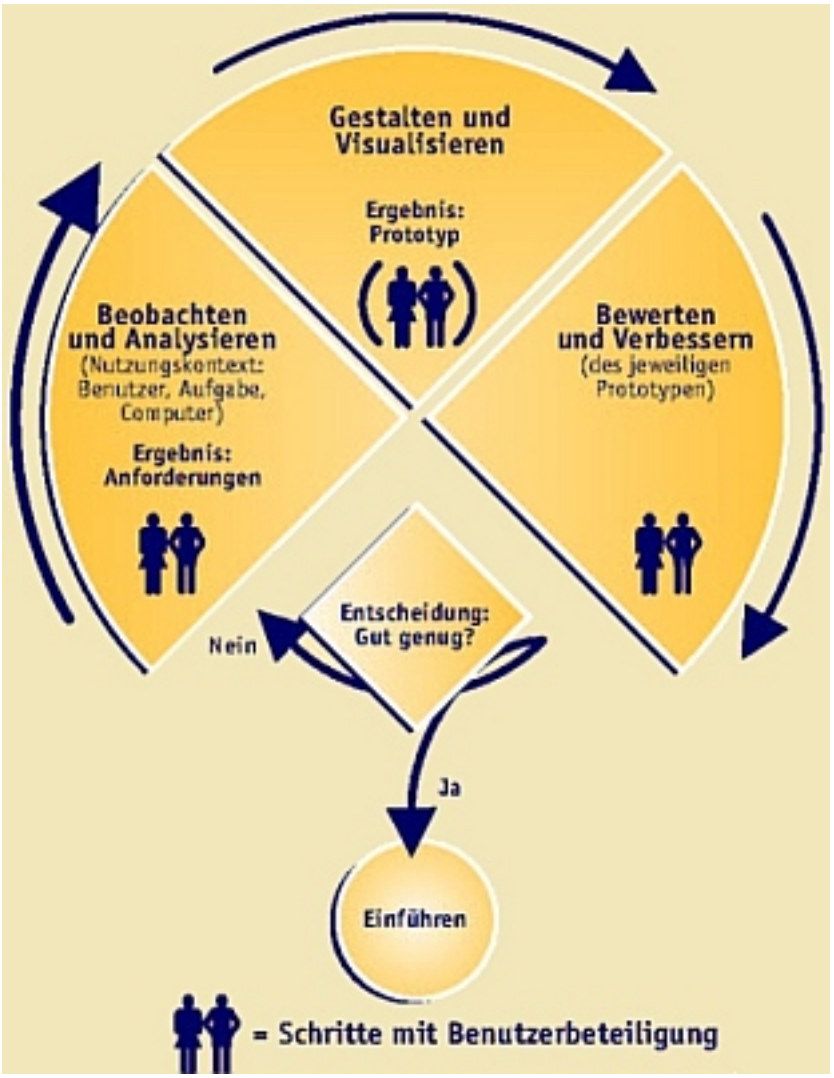


Abbildung 2.3: Evolutionäres Vorgehensmodell [Ber04]

Die Verwendung dieses Modells kann sich als sehr nachteilig erweisen, wenn eine der Kernanforderungen übersehen wurde. Dann muss nämlich ggf. die komplette Systemarchitektur überarbeitet werden. Weiterhin kann unter Umständen eine vernünftige Versionsverwaltung ernsthafte Probleme bereiten. Außerdem ist es zwingend erforderlich, die verwendeten Technologien bereits am Anfang des Projektes zu kennen [Bar06].

2.2 Zeiterfassung

Als die tägliche Arbeitszeit noch mittels Stundenzettel oder Stechuhr erfasst wurde, waren die technischen Computersysteme noch nicht so sehr verbreitet. Doch der Sinn und Zweck der Zeiterfassung hat sich seit dieser Zeit nicht geändert. Aus Sicht der Arbeitnehmer sollen die täglich geleisteten Arbeitsstunden dokumentiert sein, um lückenlos nachweisen zu können, wie lange, wann gearbeitet wurde und wieviel Lohn und Zuschläge somit zu gewähren sind. Aus Sicht der Projektleiter soll nachprüfbar sein, wie die Planung und die tatsächlich geleistete Arbeitszeit voneinander abweichen. Der Arbeitgeber will kontrollieren, wer sich nicht an die vertraglich vereinbarten Arbeitszeiten hält, um so ggf. Umsatzeinbußen aufzudecken.

2.2.1 Definition

Die Zeiterfassung, oder auch Personalzeiterfassung, ist die Erfassung der täglich geleisteten Arbeitszeiten eines jeden Beschäftigten. Sie ordnet sich in die Betriebsdatenerfassung ein, welche eine Vielzahl von Daten über Zustände und Prozesse in einem Betrieb sammelt.

Die projektbezogene Erfassung von Arbeitszeiten ist wichtigstes Element der Aufwandserfassung. Nur mit einem vollständigen Nachweis der für ein Projekt aufgewendeten Arbeitszeit kann eine sinnvolle Projektkontrolle durchgeführt werden. Dazu werden zu jedem Projektmitarbeiter die täglichen Arbeitsstunden in den jeweiligen Projekten erfasst. Mit diesen Daten lässt sich nicht nur der tatsächliche Arbeitsaufwand für geplante Vorgänge bestimmen, sondern auch ein Aufwandsnachweis gegenüber dem Auftraggeber erbringen. Außerdem können sie für eine realistische Aufwandsschätzung bei der Planung neuer Projekte herangezogen werden.

Die unternehmensbezogene Arbeitszeiterfassung bezieht sich dagegen auf das Verhalten des Mitarbeiters. Mit ihr werden z. B. die Anwesenheit am Arbeitsplatz und Überstunden erfasst. Somit besitzt sie arbeitsrechtliche Verbindlichkeit, da aus ihr Informationen über Arbeitsvertragsverletzungen ablesbar sind, welche auch Sanktionen nach sich ziehen können.

Die projektbezogene Aufwandszeiterfassung und die unternehmensbezogene Arbeitszeiterfassung stehen oft im Konflikt zueinander. Während die unternehmensbezogenen Daten die Mitarbeiter kontrollieren, werden die projektbezogenen Daten zur Bestimmung des Arbeitsaufwandes und zur Kontrolle des Projektfortschrittes verwendet [PRO07].

2.2.2 Zeiterfassungssystem

Bei Zeiterfassungssystemen handelt es sich um technische Systeme, die der Erfassung und Auswertung der Personalarbeitszeit dienen. Dazu gibt es viele verschiedene Lösungen die unter anderem mit PDAs, RFID-Transpondern, Handys, Ausweisen, Chipkarten und weiteren Erfassungsgeräten arbeiten. Neben der Erfassungsfunktion stellen diese Systeme auch eine Schnittstelle zur Auswertung der erfassten Daten zur Verfügung.

Da es sich bei einem Zeiterfassungssystem um technische Einrichtungen zur Überwachung von Verhalten und Leistung von Mitarbeitern handelt, bedarf die Einführung eines solchen Systems laut § 87 Abs. 1 Nr. 6 BetrVG der Mitbestimmung durch den Betriebsrat [Jan00].

2.3 Internetkommunikationsmodelle

Die Interaktion mit Webanwendungen entspricht dem Client/Server-Modell. Hierbei werden die beteiligten Komponenten in zwei Arten unterschieden. Der Server ist meist ein sehr leistungsstarker Computer, welcher eine Vielzahl von Diensten anbietet. Der Client ist oft ein weniger leistungsstarker Rechner, der die Dienste des Servers nutzt [Tan03].

Im Allgemeinen kommunizieren mehrere Clients über ein beliebiges Netzwerk mit dem zentralen Server (siehe Abbildung 2.4). Dieser wartet auf die Clientanfragen, welche parallel bearbeitet werden. Nach der Ausführung seiner Aufgaben schickt der Server eine Antwortnachricht mit den Ergebnissen an den jeweiligen Client zurück.

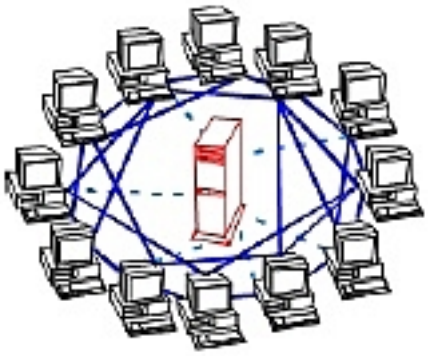


Abbildung 2.4: Kommunikationsmodell zwischen Server und Clients

2.3.1 Klassisches Modell

Bei der klassischen Kommunikation mit einer Anwendung auf einem Webserver werden die Nachrichten über das HTTP-Protokoll ausgetauscht (siehe Abbildung 2.5). Dazu wird durch die verschiedenen Benutzer eine Anfrage an den Webserver gestellt. Dieser bearbeitet dann die Anfrage, z. B. sammelt Daten aus einer Datenbank oder führt Berechnungen durch, und sendet eine HTML-Seite mit den zugehörigen CSS-Daten zum Client zurück [Gar05]. Zur Darstellung der Seite werden die Daten dann vom Browser aufbereitet.

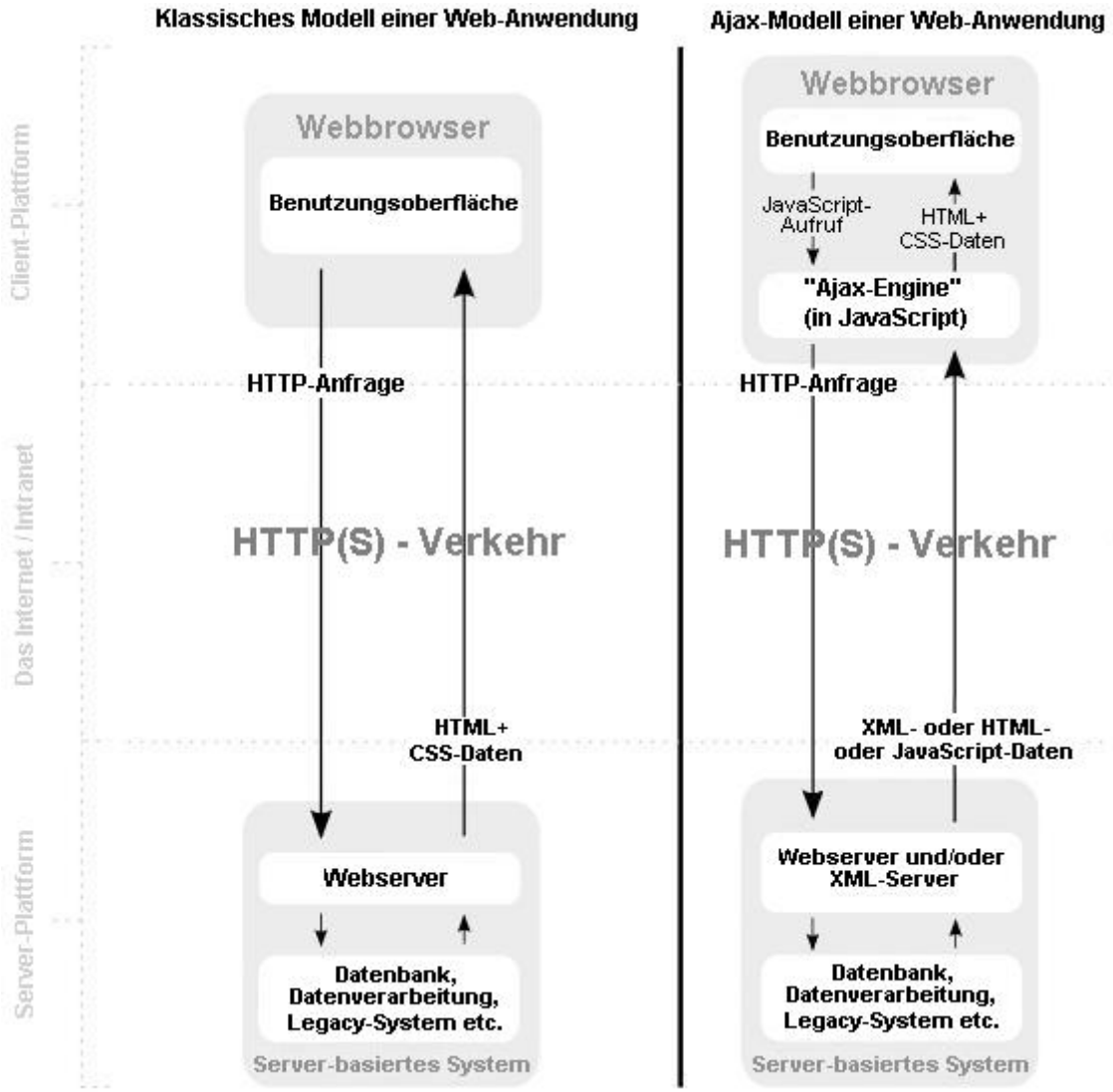


Abbildung 2.5: Vergleich der Kommunikationsmodelle (angelehnt an [Wik07a])

Der aus diesem Modell resultierende Prozessfluss (siehe Abbildung 2.6) ist absolut syn-

chron. Erst wenn die Anfrage beim Server angekommen ist, wird sie bearbeitet, und erst wenn die Abarbeitung abgeschlossen ist, wird das Ergebnis zum Client zurückgeschickt. Durch diesen festen Ablauf können lange Wartezeiten beim Client entstehen, da dieser von der Bearbeitungszeit auf dem Server abhängig ist. Außerdem werden die HTML-Seiten für den Client immer neu generiert. Wenn sich die Struktur der Seite komplett ändert, ist dies von Vorteil. Wenn die Struktur allerdings gleich bleibt und nur die Nutzdaten verändert werden, z. B. das Sortieren einer Liste nach einem anderen Kriterium, ist dieser Ablauf sehr langsam. Erst recht, wenn viele Daten angezeigt werden sollen, z. B. bei Tabellen mit mehreren 1000 Einträgen oder bei Verzeichnisbäumen mit tiefer Struktur, benötigt der Browser viel Zeit um die HTML-Seite zu rendern.

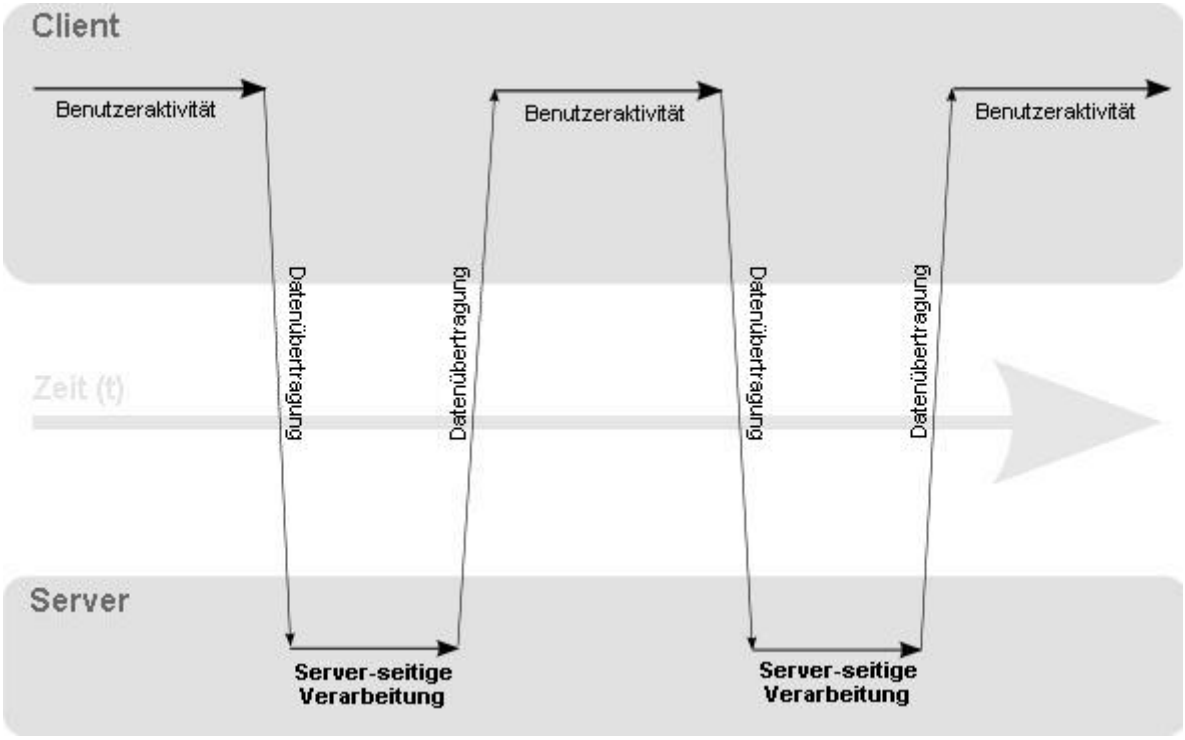


Abbildung 2.6: Prozessfluss einer traditionellen Webanwendung (angelehnt an [Wik07c])

2.3.2 Ajax-Modell

Ajax ist eine Abkürzung für die Wortfolge „Asynchronous JavaScript and XML“ und bezeichnet ein Konzept zur asynchronen Datenübertragung zwischen Server und Browser, welches es ermöglicht, HTTP-Anfragen innerhalb einer HTML-Seite durchzuführen, ohne die Seite komplett neu laden zu müssen.

„Ajax isn't a technology. It's really several technologies, each flourishing in its own right, coming together in powerful new ways." [Gar05]

Jesse James Garrett[2] definiert Ajax nicht als Technologie in dem Sinne, sondern beschreibt es als einen neuen Ansatz, verschiedene, vorhandene Technologien zu nutzen wie HTML oder XHTML, CSS, JavaScript, DOM, XML, XSLT und das XMLHttpRequest Objekt [Gar05]. Wenn diese Technologien zum Ajax-Modell kombiniert werden, können Webanwendungen schnell und einfach die HTML-Seiten im Browser verändern.

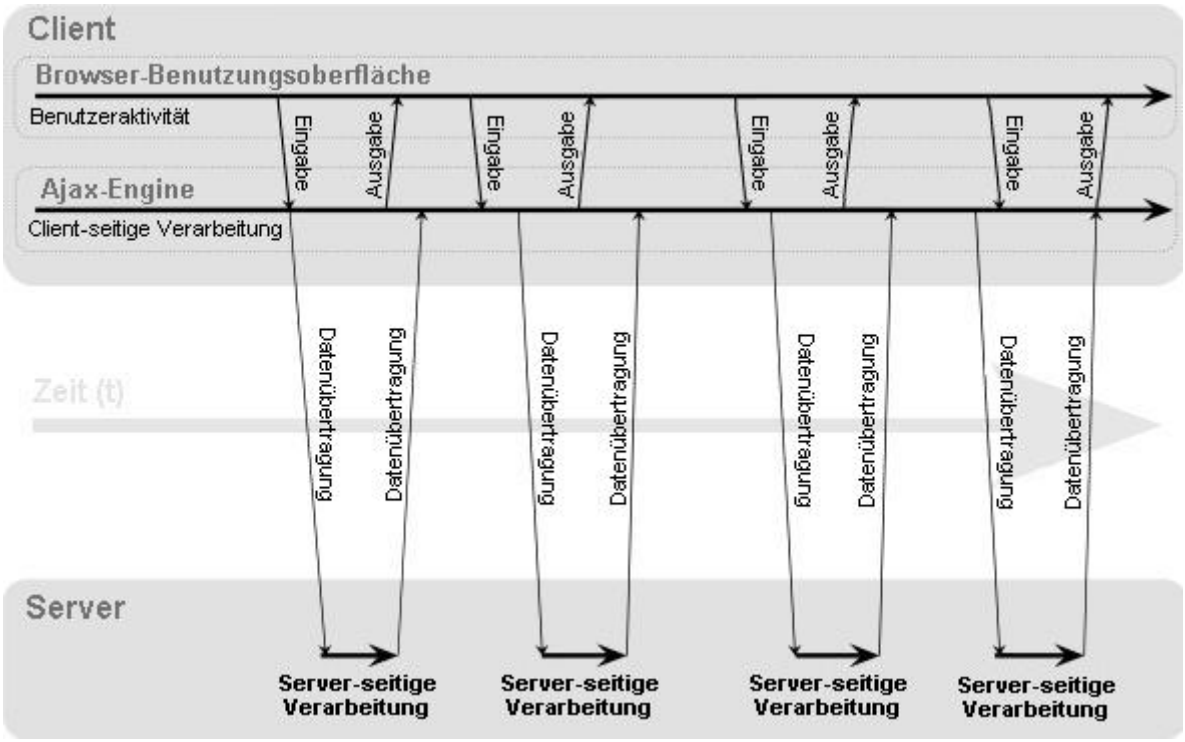


Abbildung 2.7: Prozessfluss einer Ajax-Webanwendung (angelehnt an [Wik07b])

Mit Hilfe von Ajax verliert die Interaktion mit Webapplikationen ihren Start-Stop-Start-Stop Charakter. Dafür wird zwischen Server und Client eine zusätzliche Schicht eingeführt (siehe Abbildung 2.5). Die Ajax-Engine wird beim Start der Sitzung geladen und übernimmt nun sowohl die Server-Client-Kommunikation als auch die Darstellung der Daten im Browser. Nun muss nur der aktuell notwendige Seiteninhalt beim Server angefordert werden. Um die Seite mit weiteren Daten zu füllen wird sie nicht neu geladen, sondern erhält per XMLHttpRequest die Daten und fügt sie per JavaScript in das DOM ein [Gar05]. Dies ist vor allem bei großen Datenmengen sehr sinnvoll. So muss nicht die komplette Struktur eines Baums geladen werden, sondern nur die obersten Knoten. Mittels Ajax ist es dann möglich, sukzessive die

[2] Präsident und einer der Gründer von Adaptive Path

jeweiligen Kindknoten nachzuladen und anzuzeigen. Ein weiterer Vorteil ist, dass der Benutzer kaum wahrnimmt, dass eine Kommunikation mit dem Server stattgefunden hat, da dies im Hintergrund geschieht und keine Wartezeiten mit leeren Browserfenstern entstehen.

Bei der Betrachtung des Prozessflusses (siehe Abbildung 2.7) wird sofort der Unterschied zum klassischen Kommunikationsmodell klar. Hat eine Benutzeraktion normalerweise einen HTTP-Request zur Folge, wird bei Ajax stattdessen ein JavaScript-Aufruf an die Ajax-Engine weitergeleitet. Kann diese auf eine Benutzeraktion reagieren, ohne Daten beim Server anzufrage, z. B. einfache Plausibilitätsprüfungen oder Datenvalidierung, setzt sie die Aktion direkt um. Wenn allerdings Daten vom Server benötigt werden oder dort Aktionen ausgeführt werden müssen, z. B. Interaktion mit einer Datenbank oder umfangreiche Plausibilitätsprüfungen, geschieht dies asynchron mittels XMLHttpRequest, ohne dass die Interaktion zwischen Benutzer und Anwendung gestört wird. Der Server erhält eine Anfrage, bearbeitet diese und sendet die Ergebnisse per XML an den Client zurück [Gar05].

Ajax bringt allerdings auch einige Nachteile mit sich. Unter anderem ist die Funktionalität der „Zurück“-Schaltfläche nicht mehr korrekt gewährleistet. Somit besteht immer die Gefahr, dass beim Klick auf diese Schaltfläche nicht der vorherige Anwendungszustand wieder hergestellt werden kann. Ein weiteres Problem ist, dass der Server keine asynchronen Ereignisse beim Client auslösen kann. Insofern ist es nicht möglich, einen Client zu informieren, wenn sich seine aktuellen Nutzdaten verändert haben. Um dies aber zu gewährleisten muss der Client mittels Polling ständig beim Server eventuelle Veränderungen verfolgen.

3 Analyse

Die Analyse dient der Feststellung der Anforderungen an das Zeiterfassungssystem. Um diese zu bestimmen, werden existierende Systeme untersucht und hinsichtlich ihres Funktionsumfangs und ihrer Ergonomie bewertet. Die Ergebnisse dieser Untersuchung fließen dann in die Bestimmung der Anforderungen an das System ein. Dabei sollen die allgemeinen Anforderungen den grundlegenden Funktionsumfang der Software festlegen. Die Anforderungen aus software-ergonomischer Sicht werden für diesen speziellen Anwendungsfall, einem webbasierten Zeiterfassungssystem, ermittelt.

3.1 Untersuchung existierender Systeme

Auf dem Softwaremarkt existiert bereits eine Vielzahl kommerzieller und nicht kommerzieller Zeiterfassungssysteme. Da eine vollständige Untersuchung aller Systeme nicht möglich ist, werden im Folgenden drei dieser existierenden Systeme aus dem nicht kommerziellen Bereich untersucht und bewertet. Hierbei ist festzustellen, inwiefern die wesentlichen Anforderungen an ein Zeiterfassungssystem umgesetzt wurden und wie sehr die Software-Ergonomie beachtet wurde. Es wird in allen drei Fällen eine Online-Demo getestet, um lokale Installationen zu vermeiden.

3.1.1 Gleit-Zeit.de

Programm:	Gleit-Zeit.de
Version:	12.11.2004
Webseite:	http://www.gleit-zeit.de/
Online-Demo:	http://www.gleit-zeit.de/gz
Programmiersprache:	PHP
Datenbank:	MySQL

Bei diesem webbasierten Zeiterfassungssystem handelt es sich um eine einfache und kostenlose Gleitzeit-Software zur Erfassung von Arbeits-, Krankheits- und Urlaubszeiten. Es ist für kleine bis mittelgroße Betriebe bis maximal 9999 Mitarbeitern ausgelegt. Die gesammelten Daten können mittels enthaltener Listen-Funktionen ausgewertet werden.

Die jeweilige Erfassung der Zeit startet, wenn sich ein Benutzer am System anmeldet. Dann wird der aktuelle Zeitpunkt als Arbeitsbeginn notiert. Nach Beendigung der Arbeit muss sich der Benutzer noch vom System abmelden, um den Endzeitpunkt zu notieren.

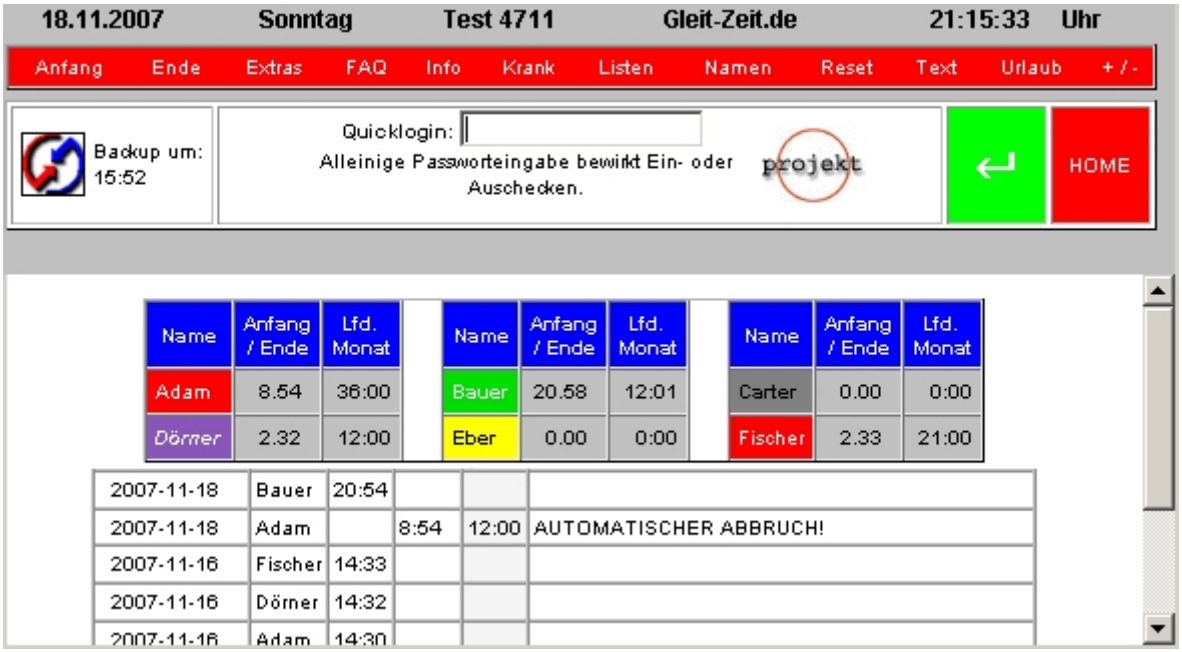


Abbildung 3.1: Benutzungsoberfläche vom Zeiterfassungssystem Gleit-Zeit.de

Alles in allem wirkt die Programmoberfläche sehr unstrukturiert und nicht wohl durchdacht. Das Menü ist überfüllt und unübersichtlich gestaltet, außerdem sind die Menüpunkte nicht aussagekräftig. Es werden auch keine Tooltips angezeigt, was es unmöglich macht die Funktionen der Links und Knöpfe zu erfahren. Weiterhin ist die Wahl der Farben nicht von Vorteil für den Benutzer. Es wird sehr viel rot verwendet, was eigentlich eine Warnfarbe ist. Ansonsten werden auch viele weitere Farben verwendet, was die Benutzungsoberfläche sehr bunt und unruhig erscheinen lässt (siehe Abbildung 3.1). Es ist auch nicht ersichtlich, welche Bedeutung die jeweiligen Farben haben.

Weiterhin werden Funktionen angeboten, die keine Auswirkung haben. So können z. B. Daten exportiert werden. Allerdings passiert beim Export nichts, außer dass die Liste der Daten angezeigt wird. Hier wird jedoch eine Datei im CSV- oder XML-Format erwartet.

Abschließend ist die Benutzungsoberfläche als sehr unergonomisch zu bewerten. Es sind vor allem starke Defizite in der Erwartungskonformität und Selbstbeschreibungsfähigkeit vorhanden. Eine Verwendung dieses Programms ist unvorstellbar und absolut nicht zu empfehlen.

3.1.2 Timeanchor

Programm:	Timeanchor
Version:	2.7
Webseite:	http://www.timeanchor.com
Online-Demo:	http://www.timeanchor.com/Timeanchor_Online_Demo.jsp
Programmiersprache:	Java
Datenbank:	MySQL, DB2, Oracle, Microsoft SQL Server, Sybase

Timeanchor ist eine webbasierte Zeit-, Projekt- und Aktivitätserfassungsanwendung. Die Software ist sowohl als Java Webstart oder als Applet startbar, was die klassische Bedienbarkeit einer Rich-Client-Anwendung bietet.

Die Erfassung der Zeiten erfolgt Stil von Microsoft Excel, wobei Summen dynamisch berechnet werden. Neben dem direkten Eintragen der Stunden für ein Projekt, kann der Benutzer auch einen Timer starten. Beim Stoppen der Uhr können dann das Projekt und die Aktivität zugeordnet werden. Die Bedieneroberfläche ist gut strukturiert und wirkt dadurch sehr übersichtlich (siehe Abbildung 3.2). Durch Minimieren oder Schließen von Übersichtsfenstern kann der Benutzer seinen Arbeitsablauf gut steuern.

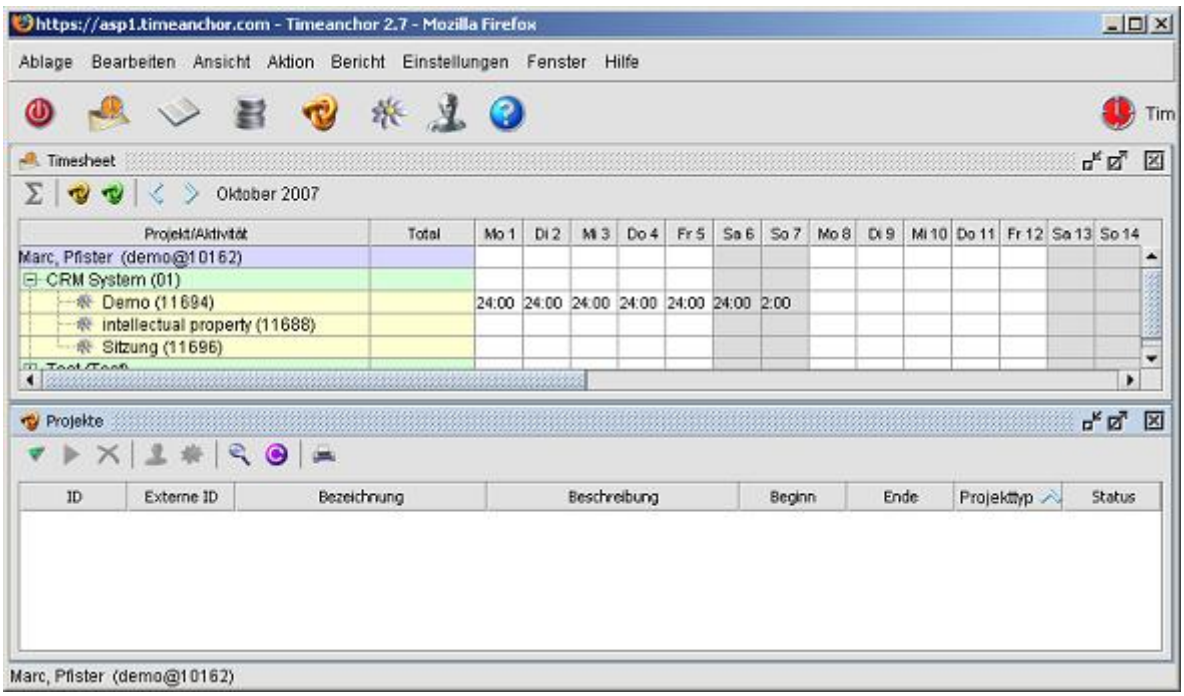


Abbildung 3.2: Benutzungsoberfläche vom Zeiterfassungssystem Timeanchor

Die Struktur der Menüleiste entspricht der von Standard Windows Applikationen, was der Erwartungskonformität zugute kommt. Außerdem sind alle Funktionen klar erkennbar und

werden mittels Tooltips kurz erklärt. Steht eine Funktion im aktuellen Kontext nicht zur Verfügung, so wird sie im Menü deaktiviert.

Eine umfangreiche Berichtsfunktion ermöglicht es neben den Arbeitszeiten auch die Aktivitäten, Projektbudgets oder Kosten auszuwerten. Diese Daten können dann auch in verschiedenen Formaten exportiert werden.

Neben dem sehr großen Funktionsumfang und der übersichtlichen Oberfläche gibt es allerdings auch einige negative Punkte in dieser Software. So wird z. B. die Hilfe in einem neuen Browserfenster geöffnet. Dadurch muss der Anwender das Programm verlassen, um sich Hilfestellungen zu einzelnen Funktionen zu holen. Weiterhin sind einige verwendete Bilder nicht selbsterklärend. Eine regenbogenfarbene Blüte steht für Aktivitäten. Dies entspricht sicher nicht der Erwartung eines Benutzers. Der Fehlertoleranz wurde auch recht wenig Beachtung geschenkt. Wird eine Fehlermeldung angezeigt, erscheint dort lediglich kryptischer Programmcode, den wahrscheinlich nur die Entwickler des Systems interpretieren können.

Alles in allem ist Timeanchor aber gut bedienbar. Die vielen Funktionen wirken sich nicht nachteilig auf die Übersichtlichkeit aus. Der Einsatz dieser Software ist sehr zu empfehlen, wenn das Preis-/Leistungsverhältnis den Vorstellungen eines Unternehmens entspricht.

3.1.3 Timesheet

Programm:	Timesheet.Php
Version:	1.2.1
Webseite:	http://www.timesheetphp.com
Online-Demo:	http://www.timesheetphp.com/timesheet/login.php
Programmiersprache:	PHP
Datenbank:	MySQL

Bei Timesheet.Php handelt es sich um eine einfache webbasierte Anwendung zur Erfassung von Projektarbeitszeiten. Dabei werden die Arbeitszeiten der Mitarbeiter einer bestimmten Aktivität in einem Projekt zugeordnet.

Die Benutzungsoberfläche ist übersichtlich und gut strukturiert. Das Menü dagegen ist nicht sofort erkennbar und wirkt auch sehr ungeordnet. Die verwendeten Farben sind einheitlich und wirken nicht aufdringlich (siehe Abbildung 3.3). Bei der Benutzung farbiger Hervorhebungen wurde allerdings rot für die Darstellung von Summen verwendet. Da dies aber eigentlich eine Warnfarbe ist, besteht hier ein Problem bei der Anzeige von Fehlern.

Der angebotene Funktionsumfang ist nicht sehr groß, bietet aber trotzdem viele Möglichkeiten. So kann der Benutzer nach der Anmeldung z. B. einen Timer starten. Wenn er diesen

Abbildung 3.3: Benutzungsoberfläche vom Zeiterfassungssystem Timesheet.Php

stoppt, kann die entsprechende Arbeitszeit einer Projektaktivität zugeordnet werden. Die gearbeitete Zeit kann allerdings auch in einer kompakten Wochenansicht eingetragen werden. Dort werden nach der Auswahl von Projekt und Aktivität die Stunden und Minuten pro Wochentag eingetragen.

Die Benutzung der Tagesansicht ist allerdings sehr kompliziert. Hier dauert das Eintragen sehr lange, da erst der Kunde, dann das Projekt und dann die Aktivität ausgewählt werden müssen. Danach werden die Startzeit und die Endzeit gewählt. Abschließend muss noch ein Text als Notiz für die Tätigkeit eingetragen werden. Sollte nun die Arbeit an verschiedenen Tagen beginnen und enden (z. B. bei Nachtarbeit) müssen dazu 2 Einträge getätigt werden, da die 00:00 Uhr Grenze nicht überschritten werden kann. Ein weiteres Problem ist die Darstellung der Monatsansicht. Diese braucht sehr lange um die Seite im Browser aufzubauen. Die Wartezeit für den Seitenaufbau wächst, je mehr Einträge pro Tag und Monat vorhanden sind.

Eine Benutzung dieser Software ist vorstellbar, wenn auch auf einigen Komfort verzichtet werden muss. Ein ergonomisches Arbeiten ist aber nur bei einigen Funktionen gegeben.

3.1.4 Zusammenfassung der Untersuchung

Zusammenfassend wurde festgestellt, dass bei allen drei Systemen ein rollenbasiertes Rechtesystem verwendet wurde. Es gibt mindestens Administratoren und Mitarbeiter, meist sogar noch andere Rollen. Ihnen stehen auf Grund ihrer jeweiligen Rechte mehr oder weniger Funktionen zur Verfügung. Zu den grundlegenden Funktionen gehören die Zeiterfassung, die Auswertung und die Systemadministration. Diese sind beliebig komplex. So kann es bei der Erfassung der Arbeitszeiten verschiedene Sichten (Monats- und Wochensicht) geben. Zum Auswerten der Daten werden meist Projektübersichten angeboten. Es gibt aber auch die Möglichkeit Soll-/Ist-Vergleiche zu machen. Im Allgemeinen gehören zur Administration des Systems die Personen- und Projektverwaltung.

Das Fazit dieser Untersuchung ist, dass es nicht auf die Programmiersprache, die Datenbank der den Preis ankommt, wenn eine Software ergonomisch sein soll. Die Gestaltung der Benutzungsoberfläche und der Arbeitsabläufe hat maßgeblichen Einfluss auf die Ergonomie einer Anwendung.

3.2 Anforderungsanalyse

Basierend auf der Untersuchung aus Kapitel 3.1 werden die Anforderungen an das hier zu entwickelnde Zeiterfassungssystem bestimmt. Dabei wird zwischen den allgemeinen Anforderungen an das System und den Anforderungen aus software-ergonomischer Sicht unterschieden.

3.2.1 Allgemeine Anforderungen

Die allgemeinen Anforderungen betreffen die grundlegenden Funktionen des Zeiterfassungssystems, welche zur Erledigung der Aufgaben notwendig sind, und seine Eigenschaften. Dazu werden neben der Bestimmung der funktionalen und nichtfunktionalen Anforderungen auch die Benutzerrollen und ihre Anwendungsfälle identifiziert.

3.2.1.1 Funktionale Anforderungen

Mit den funktionalen Anforderungen werden die Systemfunktionen bezeichnet, die dem Benutzer zur Interaktion mit der Software zur Verfügung stehen. Sie werden in 3 Prioritätsklassen eingestuft. Anforderungen die das System unbedingt anbieten muss, werden als „must“ klassifiziert, Funktionen die umgesetzt werden sollten als „should“ und alle übrigen,

die vorhanden sein können als „nice to have“. Dazu werden die Anforderungen in 4 Gruppen aufgeteilt, welche die Kernfunktionalitäten des Zeiterfassungssystems darstellen.

Zeiterfassung

Die Zeiterfassung muss dem Benutzer grundlegend die Möglichkeit bieten seine geleistete Arbeitszeit für die jeweiligen Projekte ins System einzutragen.

Eintragen (must)

Zum Eintragen der täglichen Arbeitsstunden muss der Benutzer eine Liste erhalten, in der er das Projekt und den entsprechenden Tag auswählen kann. Die dann einzugebenden Werte dürfen nur positive rationale Zahlen zwischen 0 und 24 mit maximal 2 Kommastellen sein. So ist gewährleistet, dass es sich um sinnvolle Zeiten handelt, die bis auf eine viertel Stunde genau sind.

Komfortables Eintragen (should)

Um nicht mit der Tastatur arbeiten zu müssen, kann dem Benutzer eine komfortable Funktion zum Eintragen der Arbeitsstunden angeboten werden. Diese kann sich z. B. in Form von „+“ und „-“ Schaltflächen oder eines Schiebereglers darstellen. Für das Eintragen eines Datums muss dem Benutzer ein Kalender zum Auswählen eines Tags zur Verfügung stehen.

Verwendung eines Timers (nice to have)

Um die gearbeitete Zeit nicht selbst berechnen zu müssen, kann der Benutzer einen Timer starten. Nach Beendigung der Messung muss die ermittelte Zeit einem Projekt zugeordnet werden können.

Änderung der Sicht (must)

Die angebotene Liste zum Eintragen der Arbeitszeit soll auf verschiedene Sichten reduziert werden können. Dazu stehen ein Monat, eine Woche oder nur ein Tag zur Auswahl zur Verfügung.

Summieren der Zeilen und Spalten (must)

Nach der Eingabe der Arbeitszeiten in die jeweiligen Felder müssen sowohl die zugehörige Zeile als auch die entsprechende Spalte aufsummiert werden. Somit wird dem Benutzer angezeigt, wie viel Stunden er im angezeigten Zeitraum in seinen Projekten gearbeitet hat und wie viele Arbeitsstunden an jedem Tag angefallen sind.

Auswertung

Die Auswertung muss es ermöglichen die gesammelten Daten in aufbereiteter Form auszugeben. So soll jeder einzelne Mitarbeiter seine eigenen Stunden auswerten können. Mittels dieser Funktion wird einem Projektleiter die Möglichkeit gegeben, sich einen Überblick über den bisher geleisteten Aufwand in seinem Projekt zu verschaffen.

Listen (must)

Die Ausgabe der Daten in Listenform dient der Auswertung und Zusammenfassung. Mit entsprechender Filterfunktion soll die Ausgabe auf einen bestimmten Zeitraum beschränkt werden können.

Datenexport (nice to have)

Um die erfassten Daten in anderen Anwendungen nutzen zu können, kann eine Exportfunktion angeboten werden. Diese fasst die Daten für einen bestimmten Zeitraum zusammen und stellt sie z. B. als CSV- oder XML-Datei bereit.

Diagramme (nice to have)

Die Ausgabe als Diagramme dient der Visualisierung der Daten. So können z. B. Projekte gegenübergestellt und verglichen werden. So werden auch Arbeitsanteile oder Tendenzen besser ersichtlich.

Einstellungen

Mittels Einstellungen kann sich der Benutzer seine Arbeitsoberfläche individualisieren. So ist es ihm möglich, die für ihn komfortabelste Umgebung einzurichten.

Standardansicht (must)

Das Einstellen der Standardansicht bezieht sich auf die Zeiterfassung. So kann der Benutzer wählen, ob dort bevorzugt nur ein Tag, eine Woche oder gar ein ganzer Monat angezeigt werden soll.

Projektsichtbarkeit (must)

Die Auswahl sichtbarer Projekte hat für den Benutzer Auswirkungen auf die Zeiterfassung. So können Projekte, an denen im Moment nicht oder überhaupt nie mehr gearbeitet wird, deaktiviert werden. Diese Projekte werden dann nicht mehr angezeigt, was die Liste auf die persönlich wichtigsten Einträge reduziert.

Projektordnung (must)

Mit der Angabe der Projektordnung wird die Reihenfolge der Projekte in der Liste der Zeiterfassung bestimmt. So kann das Projekt, an welchem im Moment gearbeitet wird, in der Liste ganz oben stehen. Genauso können weniger wichtige Projekte nach unten verschoben werden. Durch diese individuelle Ordnung hat der Benutzer einen schnelleren Zugriff.

Vorgaben (nice to have)

Mit Hilfe von Vorgaben kann ein Benutzer Projektarbeitszeiten für einen bestimmten Zeitraum festlegen. Das ist dann sehr sinnvoll, wenn die Personen immer pünktlich mit der Arbeit beginnen und aufhören, oder für eine gewisse Dauer täglich die selben Arbeitsstunden leisten.

Verwaltung

Zur Administration und Steuerung des Zeiterfassungssystems ist eine Verwaltung notwendig. Dort werden die Ressourcen hinzugefügt, bearbeitet oder entfernt.

Benutzerverwaltung (must)

Um die Systembenutzer zu verwalten, müssen Personen sowohl ins System aufgenommen als auch aus dem System gelöscht werden können. Beim Löschen ist zu beachten, was mit den eingetragenen Arbeitszeiten und den Zugangsdaten der Personen geschieht.

Administratorenverwaltung (must)

Um das Zeiterfassungssystem zu verwalten wird mindestens ein Administrator benötigt. Es sollen aber mehrere Personen das Verwaltungsrecht besitzen. So ist die Anwendung nicht allein von einer Person abhängig. Es kann z. B. in jeder Abteilung eines Betriebs einen Verantwortlichen und einen Stellvertreter für das Zeiterfassungssystem geben.

Projektverwaltung (must)

Da die eingetragenen Zeiten der Mitarbeiter jeweils Projekten zugeordnet werden sollen, müssen diese im System neu angelegt werden können. Neben dem Ändern des Namens muss aber auch ein Projektleiter festlegbar sein. Dabei ist wichtig, dass Projekte nie ohne Projektleiter existieren dürfen. Ein Löschen von Projekten muss möglich sein, wobei auf den Verbleib der bisherigen Projektzeiten und deren Abruf zu achten ist.

Projektmitgliederverwaltung (must)

Ein Projektleiter soll die Mitglieder in seinen Projekten verwalten können. Dazu muss er Systembenutzer den einzelnen Projekten zuordnen können. Beim Auflösen der Mitgliedschaft ist zu beachten, wie mit den Arbeitsstunden dieser Person für ein Projekt zu verfahren ist.

3.2.1.2 Nichtfunktionale Anforderungen

Mit den nichtfunktionalen Anforderungen werden die Eigenschaften des Zeiterfassungssystems definiert. Sie beeinflussen ganz wesentlich das Systemkonzept und seine Umsetzung.

Bedienbarkeit

Da es bei Software-Ergonomie auf die Bedienbarkeit einer Anwendung ankommt, ist dies die wichtigste nichtfunktionale Anforderung. In Kapitel 3.2.2 werden dazu die Anforderungen zu den einzelnen Bewertungskriterien aufgeführt.

Systemunabhängigkeit

Da die Zielsoftware ein webbasiertes Softwaresystem sein soll, muss sie unabhängig von Hardware und Betriebssystem ablaufen können. Durch den Einsatz von Webservern und Datenbanksystem, die betriebssystemunabhängig sind, ist so eine leichte Integration in eine bestehende Umgebung möglich.

Vertraulichkeit

Um die Vertraulichkeit des Systems sicherzustellen, dürfen Mitarbeiter nur ihre eigenen Arbeitszeiten angezeigt bekommen. Es muss ihnen unmöglich gemacht werden, Zeiten von Kollegen einzusehen, oder gar zu verändern. Bei der Datenauswertung dürfen die Projektleiter nur ihre eigenen Projekte einsehen. Die Auswertung fremder Projekte ist zu verbieten. Um die Überwachung von Mitarbeitern zu vermeiden, sollte ein Projektleiter auf die geleisteten Arbeitsstunden einzelner Projektmitglieder keinen Zugriff haben.

Weitere Anforderungen

Auf weitere nichtfunktionale Anforderungen wie Qualität, Performanz, Skalierbarkeit, Verfügbarkeit oder Erweiterbarkeit kann in einem webbasierten Softwaresystem eigentlich nicht verzichtet werden. Da diese Anforderungen aber weniger zu einer ergonomischen Software beitragen, werden sie im Rahmen dieser Arbeit kaum beachtet.

3.2.1.3 Benutzerrollen und Anwendungsfälle

Jeder Benutzer des Zeiterfassungssystems gehört einer oder mehrerer Rollen an. Diese können Mitarbeiter, Projektleiter oder Administrator sein. Die verschiedenen Aktivitäten der einzelnen Benutzerrollen basieren auf den folgenden Anwendungsfalldiagrammen.

Mitarbeiter

Zu den Mitarbeitern gehören die Systembenutzer, welche in den verschiedenen Projekten eines Betriebs tätig sind. Sie haben keine speziellen Verpflichtungen oder Privilegien für Sonderaufgaben in diesem Anwendungskontext.

Sie können ihre Arbeitszeit für die jeweiligen Projekte eintragen und verändern. Weiterhin ist es ihnen möglich, ihre eigenen Arbeitszeiten auszuwerten. Mit dem Anpassen der persönlichen Einstellungen kann jeder Mitarbeiter sowohl auf seine Arbeitsobfläche als auch auf seinen Arbeitsablauf Einfluss nehmen.

Abbildung 3.4 zeigt die identifizierten Anwendungsfälle für die Benutzerrolle Mitarbeiter im Zeiterfassungssystem.

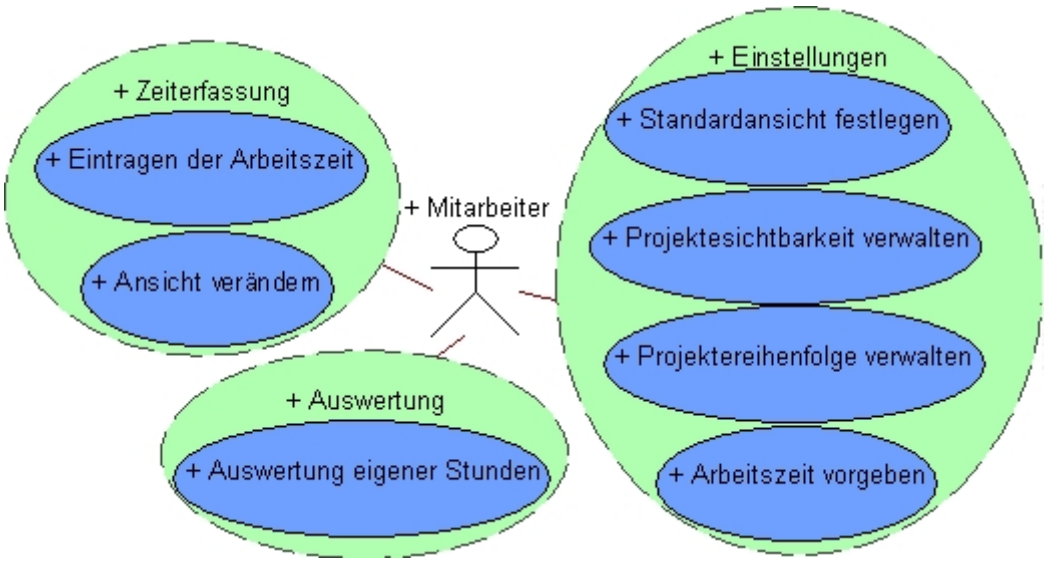


Abbildung 3.4: Anwendungsfalldiagramm für die Benutzerrolle Mitarbeiter

Projektleiter

Ein Projektleiter übernimmt primär die operativen Aufgaben der Steuerung und Planung in einem Projekt. Neben dem Erreichen der Ziele ist er auch für die Einhaltung der Termine und Fristen im Rahmen der Projektdurchführung verantwortlich. Um diese speziellen Aufgaben ausführen zu können, hat er im Zeiterfassungssystem die Möglichkeit, die Arbeitsstunden

einzelner Projekte oder auch Mitarbeiter auszuwerten. Außerdem werden ihm Funktionen zur Verwaltung der Projektmitglieder sowie zur Umbenennung seines Projektes zur Verfügung gestellt.

Abbildung 3.5 zeigt die identifizierten Anwendungsfälle für die Benutzerrolle Projektleiter im Zeiterfassungssystem.

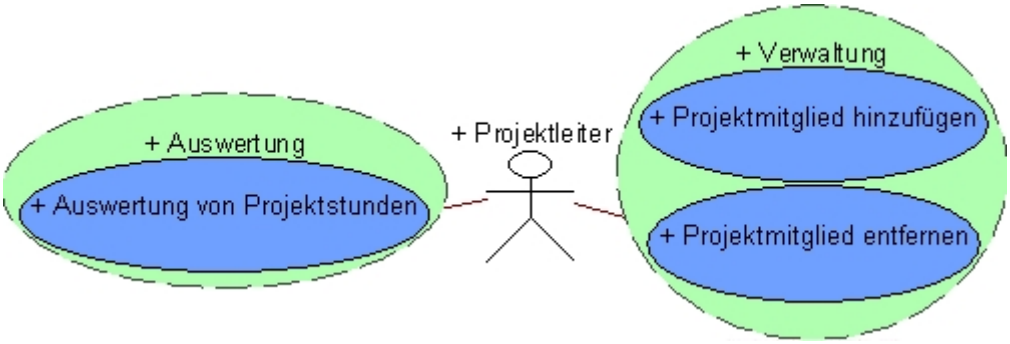


Abbildung 3.5: Anwendungsfalldiagramm für die Benutzerrolle Projektleiter

Administratoren

Administratoren sind Personen mit erweiterten Rechten. Diese privilegierten Personen übernehmen die allgemeine Verwaltung des interaktiven Softwaresystems. In dieser Rolle verwalten sie sowohl die Benutzer des Systems als auch alle weiteren Administratoren. Zur Verwaltung von Projekten können sie neue anlegen sowie bestehende umbenennen oder entfernen. Außerdem kann der jeweilige Projektleiter festgelegt werden.

Abbildung 3.6 zeigt die identifizierten Anwendungsfälle für die Benutzerrolle Administrator im Zeiterfassungssystem.

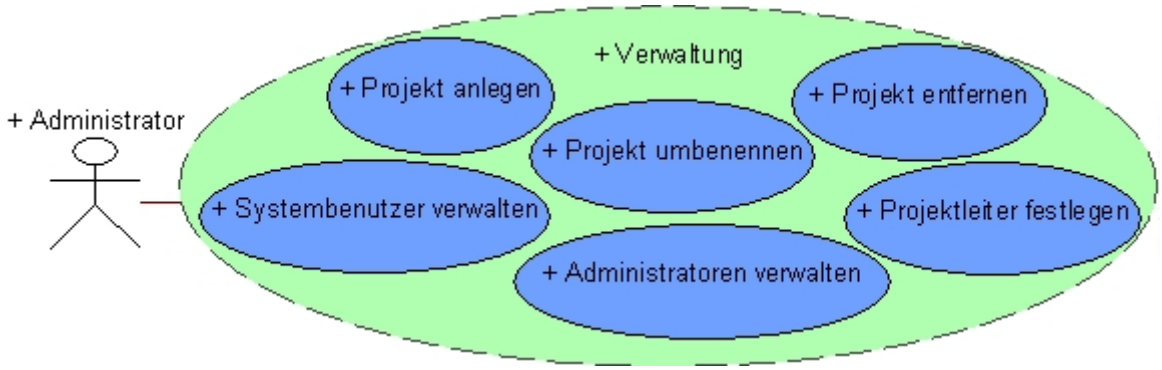


Abbildung 3.6: Anwendungsfalldiagramm für die Benutzerrolle Administrator

3.2.2 Software-ergonomische Anforderungen

Auf die software-ergonomischen Anforderungen wird im Rahmen dieser Arbeit das Hauptaugenmerk gelegt, da sie maßgeblich zur Bedienbarkeit des Zeiterfassungssystems beitragen. Im Folgenden werden die speziellen Anforderungen an das zu entwerfende System den einzelnen Bewertungskriterien (siehe Kapitel 2.1.2) der Software-Ergonomie zugeordnet.

Aufgabenangemessenheit

Um ein effektives und effizientes Arbeiten zu gewährleisten, soll das System den Benutzer bei der Erledigung seiner Arbeit unterstützen.

Im Fall eines Zeiterfassungssystems ist der aktuelle Tag eine wesentliche Information, die dem Anwender angezeigt werden muss. Dazu sollte in der Benutzungsoberfläche ein fester Bereich definiert sein, in dem immer das Datum zu sehen ist. Weiterhin muss der aktuelle Tag in der Zeiterfassungsliste visuell hervorgehoben werden. So kann sich der Benutzer leichter orientieren. Eine andere farbliche Formatierung muss für die Wochenenden benutzt werden. Dies trägt auch zu einer besseren Orientierung bei.

Zusätzlich ist der Cursor in dem Feld zu positionieren, wo die Arbeitsstunden für den aktuellen Tag im ersten Projekt eingetragen werden können. Wird vom Anwender ein fehlerhafter Wert eingegeben, z. B. ein Buchstabe oder eine Zahl größer als 24, dann muss die letzte Aktion verworfen und der Cursor im Fehlerfeld gesetzt werden. In allen anderen Bereichen ist ebenfalls das erste Bearbeitungsfeld nach dem Laden des Formulars zu fokussieren.

Außerdem muss dafür gesorgt werden, dass persistente Daten automatisch ausgelesen oder in die Datenbank geschrieben werden. Hier kann der Benutzer nicht ständig darauf achten, dass dies manuell geschieht.

Selbstbeschreibungsfähigkeit

Ein selbstbeschreibungsfähiges System zeigt dem Benutzer immer direkt an, ob seine Aktionen getätigt wurden oder nicht.

Das bedeutet, dass Fehler sofort und in der Sprache des Anwenders dargestellt werden. Nach Abschluss einer erfolgreichen Aktion muss nicht unbedingt eine Meldung gezeigt werden. Vielmehr muss nach der Eingabe einer Arbeitszeit, diese auch weiterhin in dem Feld zu finden sein. Genauso verhält es sich z. B. mit dem Anlegen von Projekten. Nach dem Hinzufügen eines neuen Projektes, sollte dieses auch unmittelbar in der Liste der Projekte zu sehen sein.

Die Selbstbeschreibungsfähigkeit einer Software sorgt aber auch dafür, dass dem Anwender stets Informationen bezüglich des aktuellen Interaktionsschritts zur Verfügung gestellt werden. So sind die einzelnen Eingabefelder in der Zeiterfassung mit kurzen Hinweistexten zu

versehen, anhand derer sich ein Benutzer informieren kann, für welches Projekt und welchen Tag dieses Feld steht. Außerdem sollen umfangreiche Hinweise eingeblendet werden, die Auskunft über die angezeigten Elemente und auszuführenden Aufgaben geben.

Bei der Beschriftung von Menüpunkten und Schaltflächen ist darauf zu achten, dass sie ihre Funktion eindeutig erklären. Unterstützend können dazu auch kurze Hinweistexte gezeigt werden.

Steuerbarkeit

Ein steuerbares System bietet dem Anwender jederzeit die Möglichkeit die Geschwindigkeit und Richtung des Dialogablaufs zu bestimmen.

Dies ist bei einer Webanwendung allerdings nicht sehr einfach, da Daten, die in Formularen eingetragen wurden, nicht automatisch gespeichert werden. So gehen die Daten verloren, sobald das Formular aktualisiert oder verlassen wird. Durch den Einsatz geeigneter Mittel ist also sicherzustellen, dass Daten sofort gespeichert werden. So kann zu jedem Zeitpunkt gewährleistet werden, dass ein Benutzer eine Aktion unterbricht, um diese nach anderen Tätigkeiten fortzusetzen, ohne einen Verlust von Daten hinnehmen zu müssen.

Ein weiterer Aspekt der Steuerbarkeit ist das Anbieten von alternativen Eingabeformen. Sind verschiedene Arten der Eingabe vorgesehen, muss der Benutzer frei wählen können welche er verwendet. Im Bereich Zeiterfassung betrifft dies die Benutzung des Nummernblocks der Tastatur oder der Schaltflächen in der Oberfläche.

Zusätzlich wäre es wichtig eine Funktion zur Verfügung zu stellen, mit welcher der Benutzer seine Aktionen rückgängig machen kann. Dies ist in Webanwendungen allerdings sehr schwierig, da dazu eine Protokollierung der Dialogschritte notwendig ist. Es wird also ein komplexes Protokollierungssystem benötigt, was nicht Ziel dieser Arbeit ist. Insofern wird auf eine solche Funktionalität verzichtet.

Erwartungskonformität

Ein erwartungskonformes System ist konsistent und zuverlässig gegenüber dem Benutzer.

Es muss also stets eine gleichbleibende Verwendung der richtigen Fachausdrücke gewährleistet sein. Das vereinfacht das Verständnis der Oberfläche und Funktionen. Das Gleiche gilt für den Gebrauch unterstrichener Wörter. Diese sind immer Hypertext-Links.

Da in der Zeiterfassung eine automatisierte Datenspeicherung verwendet werden soll, muss sich diese auch in allen anderen Bereichen wiederfinden. Es darf nicht sein, dass z. B. in der Verwaltung alle Aktionen durch manuelles Speichern abgeschlossen werden müssen.

Die Verwendung der Tabulator-Taste muss der Konvention entsprechen. Durch Betätigung dieser Taste wird der Cursor immer in dem Eingabefeld des Dialogs positioniert, welches im Arbeitsablauf das Nächste ist.

Im Allgemeinen muss der Cursor immer an der Stelle zu finden sein, wo die nächste Eingabe erwartet wird. So muss er z. B. beim Auftritt eines Fehlers im entsprechend zugehörigen Fehlerfeld stehen. Eine einheitliche Oberfläche gehört genauso zu einer erwartungskonformen Software. So sind die Fehler immer an der selben Stelle in gleicher Art und Weise darzustellen.

Außerdem sollen Dialoge für ähnliche Aufgaben immer ähnlich aufgebaut sein. Dies vereinfacht ebenfalls die Bedienung für den Benutzer.

Fehlertoleranz

Ein fehlertolerantes Softwaresystem erkennt falsche Eingaben und korrigiert diese automatisch oder lässt sie mit geringstem Aufwand durch den Benutzer beseitigen.

So ist bei der Eingabe der Zeiten sowohl der Punkt als auch das Komma zu akzeptieren. Da rationale Zahlen im Computer aber mit einem Punkt als Trennzeichen von Vor- und Nachkommastellen dargestellt werden, würden bei Berechnungen von Zahlen mit Komma Fehler auftreten. Es muss also jedes eingegebene Komma durch einen Punkt ersetzt werden. Nur so ist gewährleistet, dass die Eingaben auf dem Nummernblock der Tastatur keine Formatfehler hervorrufen. Dazu muss nach jedem Tastendruck eine Plausibilitätsprüfung stattfinden.

Bei dieser Verifikation der Daten ist zusätzlich sicherzustellen, dass weder Buchstaben noch andere Zeichen außer Punkt und Komma eingegeben werden können. Die Länge der Eingaben darf 5 Zeichen nicht überschreiten, wobei maximal 2 Ziffern vor und 2 Ziffern nach dem Komma erlaubt sind. Insgesamt darf die Zahl nicht kleiner als 0 und nicht größer als 24 sein, wobei die Summe mehrerer Arbeitszeiten an einem Tag nicht 24 Stunden übersteigen darf. Diese Überprüfung betrifft nicht nur das Eintragen von Zahlen über die Tastatur, sondern auch das Einfügen von Werten aus dem Clipboard.

Allgemein müssen Fehler zu Korrekturzwecken in der Sprache des Benutzers erläutert werden. Die Meldungen dürfen nicht technisch verklausuliert sein oder nur als Nummer angezeigt werden. Um dann den Aufwand zur Behebung des Fehlers zu minimieren, sind die fehlerhaften Zeichen der letzten Eingabe zu löschen und der Cursor muss im Fehlerfeld positioniert werden.

Individualisierbarkeit

Eine individualisierbare Benutzungsoberfläche bietet dem Anwender die Möglichkeit, seinen Arbeitsbereich nach persönlichen Wünschen und Bedürfnissen zu gestalten. Zu diesem

Zweck müssen Funktionen angeboten werden, mit denen die Einstellungen (siehe Kapitel 3.2.1.1) angepasst werden können.

So soll ein Benutzer seine bevorzugte Sicht für die Zeiterfassung festlegen können. Er kann die Liste auf einen Tag reduzieren oder sich eine Woche oder einen Monat anzeigen lassen. Weiterhin muss die Sichtbarkeit und Reihenfolge der Projekte festlegbar sein. Dies ist vor allem auch für körperlich eingeschränkte Benutzer von Vorteil. Sehbehinderte können so z. B. ihre Anzeige auf die minimal notwendigsten Inhalte beschränken und dann die Schriftgröße im Browser anpassen. Dann wird der Text besser lesbar, der Arbeitsbereich jedoch nicht zu groß.

Anwender, die eine gleichbleibende Arbeitszeit für eine gewisse Dauer haben, sollen ihre Arbeitsstunden für diesen Zeitraum vorgeben können. So müssen sie sich nicht täglich am System anmelden, sondern brauchen nur noch eventuelle Abweichungen anpassen.

Zum Eintragen der Arbeitszeit sollen verschiedene Interaktionsformen angeboten werden. Es muss möglich sein, die Daten nur mittels der Maus zu erfassen. Dazu werden Schaltflächen benötigt, mit denen der Benutzer seine Arbeitszeiten eintragen kann. Genauso muss der Anwender das System aber auch komplett über die Tastatur bedienen können. Dazu sollte er nicht nur die Arbeitszeiten eingeben, sondern auch mittels Tabulator-Taste in den Formularen navigieren können.

Lernförderlichkeit

Eine lernförderliche Software zeichnet sich dadurch aus, dass sie den Anwender beim Erlernen der Funktionen unterstützt.

Dazu müssen alle Schaltflächen mit kurzen Hinweistexten versehen werden, die Auskunft über die jeweilige Funktionalität geben.

Weiterhin sollen bei allen Aufgaben umfangreiche Hinweise zur Verfügung gestellt werden, mit denen der Arbeitsablauf und das Ziel kurz erläutert werden. Diese sollen immer in gleicher Weise dargestellt werden und auch einheitlich abrufbar sein.

Zusätzlich ist eine Onlinehilfe anzubieten, die sowohl allgemeine Unterstützung zur Bedienung des System gibt, als auch aufgabenspezifische Hinweise bereitstellt.

4 Design

Bei dem zu erstellenden Zeiterfassungssystem handelt es sich um eine webbasierte Software, welche einem Mitarbeiter ermöglicht, seine verrichtete Arbeitszeit digital zu speichern. Die Daten werden projektbezogen erfasst, um so z. B. wöchentlich die aktuellen Aufwände der jeweiligen Projekte ermitteln zu können.

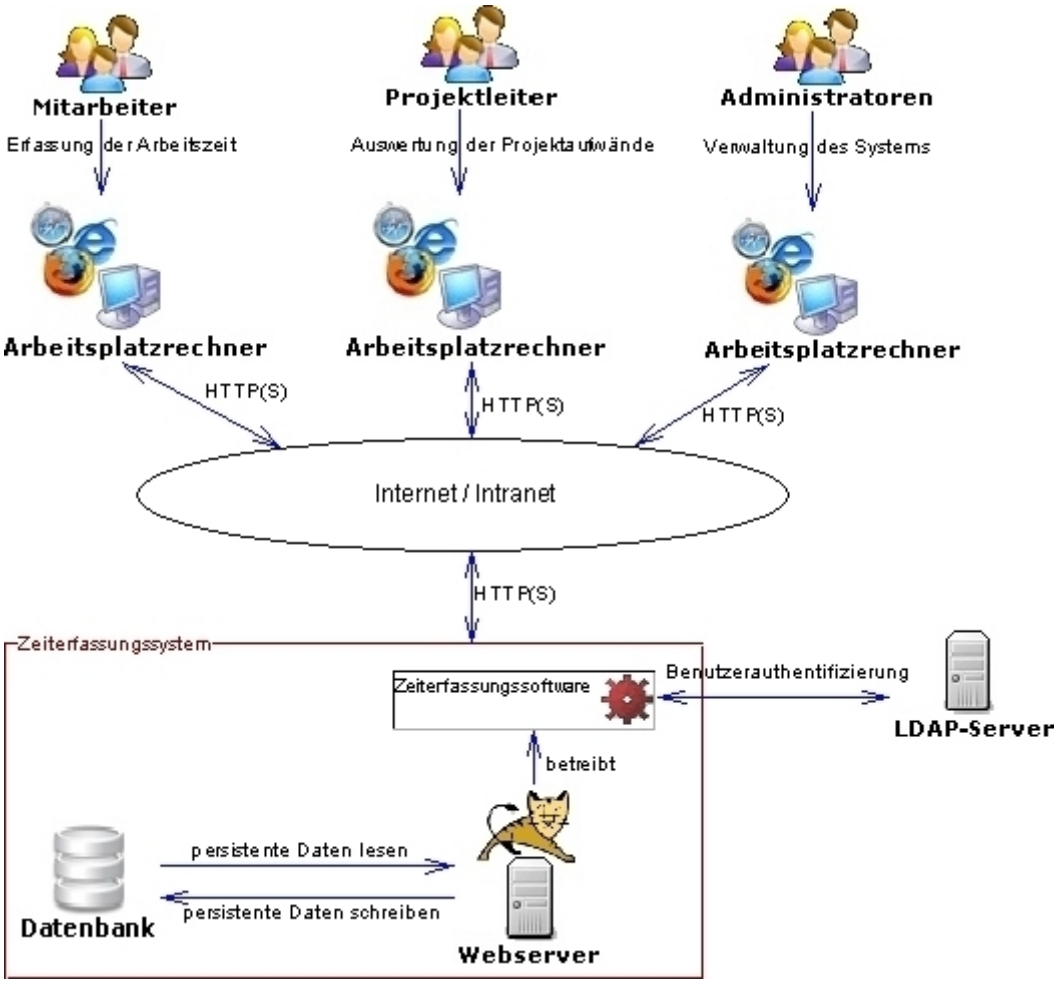


Abbildung 4.1: Übersicht über die komplette Systemarchitektur

Die Zeiterfassungssoftware soll auf einem zentralen Server ablaufen. So ist sie über das Netzwerk erreichbar und kann von den Arbeitsplatzrechnern im Internet/Intranet genutzt werden (siehe Abbildung 4.1). Diese müssen lediglich über einen Browser verfügen der JavaScript unterstützt und aktiviert hat. So ist es jedem Benutzer jederzeit möglich, sich am System anzumelden, um dort Zeiten einzutragen oder auszuwerten. Die Authentifizierung am System wird mittels eines LDAP-Servers durchgeführt.

4.1 Fachliche Architektur

Aus fachlicher Sicht besteht das Zeiterfassungssystem aus vier Komponenten (siehe Abbildung 4.2). Die Datenspeicherung übernimmt die Verwaltung der persistenten Daten. Dort werden die Arbeitszeiten und Informationen zu den Systemressourcen, z. B. Personen- oder Projektdaten, abgelegt. Mittels der anderen drei Komponenten kann entsprechend lesend oder schreibend auf die Daten zugegriffen werden.

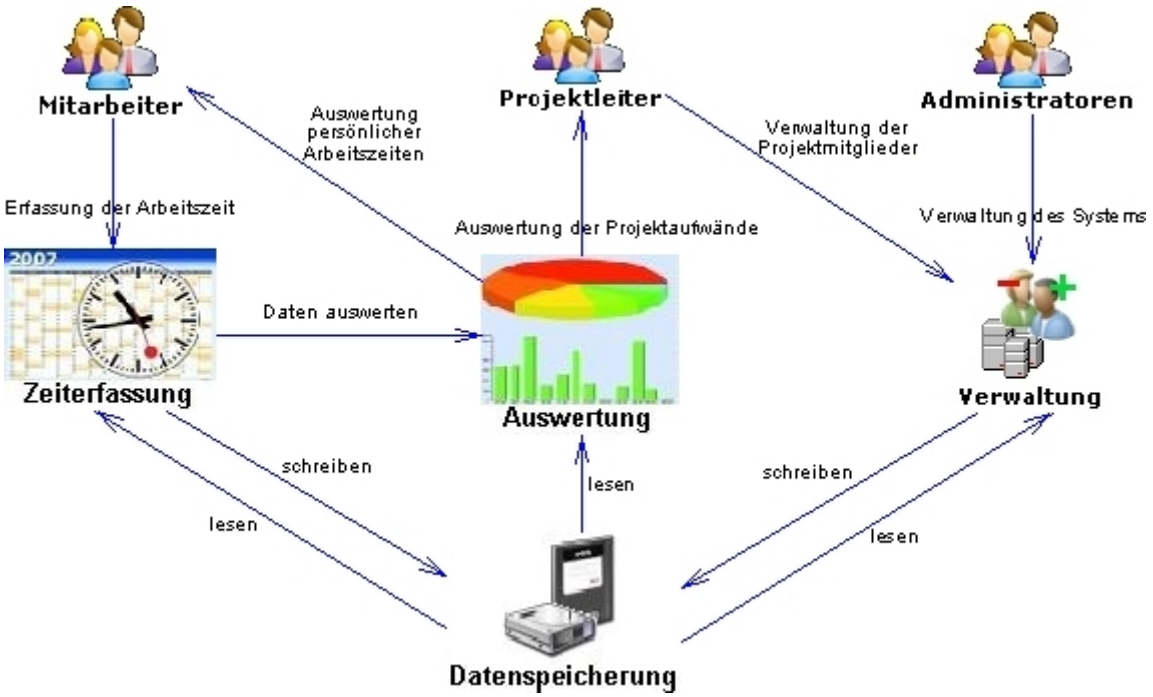


Abbildung 4.2: Fachliche Architektur des Zeiterfassungssystems

Die Zeiterfassung liesst die Arbeitszeiten aus, welche schon gespeichert wurden und für die aktuelle Ansicht notwendig sind oder schreibt geänderte und neu hinzugefügte Zeiten in die Datenspeicherung.

Der Auswertung ist nur ein lesender Zugriff auf die persistenten Daten erlaubt. Sie fasst die gesammelten Daten aus der Zeiterfassung zusammen und bereitet sie ggf. auf. So können die Projektleiter die Aufwände ihrer Projekte auswerten. Außerdem kann sich jeder Mitarbeiter einen Überblick über seine persönlichen Arbeitszeiten verschaffen.

In der Verwaltung können die Administratoren die Systembenutzer mit ihren Rechten festlegen, oder Projekte hinzufügen, bearbeiten und löschen. Außerdem verwalten hier die Projektleiter die Mitglieder ihrer Projekte.

4.1.1 Entity-Relationship-Modell

Bei dem Entity-Relationship-Modell (siehe Abbildung 4.3) handelt es sich um ein sehr abstraktes Modell, was die Grundlage für das eingesetzte Datenbankmodell (siehe Kapitel 4.2.5) liefert. So wird eine konkrete Struktur festgelegt die bestimmt, auf welche Art und Weise die Daten gespeichert werden.

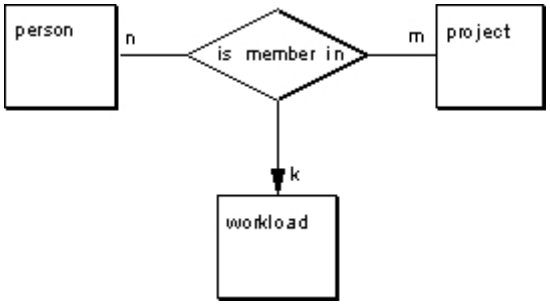


Abbildung 4.3: Abstraktes Datenbankschema

Bei dem Zeiterfassungssystem werden drei Entitäten festgelegt. Die Daten zu den einzelnen Systembenutzern werden in der Entität „person“ gespeichert. Für die Informationen zu den hinzugefügten Projekten ist die Entität „project“ vorgesehen. Da eine Person in einem Projekt Mitglied sein kann und Projekte viele Mitglieder haben können, werden diese beiden Entitäten über die n:m-Beziehung „is member in“ miteinander verbunden. Aus dieser Verknüpfung ergeben sich eine Vielzahl von Arbeitszeiten, welche sich in der Entität „workload“ wiederfinden.

4.1.2 Aktivitäten

Im zu entwickelnden Zeiterfassungssystem gibt es eine Reihe von Aktivitäten, welche von Benutzern oder dem System ausgeführt werden. Diese sind teilweise sehr einfach und kurz,

können aber auch recht umfangreich sein. Im Folgenden wird nur auf die komplexen Aktivitäten eingegangen. Dazu zählen neben dem Eintragen der Arbeitszeit auch die Designentscheidungen zum Entfernen von Systemressourcen bzw. den Verbleib ihrer Arbeitszeiten.

Eintragen der Arbeitszeit

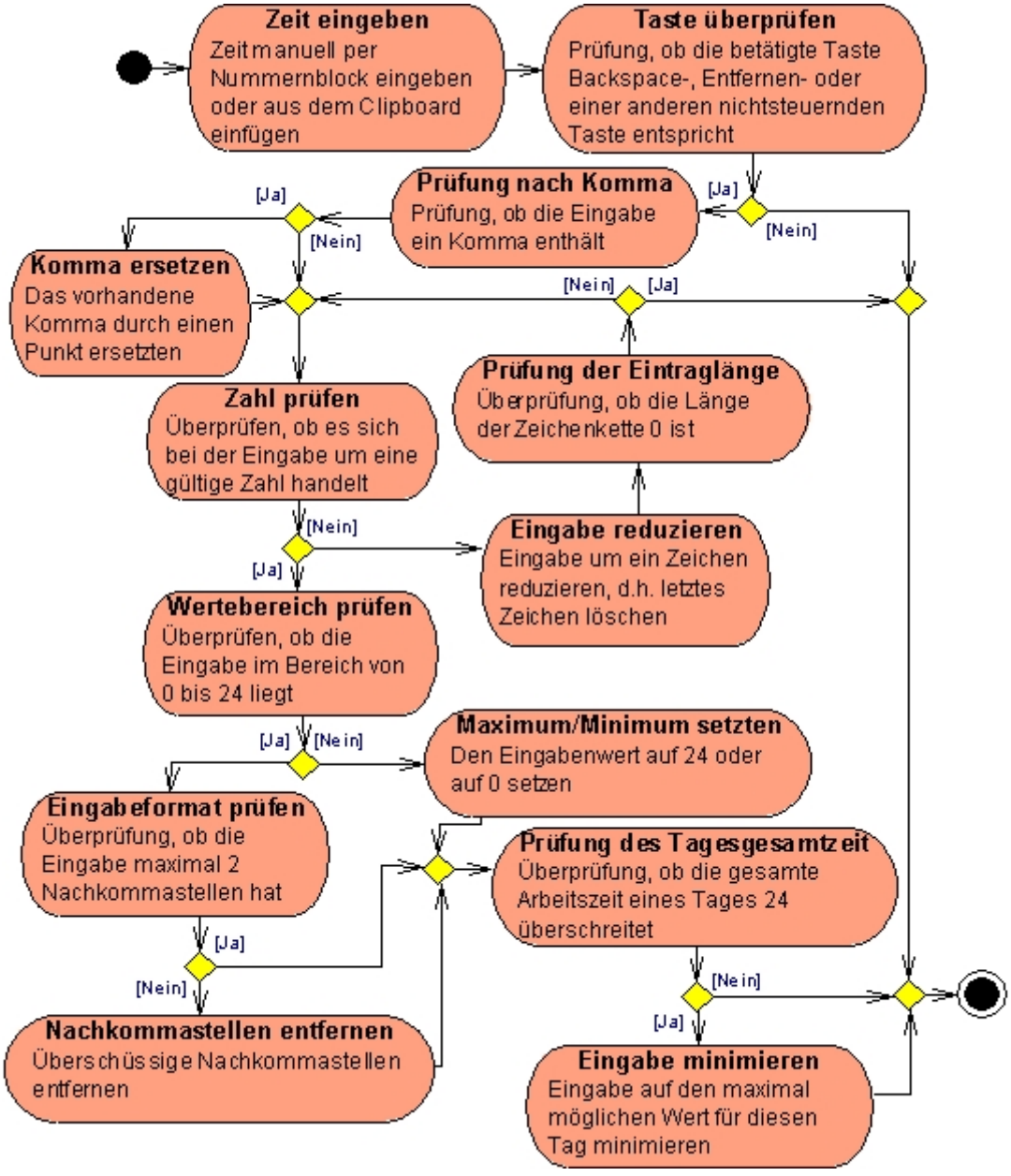


Abbildung 4.4: Aktivitätsdiagramm zum Eintragen der Arbeitszeit

Das Eintragen von Arbeitszeiten (siehe Abbildung 4.4) weist eine besondere Komplexität auf,

da die Eingabe mehrfach geprüft werden muss. Die Einträge können sowohl über die Tastatur getätigt werden, als auch per Einfügeoperation aus dem Clipboard stammen. Als erstes wird der Tastencode abgefragt. Entspricht dieser der Backspace-, Entfernen- oder einer anderen nichtsteuernden Taste wird die Überprüfung fortgesetzt. Nur dann ist gewährleistet, dass sich der Eintrag verändert hat. Als nächstes wird die Existenz von Kommas kontrolliert. Sollte eines vorhanden sein, wird es durch einen Punkt ersetzt. So wird sichergestellt, dass der Eintrag ein Zahlenformat annimmt. Anschließend wird untersucht, ob der Eintrag eine Zahl ist. Sollte das nicht der Fall sein, wird das letzte Zeichen gelöscht und der Eintrag wieder untersucht. Dieser Vorgang wiederholt sich solange, bis der restliche Eintrag eine Zahl ist oder die Länge der Zeichenkette 0 beträgt. Die Reduktion der Zeichenkette wird durchgeführt, um den maximalen Zeichenanteil des Eintrags zu behalten, der eine Zahl darstellt. So wird der Korrekturaufwand für den Benutzer minimiert. Im folgenden Schritt wird überprüft, ob sich die Zahl im vorgegebenen Wertebereich befindet. Beim Über- oder Unterschreiten der definierten Grenzen wird das Maximum bzw. das Minimum gesetzt. Das heißt konkret, dass ein Eintrag, der größer als 24 ist, auf 24 gesetzt wird und der kleiner als 0 ist, erhält den Wert 0. Befindet sich die Zahl im Wertebereich wird kontrolliert, ob sie maximal 2 Nachkommastellen hat. Da alle Zahlen mit mehr als 2 Vorkommastellen bereits bei der Wertebereichsprüfung eliminiert werden, muss nun nur noch diese Kontrolle durchgeführt werden. Wird die festgelegte Länge der Nachkommasstellen überschritten, werden alle überschüssigen Zeichen gelöscht. Als letztes wird die Gesamtarbeitszeit eines Tages überprüft. Da es nicht möglich sein kann, an einem Tag mehr als 24 Stunden zu arbeiten, darf die Summe aller Arbeitszeiten für einen Tag diesen Wert auch nicht übersteigen. Ist das doch der Fall, wird der aktuelle Eintrag auf den maximal möglichen Restwert, nämlich 24 abzüglich Summe der anderen Arbeitszeiten an diesem Tag, gesetzt.

Der gesamte Prüfalgorithmus kann auch verallgemeinert werden. Der reguläre Ausdruck

24 | (1 ? [0 - 9] | 2 [0 - 3]) (, [0 - 9] { 1 , 2 }) ?

beschreibt die validen Arbeitszeiten. Mittels vorhandener Methoden, welche von den Programmiersprachen zur Verfügung gestellt werden, können so viele Benutzereingaben durch entsprechende reguläre Ausdrücke überprüft werden.

Entfernen von Systembenutzern

Beim Entfernen von Systembenutzern (siehe Abbildung 4.5) wird als erstes überprüft, ob die ausgewählte Person Leiter eines oder mehrerer Projekte ist. In diesem Fall wird der Löschvorgang abgebrochen und der Administrator wird aufgefordert der ausgewählten Person die Leitungsrolle in allen Projekten zu entziehen. Handelt es sich nicht um einen Projektleiter, wird geprüft, ob die ausgewählte Person bereits Arbeitszeiten zu irgendeinem Projekt eingetragen hat. Bei vorhandenen Zeiten kann die Löschaktion nur durchgeführt werden, wenn der Administrator zustimmt, dass die Person mit allen ihren Arbeitszeiten gelöscht werden soll. Eine Ablehnung hat keine weiteren Handlungsschritte zur Folge.

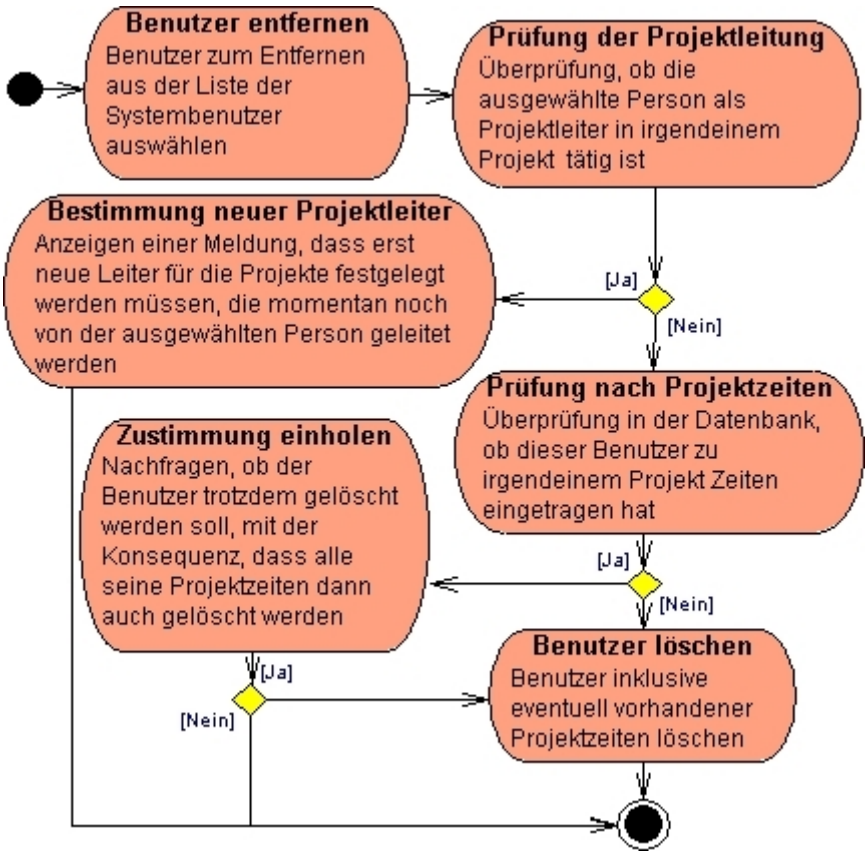


Abbildung 4.5: Aktivitätsdiagramm zum Entfernen von Systembenutzern

Entfernen von Administratoren

Das Entfernen von Systemadministratoren (siehe Abbildung 4.6) ist nicht sehr umfangreich, da es sich lediglich auf das Entziehen des administrativen Rechts bezieht. Hierbei muss allerdings sichergestellt werden, dass die aktuell angemeldete Person sich nicht selbst dieses

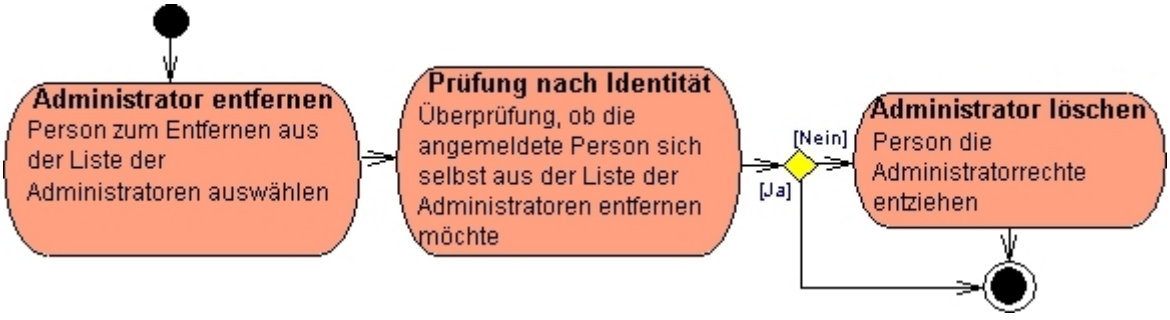


Abbildung 4.6: Aktivitätsdiagramm zum Entfernen von Administratoren

Recht einzieht. Stimmt also die Identität der angemeldeten Person mit der ausgewählten überein, darf keine Aktion durchgeführt werden.

Entfernen von Projekten

Nach der Auswahl des zu löschenden Projektes (siehe Abbildung 4.7) wird zuerst die Zustimmung des Benutzers eingeholt, ob es sich bei dem gewählten Projekt um das Richtige handelt. Bestätigt die Person ihre Wahl, wird geprüft, ob in der Datenbank Arbeitszeiten zu diesem Projekt gespeichert sind. Ist das nicht der Fall, wird das Projekt sofort gelöscht. Existieren jedoch bereits Zeiten, wird eine weitere Zustimmung vom Benutzer eingefordert. Hier muss er angeben, ob das Projekt trotz vorhandener Zeiten gelöscht werden soll. Wird dies bestätigt, wird das Projekt mit allen seinen eingetragenen Zeiten aus der Datenbank entfernt. Bei einer Ablehnung wird keine Aktion durchgeführt.

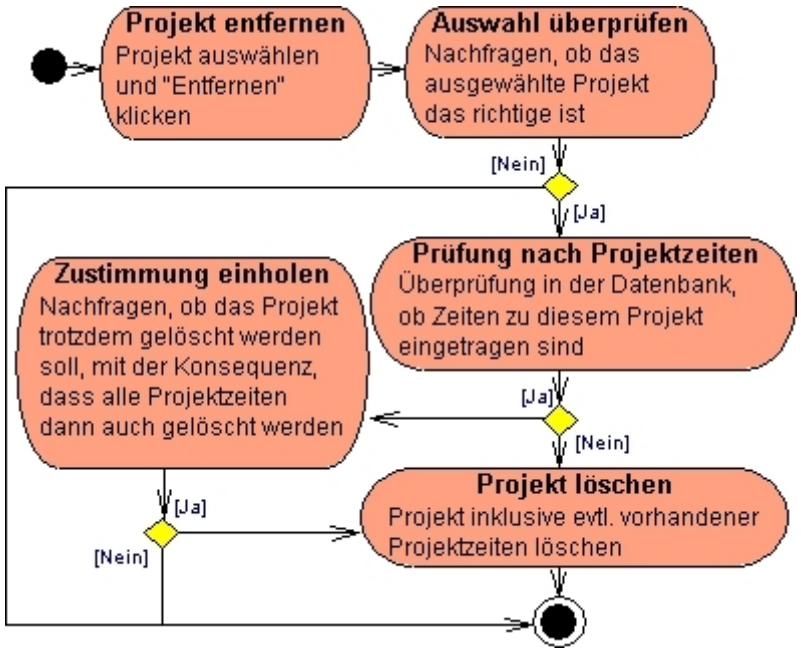


Abbildung 4.7: Aktivitätsdiagramm zum Entfernen von Projekten

Entfernen von Projektmitgliedern

Das Entfernen von Projektmitgliedern (siehe Abbildung 4.7) wird von den jeweiligen Projektleitern durchgeführt. Dazu müssen sie das entsprechende Projekt und die zu löschende Person auswählen. Danach wird überprüft, ob die Person für das gewählte Projekt bereits Arbeitszeiten eingetragen hat. Wenn das der Fall ist, muss der Projektleiter zustimmen, dass beim Entfernen der Person alle ihre Arbeitszeiten für dieses Projekt gelöscht werden. Bei einer Ablehnung wird keine Aktion durchgeführt.

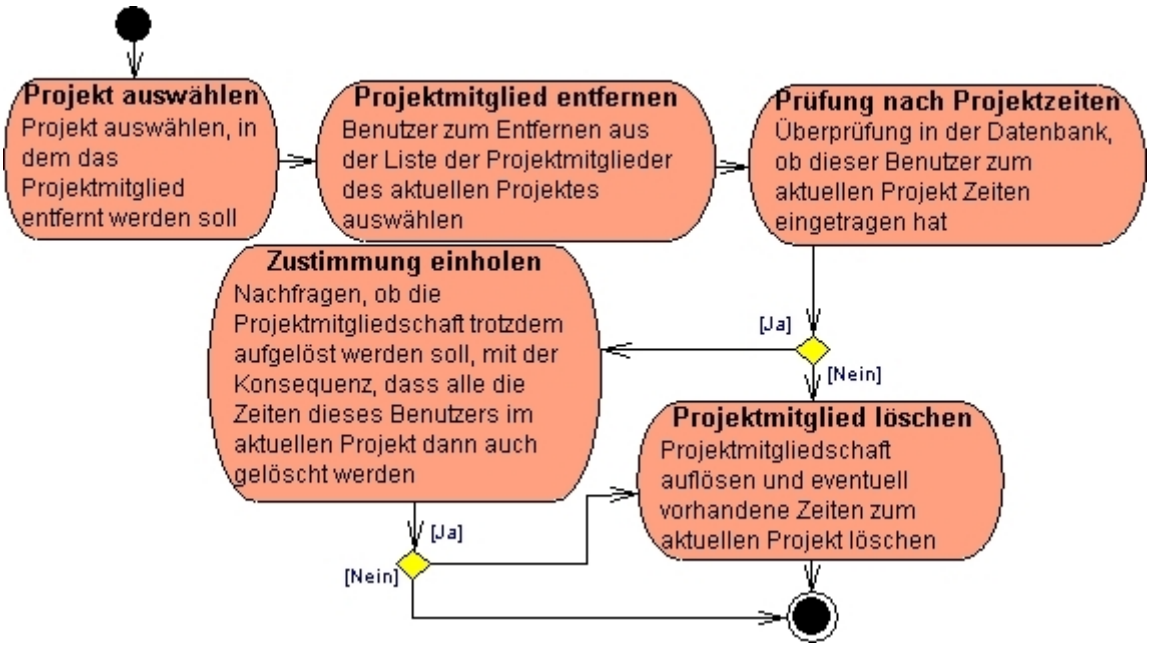


Abbildung 4.8: Aktivitätsdiagramm zum Entfernen von Projektmitgliedern

4.2 Technische Architektur

In der technischen Architektur werden die Funktionalitäten des Systems auf die Software-Komponenten abgebildet. Dazu wird sowohl auf die Aufteilung des Systems als auch auf die Abhängigkeiten der Implementierungskomponenten und die Systemteile zur Laufzeit eingegangen.

4.2.1 Infrastrukturelle Voraussetzungen

Um das Zeiterfassungssystem betreiben zu können, wird ein Rechner benötigt, auf dem ein Tomcat-Server installiert ist. Des Weiteren muss eine Datenbank zur Verfügung stehen, in welcher die Daten gespeichert werden können.

Für die Authentifizierung und Verwaltung der Systembenutzer wird ein LDAP-Server vorausgesetzt, da dort die Personen- und Adressdaten zentral gespeichert sind. Mittels bestehender Bibliotheken können Personendaten gelesen oder Benutzernamen und Passwörter überprüft werden.

4.2.2 3-Schichten-Architektur

Durch die Verwendung einer 3-Schichten-Architektur wird ein Softwaresystem in die einzelnen Komponenten zerlegt, die in ihrem Aufgabenbereich eingeschränkt und von anderen Komponenten abhängig sind. Die drei resultierenden Schichten sind die Präsentation, die Anwendungslogik und die Datenhaltung (siehe Abbildung 4.9). Sie unterliegen einer strikten Ordnung, d.h. es dürfen keine Schichten übersprungen werden.

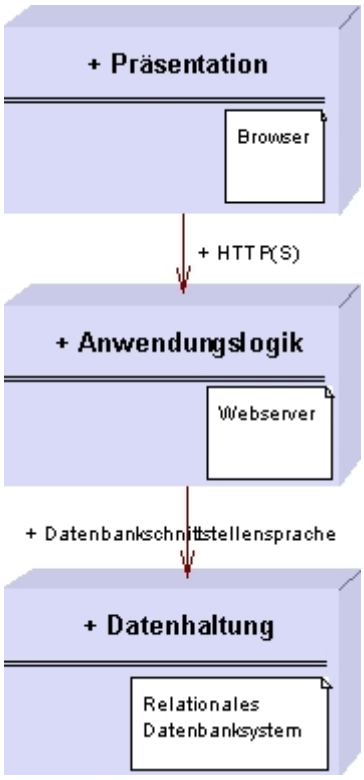


Abbildung 4.9: 3-Schichten-Architektur

Die Vorteile dieser Architektur sind die gute Skalierbarkeit und Wartbarkeit. So muss die Anwendung nicht grundlegend geändert werden, wenn die einzelnen Schichten auf verschiedenen Rechnern ablaufen sollen. Es ist möglich die Datenhaltung sowohl auf einen großen zentralen Datenbankserver zu verlagern, als auch auf dem eigens für das Zeiterfassungssystem zur Verfügung stehenden Computer zu installieren. Die Anwendungslogik kann von einem komplexen Applicationserver bereitgestellt werden oder von einem einfachen Webserver. Bei der Präsentation kann HTML oder ein Applet zum Einsatz kommen. Die Veränderung in einer Schicht hat immer nur kleine Anpassungen der umgebenden Schichten zur Folge, aber selten umfangreiche Modifikation.

Im Fall des Zeiterfassungssystems wird für die Datenhaltung ein relationales Datenbanksystem eingesetzt und für die logischen Anwendungsprozesse ein Webserver. Diese beiden Schichten werden über eine geeignete Datenbankschnittstelle verbunden. Die Benutzungsoberfläche wird dann in einem Browser dargestellt.

4.2.3 Klassenmodell

Das folgende Klassendiagramm (siehe Abbildung 4.10) zeigt die einzelnen Klassen des Systems und deren Beziehungen untereinander.

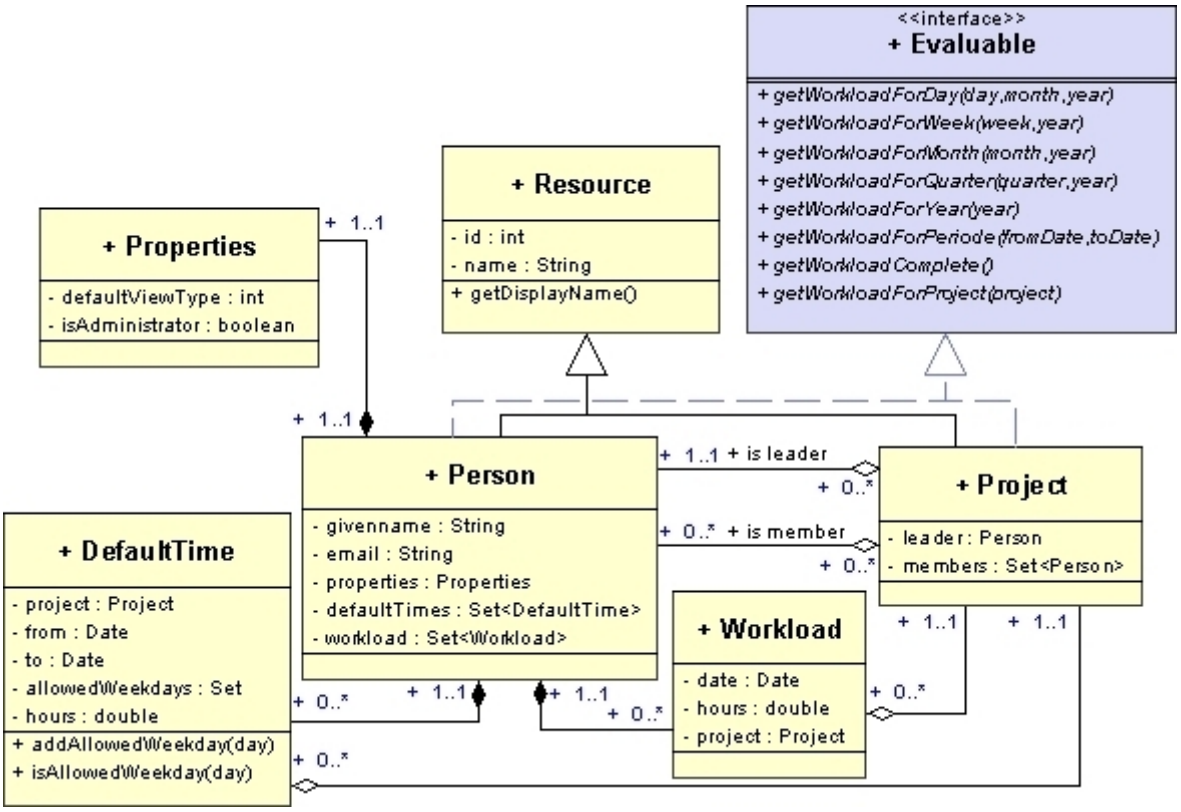


Abbildung 4.10: Klassenmodell des Systems

Klasse Resource

Die generelle Klasse *Resource* legt die grobe Struktur der Systemressourcen fest. Sie enthält die allgemeinen Attribute und eine Methode zum Anzeigen des Ressourcennamens.

Interface Evaluable

Das Interface *Evaluable* definiert Operationen zum Auswerten der Arbeitszeiten. Diese können tageweise, wöchentlich, monatlich, quartalsweise, jährlich oder komplett abgerufen werden. Zusätzlich ist es möglich, die Arbeitszeit für einen frei definierbaren Zeitraum zu ermitteln. Die Methode *getWorkloadForProject(...)* soll eine projektbezogene Auswertung liefern. Da sie deshalb in Projekten nicht notwendig ist, muss sie in der Klasse *Project* so implementiert sein, dass sie beim Aufruf eine Exception auslöst.

Klasse Person

Die Klasse *Person* bildet einen Systembenutzer der realen Welt auf das System ab. Sie spezialisiert die Klasse *Resource* und realisiert die Methoden des Interfaces *Evaluable*. Außerdem werden zusätzliche Attribute für den Vornamen und die Emailadresse des Benutzers festgelegt. Über eine Komposition wird sie mit der Klasse *Properties* verbunden, welche zusätzliche Einstellungen einer Person speichert. Weiterhin werden zwei *Collections* für die Vorgaben und die Arbeitszeiten definiert. Sie sind vom Typ *Set*, um Duplikate zu eliminieren. Ihre enthaltenen Werte (Objekte der Klassen *DefaultTime* oder *Workload*) sind ebenfalls über eine Komposition gebunden, da ihre Lebensdauer vom umschließenden Objekt (Instanz einer Person) abhängig sind. Wird also eine Person gelöscht, werden auch ihre Arbeitszeiten und ihre Vorgaben gelöscht.

Klasse Project

Die Klasse *Project*, welche auch *Resource* spezialisiert und *Evaluable* implementiert, ist eine Abbildung der Projekte auf das System. Neben einem Projektleiter legt sie eine Menge von Mitgliedern fest. Diese werden in einem *Set* gesammelt, um Duplikate auszuschließen. Sowohl die Mitglieder als auch der Projektleiter sind mittels Aggregationen verbunden, da sie unabhängig von den Projekten existieren können.

Klasse Workload

Die Arbeitszeiten werden in Objekten der Klasse *Workload* erfasst. Für die jeweilige Arbeitszeit definiert sie das Datum und die erbrachten Arbeitsstunden. Weiterhin legt sie über eine Aggregation das jeweilige Projekt fest, in welchem gearbeitet wurde.

Klasse DefaultTime

Mit der Klasse *DefaultTime* werden Vorgaben gespeichert. Dazu müssen das Projekt, der Start und das Ende des Vorgabezeitraums, die jeweiligen Wochentage und die geleistete Arbeitszeit angegeben werden. Die Assoziation zu *Project* ist eine Aggregation, da das Projekt unabhängig von der Vorgabe existiert.

Klasse Properties

Die Klasse *Properties* speichert zusätzliche Systemeinstellungen einer Person. Das sind die bevorzugte Ansicht für die Zeiterfassung und ein eventuelles Administrationsrecht.

4.2.4 Kommunikationsablauf der Systemkomponenten

Im Folgenden werden die Kommunikationswege (siehe Abbildung 4.11) zwischen den einzelnen Systemkomponenten erläutert.

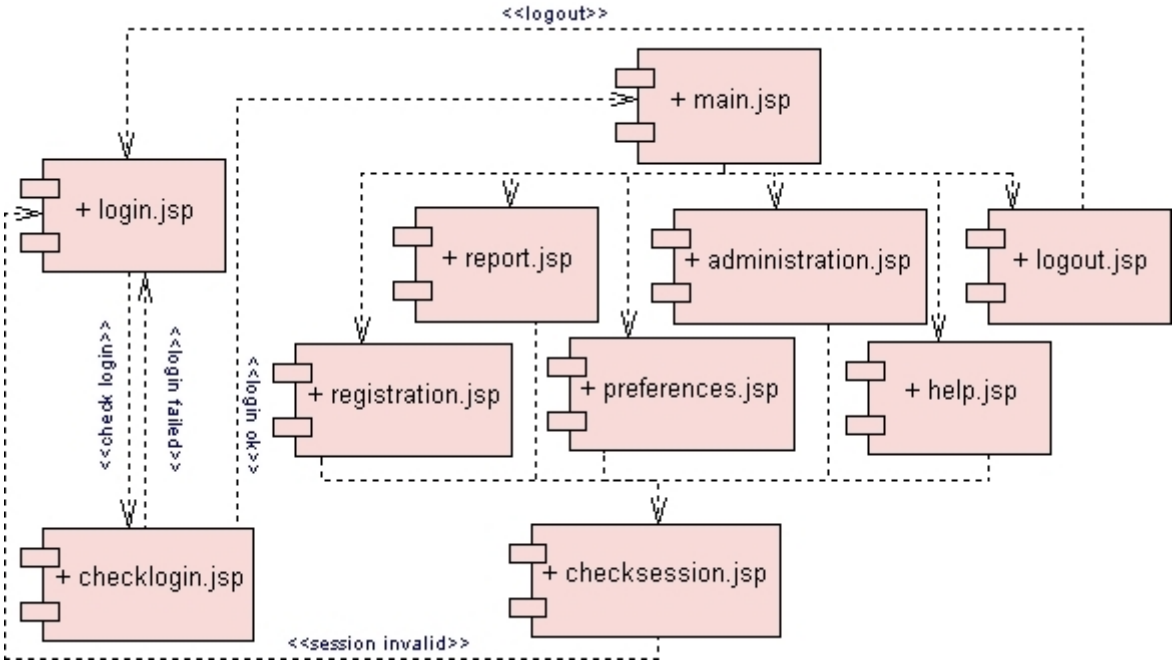


Abbildung 4.11: Kommunikationsablauf der Systemkomponenten

Beim Start der Anwendung wird zuerst die Komponente *login.jsp* geladen. Dort muss die Person ihren Benutzernamen und ihr Passwort angeben. Die Daten werden dann an *checklogin.jsp* übergeben, wo die Benutzerauthentifikation gegen einen LDAP Server stattfindet. Bei einem Misserfolg wird wieder die Loginseite mit einer entsprechenden Fehlermeldung angezeigt. Bei erfolgreicher Anmeldung wird eine neue Session erzeugt und es erscheint *main.jsp*. Von dort aus kann auf die Funktionen des Systems zugegriffen werden. Jede der Komponenten *registration.jsp*, *report.jsp*, *preferences.jsp*, *administration.jsp* oder *help.jsp* bindet *checksession.jsp* ein, welche die Gültigkeit der Session überprüft. Ist diese valide, wird die gewählte Seite angezeigt. Im Fall einer ungültigen Session gelangt der Benutzer zurück auf die Loginseite, wo eine entsprechende Fehlermeldung angezeigt wird.

4.2.5 Relationales Datenbankmodell

Die Datenbank dient der persistenten Speicherung der gesamten Daten des Zeiterfassungssystems. Dazu wird das ERM (siehe Kapitel 4.1.1) in ein relationales Datenbankmodell (siehe Abbildung 4.12) überführt. Nachfolgend werden die wichtigsten Relationen und ihre Beziehungen untereinander erläutert.

Zur Entfernung von Daten wird teilweise kaskadierendes Löschen verwendet. Dies hat den Vorteil, dass die Datenbank ebenfalls die Tupel entfernt, welche den zu löschenden Datensatz über einen Fremdschlüssel referenzieren. So muss dies nicht in der Applikation durchgeführt werden. Wo kaskadierendes Löschen eingesetzt wird, beruht auf den Designentscheidungen zu den Aktivitäten (siehe Kapitel 4.1.2).

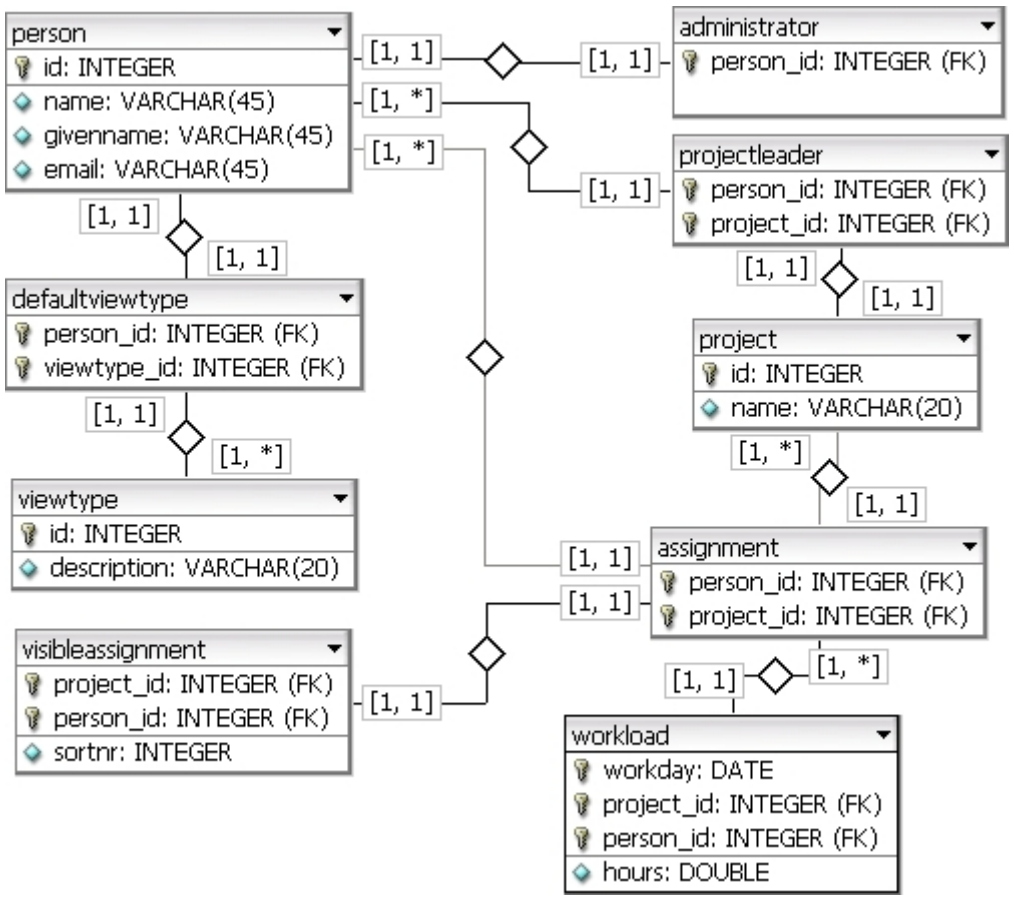


Abbildung 4.12: Relationales Datenbankmodell des Zeiterfassungssystems

Relation person

Alle Daten einer Person werden in der Relation *person* gespeichert. Die `id` wird nicht automatisch erzeugt, sondern entspricht der Benutzernummer im LDAP Server. So können

nach der Authentifikation die korrekten Benutzerdaten aus der Datenbank gelesen werden. Um dem Benutzer eine Standardansicht zuzuordnen, wird die `id` als Fremdschlüssel in die Relation *defaultviewtype* eingetragen. Besitzt die Person administrative Rechte, wird ihre `id` in der Relation *administrator* gespeichert. Für die Tätigkeit als Projektleiter existiert eine Verknüpfung zu *projectleader* und die Projektmitgliedschaften befinden sich in *assignment*. Im Gegensatz zur Verbindung zu *projectleader* kommt bei den anderen drei Beziehungen kaskadierendes Löschen zum Einsatz. Da das Entfernen von Personen, welche Projekte leiten, aber nicht erlaubt ist, darf dort diese Art des Löschens nicht verwendet werden.

Relation project

Die Relation *project* speichert die Daten eines Projektes. Dazu zählt die `id`, welche automatisch inkrementiert wird, sowie der Projektname. Alle angrenzenden Relationen sind über Fremdschlüsselbeziehungen mit kaskadierendem Löschen verbunden, da beim Entfernen eines Projektes sowohl die Verknüpfung zum Leiter als auch die zu jedem Mitglied aufgelöst werden muss.

Relation assignment

Um Projektmitgliedschaften zu verwalten, sind `person_id` und `project_id` in der Relation *assignment* miteinander zu verknüpfen. Das Auflösen dieser Verknüpfung hat ein Entfernen aller referenzierenden Tupel zur Folge, da bei den Beziehungen zu *visibleassignment* und *workload* kaskadierend gelöscht wird.

Relation visibleassignment

Die Relation *visibleassignment* dient der Speicherung der Benutzereinstellungen. Hier werden die Projekte abgelegt, in denen eine Person Mitglied ist und die sie in der Zeiterfassung sehen möchte. Mittels `sortnr` wird die vom Benutzer festgelegte Projektreihenfolge gespeichert.

Relation workload

Für das Sichern der persönlichen Arbeitszeiten ist der Relation *workload* verantwortlich. Der Primärschlüssel setzt sich hierbei aus `workday`, `project_id` und `person_id` zusammen, da eine Kombination dieser Attribute eindeutig sein muss.

4.2.6 Oberflächenstruktur

Um die Benutzungsoberfläche zu strukturieren, wird sie in drei Bereiche unterteilt (siehe Abbildung 4.13). Der Informationsbereich dient der Anzeige des Namens der angemelde-

ten Person und des aktuellen Datums. Zur Navigation in den einzelnen Hauptfunktionen im System werden Reiter angeboten. Dabei ist der aktuell ausgewählte immer farblich abgesetzt, um eine bessere Orientierung zu gewährleisten. Unterhalb der Menüleiste liegt der Arbeitsbereich mit den einzelnen Aufgaben.

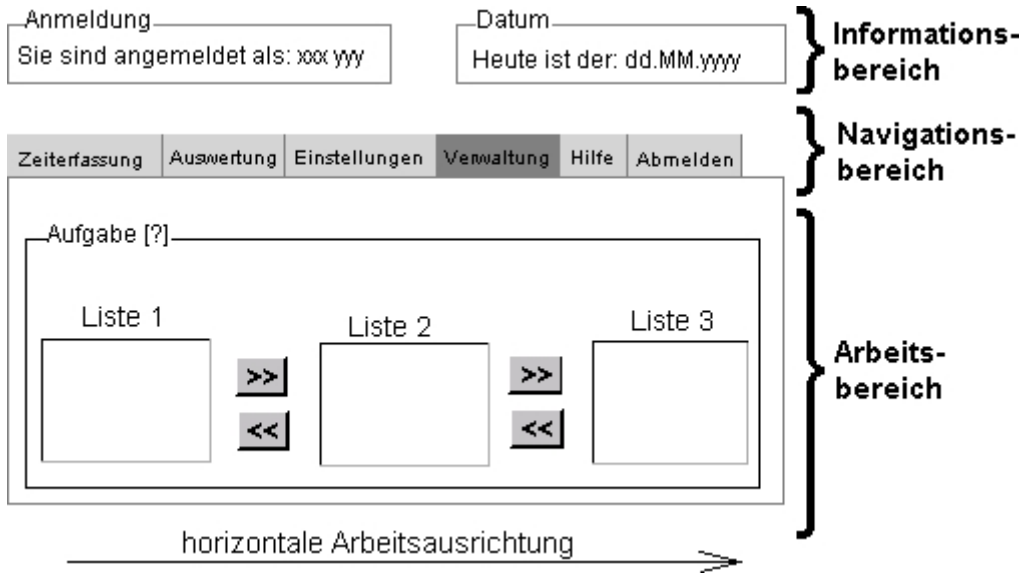


Abbildung 4.13: Struktur der Benutzungsoberfläche

4.2.7 Software-ergonomische Umsetzung

Den maßgeblichen Anteil an der Usability des Zeiterfassungssystems trägt die Umsetzung der software-ergonomischen Anforderungen. Sie wird im Folgenden den einzelnen Bewertungskriterien (siehe Kapitel 2.1.2) der Software-Ergonomie zugeordnet.

Aufgabenangemessenheit

Da das aktuelle Datum eine wesentliche Information für den Benutzer ist, wird es immer an dem vorgesehenen Platz im Informationsbereich angezeigt (siehe Abbildung 4.13). Um zusätzlich die einzelnen Tage im Tabellenkopf der Zeiterfassungsliste zu unterscheiden, werden Wochentage hellgrau, Wochenende dunkelgrau und der aktuelle Tag hellblau hinterlegt.

Per JavaScript können Felder in HTML-Formularen fokussiert werden. Diese Funktion kommt zum Einsatz, um beim Laden einer Seite oder im Fehlerfall den Cursor ins entsprechende Eingabefeld zu setzen.

Das automatische Speichern wird mittels Ajax (siehe Kapitel 2.3.2) umgesetzt. So können die Daten sofort im Hintergrund asynchron in die Datenbank übertragen werden, ohne ständig eine manuelle Bestätigung vom Benutzer einzufordern.

Eine Kalenderfunktion in der Zeiterfassung ermöglicht dem Benutzer das Einstellen des gewünschten Datums. Dazu werden ihm Listen für den Tag, Monat und Jahr angegeben. Bei der Auswahl in einer der Listen wird sofort die Ansicht angepasst. Außerdem werden Buttons angeboten mit denen jeweils ein Tag / eine Woche / ein Jahr weiter oder zurück navigiert werden kann. Die Beschriftung und die Funktion dieser Buttons hängt von der aktuell ausgewählten Sicht ab.

Selbstbeschreibungsfähigkeit

Ein wichtiger selbstbeschreibender Teil einer Anwendung ist die Menüleiste. Sie muss dem Benutzer stets anzeigen, wo er sich befindet und wohin er navigieren kann. Im Fall des Zeiterfassungssystems werden Reiter eingesetzt. Diese bieten dem Benutzer einen Überblick über die Grundfunktionen der Anwendung, wobei der aktive Reiter immer farblich abgesetzt ist. Die Beschriftung ist eindeutig und im System konsistent, wird aber zusätzlich mit zusammenfassenden Erklärungen in Form von Tooltips unterstützt.

Um dem Benutzer in der Zeiterfassung anzuzeigen, dass seine Aktionen erfolgreich waren, werden nach jeder Eingabe die entsprechenden Zeilen und Spalten neu summiert. So erhält der Anwender eine Rückmeldung. Beim Anlegen eines neuen Projekts in der Verwaltung wird in der Tabelle eine neue Zeile mit den Daten erzeugt. Sie wird alphabetisch korrekt einsortiert und in den sichtbaren Bereich gescrollt. Zum Verwalten der Benutzer werden die Einträge aus einer Liste in eine andere verschoben. So bekommt der Benutzer auch hier sofort eine Rückmeldung über seine Aktion.

Steuerbarkeit

Um eine gute Steuerbarkeit zu gewährleisten, wird automatisches Speichern mittels Ajax (siehe Kapitel 2.3.2) eingesetzt. Dadurch werden alle Aktionen sofort im Hintergrund in die Datenbank übertragen und können nicht verloren gehen. Das Eingeben der Arbeitszeit kann somit stets unterbrochen werden, um dort nach der Erledigung anderer Aufgaben fortzufahren.

Verschiedene Eingabeformen tragen ebenfalls zur Steuerbarkeit bei. Für die ausschließliche Arbeit mit der Tastatur ist die Navigation mittels Tabulator-Taste möglich. Wenn ein Benutzer hingegen nur die Maus verwenden will, werden ihm Schaltflächen zum Addieren und Subtrahieren der Arbeitsstunden angeboten.

Erwartungskonformität

Um die Konsistenz der Fachausdrücke zu gewährleisten, wird eine zentrale Stelle zur Verwaltung der Begriffe festlegt. Diese können dann über einen Schlüssel referenziert werden. Weiterhin sind alle Informationen bzgl. der Darstellung der HTML-Seiten in CSS-Dateien auszulagern.

Das automatische Speichern wird in allen Bereichen durchgeführt. Das betrifft sowohl die Zeiterfassung als auch die Verwaltung und die persönlichen Einstellungen.

Um die Oberfläche einheitlich zu gestalten, sind zusammengehörende Daten immer zeilenweise dargestellt. Dies legt einen horizontalen Arbeitsablauf fest (siehe Abbildung 4.13). Zusätzlich werden die Formularfelder einer Aufgabe durch einen beschrifteten Rand gruppiert. Das trägt zu einer besseren Orientierung und Lesbarkeit bei.

Beim Auftreten von Fehlern werden die Meldungen immer in einem JavaScript Alert angezeigt. Der hat stets die gleiche Erscheinung und Bildschirmposition, was auch den Erwartungen des Benutzers entspricht.

Fehlertoleranz

Auf die Vorgehensweise bei der Eingabe fehlerhafter Arbeitszeiten wurde bereits im Abschnitt „Eintragen der Arbeitszeit“ im Kapitel 4.1.2 näher eingegangen.

Individualisierbarkeit

Zum Individualisieren der Benutzungsoberfläche können persönliche Einstellungen festgelegt werden. Dazu wird ein entsprechender Reiter zur Verfügung gestellt. Dieser ermöglicht es die Standardansicht auszusuchen und die sichtbaren Projekte und deren Reihenfolge zu bestimmen. Weiterhin können dort Vorgaben einer Arbeitszeit eingetragen werden. Dazu sind das Projekt, der Zeitraum, die Wochentage und die Arbeitszeit zu wählen. Diese Funktion erfordert eine manuelle Bestätigung des Benutzers, da er sonst fehlerhafte Eingaben nicht kontrollieren kann.

Im Zeiterfassungsformular kann der Benutzer ebenfalls seine Sicht verändern. Außerdem ist es ihm möglich verschiedene Interaktionsformen zu nutzen. Für die Arbeit mit der Maus werden Schaltflächen mit „+“ und „-“ angeboten, welche den aktuellen Wert im zugehörigen Feld jeweils um eine Stunde erhöhen oder verringern. Zur alleinigen Navigation mit der Tastatur, ist die Tabulator-Taste zu benutzen. Der Cursor wird dabei zeilenweise weiter gesetzt, da so die Arbeitszeiten für einen Tag in verschiedenen Projekten besonders gut eingetragen werden können. Bei Erreichen der letzten Zeile ist die neue Cursorposition das erste Feld der folgenden Spalte.

Lernförderlichkeit

Um den Benutzer mit Hinweisen zu unterstützen, werden zu allen Buttons und Links Tooltips angezeigt, welche die Funktion kurz zusammenfassen. Das betrifft auch die Reiter im Navigationsbereich.

Weiterhin werden dem Benutzer umfangreiche Aufgabenbeschreibungen zur Verfügung gestellt. Dazu befindet sich hinter jeder Aufgabe ein Fragezeichen in eckigen Klammern ([?]). Der Hint wird dann eingeblendet, wenn die Maus auf diese Zeichen zeigt und verschwindet wieder, wenn der Mauszeiger den Bereich verlässt.

Zusätzlich wird ein Reiter in den Navigationsbereich eingefügt, der den Benutzer zu einer Onlinehilfe führt. Diese enthält ein Inhaltsverzeichnis und eine Suchfunktion.

5 Realisierung

Um das entworfene Konzept überprüfen zu können, wird ein Prototyp entwickelt. Dieser wird dann mittels der Usability-Analyse auf seine Ergonomie untersucht. Bei der Realisierung wird das Hauptaugenmerk auf die Umsetzung der software-ergonomischen Anforderungen (siehe Kapitel 4.2.7) gelegt.

Der realisierte Prototyp und die verwendete Software befinden sich auf der beiliegenden CD-ROM (siehe Anhang A).

5.1 Ablaufumgebung

Zum Betreiben einer datenbankgestützten Webanwendung wird eine entsprechende Ablaufumgebung benötigt. Im Folgenden wird kurz auf die verwendete Programmiersprache, den Webserver und die Datenbank eingegangen.

5.1.1 Programmiersprache

Die Wahl der Programmiersprache ist abhängig vom Softwaresystem. So kann für eine Webanwendung wie das Zeiterfassungssystem jede webfähige Sprache verwendet werden. Für den Prototypen kommt Java 5.0 zum Einsatz. Diese Version wird unter anderem vom eingesetzten Webserver (siehe Kapitel 5.1.2) benötigt, bietet aber auch viele Vorteile aus programmiertechnischer Sicht. So kann einfacher über Collections iteriert werden und das Umkonvertieren bei der Arbeit mit Primitiven und Wrapper-Klassen fällt weg. Außerdem stehen durch den Gebrauch der Generics unter anderem typsichere Listen zur Verfügung. Allerdings benötigt Java zur Ausführung der Programme eine Laufzeitumgebung, welche zusätzlich installiert werden muss.

5.1.2 Webserver

Der eingesetzte Webserver hängt vor allem von der verwendeten Programmiersprache ab. Für Java Webanwendungen eignet sich der Apache Tomcat besonders gut. Dabei handelt es sich um einen in Java realisierten Servlet Container der mittels des JSP-Compilers Jasper auch Java Server Pages in Servlets übersetzen und ausführen kann. In der eingesetzten Version 5.5 erfüllt der Tomcat aber auch alle Anforderungen an einen Webserver.

5.1.3 Datenbank

Bei der Wahl der Datenbank sind viele Faktoren (z. B. aufkommende Datenmenge, Fremdschlüssel, etc.) zu berücksichtigen. Im Prototyp übernimmt eine MySQL-Datenbank in der Version 5.0.22 die persistente Datenspeicherung. Um die referenzielle Integrität der Fremdschlüsselbeziehungen zu sichern, wird die InnoDB Engine verwendet. Somit ist auch die Nutzung von kaskadierendem Löschen (siehe Kapitel 4.2.5) gewährleistet.

5.2 Programmierung

Bei der Entwicklung des Prototyps wurde nicht nur auf die Umsetzung der softwareergonomischen Anforderungen geachtet. Um eine Trennung des Interaktions- und Funktionskomponenten zu gewährleisten, werden JSP-Dateien für die Erstellung der Benutzungsoberfläche verwendet und Servlets für die technische Ausführung der Aufgaben. Weiterhin sind JavaScript- und CSS-Daten in separate Dateien ausgelagert.

Für die Interaktion mit der Webanwendung wird eine Kombination aus klassischem und Ajax-Kommunikationsmodell (siehe Kapitel 2.3) verwendet. Für den Wechsel zwischen den Reitern ist das klassische Modell verantwortlich, für die Datenmanipulation innerhalb der Seiten das Ajax-Modell.

5.2.1 Datenbankschnittstelle

Die Datenbankschnittstelle ist durch eine Java Klasse realisiert, welche eine Datenbankverbindung über JDBC herstellt. Entsprechende Methoden ermöglichen das Auslesen und Aufbereiten sowie das Einfügen und Ändern der Daten.

Beim Anmelden an die Anwendung wird eine Instanz dieser Klasse in der Benutzersession gespeichert und somit in den Interaktions- und Funktionskomponenten zur Verfügung gestellt.

5.2.2 Ajax

Um die Daten aus den Formularen asynchron im Hintergrund an die Datenbank übertragen zu können, kommt Ajax zum Einsatz. Dazu steht eine entsprechende JavaScript-Funktion (siehe Quelltext 5.1) zur Verfügung. Ihr werden der URL der Funktionskomponente, ein Array von Parametern und der Name der Handlerfunktion übergeben. Nach der erfolgreichen Erzeugung des Request-Objekts, wird die Zieladresse zusammengesetzt (siehe Quelltext 5.1, Z.25ff). Diese ergibt sich aus dem allgemeinen URL für die Applikation, dem übergebenen URL und den Parametern. Zum Schluss wird die Handlerfunktion, welche den Response verarbeitet, gesetzt und der Request abgeschickt.

```
1  var req = null;
2  function doAjaxRequest(url, params, handlerFunction){
3     try{
4        req = new XMLHttpRequest();
5     }
6     catch (e){
7        try{
8           req = new ActiveXObject("Msxml2.XMLHTTP");
9        }
10       catch (e){
11          try{
12             req = new ActiveXObject("Microsoft.XMLHTTP");
13          }
14          catch (failed){
15             req = null;
16          }
17       }
18    }
19    if (req == null){
20       alert("Error creating request object!");
21    }
22    else
23    {
24       //applicationUrl has to be set in calling jsp file
25       var completeUrl = applicationUrl + '/' + url;
26       if (params.length > 0){
27          completeUrl += "?";
28          for (var i = 0; i < params.length; i++){
29             completeUrl += params[i][0] + "=" + params[i][1];
30
31             if (i < (params.length - 1)){
32                completeUrl += "&";
33             }
34          }
35       }
36       req.open("POST", completeUrl, true);
37       req.onreadystatechange = handlerFunction;
38       req.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
39       req.send(null);
40    }
41 }
```

Quelltext 5.1: JavaScript Funktion für den Ajax Request

5.2.3 Beispiel

Im folgenden Beispiel wird exemplarisch gezeigt, welche Methoden an der Abarbeitung einer Systemfunktion, hier das Hinzufügen eines oder mehrerer Administratoren, beteiligt sind.

Zuerst wird aus der Interaktionskomponente eine JavaScript Funktion (siehe Quelltext 5.2) aufgerufen, welche die benötigten Parameter für die Abarbeitung der Aufgabe sammelt. Danach folgt das Absenden des Ajax Request.

```
1  function addAdministrator (){
2     var params = new Array();
3     for (var i = 0; i < document.form.systemusers.options.length; i++){
4        if (document.form.systemusers.options[i].selected){
5           params.push(["userId", document.form.systemusers.options[i].value]);
6        }
7     }
8     if (params.length > 0){
9        doAjaxRequest("addAdministrator", params, addAdministratorHandler);
10    }
11 }
```

Quelltext 5.2: JavaScript Funktion zum Hinzufügen eines Administrators

Die Funktionskomponente (siehe Quelltext 5.3) empfängt den Request, wertet ihn aus und ruft eine entsprechende Methode in der Datenbankschnittstelle auf (siehe Quelltext 5.3, Z.12). Mittels der Klasse `AjaxXmlBuilder` werden die Antwortdaten in eine geeignete XML-Struktur gebracht.

```
1  public String getXmlContent (HttpServletRequest request, HttpServletResponse response)
       throws Exception{
2     AjaxXmlBuilder ajaxXmlBuilder = new AjaxXmlBuilder();
3     String[] userIds = request.getParameterValues("userId");
4     try{
5        if (userIds == null || userIds.length < 1){
6           throw new SQLException("Es wurden nicht alle Parameter übertragen!", "10000");
7        }
8        List<Person> administratorsToAdd = new ArrayList<Person>();
9        for (int i = 0; i < userIds.length; i++){
10          administratorsToAdd.add(new Person(Integer.parseInt(userIds[i])));
11       }
12       ((PersistenceManager) request.getSession().getAttribute("persistenceManager")).
             addAdministrator(administratorsToAdd);
13       ajaxXmlBuilder.addItem("success", "success");
14    }
15    catch (SQLException e){
16       ajaxXmlBuilder.addItem("errorMsg", e.getMessage());
17    }
18    return ajaxXmlBuilder.toString();
19 }
```

Quelltext 5.3: Servlet zum Hinzufügen eines Administrators

In der Funktion der Datenbankschnittstelle (siehe Quelltext 5.4) werden die Daten in SQL-Statements übertragen. Ihre Ausführung findet in einer Transaktion statt, um im Fehlerfall alle bisherigen Aktionen rückgängig machen zu können (siehe Quelltext 5.4, Z.11).

```
1  public void addAdministrator (List<Person> administratorsToAdd) throws SQLException{
2     this.commandList.clear();
3     for (Person person : administratorsToAdd){
4        this.commandList.add("INSERT INTO administrator (person_id) VALUES (" + person.getId
             () + ");");
5     }
6     try{
7        this.connectDbAndExecute();
8        this.connection.commit();
9     }
10    catch (SQLException e){
11       this.connection.rollback();
12       throw e;
13    }
14 }
```

Quelltext 5.4: Funktion in der Datenbankschnittstelle

Zum Schluss wird die Handlerfunktion (siehe Quelltext 5.5) von der Ajax-Engine aufgerufen. Sie übernimmt das Parsen der übertragenen XML-Daten. Trägt das erste Element den Namen „errorMsg“ wird die Fehlermeldung ausgegeben (siehe Quelltext 5.5, Z.14), andernfalls verschiebt eine Funktion die markierten Einträge aus der Liste der Systembenutzer in die der Administratoren (siehe Quelltext 5.5, Z.17).

```
1  function addAdministratorHandler (){
2     switch(req.readyState){
3        case 4:
4           if(req.status!=200){
5              showStatusError(req.status);
6           }
7           else{
8              xml = req.responseXML;
9              var parser = new ResponseXmlParser();
10             parser.load(req);
11             var results = parser.itemList;
12             if (results.length > 0){
13                if (results[0][0] == "errorMsg"){
14                   showErrorMsg(results[0][1]);
15                }
16                else{
17                   moveSelectedOptions(document.form.systemusers,
                         document.form.administrators, true);
18                }
19             }
20          }
21          break;
22       default: return false; break;
23    }
24 }
```

Quelltext 5.5: JavaScript Handlerfunktion für das Hinzufügen eines Administrators

5.3 Prototyp

Im Folgenden werden einige Funktionen und Oberflächen des Prototyps zur Visualisierung der technischen Umsetzung vorgestellt.

Nach der erfolgreichen Installation des Zeiterfassungssystems, muss es zuerst initialisiert werden. Dazu ist ein Administrator festzulegen, welcher die Systemressourcen verwaltet. Hierfür steht die Initialisierungsseite zur Verfügung. Diese prüft, ob es bereits Administratoren im System gibt. Ist das nicht der Fall, bietet sie eine Liste aller LDAP-Benutzer an, aus welcher der erste Systemadministrator auzuwählen ist. Nach erfolgreicher Speicherung, kann sich die festgelegte Person am System anmelden und dort alle weiteren Einstellungen vornehmen.

So müssen unter anderem die Systembenutzer verwaltet werden. Die Oberfläche (siehe Abbildung 5.1) bietet dafür 3 Listen an. Hier muss der Administrator die entsprechenden Einträge markieren und mit den Buttons in die gewünschten Listen verschieben. Alle Elemente zur Abarbeitung der Aufgabe sind von einem HTML-Fieldset (beschrifteter Rand) eingeschlossen und Hilfestellungen (Hinweistexte die beim Überfahren von „[?]“ mit der Maus aufblenden) werden sowohl bei der Aufgabe als auch bei den einzelnen Elementen bereitgestellt. Diese Darstellung findet sich in allen Aufgaben wieder, die mehrere Listen verwenden.

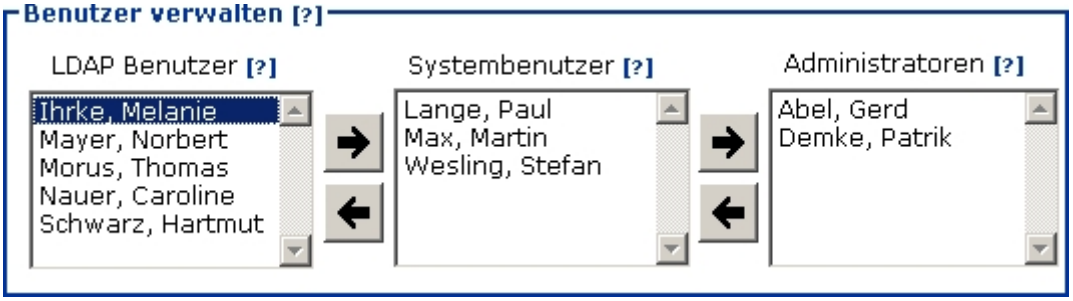


Abbildung 5.1: Screenshot der Benutzerverwaltung

Zur Verwaltung der Projekte steht eine weitere Oberfläche zur Verfügung (siehe Abbildung 5.2). Hier werden die aktuell vorhandenen Projekte inklusive des jeweiligen Projektleiters in einer alphabetisch geordneten Tabelle abgebildet. Eine Funktion zum Anlegen neuer Projekte befindet sich direkt darunter. Sie sortiert neue Projekte automatisch in die Tabelle ein.

Zur Gruppierung der Aufgabe wird ebenfalls ein Fieldset verwendet und der zugehörige Hinweistext zur Aufgabenerklärung ist ebenfalls vorhanden. Um ein Projekt zu bearbeiten, muss der Editierbutton in der entsprechenden Zeile gewählt werden. Dann stehen dort ein Textfeld und eine Dropdown-Liste mit den aktuellen Werten zur Verfügung. Für das Speichern der Änderungen muss der Button mit der Diskette geklickt werden. Verändert sich der Name des Projekts, wird die Zeile an die alphabetisch richtige Stelle verschoben.

In der Dropdown-Liste für den Projektleiter ist kein leerer Eintrag vorhanden. Dies gewährleistet, dass immer ein Leiter für ein Projekt festgelegt wird.

Abbildung 5.2: Screenshot der Projektverwaltung

Abbildung 5.3 zeigt die Applikation aus Sicht eines Mitarbeiters. Der aktive Reiter hat keinen unteren Rand und hebt sich farblich von den anderen ab. Im Formular werden eine Wechselmöglichkeit für Tag-, Wochen- oder Monatsansicht und eine Kalenderfunktion angeboten. Außerdem stehen zur Erfassung der Arbeitszeit neben jedem Feld die Schaltflächen mit „+“ und „-“ zur Verfügung.

Abbildung 5.3: Screenshot vom Zeiterfassungsformular

5.4 Probleme

Bei der Entwicklung des Zeiterfassungssystems traten einige Probleme und Schwierigkeiten auf. Vor allem im Bereich JavaScript und CSS sind diese zu verzeichnen. Im Folgenden werden die Komplikationen kurz erklärt.

5.4.1 Browserkompatibilität

Die Browserkompatibilität ist ein weit verbreitetes Problem bei der Entwicklung von Webanwendungen. Auf der einen Seite stehen die vielen verschiedenen Benutzer der Applikation, die ihren persönlich favorisierten Browser benutzen wollen. Ihre Wahl hängt dabei einerseits vom Betriebssystem und andererseits von den persönlichen Vorlieben ab. Da es also viele Benutzer gibt, welche unterschiedliche Browser nutzen, muss eine hohe Kompatibilität gewährleistet sein. Dem steht allerdings der eingeschränkte Funktionsumfang jedes einzelnen Browsers gegenüber. Es gibt zwar eine Vielzahl gemeinsamer Funktionen, allerdings auch viele browserabhängige. Hier stehen also Kompatibilität und Funktionsumfang im Konflikt (siehe Abbildung 5.4).

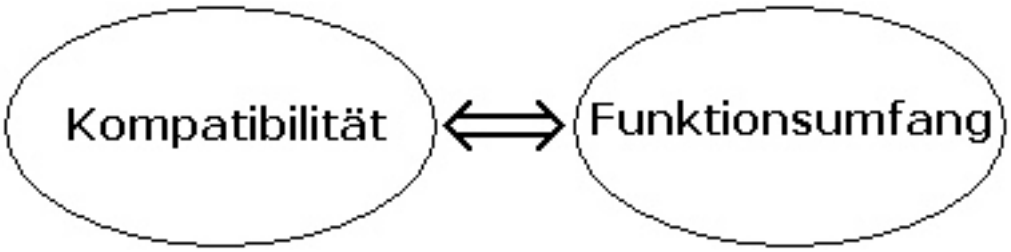


Abbildung 5.4: Kompatibilität vs. Funktionsumfang bei Browsern

Im Prototyp wurde die Kompatibilität auf einen Browser (Mozilla Firefox) beschränkt. Dadurch stehen Funktionen für JavaScript (siehe Kapitel 5.4.2) und CSS (siehe Kapitel 5.4.4) zur Verfügung, die andere Browser gar nicht erst bieten. Da jeder Browserhersteller eigene spezielle Funktionen in seinem Programm unterbringt und sich nicht hundertprozentig an einen Standard hält, ist das Kompatibilitätsproblem nicht so einfach lösbar.

5.4.2 JavaScript

Im Bereich JavaScript gibt es sehr viele Unterschiede zwischen den Browsern. Werden die Funktionen in dem einen Browser korrekt ausgeführt, liefern sie in einem anderen falsche Werte oder sind erst gar nicht bekannt. Um also in allen Browsern die gleiche Funktionalität

zu erreichen, muss ständig auf Browserweichen zurückgegriffen werden. Mit Hilfe dieser Alternativen kann der aktuelle Browsertyp und dessen Version abgefragt werden, um dann die entsprechenden browserkompatiblen JavaScript Funktionen auszuführen. Da aber viele Funktionalitäten browserspezifisch sind, wird der Funktionsumfang stark eingeschränkt. Hier nutzen auch keine Alternativen, sondern nur noch ein entsprechender Kompromiss zwischen Kompatibilität und Funktionsumfang.

Ein weiteres Problem ist, dass JavaScript im Browser deaktiviert werden kann. Eine komplette Deaktivierung hat zur Folge, dass die Anwendung nicht mehr bedienbar ist. Der Benutzer wird dann aufgefordert JavaScript zu aktivieren. Im Firefox ist es aber zusätzlich möglich, einige JavaScipt Funktionen zu erlauben oder zu verbieten. Diese beziehen sich zwar eigentlich nur auf das Browserfenster, haben aber Auswirkung auf die Ausführung mancher Scripte im Zeiterfassungssystem. So werden Eingaben über die Tastatur nicht überprüft und korrigiert. Es reicht also nicht nur eine Aktivierung von JavaScript aus, sondern es müssen auch alle Funktionen der erweiterten JavaScript-Einstellungen erlaubt sein.

Für den Prototyp wurde die Kompatibilität auf den Firefox beschränkt. So kann hier beim Erzeugen eines neuen Buttons die auszuführende Aktion als Attribut gesetzt werden. Die Verwendung eines EventListeners, wie es für den Internet Explorer der Fall wäre, ist hier nicht notwendig.

5.4.3 DOM

Das Einfügen neuer Elemente in den DOM-Baum ist nicht sehr schwierig, abgesehen davon, dass dafür von den verschiedenen Browsern unterschiedliche Funktionen zur Verfügung gestellt werden. Ein größeres Problem ist das Adressieren vorhandener Elemente. Da Zeilenumbrüche oder auch Leerzeichen schon als neues DOM-Element erkannt werden, kommt es z. B. vor, dass eine Dropdown-Liste mit 3 Auswahlpunkten insgesamt 7 Kindknoten hat. Konkret sind das dann 3 Optionen und 4 leere Textknoten, die abwechselnd auftreten. Die Ursache liegt im JSP-Dokument, wo nach dem Erzeugen eines Auswahlpunktes eine neue Zeile begonnen wird. Der Zeilenumbruch ist nun ein neuer Textknoten. Bei Punkten die per JavaScript erzeugt und in den DOM-Baum eingefügt werden, fehlen allerdings die Zeilenumbrüche und somit entstehen keine unnötigen Knoten. Das hat zur Folge, dass nicht mehr mit einer Regelmäßigkeit (z. B. jeder zweite Kindknoten) auf die Elemente zugegriffen werden kann. Um dieses Problem zu lösen, wird entweder der Knotenname überprüft, dieser entspricht dem HTML-Tag-Namen, oder mittels Id auf einzelne Knoten zugegriffen.

5.4.4 CSS

Die Nutzung von CSS sollte eigentlich deutlich unkomplizierter verlaufen, da es verschiedene Versionen gibt, die von den Browsern unterstützt werden. Allerdings halten sich die Hersteller auch hier nicht an alle Vorgaben. So kann im Firefox ein Tabellenrumpf mit einer Scrollleiste versehen werden, wobei der Tabellenkopf feststeht. Der Internet Explorer ignoriert diese Anweisung komplett. Weitere Probleme entstehen, wenn neue DOM-Elemente erzeugt werden, deren Formatierung in einer entsprechenden CSS-Klasse festgelegt sind. Dies kann der IE ebenso nicht umsetzen. Hier kann zwar mit einer Id gearbeitet werden, das verbietet aber die allgemeine Formatierung einer Klasse von Elementen. Für den Prototypen wurde nur die Firefoxkompatibitität hergestellt.

6 Usability-Analyse

Mittels einer Usability-Analyse ist es möglich, die Ergonomie des realisierten Prototyps zu bewerten. So können Defizite festgestellt und positive bzw. negative Merkmale der Software identifiziert werden. Im Folgenden werden die Vorbereitungen und die Vorgehensweise bei der Durchführung dieser Untersuchung erläutert. Nach einer Auswertung des Tests wird der Prototyp anhand der Ergebnisse bewertet.

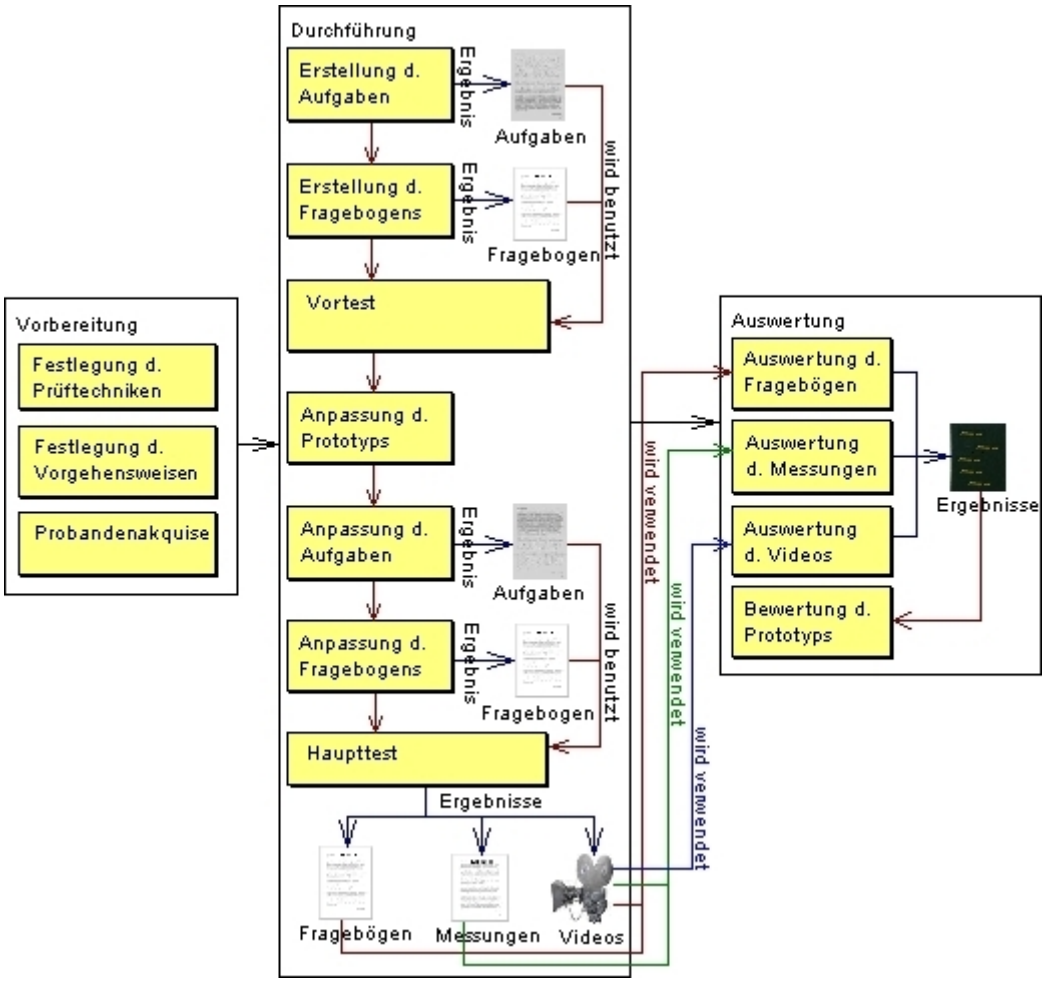


Abbildung 6.1: Ablauf der Usability-Analyse

6.1 Vorbereitung

Um eine Usability-Analyse durchführen zu können, müssen gewisse Vorbereitungen getroffen werden. Zur Ermittlung der anwendbaren Prüftechniken ist der Ort der Untersuchung festzulegen. Weiterhin sind die Bestimmung der Testpersonen und die Definition der Vorgehensweise notwendig.

6.1.1 Festlegung der Prüftechniken

Zur Festlegung der Prüftechniken muss zuerst ein geeigneter Ort für die Untersuchung ausgewählt werden. Dies können die Büroräume eines Kunden oder auch ein Usability-Labor sein. Danach ist es möglich die einzusetzenden Prüftechniken zu bestimmen. Die Usability-Analyse im Rahmen dieser Arbeit wurde im Usability-Labor der Hochschule für Angewandte Wissenschaften Hamburg durchgeführt. Da dort entsprechendes Equipment zur Verfügung steht, können Videodokumentation und Think Aloud als Prüftechniken angewendet werden. Weiterhin kommt ein Fragebogen zum Einsatz, welcher ebenso auf Video aufgezeichnet wird. So ist auch die persönliche Meinung der Probanden digital dokumentiert. Der Einsatz von Eye Tracking war zum Zeitpunkt der Analyse nicht möglich.

6.1.2 Probandenakquise

Nach der Terminfindung mit den zuständigen Mitarbeitern des Usability-Labors muss die Probandenakquise stattfinden. Dazu werden Personen bestimmt, welche den Test durchführen sollen. Hierbei ist es wichtig, das Fachwissen der Probanden zu beachten. Im Rahmen dieser Analyse waren ausschließlich Tester aktiv, die Informatik studieren und denen der Umgang mit Webapplikationen geläufig ist.

6.1.3 Vorgehensweise

Eine weitere vorbereitende Maßnahme ist die Festlegung der Vorgehensweise. Da es sich bei dem Prototypen um die erste Entwicklung handelt, eignet sich die Kombination aus Vergleichstest und explorativem Test besonders gut. So kann die entwickelte Anwendung unter ergonomischen Gesichtspunkten mit einer Referenzsoftware verglichen werden. Weiterhin ist es möglich die Effizienz und Effektivität der Aufgabenerledigung in dieser Applikation zu untersuchen.

6.2 Durchführung

Bei der Durchführung der Usability-Analyse ist die Reihenfolge der Schritte sehr wichtig. So müssen die erstellten Aufgaben und der Fragebogen zuerst in einem Vortest überprüft werden. Anhand der gewonnenen Ergebnisse können diese dann abgestimmt und verfeinert werden, bevor sie im Haupttest zum Einsatz kommen.

Für den Vergleichstest wird die Anwendung „Gleit-Zeit.de“ (siehe Kapitel 3.1.1) als Referenzsoftware genutzt. Das entwickelte System erhält den Namen „Arbeitszeiterfassung“.

6.2.1 Erstellung der Aufgaben

Um den Prototypen testen zu können, werden Aufgaben benötigt, welche die Probanden auszuführen haben. Da die Testdauer eine halbe Stunde nicht übersteigen soll, sind pro Vorgehensweise 2 bis 3 Aufgaben festgelegt. Besondere Beachtung kommt dabei der abstrakten Aufgabenformulierung zuteil. Eine Schritt-für-Schritt-Anleitung würde keine repräsentativen Ergebnisse liefern, da so die Selbstbeschreibungsfähigkeit und die Lernförderlichkeit des Systems nicht überprüft werden kann. Abstrakte Aufgaben dagegen fordern den Probanden auf, sich Zusammenhänge zu erschließen und die angebotenen Informationen der Software zu nutzen.

Der Vergleichstest soll dabei ermitteln, bei welcher Software eine effektivere Arbeitserledigung möglich ist. Dazu werden für jede Teilaufgabe die Dauer, die Anzahl der Maus- und Tastaturklicks und die zurückgelegte Mausstrecke gemessen, welche für die Bewältigung benötigt wurden. Weiterhin gilt es festzustellen, wie gut eine Orientierung in den unterschiedlichen Systemen möglich ist, ob das gewünschte Ergebnis erreicht werden kann und ob es den Erwartungen des Probanden entspricht.

Der explorative Test soll dazu dienen, die ergonomischen Merkmale des Prototypen zu prüfen. Durch die abstrakten Aufgabenstellungen werden keine konkreten Hinweise gegeben, welche Menüpunkte zu wählen sind. So kann festgestellt werden, ob die gewählten Begriffe und Bezeichnungen selbsterklärend sind. Weiterhin soll sich herausstellen, ob die verschiedenen Standards und Interaktionsformen situationsspezifisch eingesetzt werden.

Im Zeiterfassungssystem wird dazu konkret die Benutzung der Shift- oder Strg-Taste zur Mehrfachauswahl bei Listen, die Verwendung der Tabulator-Taste und des Nummernblocks bei der Eingabe rationaler Zahlen oder der Gebrauch der Maus und der angebotenen Schaltflächen bei der Erfassung viele Arbeitsstunden beobachtet. Zusätzlich soll ermittelt werden, ob die bereitgestellten Funktionen und Hilfestellungen von den Probanden angenommen werden.

6.2.2 Erstellung des Fragebogens

Der Fragebogen soll dazu dienen, zusätzliche Informationen von den Probanden zu erhalten. Dazu wurde der ISONORM-Fragebogen (siehe Anhang C) als Vorlage verwendet. Dieser ordnet die Fragen den einzelnen Bewertungskriterien der Software-Ergonomie zu. So wird explizit auf die ergonomischen Merkmale der Software eingegangen. Diese Vorlage kam beim Vortest zum Einsatz, um sie mit Hilfe der Probanden zu verfeinern.

6.2.3 Ablauf des Vortests

Der Vortest dient der Abstimmung und Verfeinerung der Aufgaben und des Fragebogens. Dazu wird zuerst der Prototyp im Usability-Labor installiert und auf seine korrekte Funktionsweise überprüft. Dann werden mindestens zwei Probanden aufgefordert, die Aufgaben jeweils abzuarbeiten und ihre Probleme mitzuteilen. Der Testleiter ist während des gesamten Testlaufs im Testraum anwesend, um eventuell Hilfestellung zu geben. So wird gleichzeitig ermittelt, ob die Aufgaben verständlich formuliert sind und durchgeführt werden können. Im Anschluss an jeden Testlauf wird der Fragebogen gemeinsam von Testleiter und Proband durchgearbeitet. Auf diese Weise ist es möglich, den Fragebogen an die Sprache der Probanden anzupassen, unnötige Fragen zu streichen und neue Fragen zu ermitteln.

6.2.4 Anpassung des Prototyps

Während des Vortests sind schon Abweichungen von den Erwartungen des Testleiters erkennbar. Trotz allem sollten nur noch kleine Verbesserungen an dem Prototypen stattfinden. Diese betrafen bei der verwendeten Anwendung einige Rechtschreibfehler und unvollständige Hinweistexte.

6.2.5 Abstimmung der Aufgaben

Zur Verfeinerung der Aufgaben hat der Vortest ein entscheidendes Ergebnis geliefert. Die Reihenfolge der Aufgaben musste geändert werden. In der ursprünglichen Version sollte zuerst der explorative Test und dann der Vergleichstest durchgeführt werden. Da die Probanden so aber schon vor dem Vergleich Bekanntschaft mit einem der Softwaresysteme machen, würde diese Reihenfolge den Test verfälschen. Somit wird im Haupttest erst der Vergleichstest und dann der explorative Test durchgeführt. Weiterhin mussten noch einige Formulierungen angepasst werden, da sie für die Probanden unverständlich waren. Die

schließlich verwendeten Aufgaben (siehe Anhang B) konnten somit genau auf die Tester und die Testprodukte abgestimmt werden.

6.2.6 Abstimmung des Fragebogens

Zur Anpassung des Fragebogens wurden zuerst die Fragen gestrichen, welche die Probanden gar nicht beantworten konnten. Diese bezogen sich meist auf Funktionen, die die Software in ihrem Anwendungskontext gar nicht anbietet (z. B. Erweiterbarkeit). Aus den übrigen und zusätzlich notwendigen Fragen wurde ein neuer Fragebogen mit einer ähnlichen Struktur entworfen (siehe Anhang D). So wird nun sowohl auf den Vergleich eingegangen als auch auf den explorativen Test. Zur Bewertung der software-ergonomischen Kriterien kommt das Schulnotensystem zum Einsatz. Dies ermöglichte den Probanden im Vortest eine leichtere Bewertung, als eine Einstufung in positive und negative Merkmale. Der Vortest half auch hier, den Fragebogen auf den Test abzustimmen.

6.2.7 Ablauf des Haupttests

Die eigentliche Untersuchung der Software fand dann im Haupttest statt. Dazu wurden sechs Probanden ermittelt, welche nach einer kurzen Einweisung in den Test die Aufgaben ausführten und im Anschluss den Fragebogen beantworteten. Während des Testlaufs war der Testleiter nicht im Untersuchungsraum anwesend, um eine Interaktion mit den Probanden zu vermeiden. So wurde gewährleistet, dass die Testpersonen allein die Mittel der Software nutzen, um die Funktionen und Handlungsabläufe zur Arbeitsbewältigung zu erschließen. Die anschließende Befragung der Probanden fand in Form eines Interviews im Testraum statt. So konnte alles aufgezeichnet werden und der Testleiter hatte die Möglichkeit auf die Aussagen der Probanden einzugehen.

6.3 Auswertung

Zur Auswertung der Usability-Analyse standen neben den ausgefüllten Fragenbögen und den ermittelten Messdaten auch die zusammengeschnittenen Videoaufzeichnungen zur Verfügung. In diesen Filmen sind der jeweilige Desktop und die Testperson aus allen 4 Kameraperspektiven synchron zu sehen.

Die Probanden waren alle männlich, im Alter zwischen 22 und 35 Jahren. Die ersten Kontakte mit dem Computer liegen mindestens 10 Jahre zurück, wobei alle Befragten zur Zeit täglich damit arbeiten.

6.3.1 Vergleichstest

Der Vergleichstest hat einstimmig ergeben, dass die im Rahmen dieser Arbeit entwickelte Software im Vergleich besser bewertet wurde. Die Gründe dafür sind vor allem die intuitive Bedienung und die übersichtlich gestaltete Benutzungsoberfläche.

Die Vergleichssoftware dagegen wird als verwirrend, unlogisch und unübersichtlich bewertet. Das schlechte Look and Feel und die teilweise kryptischen Meldungen haben sich sehr negativ ausgewirkt.

In Abbildung 6.2 ist der Vergleich der Abarbeitungsdauer jeder Aufgabe zu sehen. Über 75 % der Zeit wurde mit Gleit-Zeit.de benötigt und weniger als ein Viertel mit der Arbeitszeiterfassung. Daraus ist zu schließen, dass die übersichtliche Oberfläche und die selbsterklärenden Begriffe zu einem gut bedienbaren System führen. Zusätzlich ist zu bemerken, dass keiner der Probanden die Aufgaben mit Gleit-Zeit.de ohne die Hilfestellungen des Testleiters erfüllen konnte.

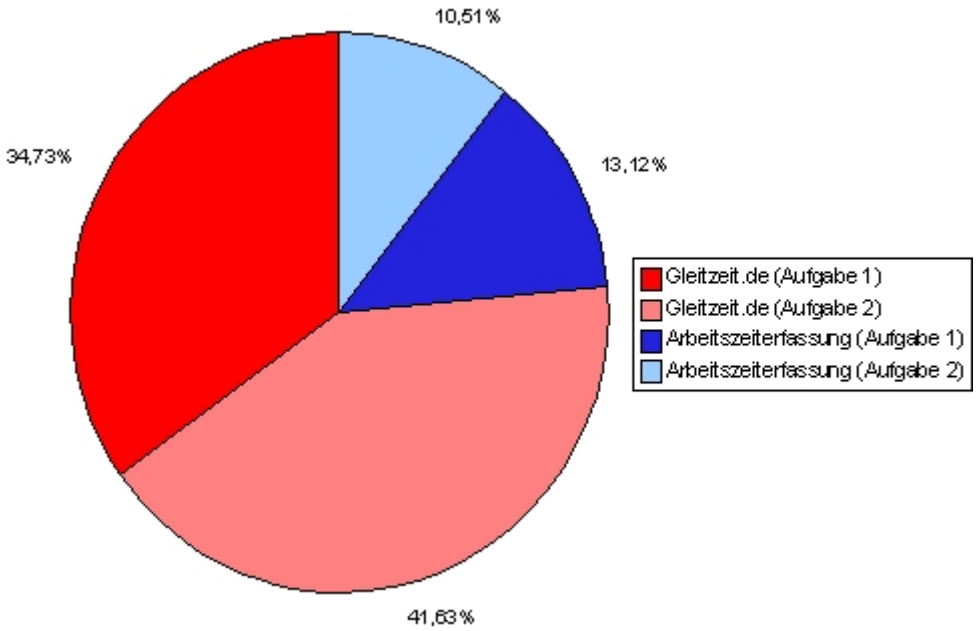


Abbildung 6.2: Durchschnittliche Dauer zur Bewältigung der Aufgaben

Die Abbildung 6.3 zeigt eine Gegenüberstellung weiterer Messwerte des Tests. So ist zu erkennen, dass die zurückgelegten Mausstrecken in Gleit-Zeit.de bis zu 3 mal länger sind als in der Arbeitszeiterfassung, was auch auf die Unübersichtlichkeit der Benutzungsoberfläche zurückzuführen ist. Die Videoaufzeichnungen manifestieren dies. Dort ist zu sehen, dass sich der Benutzer mit Hilfe des Mauszeigers auf dem Bildschirm orientiert. Eine schlecht

strukturierte und verwirrende Oberfläche erfordert dabei mehr Aufwand zum Suchen, was sich in der Mausstrecke wiederspiegelt.

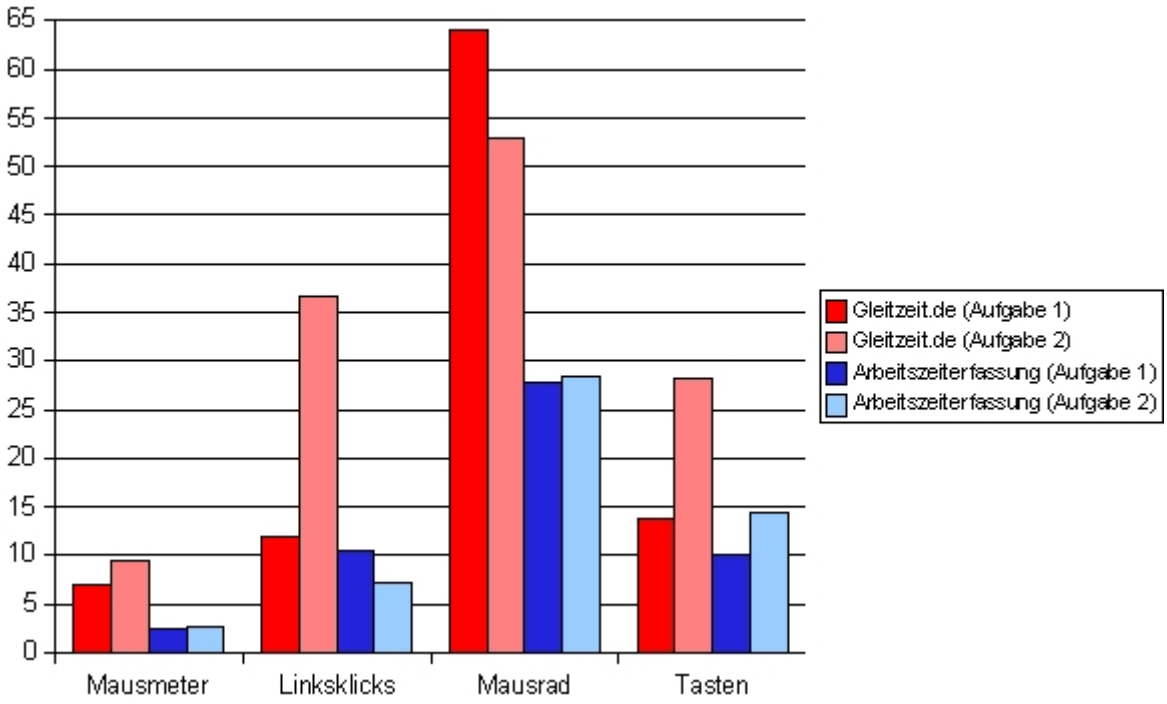


Abbildung 6.3: Gegenüberstellung der weiteren Messwerte

Die Anzahl der Klicks mit der linken Maustaste ist in der Arbeitszeiterfassung auch deutlich weniger. Hier zeigen wieder die Videos den Grund. Während die Benutzer bei Gleit-Zeit.de oft einen Menüpunkt wählten, um die damit verknüpfte Funktionalität zu ergründen, standen in der Arbeitszeiterfassung selbsterklärende Begriffe und eingeblendete Tooltips zur Verfügung.

Die Menge der Tastatureingaben ist zwar bei der Auswertung von Arbeitszeiten (Aufgabe 1) in beiden Anwendungen fast gleich, bei der Erfassung der Stunden (Aufgabe 2) aber in Gleit-Zeit.de doppelt so hoch wie in der Arbeitszeiterfassung. Das ist auf ständig wiederholte Passworteingaben zurückzuführen, was auch die Videoaufzeichnungen belegen.

Weiterhin wurden für Gleit-Zeit.de mehr als doppelt so viele Mausradeinheiten gezählt, welche durch den zu großen Arbeitsbereich verursacht wurden.

6.3.2 Explorativer Test

Der explorative Test ergab eine sehr gute Bewertung der Arbeitszeiterfassung (siehe Tabelle 6.1). Am schlechtesten hat dabei die Erwartungskonformität abgeschnitten. Das ist auf die

fehlenden Rückmeldungen beim Erfassen der Arbeitszeiten zurückzuführen, was ein wenig Unsicherheit bei den Probanden auslöste.

Bewertungskriterium	**Note**
Aufgabenangemessenheit	1,36
Selbstbeschreibungsfähigkeit	1,34
Steuerbarkeit	1,29
Erwartungskonformität	1,61
Fehlertoleranz	1,2
Individualisierbarkeit	1,37
Lernförderlichkeit	1,3
Gesamtnote	**1,35**

Tabelle 6.1: Durchschnittsnoten des explorativen Tests

Wie vermutet war das automatische Speichern neu. Die Tester suchten nach der Eingabe der Arbeitszeiten einen Button zum Speichern. Nach kurzer Zeit gewöhnten sie sich allerdings daran, wobei mehrfach eine Rückmeldung gewünscht wurde.

Entgegen der Erwartung benutzten die Probanden die verschiedenen Interaktionsformen nicht situationsabhängig. Personen, welche die Maus bevorzugen, gebrauchten die Tastatur lediglich um rationale Zahlen einzugeben. Tastaturbenutzer hingegen versuchten alle Arbeiten ohne Maus zu erledigen. Die Gründe dafür waren nicht die unterschiedlichen Situationen, sondern die Gewohnheiten der Anwender.

Weiterhin wurde angenommen, dass die umfangreichen Hilfetexte zur Erklärung der Aufgaben kaum genutzt werden. Dies bestätigt auch der Test. Der Grund ist vor allem die intuitive Bedienbarkeit der Anwendung und der Erfahrungen der Tester im Umgang mit Computern. Unerfahrene Benutzer würden eine solche Funktion sicher mehr in Anspruch nehmen.

Insgesamt können sich alle Probanden die Software in einer realen Umgebung vorstellen, wobei sie allerdings vorher noch verfeinert werden muss. Dies betrifft unter anderem die Verwendung von Aktivitäten für die einzelnen Projekte und die Erweiterung der Auswertungsfunktion.

6.4 Bewertung des Prototyps

Da die Probanden den Test als repräsentativ einschätzten und die Noten alle sehr gut sind, ist die Anwendung als positiv zu bewerten. Für ein marktreifes Produkt muss die Software allerdings noch verfeinert und um einige Details erweitert werden.

Das automatische Speichern benötigt zwar eine kurze Eingewöhnungsphase und ein gewisses Vertrauen in die Software, trägt aber maßgeblich zu der komfortablen Benutzungsoberfläche bei. Da alle Probanden diese Eigenschaft des Systems als sehr positiv empfanden, kann sie richtungsweisend für Webapplikationen sein.

Das Anbieten verschiedener Interaktionsformen ist unbedingt notwendig, da die Benutzer nicht gezwungen werden, entgegen ihrer Gewohnheiten mit der Oberfläche zu arbeiten. So wird die Anpassung der Software an die menschlichen Bedürfnisse unterstützt.

7 Zusammenfassung

Um die Arbeit zusammenzufassen, wird auf den aktuellen Stand der Entwicklung eingegangen. Danach folgt ein Ausblick auf die Weiterentwicklung der Anwendung und zum Schluss eine Bewertung der Arbeit.

7.1 Aktueller Entwicklungsstand

Bis zum Abschluss dieser Arbeit sind sowohl die software-ergonomischen Anforderungen (siehe Kapitel 3.2.2) als auch die funktionalen Anforderungen (siehe Kapitel 3.2.1.1) der Prioritätsklasse „must“ und „should“ konzeptionell entwickelt und prototypisch realisiert worden. Bei der softwaretechnischen Umsetzung wurde teilweise auf sehr einfache Lösungen zurückgegriffen, da die Ergonomie der Benutzungsoberfläche im Vordergrund stand.

7.2 Ausblick

Um ein marktreifes Produkt zu erstellen, muss die Anwendung ingenieursmäßig entwickelt werden. Dazu sind vorhandene Technologien und Frameworks einzusetzen. Für die Oberflächengestaltung bietet sich die Verwendung des Google Widged Toolkit an. Dies gewährleistet unter anderem die Kompatibilität zu verschiedenen Browsern. Als Persistenzschicht ist Hibernate besonders geeignet. Dies verbessert auch die Erweiterbarkeit und Wartbarkeit des Systems, was aus Sicht des Software Engeneering unbedingt notwendig ist.

Da die Ergonomie der Software aber auch von der Performanz der Datenübertragung und der Verfügbarkeit der Teilsysteme abhängt, müssen sowohl die Server Hardware als auch der eingesetzte Webserver und die Datenbank auf die jeweiligen Anforderungen abgestimmt werden.

Zur Verbesserung der Anwendung muss in erster Linie die Auswertungsfunktion angepasst werden, da diese bisher nur sehr rudimentär umgesetzt ist. Weiterhin sind zusätzliche Individualisierungen, wie das Einstellen der Sprache und Farbe oder die Festlegung der Schrittweite beim Addieren und Subtrahieren der Arbeitsstunden mittels Schaltflächen, denkbar.

Außerdem ist der Verbleib der Arbeitszeiten beim Entfernen von Systemressourcen an die Anforderungen der Einsatzumgebung anzupassen. So dürfen bei ständiger Fluktuation der Projektmitglieder nicht deren gearbeitete Stunden gelöscht werden, da diese als Projektaufwände angefallen sind und eventuell einem Kunden in Zahlung gestellt werden müssen.

7.3 Bewertung der Arbeit

Diese Arbeit veranschaulicht, welchen Stellenwert die Software-Ergonomie bei der Entwicklung von Zeiterfassungssystemen hat und wie gravierend sich die Missachtung der Normen und Standards auf den Arbeitsablauf auswirken.

Weiterhin zeigt sich, dass Möglichkeiten existieren, welche die ergonomische Gestaltung von Webapplikationen unterstützen. So kann ihnen z. B. durch Ajax der Charakter einer Rich-Client-Anwendung verliehen werden. Dies wirkt sich vorteilhaft auf die Bedienbarkeit aus, erzeugt allerdings auch Unsicherheit und Misstrauen beim Benutzer, wenn das System nicht genügend Rückmeldungen bei Zustandsänderungen liefert.

Darüber hinaus ist auch ersichtlich, dass die Ergonomie einer Software von vielen verschiedenen Faktoren beeinflusst wird und somit kaum allgemeine Aussagen für alle Softwaresysteme getroffen werden können. Es wird aber sehr deutlich, dass der Autor-Kritiker-Zyklus (siehe Kapitel 2.1.4) maßgeblich zur Ergonomie beiträgt, denn nicht alle Vorschläge der Entwickler werden von den Benutzern angenommen. Deshalb ist es notwendig die Software während des Entwicklungsprozesses rechtzeitig zu untersuchen, um Probleme zu identifizieren und zu beseitigen. Da so das Produkt auf den Benutzer und dessen Bedürfnisse zugeschnitten wird, ist sichergestellt, dass er es auch verwendet.

Eine gute Software-Ergonomie hat also wesentlichen Einfluss auf den Erfolg eines Produktes. Ist es schlecht bedienbar, wird es nicht eingesetzt und das bedeutet: das Projekt ist gescheitert.

A Beiliegende CD-ROM

Dieser Arbeit liegt eine CD mit folgendem Inhalt bei:

- `/bachelorarbeit.pdf`: dieses Dokument
- `/prototyp.zip`: die Ablaufumgebung mit dem funktionierenden Prototyp (wichtige Hinweise zum Betrieb des Prototyps in der Datei „readme.txt“)
  - `apache-tomcat-5.5.17`: der Apache Tomcat Server
  - `jdk1.5.0_07`: das Java SDK 5.0
  - `ldapserver`: eine LDAP Server Simulation
  - `mysql-5.0.22-win32`: das MySQL Datenbanksystem
- `/quellen`: der Quellcode des Prototyps
  - `css`: die CSS Dateien
  - `db`: die Skripte zum Erstellen und Löschen der Datenbank
  - `doc`: die Dokumentation des Quelltextes (Javadoc)
  - `image`: die verwendeten Bilder
  - `javascript`: die JavaScript Dateien
  - `jsp`: die JSP Dateien
  - `lib`: die verwendeten Java-Bibliotheken
  - `src`: die Java-Quellen
  - `web.xml`: die Konfigurationsdatei der Anwendung für den Tomcat
- `/usability_analyse`: die Videos und die Auswertung der Usability-Analyse
  - `messdaten`: die ermittelten Messdaten und eine Zusammenfassung
  - `videos`: die aufgezeichneten Videos
  - `auswertung.xls`: die Auswertung der Messdaten und Fragebögen

B Aufgaben der Usability-Analyse

Für die Usability-Analyse wurde die folgenden Aufgaben eingesetzt, welche die Probanden auszuführen hatten.

Aufgaben

Vergleichstest

Vergleichen Sie GleitZeit.de mit der Arbeitszeiterfassung.

1. Erstellen Sie für den Oktober 2007 eine Auswertung Ihrer gesamten Arbeitszeiten in Listenform.

   Benutzen Sie die folgenden Daten zur Anmeldung:

GleitZeit.de	**Arbeitszeiterfassung**
Benutzername: Adam Passwort: Adam	Benutzername: lange Passwort: lange

   Melden Sie sich von beiden Systemen ab.

2. Erfassen sie für den heutigen Tag eine Arbeitszeit von 8 Stunden.

   Benutzen Sie die folgenden Daten zur Anmeldung:

GleitZeit.de	**Arbeitszeiterfassung**
Benutzername: Bauer Passwort: Bauer	Benutzername: wesling Passwort: wesling

   Melden Sie sich von beiden Systemen ab.

Explorativer Test

Dieser Test bezieht sich nur auf die Anwendung „Arbeitszeiterfassung“.

1. Melden Sie sich als Gerd Abel in der Arbeitszeiterfassung an (Benutzername: abel, Passwort: abel).

   Sie sind nun ein Systemadministrator!

   Erstellen Sie ein neues Projekt mit dem Namen „Usability-Analyse“ und sich selbst als Projektleiter.

   Legen Sie dann die folgenden Personen für dieses Projekt als Mitglieder fest: Patrik Demke, Paul Lange, Melanie Ihrke, Norbert Mayer und Sie selbst.

   Tragen Sie zum Schluss für sich selbst in dem angelegten Projekt für den aktuellen Tag eine Stunden ein.

   Melden Sie sich vom System ab.

2. Melden Sie sich als Paul Lange in der Arbeitszeiterfassung an (Benutzername: lange, Passwort: lange).

   Gestalten Sie nun Ihre Benutzungsoberfläche entsprechend dem folgenden Bild. Stellen Sie dazu alles Notwendige so ein, dass die Ansicht Ihrer Zeiterfassungsseite immer dem folgenden Bild entspricht.

Projekt	Mo 29.	Di 30.	Mi 31.	Do 01.	Fr 02.	Sa 03.	So 04.	Σ
Aufgabenanalyse	2 + -	1 + -	2 + -	2 + -	aa + -	+ -	+ -	7
Konzeption	4 + -	4 + -	1 + -	+ -	4 + -	+ -	+ -	13
Produktspezifikation	2 + -	+ -	3 + -	7 + -	+ -	+ -	+ -	12
TestProjekt	+ -	2 + -	2 + -	+ -	4 + -	+ -	+ -	8
Σ	8	7	8	9	8			40

   Erfassen Sie auch die Zeiten, welche im Bild erkennbar sind.

   Melden Sie sich vom System ab.

3. Melden Sie sich als Martin Max am Zeiterfassungssystem an (Benutzername: max, Passwort: max).

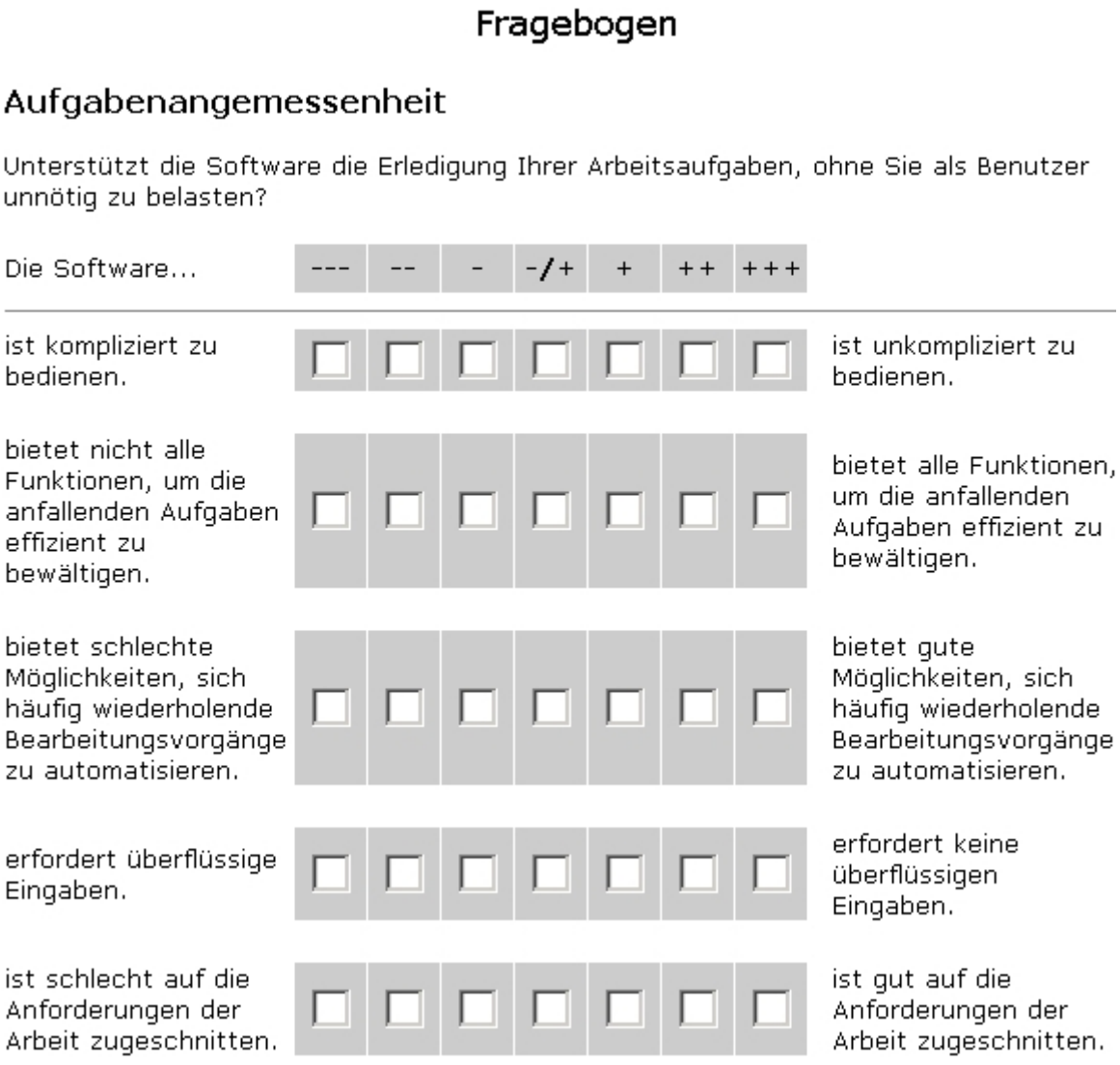
   Erfassen Sie folgenden Zeiten für Ihre Projekte für den aktuellen Tag.

Projekt	**Arbeitszeit**
TestProjekt	3,75
Produktspezifikation	1,5
Aufgabenanalyse	3,0
Konzeption	0,25

   Melden Sie sich vom System ab.

C Alter Fragebogen

Diese Vorlage eines ISO-NORM-Fragebogens wurde im Vortest ein.gesetzt, um den Fragebogen für den Haupttest zu erstellen.



Fragebogen

Aufgabenangemessenheit

Unterstützt die Software die Erledigung Ihrer Arbeitsaufgaben, ohne Sie als Benutzer unnötig zu belasten?

Die Software...	---	--	-	-/+	+	++	+++	
ist kompliziert zu bedienen.	☐	☐	☐	☐	☐	☐	☐	ist unkompliziert zu bedienen.
bietet nicht alle Funktionen, um die anfallenden Aufgaben effizient zu bewältigen.	☐	☐	☐	☐	☐	☐	☐	bietet alle Funktionen, um die anfallenden Aufgaben effizient zu bewältigen.
bietet schlechte Möglichkeiten, sich häufig wiederholende Bearbeitungsvorgänge zu automatisieren.	☐	☐	☐	☐	☐	☐	☐	bietet gute Möglichkeiten, sich häufig wiederholende Bearbeitungsvorgänge zu automatisieren.
erfordert überflüssige Eingaben.	☐	☐	☐	☐	☐	☐	☐	erfordert keine überflüssigen Eingaben.
ist schlecht auf die Anforderungen der Arbeit zugeschnitten.	☐	☐	☐	☐	☐	☐	☐	ist gut auf die Anforderungen der Arbeit zugeschnitten.

Selbstbeschreibungsfähigkeit

Gibt Ihnen die Software genügend Erläuterungen und ist sie in ausreichendem Masse verständlich?

Die Software...	---	--	-	-/+	+	++	+++	
bietet einen schlechten Überblick über ihr Funktionsangebot.	☐	☐	☐	☐	☐	☐	☐	bietet einen guten Überblick über ihr Funktionsangebot.
verwendet schlecht verständliche Begriffe, Bezeichnungen, Abkürzungen oder Symbole in Masken und Menüs.	☐	☐	☐	☐	☐	☐	☐	verwendet gut verständliche Begriffe, Bezeichnungen, Abkürzungen oder Symbole in Masken und Menüs.
liefert in unzureichendem Masse Informationen darüber, welche Eingaben zulässig oder nötig sind.	☐	☐	☐	☐	☐	☐	☐	liefert in zureichendem Masse Informationen darüber, welche Eingaben zulässig oder nötig sind.
bietet auf Verlangen keine situationsspezifischen Erklärungen, die konkret weiterhelfen.	☐	☐	☐	☐	☐	☐	☐	bietet auf Verlangen situationsspezifische Erklärungen, die konkret weiterhelfen.
bietet von sich aus keine situationsspezifischen Erklärungen, die konkret weiterhelfen.	☐	☐	☐	☐	☐	☐	☐	bietet von sich aus situationsspezifische Erklärungen, die konkret weiterhelfen.

Steuerbarkeit

Können Sie als Benutzer die Art und Weise, wie Sie mit der Software arbeiten, beeinflussen?

Die Software...	---	--	-	-/+	+	++	+++	
bietet keine Möglichkeit, die Arbeit an jedem Punkt zu unterbrechen und dort später ohne Verluste wieder weiterzumachen.	☐	☐	☐	☐	☐	☐	☐	bietet die Möglichkeit, die Arbeit an jedem Punkt zu unterbrechen und dort später ohne Verluste wieder weiterzumachen.
erzwingt eine unnötig starre Einhaltung von Bearbeitungsschritten.	☐	☐	☐	☐	☐	☐	☐	erzwingt keine unnötig starre Einhaltung von Bearbeitungsschritten.
ermöglicht keinen leichten Wechsel zwischen einzelnen Menüs oder Masken.	☐	☐	☐	☐	☐	☐	☐	ermöglicht einen leichten Wechsel zwischen einzelnen Menüs oder Masken.
ist so gestaltet, dass der Benutzer nicht beeinflussen kann, wie und welche Informationen am Bildschirm dargeboten werden.	☐	☐	☐	☐	☐	☐	☐	ist so gestaltet, dass der Benutzer beeinflussen kann, wie und welche Informationen am Bildschirm dargeboten werden.
erzwingt unnötige Unterbrechungen der Arbeit.	☐	☐	☐	☐	☐	☐	☐	erzwingt keine unnötigen Unterbrechungen der Arbeit.

Erwartungskonformität

Kommt die Software durch eine einheitliche und verständliche Gestaltung Ihren Erwartungen und Gewohnheiten entgegen?

Die Software...	---	--	-	-/+	+	++	+++	
erschwert die Orientierung, durch eine uneinheitliche Gestaltung.	☐	☐	☐	☐	☐	☐	☐	erleichtert die Orientierung, durch eine einheitliche Gestaltung.
lässt einen im Unklaren darüber, ob eine Eingabe erfolgreich war oder nicht.	☐	☐	☐	☐	☐	☐	☐	lässt einen nicht im Unklaren darüber, ob eine Eingabe erfolgreich war oder nicht.
informiert in unzureichendem Masse über das, was sie gerade macht.	☐	☐	☐	☐	☐	☐	☐	informiert in ausreichendem Masse über das, was sie gerade macht.
reagiert mit schwer vorhersehbaren Bearbeitungszeiten.	☐	☐	☐	☐	☐	☐	☐	reagiert mit gut vorhersehbaren Bearbeitungszeiten.
lässt sich nicht durchgehend nach einem einheitlichen Prinzip bedienen.	☐	☐	☐	☐	☐	☐	☐	lässt sich durchgehend nach einem einheitlichen Prinzip bedienen.

Fehlertoleranz

Bietet Ihnen die Software die Möglichkeit, trotz fehlerhafter Eingaben das beabsichtigte Arbeitsergebnis ohne oder mit geringem Korrekturaufwand zu erreichen?

Die Software...	---	--	-	-/+	+	++	+++	
ist so gestaltet, dass kleine Fehler schwerwiegende Folgen haben können.	☐	☐	☐	☐	☐	☐	☐	ist so gestaltet, dass kleine Fehler keine schwerwiegenden Folgen haben können.
informiert zu spät über fehlerhafte Eingaben.	☐	☐	☐	☐	☐	☐	☐	informiert sofort über fehlerhafte Eingaben.
liefert schlecht verständliche Fehlermeldungen.	☐	☐	☐	☐	☐	☐	☐	liefert gut verständliche Fehlermeldungen.
erfordert bei Fehlern im grossen und ganzen einen hohen Korrekturaufwand.	☐	☐	☐	☐	☐	☐	☐	erfordert bei Fehlern im grossen und ganzen einen geringen Korrekturaufwand.
gibt keine konkreten Hinweise zur Fehlerbehebung.	☐	☐	☐	☐	☐	☐	☐	gibt konkrete Hinweise zur Fehlerbehebung.

Individualisierbarkeit

Können Sie als Benutzer die Software ohne grossen Aufwand auf Ihre individuellen Bedürfnisse und Anforderungen anpassen?

Die Software...	---	--	-	-/+	+	++	+++	
lässt sich von dem Benutzer schwer erweitern, wenn für ihn neue Aufgaben entstehen.	☐	☐	☐	☐	☐	☐	☐	lässt sich von dem Benutzer leicht erweitern, wenn für ihn neue Aufgaben entstehen.
lässt sich von dem Benutzer schlecht an seine persönliche, individuelle Art der Arbeitserledigung anpassen.	☐	☐	☐	☐	☐	☐	☐	lässt sich von dem Benutzer gut an seine persönliche, individuelle Art der Arbeitserledigung anpassen.
eignet sich für Anfänger und Experten nicht gleichermassen, weil der Benutzer sie nur schwer an seinen Kenntnisstand anpassen kann.	☐	☐	☐	☐	☐	☐	☐	eignet sich für Anfänger und Experten gleichermassen, weil der Benutzer sie leicht an seinen Kenntnisstand anpassen kann.
lässt sich – im Rahmen ihres Leistungsumfangs – von dem Benutzer schlecht für unterschiedliche Aufgaben passend einrichten.	☐	☐	☐	☐	☐	☐	☐	lässt sich – im Rahmen ihres Leistungsumfangs – von dem Benutzer gut für unterschiedliche Aufgaben passend einrichten.
ist so gestaltet, dass der Benutzer die Bildschirmdarstellung schlecht an seine individuellen Bedürfnisse anpassen kann.	☐	☐	☐	☐	☐	☐	☐	ist so gestaltet, dass der Benutzer die Bildschirmdarstellung gut an seine individuellen Bedürfnisse anpassen kann.

Lernförderlichkeit

Ist die Software so gestaltet, dass Sie sich ohne grossen Aufwand in sie einarbeiten konnten und bietet sie auch dann Unterstützung, wenn Sie neue Funktionen lernen möchten?

Die Software...	---	--	-	-/+	+	++	+++	
erfordert viel Zeit zum Erlernen.	☐	☐	☐	☐	☐	☐	☐	erfordert wenig Zeit zum Erlernen.
ermutigt nicht dazu, auch neue Funktionen auszuprobieren.	☐	☐	☐	☐	☐	☐	☐	ermutigt dazu, auch neue Funktionen auszuprobieren.
erfordert, dass man sich viele Details merken muss.	☐	☐	☐	☐	☐	☐	☐	erfordert nicht, dass man sich viele Details merken muss.
ist so gestaltet, dass sich einmal Gelerntes schlecht einprägt.	☐	☐	☐	☐	☐	☐	☐	ist so gestaltet, dass sich einmal Gelerntes gut einprägt.
ist schlecht ohne fremde Hilfe oder Handbuch erlernbar.	☐	☐	☐	☐	☐	☐	☐	ist gut ohne fremde Hilfe oder Handbuch erlernbar.

Zum Schluss

Wie gut beherrschen Sie die beurteilte Software?

	---	--	-	-/+	+	++	+++	
sehr schlecht	☐	☐	☐	☐	☐	☐	☐	sehr gut

Wie lange benutzen Sie ungefähr schon...

Computer? ☐ Jahre ☐ Monate

Wie häufig benutzen Sie durchschnittlich...

Computer? ☐ Stunden pro Woche

Ihr Geschlecht? ☐ weiblich ☐ männlich

Wie alt sind Sie? ☐ Jahre

D Fragebogen der Usability-Analyse

Für die Befragung der Probanden im Rahmen der Usability-Analyse wurde der folgende Fragebogen eingesetzt.

Fragebogen

Vergleichstest

Bewerten Sie die beiden Anwendungen, die im Vergleichstest gegenüber gestellt wurden!

1. Welche Anwendung hat Ihnen besser gefallen?
2. Was sind die ausschlaggebenden Kriterien für Ihre Entscheidung?
3. Was ist Ihre Kritik an der anderen Software?

Explorativer Test

Im Folgenden ist das System „Arbeitszeiterfassung“ zu bewerten. Vergeben Sie dazu Schulnoten von 1 (sehr gut) bis 6 (ungenügend)!

Aufgabenangemessenheit

1. Wie bewerten Sie die Bedienbarkeit der Software?
2. Wie gut ist der angebotene Funktionsumfang um die Aufgaben zu bewältigen?
3. Wie gut werden überflüssige Eingaben vermieden?
4. Wie gut ist die Anwendung auf die Anforderungen zugeschnitten?

Selbstbeschreibungsfähigkeit

1. Wie gut bietet Ihnen die Software einen Überblick über den Funktionsumfang?
2. Wie gut verstehen Sie die verwendeten Begriffe und Bezeichnungen?
3. Wie gut werden Sie darüber informiert, welche Eingaben zulässig sind?
4. Wie gut hilft Ihnen die Software mit situationsspezifischen Erklärungen weiter?

Steuerbarkeit

1. Wie gut wird es Ihnen ermöglicht, die Arbeit an einem Punkt zu unterbrechen, und dort später ohne Verlust weiterzumachen?
2. Wie gut ist das Wechseln zwischen Menüs oder Eingabemasken möglich?
3. Wie gut können Sie beeinflussen, welche Informationen am Bildschirm dargeboten werden?
4. Wie gut werden unnötige Unterbrechungen der Arbeit vermieden?

Erwartungskonformität

1. Wie gut können Sie sich in der Benutzungsoberfläche orientieren?
2. Wie gut werden Sie darüber informiert, ob Eingaben erfolgreich waren, oder nicht?
3. Wie sehr entsprechen die ausgeführten Aktionen ihren Erwartungen?
4. Wie sehr ist eine einheitliche Bedienung der Software möglich?

Fehlertoleranz

1. Wie gut reagiert die Software auf kleine Fehler?
2. Wie gut werden Sie über Fehler informiert?
3. Wie verständlich sind die Fehlermeldungen formuliert?
4. Wie gut werden Sie bei der Korrektur eines Fehlers unterstützt?

Individualisierbarkeit

1. Wie können Sie die Software an Ihre persönliche Art der Arbeitserledigung anpassen?
2. Wie können Sie die Bildschirmdarstellung an Ihre individuellen Bedürfnisse anpassen?
3. Wie sehr können Sie die Software an Ihren Kenntnisstand anpassen?
4. Wie gut werden Ihnen verschiedene Interaktionsformen angeboten?

Lernförderlichkeit

1. Wie gut können Sie sich im System ohne fremde Hilfe und Handbuch einarbeiten?
2. Wie sehr ermutigt Sie die Software, neue Funktionen auszuprobieren?
3. Wie sehr müssen Sie sich Details merken?
4. Wie unterstützen Sie die angebotenen Hilfestellungen?

Bei den weiteren Fragen soll die Anwendung nicht mehr benotet werden. Bitte geben Sie, wenn möglich, ausführliche Antworten!

Allgemein

1. Was war für Sie ungewöhnlich oder neu?
2. Wie kamen Sie mit dem automatischen Speichern zurecht?
3. Können Sie sich einen Einsatz der „Arbeitszeiterfassung“ in einer realen Umgebung vorstellen?
4. Würden Sie die Software weiterempfehlen?
5. Haben Sie Verbesserungsvorschläge?

Kritik am Test

1. Wie sind Sie mit den Aufgaben zurecht gekommen?
2. Bei welchen Aufgaben sind Probleme aufgetreten?
3. Liefert dieser Test, ihrer Meinung nach, eine repräsentative Bewertung der Software?

Zusatzinformationen

1. Seit wievielen Jahre benutzen Sie einen Computer?
2. Wie oft benutzen Sie einen Computer (täglich/selten/nie)?
3. Wie ist Ihr Geschlecht (weiblich/männlich)?
4. Wie alt sind Sie?

Abkürzungsverzeichnis

AJAX	Asynchronous JavaScript and XML
API	Application Programmers Interface
BetrVG	Betriebsvereinbarungsgesetz
CSS	Cascading Style Sheets
CSV	Comma Separated Values
DOM	Document Object Model
ERD	Entity Relationship Diagram
FAQ	Frequently Asked Questions
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IE	Microsoft Internet Explorer
JDBC	Java Database Connectivity
JSP	Java Server Pages
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
PC	Personal Computer
PDA	Personal Digital Assistant
PHP	Hypertext Preprocessor
RFID	Radio Frequency Identification
SUMI	Software Usability Measurement Inventory
URL	Uniform Resource Locator
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language
XSLT	Extensible Stylesheet Language Transformation

Literaturverzeichnis

AK06 ALCHAER, Maneifa ; KHOSHRAFTAR, Said D.: *Usability: Gestaltungsrichtlinien für die Entwicklung benutzerfreundlicher Internet-Angebote*, Hochschule für Angewandte Wissenschaften Hamburg, Bachelorarbeit, 2006. http://users.informatik.haw-hamburg.de/~use-lab/papers/BA-Alchaer-Djavan.pdf

Bal88 BALZERT, Helmut (Hrsg.): *Einführung in die Software-Ergonomie*. 2. Aufl. Berlin : de Gruyter, 1988. – ISBN 3–11–011939–0

Bal04 BALZERT, Heide (Hrsg.): *Webdesign & Web-Ergonomie : Websites professionell gestalten*. 1. Aufl. Herdecke : W3L, 2004. – ISBN 3–937137–02–5

Bar06 BARTUSSEK, Wolfram: *Software Engineering : Vorgehensmodelle*, Fachbereich Informatik, Hochschule Darmstadt, Script, 2006. http://www.fbi.h-da.de/fileadmin/personal/w.bartussek/Skripten/SE11.pdf

Ber04 BERUFSFORSCHUNGS- UND BERATUNGSINSTITUT FÜR INTERDISZIPLINÄRE TECHNIKGESTALTUNG E.V. (Hrsg.): *Umsetzung software-ergonomischer Anforderungen*. Version: 2004. http://www.ergusto.de/download/produktblaetter/Ergonomie_Beratung.pdf, Abruf: 18. September 2007

Bun06 BUNDESMINISTERIUM DER JUSTIZ (Hrsg.): *Verordnung über Sicherheit und Gesundheitsschutz bei der Arbeit an Bildschirmgeräten (Bildschirmarbeitsverordnung - BildscharbV)*. Version: Oktober 2006. http://bundesrecht.juris.de/bundesrecht/bildscharbv/gesamt.pdf, Abruf: 22. September 2007

Dah06 DAHM, Markus (Hrsg.): *Grundlagen der Mensch-Computer-Interaktion*. 1. Aufl. München : Pearson Studium, 2006. – ISBN 3–8273–7175–9

EOO94 EBERLEH, Edmund (Hrsg.) ; OBERQUELLE, Horst (Hrsg.) ; OPPERMANN, Reinhard (Hrsg.): *Einführung in die Software-Ergonomie : Gestaltung graphisch-interaktiver Systeme ; Prinzipien, Werkzeuge, Lösungen*. 2. Aufl. Berlin : de Gruyter, 1994. – ISBN 3–11–013814–X

Fäh87 FÄHNRICH, Klaus P. (Hrsg.): *Software Ergonomie : State of the Art 5*. 1. Aufl. München : Oldenbourg, 1987. – ISBN 3–486–20523–4

Fra05 FRAUNHOFER INSTITUT FIT (Hrsg.): *Usability 1x1*. Version: März 2005. http://www.fit-fuer-usability.de/1x1/uebersicht.html, Abruf: 15. Oktober 2007

Gar05 GARRETT, Jesse J.: *Ajax: A New Approach to Web Applications*, Adaptive Path, Abhandlung, 2005. http://www.adaptivepath.com/ideas/essays/archives/000385.php

GBK99 GÖRNER, Claus (Hrsg.) ; BEU, Andreas (Hrsg.) ; KOLLER, Franz (Hrsg.): *Der Bildschirmarbeitsplatz : Softwareentwicklung mit DIN EN ISO 9241*. 1. Aufl. Berlin : Beuth, 1999. – ISBN 3–410–14339–4

Gre01 GREEVE, Gina: *Der EDV-Arbeitsplatz und seine straf- und bußgeldrechtlichen Risiken*, Arbeitsgemeinschaft Strafrecht des DAV (Deutscher Anwaltverein) e.V., Vortrag / Referat, 2001. http://www.ag-strafrecht.de/archivver/beitraggreeve.htm

Hel00 HELLBARDT, Günter (Hrsg.): *Vorlesung Software-Ergonomie*. Version: 2000. http://www1.informatik.uni-jena.de/Lehre/SoftErg/vor_e100.htm, Abruf: 17. September 2007

Hel05 HELLBUSCH, Jan E. (Hrsg.): *Barrierefreies Webdesign : Praxishandbuch für Webgestaltung und grafische Programmoberflächen*. 1. Aufl. Heidelberg : dpunkt-Verlag, 2005. – ISBN 3–89864–260–7

Her94 HERCZEG, Michael (Hrsg.): *Software Ergonomie : Grundlagen der Mensch-Computer-Kommunikation*. 1. Aufl. Bonn : Addison-Wesley, 1994. – ISBN 3–89319–615–3

HV03 HEINSEN, Sven (Hrsg.) ; VOGT, Petra (Hrsg.): *Usability praktisch umsetzen : Handbuch für Software, Web, Mobile Devices und andere interaktive Produkte*. 1. Aufl. München : Hanser, 2003. – ISBN 3–446–22272–3

Ins02 INSTITUT FÜR SOFTWARE-ERGONOMIE UND USABILITY (Hrsg.): *Was ist Usability?* Version: 2002. http://www.usability.ch/Deutsch/usab.htm, Abruf: 18. September 2007

Jan00 JANSSEN, Axel (Hrsg.): *Muster-Betriebsvereinbarung Zeiterfassung und Arbeitszeitkonten/gleitende Arbeitszeit*. Version: 2000. http://www.jes-beratung.de/zeiterfassung.html, Abruf: 28. September 2007

KRZ05 KAHLBRANDT, Bernd ; RAASCH, Jörg ; ZUKUNFT, Olaf: *Software Engineering : 047v-WAM*, Hochschule für Angewandte Wissenschaften Hamburg, Folien, 2005

Lau87 LAUTER, Barbara (Hrsg.): *Software-Ergonomie in der Praxis*. 1. Aufl. München : Oldenbourg, 1987. – ISBN 3–486–20459–9

Med07 MEDIA SUPERVISION, SOFTWARE CONSULTING GMBH (Hrsg.): *Usability Services*. Version: April 2007. http://www.mediasupervision.de/usabilityservices, Abruf: 10. Oktober 2007

Mek03 MEKELBURG, Hans-G. (Hrsg.): *Qualitätssicherung der Ergonomie*. Version: Oktober 2003. http://home.nordwest.net/hgm/ergo/kap-qs.htm, Abruf: 13. Oktober 2007

PRO07 PROJECT CONSULT GMBH (Hrsg.): *Wissen/PM-Glossar*. Version: April 2007. http://www.project-consult.net/portal.asp?UR=85&SA=Z, Abruf: 28. September 2007

Ram07 RAMPL, Hansjörg (Hrsg.): *Usability Testing - Wahl der Testform*. Version: 2007. http://www.handbuch-usability.de/usability-testing.html, Abruf: 13. Oktober 2007

Sch83 SCHMITT, Alfred A. (Hrsg.): *Dialogsysteme : kommunikative Schnittstellen, Software-Ergonomie und Systemgestaltung*. 1. Aufl. Mannheim : Bibliographisches Institut, 1983. – ISBN 3–411–01663–9

SP05 SHNEIDERMAN, Ben (Hrsg.) ; PLAISANT, Catherine (Hrsg.): *Designing the user interface : strategies for effective human-computer interaction*. 4. Aufl. Boston : Addison-Wesley, 2005. – ISBN 0–321–26978–0

SRC99 STARY, Ch. (Hrsg.) ; RIESENECKER-CABA, Th. (Hrsg.): *EU-CON II : Softwareergonomische Bewertung und Gestaltung von Bildschirmarbeit*. 1. Aufl. Dortmund : Wirtschaftsverlag NW, 1999. – ISBN 3–89701–272–3

Sta96 STARY, Christian (Hrsg.): *Interaktive Systeme : Software-Entwicklung und Software-Ergonomie*. 2. Aufl. Braunschweig : Vieweg, 1996. – ISBN 3–528–15384–9

Sta06 STATISTISCHES BUNDESAMT DEUTSCHLAND (Hrsg.): *Moderne Informations- und Kommunikationstechnologien in Deutschland*. Version: Januar 2006. http://www.destatis.de/jetspeed/portal/cms/Sites/destatis/Internet/DE/Content/Publikationen/Querschnittsveroeffentlichungen/WirtschaftStatistik/Informationsgesellschaft/Moderneinfokommunikation,property=file.pdf, Abruf: 20. September 2007

Sta07 STATISTISCHES BUNDESAMT DEUTSCHLAND (Hrsg.): *58 Prozent der Beschäftigten nutzen im Arbeitsalltag einen PC*. Version: Februar 2007. http://www.destatis.de/jetspeed/portal/cms/Sites/destatis/Internet/DE/Presse/pm/2007/02/PD07__047__ikt.psml, Abruf: 20. September 2007

Szw06 SZWILLUS, Gerd: *Usability Engineering : Benutzertests*, Universität Paderborn, Folien, 2006. http://wwwcs.uni-paderborn.de/cs/ag-szwillus/lehre/ws05_06/UE/UE_Development_2.pdf

Tan03 TANENBAUM, Andrew S. (Hrsg.): *Computernetzwerke*. 4. Aufl. München : Pearson Studium, 2003. – ISBN 3–8273–7046–9

Usa USABILITY-LABOR, HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG (Hrsg.): *Rahmen-Konzept des Usability-Labors*. http://users.informatik.haw-hamburg.de/~use-lab/konzept.html, Abruf: 09. Oktober 2007

Wik07a WIKIMEDIA COMMONS (Hrsg.): *Ajax-vergleich.svg*. Version: Juni 2007. http://upload.wikimedia.org/wikipedia/commons/d/d8/Ajax-vergleich.svg, Abruf: 08. Oktober 2007

Wik07b WIKIMEDIA COMMONS (Hrsg.): *Prozessfluss-ajax.svg*. Version: Juni 2007. http://upload.wikimedia.org/wikipedia/commons/4/4a/Prozessfluss-ajax.svg, Abruf: 08. Oktober 2007

Wik07c WIKIMEDIA COMMONS (Hrsg.): *Prozessfluss-traditionell.svg*. Version: Juni 2007. http://upload.wikimedia.org/wikipedia/commons/8/8f/Prozessfluss-traditionell.svg, Abruf: 08. Oktober 2007

Wir05 WIRTH, Thomas (Hrsg.): *Die EN ISO 9241 - 10*. Version: Oktober 2005. http://www.kommdesign.de/texte/din.htm, Abruf: 17. September 2007

ZZ94 ZEIDLER, Alfred (Hrsg.) ; ZELLNER, Rudolf (Hrsg.): *Software-Ergonomie : Techniken der Dialoggestaltung*. 2. Aufl. München : Oldenbourg, 1994. – ISBN 3–486–22720–3

Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §22(4) ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 22. November 2007	
Ort, Datum	Unterschrift