Stefan Schumacher
Dateiversion 1.10

Hausarbeit

Freie Software zum Projektmanagement

Stefan Schumacher

31.03.2010

Zusammenfassung

In dieser Hausarbeit beschreibe ich die Spezifikationen verschiedener freier Projektmanagementprogramme, deren Gebrauch sowie allgemeine Vor- und Nachteile von Projektmanagementsoftware.

Die Hausarbeit entstand im Rahmen des Seminars »Einführung in das Projektmanagement«

Schlüsselwörter

Projektmanagement, Projektmanagementsoftware, Open Source, TaskJuggler

INHALTSVERZEICHNIS

ABBILDUNGSVERZEICHNIS 5
1 EINFÜHRUNG 6
1.1 Vom Zwecke dieses Werkes 6
2 SOFTWARE 8
2.1 Zeit- und Terminplanung 8
2.2 Kommunikation 9
2.2.1 synchrone Kommunikation mit IRC, Jabber und Co. 10
2.2.2 Mailinglisten, Foren und Wikis 11
2.3 Projektmanagementsoftware 13
2.3.1 GNOME Planner 13
2.3.2 Redmine 13
2.3.3 dotProject 16
2.3.4 TaskJuggler 16
3 CODA 26
3.1 Kritik an und Nachteile von Projektmanagementsoftware 26
3.2 Fazit 28
4 LITERATUR 29

ABBILDUNGSVERZEICHNIS

Abbildung 2.1 Mozilla Sunbird auf Windows XP *Quelle: http://upload.wikimedia.org/wikipedia/commons/8/83/Mozilla_Sunbird.png* 8

Abbildung 2.2 Das Portal von Horde Groupware *Quelle: http://upload.wikimedia.org/wikipedia/commons/6/6b/Horde-portal.png* 9

Abbildung 2.3 Zitat aus dem PhatNet http://german-bash.org/11488 10

Abbildung 2.4 Der Mailclient Mutt zeigt einen Diskussionsfaden *Quelle: selbst erstellt* 12

Abbildung 2.5 GNOME Planner mit einem Gantt-Diagramm. *Quelle: selbst erstellt* 14

Abbildung 2.6 Die Startseite eines Redmine-Projektes. *Quelle: selbst erstellt* 15

Abbildung 2.7 Der Editor von TaskJugglerUI mit einem leeren Projekt *Quelle: selbst erstellt* 17

Abbildung 2.8 Der GanttReport zeigt die einzelnen Aufgaben als Baum und als Gantt-Diagramm an. *Quelle: selbst erstellt* 18

Abbildung 2.9 Projektdefinition 18

Abbildung 2.10 Mitarbeiterdefinition 19

Abbildung 2.11 Kontendefinition 19

Abbildung 2.12 Definition der Aufgaben 21

Abbildung 2.13 Definition der Berichte 23

Abbildung 2.14 Ressourcenverbrauch buchen 24

1

EINFÜHRUNG

Nur wenige wissen, wie viel man wissen muss,
um zu wissen, wie wenig man weiß.

(Werner Heisenberg)

1.1 VOM ZWECKE DIESES WERKES

Ziel dieser Hausarbeit ist es, einige freie[1] Programme vorzustellen, die im Rahmen des Projektmanagements eingesetzt werden können.

Dazu möchte ich zeigen, in welchen Gebieten die Programme wie eingesetzt werden können, welche Vor- und Nachteile sie haben sowie allgemeine Nachteile und Probleme beim Einsatz von Projektmanagementsoftware beschreiben.

Der Begriff Projektmanagementsoftware ist nicht klar umrissen, es existiert keine brauchbare Definition. Dies dürfte daher rühren, das jedes Projekt per definitionem ein einmaliger Prozess ist, jedes Projekt also eigene Managementsoftware benötigt. Darüberhinaus sagt Projektmanagement noch nichts über den eigentlichen Inhalt des Projektes und damit über die Anforderungen ans Management aus, es kann sich bspw. um eine Hausarbeit, einen Kindergeburtstag oder den Bau einer internationalen Raumstation handeln. Daher ist das Spektrum an Projektmanagementsoftware recht breit.

Neben Software die explizit für das Projektmanagement entworfen wurde, gibt es auch andere Software, die sich in das Projektmanagement eingliedern oder zweckentfremden lässt, bspw. Texteditoren oder Mailclients.

In dieser Hausarbeit befasse ich mich mit einigen ausgewählten größeren Projektmanagementprogrammen sowie einigen kleineren Kommunikationsprogrammen, die sich nach meiner Erfahrung gut für das Projektmanagement eignen.

Die beschriebenen Programme habe ich 2007/08 zur Organisation des Magdeburger Open-Source-Tages[2] 2008 eingesetzt.

Die Vorteile der Software liegen im größten Teil darin, das auf die Daten via Internet zugegriffen werden kann, ein Projektteam daher problemlos weltweit verteilt arbeiten kann und jederzeit Zugriff auf die Daten hat. Außerdem können durch den Einsatz zentraler Server Daten beliebig hin und her kopiert bzw. synchronisiert werden, so das z. B. ein Mitarbeiter ein Datum ändert und diese Änderung sofort allen anderen Teammitgliedern zur Verfügung steht.

Ein weiterer Punkt ist die Archivierung der Daten, d. h. alle Änderungen und Zwischenschritte können beispielsweise in einem sogenannten Repository vorgehalten und später analysiert werden. Dies ermöglicht eine ausführliche Meta-Analyse des abgeschlossenen Projektes und eine sinnvolle Ergänzung zum Projekttagebuch, da so alle Schritte und Änderungen beliebig oft nachvollzogen und analysiert werden können.

Besonders nützlich ist die Automatisierung algorithmisierbarer Prozesse, so können einige Programme beispielsweise selbständig Gantt-Diagramme berechnen und plotten oder Kosten bestimmten Kostenstellen zuordnen und aufrechnen. Das heißt, das sich die Projektleitung nicht mit solch einfachen Aufgaben befassen muss, sondern diese automatisiert erledigen lassen kann.

1 Frei wie in Freiheit, nicht frei wie in Freibier.
2 http://www.open-source-tag.de

Außerdem kann die Projektleitung in der Planungsphase Planspiele durchführen und verschiedene Varianten simulieren, in dem sie beispielsweise die Anzahl der Projektmitarbeiter variiert.

Enthält das Projektmanagement auch Wissensmanagement bzw. verknüpft man beides miteinander, lassen sich Erfahrungen aus einzelnen Projekten aufbereiten und kondensieren. Insbesondere Organisationen die viele Projekte durchführen, wie Unternehmensberatungen, profitieren von dieser Maßnahme erheblich. Durch den Einsatz von Client-Server-Systemen kann jeder Mitarbeiter auf das Projekt-/Wissensmanagementsystem zugreifen und sein »Wissen« einspeisen bzw. anderes »Wissen« auslesen. Somit kann die Organisation über längere Zeiträume hinweg einen Erfahrungsschatz aufbauen und ihn tradieren. Dabei ist aber zu beachten, dass auch hier die Computersysteme keine Wunderwaffe sind, die allerlei organisationale Probleme lösen. Wenn eine Organisation keinen Wert auf Erfahrung und Lernen legt, wird dies nicht schlagartig durch den Einsatz von Computern anders. Freie Software hat außerdem den entscheidenden Vorteil das der Programmquellcode sowie die Datenstrukturen offen liegen. Das heißt die Daten der Benutzer liegen in einem änderbaren Format vor und können beliebig weiterbearbeitet, archiviert oder migriert werden. Einige kommerzielle Programme bieten diese Möglichkeit so nicht, da sie den Benutzer so an der Programm binden wollen.

2

SOFTWARE

2.1 ZEIT- UND TERMINPLANUNG

Eines der wichtigsten Gebiete des Projektmanagements ist die Zeit- und Terminplanung. Es gibt daher innerhalb größerer Projektmanagementpakete entsprechende Module, die der Zeit-/Terminplanung dienen. Daneben gibt es aber auch viele kleinere Programme, die insbesondere der persönlichen Zeitplanung eines einzelnen Benutzers dienen oder die Terminplanung einer Gruppe ermöglichen.

Ein bekanntes und verbreitetes Zeitplanungswerkzeug ist Mozilla Sunbird, ein sogenannter Personal-Information-Manager (PIM). PIM dienen dazu, Termine, Aufgaben, Notizen und Kontakte zu organisieren. Sunbird bietet dazu ein Adressbuch, einen Kalender sowie eine Aufgaben- und Notizverwaltung.

Innerhalb des Kalenders können Aufgaben und Termine eingetragen werden, dazu erfasst Sunbird unter anderem Titel, Ort, Uhrzeit, Beschreibung, Teilnehmer, Termin-Kategorien (z. B. privat, geheim etc.), eine URL und diverse Status.

Sinnvoll für den Einsatz in Gruppen ist die Synchronisationsfunktion. So kann Sunbird den eigenen Kalender bzw. Teile daraus, wie die Kategorie »dienstlich« oder ähnliches, exportieren. Über die Import-Funktion können dann Kollegen diesen Kalender importieren und bekommen so die enthaltenen Termine angezeigt.

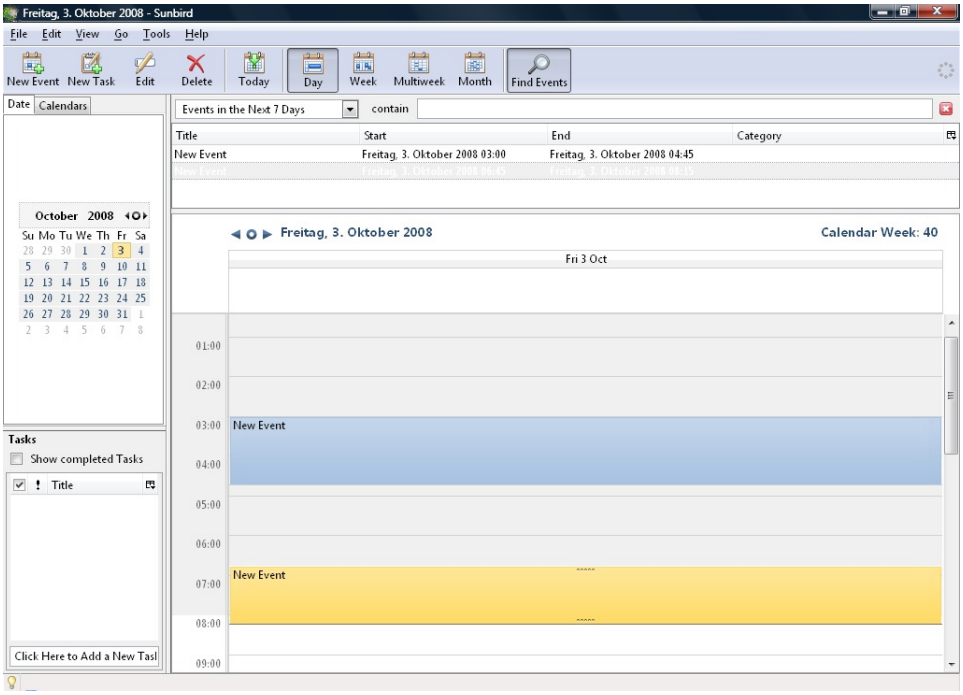


Abbildung 2.1: Mozilla Sunbird auf Windows XP

Trotzdem richtet sich Sunbird eher an den Einzelbenutzer, der einige Daten synchronisieren möchte. Speziell für Arbeitsgruppen gibt es daher Groupware-Programme, die der Gruppenkoordination dienen.

Eines dieser Programme bzw. Programmpakete ist »Horde« bzw. »Horde Groupware«. Horde selbst ist ein Framework um Webanwendungen zu

erstellen. Horde Groupware ist eine Sammlung von Groupwareprogrammen wie Kronolith (Zeitplanung), Mnemo (Notizbuch), Turba (Adressbuch) oder Nag (Aufgabenmanagement).

Horde ist webbasiert, das heißt es läuft auf einem Webserver, der das Frontend als Webseite generiert sowie die Verbindung zu einer Datenbank herstellt, in der die Daten abgelegt werden. Horde Groupware ermöglicht es so einer Arbeitsgruppe Daten wie Kontakte, Termine und Aufgaben zu synchronisieren und koordinieren. Die Einstiegshürde für den Benutzer ist vergleichsweise gering, er benötigt lediglich einen aktuellen Webbrowser und allgemeine Kenntnisse über Horde. Da Horde auf einem Webserver läuft sind die Daten jederzeit an jedem Ort verfügbar, solange der Benutzer einen Internetzugang hat.

Ein Nachteil kann der Funktionsumfang von Horde sein, da dieser neue Benutzer überfordern kann. Daher ist es sinnvoll, Horde vor dem Einsatz an die Gegebenheiten im Projekt anzupassen und bestimmte Funktionen zu deaktivieren bzw. nur notwendige Module zu installieren.

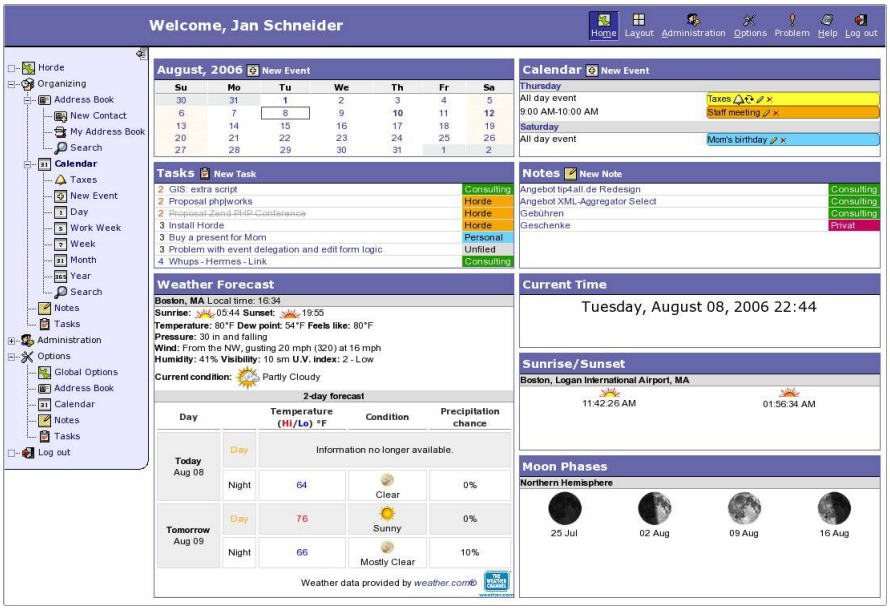


Abbildung 2.2: Das Portal von Horde Groupware

2.2 KOMMUNIKATION

Werden zur Kommunikation zwischen Individuen Computerprogramme und Netzwerke eingesetzt, wird dies in der pschologischen Forschung als computervermittelter Kommunikation bezeichnet. Es existieren verschiedene Taxonomien, umd computervemittelte Kommunikation nach unterschiedlichen Kriterien, bspw. dem Zeitabstand zwischen den Antworten oder der Anzahl der Empfänger zu kategorisieren.

Für den Rahmen dieser Arbeit beschränke ich mich auf folgende Kategorien (vgl. Döring, 2003, Kap. 2):

SYNCHRON VS. ASYNCHRON Synchrone Kommunikation erlaubt spontane, dialogische Äußerungen, die in der Regel sofort übertragen werden. In der asynchronen Kommunikation werden größere Botschaften innerhalb längerer Zeitabstände übertragen. Daher haben die einzelnen Botschaften eher einen monologischen abgeschlossenen Charakter.

INDIVIDUAL-, GRUPPEN- UND MASSENKOMMUNIKATION Diese Kategorien bestimmen die Anzahl der Sender bzw. Empfänger. Individualkommunikation findet zwischen 2 Individuen in einer 1 : 1-Beziehung statt, Gruppenkommunikation in einer m : n-Beziehung und Massenkommunikation in einer 1 : n-Beziehung.

```
<Oasis> brb
<passi> ok
<Oasis> re
<passi> wb
<Oasis> thx
<passi> np
<Oasis> cs?
<passi> ip?
<spanky> ihr spinnt doch
```

Abbildung 2.3: Zitat aus dem PhatNet

	synchron	asynchron
Individual~	Instant Messaging (ICQ)	E-Mail
Gruppen~	Chat (IRC)	Mailinglisten
Massen~	–	Webseite

Tabelle 2.1: Kategorisierung der Kommunikationsmedien

Jedes Medium bzw. jede Kategorie hat ihre Besonderheiten sowie Vor- und Nachteile, die im folgenden anhand von entsprechenden Programmen vorgestellt werden sollen.

2.2.1 *synchrone Kommunikation mit IRC, Jabber und Co.*

Synchrone computervermittelte Kommunikation verläuft vergleichsweise schnell. So kann man via Instant Messager oder Chat Diskussionen in Echtzeit führen. Die Computervermittlung ermöglicht dabei, das sich die Kommunikationspartner an verschiedenen Orten befinden.

Zu beachten ist dabei, das die synchrone Kommunikation in der Regel wie ein »normales« Gespräch von Angesicht zu Angesicht geführt wird, meist also informellen Regeln wie Abkürzungen (»LOL«, »-v«, »?« oder »BRB«), Smileys (\o/, ^_^) oder Onomatopoeitika (*autsch*, *patsch*) folgt.

Abb. 2.3 zeigt ein Zitat aus der German-Bash-Sammlung, *Oasis* und *passi* unterhalten sich im IRC-Jargon, *spanky* kommentiert dies am Ende.

Für Außenstehende ist es schwierig diese Regeln zu erlernen, wenn sie nicht expliziert wurden. Soll die Kommunikation archiviert und später durchsucht werden können, ist es notwendig die Regeln bspw. in einem Glossar zu explizieren und sie auch einzuhalten.

Neben dem klassischen IRC-Chat gibt es inzwischen eine Vielzahl an Alternativen, unter denen die Instant-Messenger besonders hervorstechen. Hierbei muss der Benutzer einen Client auf seinem Rechner installieren, der sich mit einem Server verbindet. Meldet sich der Benutzer mit Nutzername und Passwort am Server an, kann er mit seinen Kontakten Text-, Video- oder Bildnachrichten austauschen und so direkt mit ihnen kommunizieren. Die gesendeten Nachrichten werden dabei sofort übertragen und beim Kommunikationspartner angezeigt. Der größte Vorteil des Instant-Messenging ist die sofortige Übertragung der Daten wie bei einem Telefonat und die Möglichkeit zu sehen, wann ein Kontakt online und damit erreichbar ist. Somit können schnell Daten ausgetauscht und Fragen geklärt werden.

Die bekanntesten proprietären Instant-Messenging-Programme sind ICQ, Skype, Yahoo Messenger! und Windows Live Messenger. Aus Sicherheitsgründen ist hierbei zu beachten, das die Daten oftmals in unbekannten bzw. geheimgehaltenen Formaten ausgetauscht werden und nicht bekannt ist, ob die Daten abgehört werden können und ob dies geschieht. Außerdem setzen

einige kommerzielle Anbieter fragwürdige AGB ein, die beispielsweise die Verwertungsrechte an gesendeten Daten dem Anbieter übertragen[1].

Daher ist der Einsatz freier Alternativen empfehlenswert, diese sind oftmals auch technisch überlegen. Das Extensible Messaging and Presence Protocol, kurz XMPP, früher Jabber genannt, beschreibt ein Protokoll, mit dem Instant-Messenging-Programme implementiert werden können. Der Standard[2] ermöglicht es, die Verbindungen zwischen den Clients und Servern per SSL/TLS zu verschlüsseln bzw. zwischen den einzelnen Clients sogenannte OpenPGP-Schlüssel einzusetzen.

SSL/TLS sorgt dafür, das die Übertragungswege durch einen verschlüsselten Tunnel geleitet werden und die übertragenen Daten somit nicht bzw. nur mit erheblichem Aufwand mitgeschnitten und analysiert werden können. Setzen die Kommunikationspartner OpenPGP-Schlüssel ein, werden die Daten in den jeweiligen Clients verschlüsselt und übertragen, d. h. das selst der Administrator eines zwischengeschalteten Servers nicht auf die Kommunikation zugreifen kann. Außerdem können sich die Kommunikationspartner anhand der eingesetzten Schlüssel gegenseitig authentifizieren, also ihrer Identität vergewissern (**schumacher-crypt**).

Aufgrund des offenen Protokolls und der implementierten Verschlüsselungsfunktionen ist XMPP/Jabber der zur Zeit beste Instant-Messenger.

2.2.2 *Mailinglisten, Foren und Wikis*

Mailinglisten sind die Erweiterung des 1 : 1-Kommunikationsmediums E-Mail zur Gruppenkommunikation. Eine Mailingliste ist ein Verteiler, an dem sich Interessenten anmelden können. Schickt einer der Empfänger eine Email an die Adresse der Liste, wird sie vom Mailserver an alle angemeldeten Adressen weitergeleitet. Diese Technik ermöglicht es einer Gruppe, untereinander via E-Mail in Kontakt zu bleiben, ohne das jedes Gruppenmitglied alle anderen Mitglieder bzw. deren E-Mailadressen kennen muss. Desweiteren ist der administrative Aufwand in der Regel recht gering, da sich die Mitglieder selbständig beim Mailinglistenserver an- und abmelden können. Außerdem können mit den E-Mails Anhänge verschickt werden, bspw. Quellcode, Bilder oder PDF-Dateien.

Problematisch ist bei Mailinglisten die Archivierung der Beiträge. Normalerweise werden die E-Mails an die Abonnenten der Liste verschickt und können von diesen persönlich archiviert werden. Für öffentliche Mailinglisten gibt es aber oftmals Archivseiten im WWW, auf denen die einzelnen Beiträge archiviert, geordnet und damit sortierbar werden.

Abb. 2.4 zeigt meinen Mailclient Mutt und eine Email aus dem Diskussionsfaden über japanische Typen der Mailingliste `pkgsrc-users@netbsd.org`.

Mailinglisten haben eine vergleichsweise hohe technische Einstiegshürde und Lernkurve, die die Benutzung insbesondere für ungeübte Benutzer erschwert. Als Alternative haben sich insbesondere im WWW Diskussionsforen etabliert. Dabei wird auf einem Webserver eine Software betrieben, die das Diskussionsforum als Webseite darstellt. Das heißt, ein Benutzer kann mit seinem Webbrowser auf eine Webseite zugreifen, sich dort einloggen und über Formular- und Textfelder Diskussionen durchsuchen und neue Beiträge verfassen.

Die Forensoftware bietet oftmals auch die Möglichkeit Binärdaten (Bilder, PDFs etc.) hochzuladen und im Forum mit einem Eintrag zu verknüpfen und die gespeicherten Diskussionen zu durchsuchen. Die meisten Foren haben eine vergleichsweise geringe Einstiegshürde und eine niedrige Lernkurve, sie bieten sich daher für eine breite Masse von Benutzern an.

Ein weiteres nützliches Werkzeug sind Wikis. Dieses nach dem Zettelkasten-Prinzip modellierte Programm eignet sich besonders als Grundlage für eine

1 http://www.mynethome.de/?postid=176
2 IETFRFC3920\protect\protect\leavevmode@ifvmode\kern+.16667em\relax--\protect\protect\leavevmode@ifvmode\kern+.16667em\relax3923

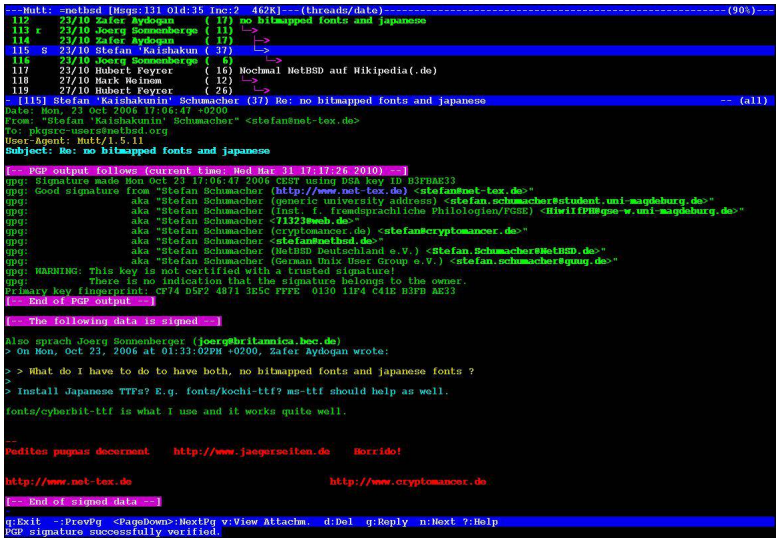


Abbildung 2.4: Der Mailclient Mutt zeigt einen Diskussionsfaden

Art Lexikon oder als strukturiertes und durchsuchbares Notizbuch. In einem Wiki können beliebige Benutzer neue Seiten anlegen, auf denen wie in einem Lexikon einzelne Lemmata erläutert werden. Man kann ein Wiki aber auch innerhalb der Projektgruppe als zentralisiertes Notizbuch verwenden, in dem Notizen abgelegt und damit der gesamten Benutzerschaft zugänglich gemacht werden.

So haben wir im Organisationsteam des Magdeburger Open-Source-Tages die Netzwerkverkabelung und Einrichtung des WLANs in einem Wiki koordiniert.

Um mit den (potenziellen) Besuchern des Open-Source-Tages zu kommunizieren, haben wir eine Webseite[3] eingerichtet. Auf dieser Webseite wurden notwendige Informationen bekanntgegeben, beispielsweise das Vortragpsprogramm, Anfahrtswege, Kontaktdaten usw.

Damit die einzutragenden Daten nicht nur von einer Person bearbeitet werden können, ist etwas Management nötig. Daher wurden unter anderem die Daten zum Vortragsprogramm (Referent, Titel, Zeit, Raum, Beschreibung usw.) in einer Datenbank eingetragen und über von mir selbst entwickelte Webformulare verwaltet. Die Datenbank bereitete die Daten entsprechend auf und gab während der Organisationsphase unter anderem die gerade aktuelle Raumplanung aus. Außerdem war es so möglich, über Webformulare auf die Datenbank zuzugreifen. Diese Zugriffsmöglichkeit wurde mehreren Benutzern gewährt, die damit die Daten aus ihrem Aufgabenbereich anpassen und aktualisieren konnten. Somit konnte die Webseite von mehreren Personen erstellt und verwaltet werden.

Für größere Projektseiten oder speziellere Anwendungen können Redaktionssysteme bzw. sogenannte Content-Management-Systeme eingesetzt werden. Diese Systeme verwalten die Inhalte zentral und ermöglichen es einer Arbeitsgruppe an ihnen zu arbeiten. Außerdem übernehmen bzw. regeln die Systeme Managementaufgaben wie Versionierung, Benutzerverfolgung oder interne Kommentarfunktionen. Neben Wikis oder Foren, die ein sehr einfaches Redaktionssystem sind, gibt es noch spezielle Systeme wie Typo3, Drupal oder Joomla. Diese bieten einen mächtigen Funktionsumfang, sind dadurch aber auch sehr komplex und schwierig zu erlernen.

3 http://www.open-source-tag.de/

2.3 PROJEKTMANAGEMENTSOFTWARE

2.3.1 *GNOME Planner*

Planner ist eine Projektmanagementsoftware des GNOME-Projekts. Sie bietet bisher Kalender- und Ressourcenmanagement, die Verknüpfung von Aufgaben und Möglichkeiten zur Fortschrittsverfolgung im Projekt. Die Daten können im XML-Format oder in einer PostgreSQL-Datenbank abgelegt sowie als PDF oder HTML exportiert werden.

Planner ist ein vergleichsweise kleines Projektmanagementwerkzeug mit graphischer Oberfläche. Es kann Aufgaben und Ressourcen (Arbeitskräfte und Material) erfassen und den Ressourcenverbrauch berechnen.

Ein Projekt wird geplant, in dem Sie Aufgaben und Ressourcen in der Oberfläche erfassen und miteinander in Beziehung setzen. Planner kann aus diesen Daten einfache Listen oder ein Gantt-Diagramm berechnen. In der Aufgabenliste bzw. im Ressourcendiagramm berechnet Planner automatisch die anfallenden Kosten und zeigt diese an. Es kann außerdem Mitarbeiter und Aufgaben in einem Kalender verwalten und somit auf einfache Weise Termine steuern.

Abb. 2.5 zeigt Planner mit dem Gantt-Diagramm eines Beispielprojektes.

Planner ist ein recht einfaches Programm, das für kleinere Projekte hinreichend ist. Problematisch kann es werden, wenn die Projektleitung aus mehreren Mitarbeitern besteht und diese Daten synchronisieren müssen. Es bietet mit den Projektphasen einfache Werkzeuge zur Verfolgung und Steuerung eines laufenden Projektes. Außerdem ist es sehr leicht zu erlernen und schnell einzusetzen.

2.3.2 *Redmine*

Redmine (siehe Abb. 2.6) ist ein Projektmanagementwerkzeug das in Ruby on Rails geschrieben wurde und auf einem Webserver läuft. Es ist plattform- und datenbankunabhängig, der Zugriff auf das Portal erfolgt über eine Webseite.

Redmine kann Projekte planen und managen, also auch im laufenden Betrieb verfolgen. Es arbeitet dabei ebenfalls mit Aufgabenpaketen aus denen Kalender, Gantt-Diagramme oder Zeitberichte erstellt werden.

Da Redmine für das Management von Softwareprojekten gedacht ist, verfügt es über ein Repository zur Quellcodeverwaltung, das Dateien archiviert und versioniert. Außerdem können externe Repositories wie CVS und Git eingebunden werden. Die Metadaten des Projektes (Termine, Aufgabe Ressourcen u.ä.) werden persistent in einer Datenbank abgelegt. Diese kann zum Beispiel PostgreSQL, Oracle oder SQLite sein.

Redmine unterstützt ein Mandanten- und Rollensystem, mit dem verschiedene Benutzer und deren Zugriffsrechte verwaltet werden. Dabei kann ein einzelner Benutzer Mitglied in mehreren Projekten sein und dort unterschiedliche Rollen und Rechte innehaben.

Zur Kommunikation der Projektmitglieder bietet Redmine Wikis und Diskussionsforen an. Außerdem können Benutzer Nachrichten an andere Benutzer schicken und per RSS-Feed oder Email verteilen. Redmine kann auch automatisch Nachrichten erstellen, beispielsweise wenn eine Datei bearbeitet wird. Dabei unterstützt Redmine mehrere Sprachen, von Deutsch über Englisch bis Japanisch und Thai.

Über ein eingebautes Trouble-Ticket-System können Benutzer Tickets absetzen und verfolgen. Das Ticketsystem dient dazu, den Empfang sowie die Bestätigung, Klassifizierung und Bearbeitung von Kundenanfragen (Tickets) zu verwalten. Das System ist eine Form des Wissensmanagements, es soll verhindern das Anfragen verloren gehen oder nicht beantwortet werden, außerdem können mit Hilfe des Systems Antworten auf die Anfragen gespeichert und für zukünftige Problemlösungen archiviert werden.

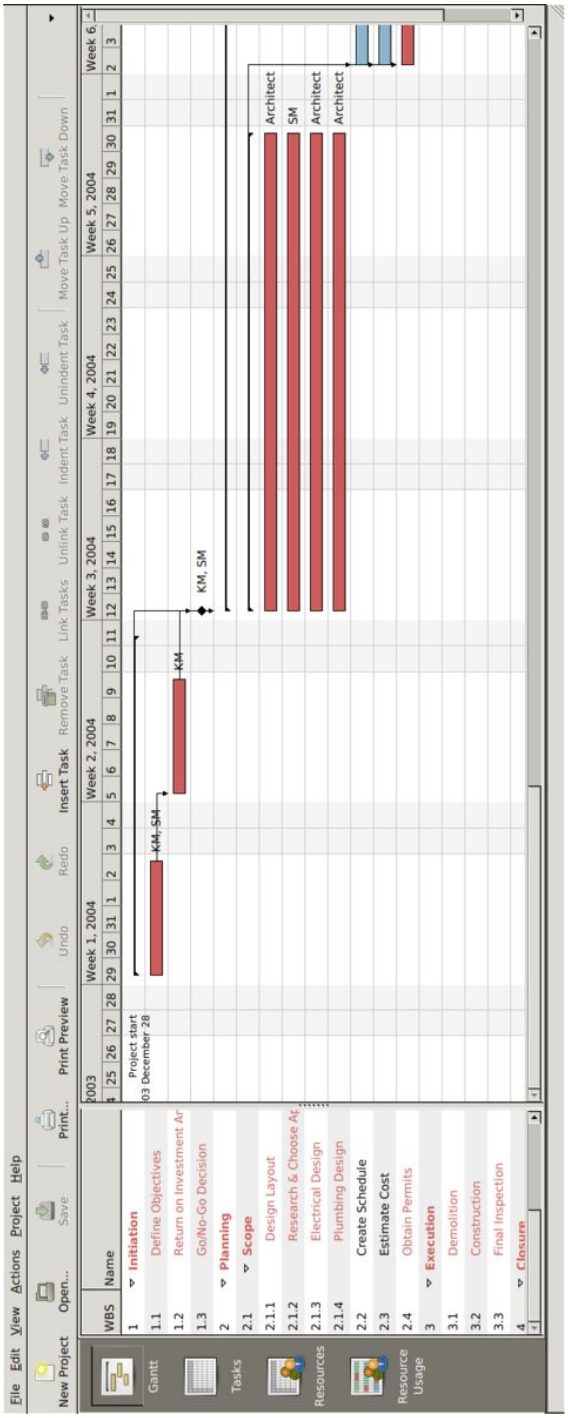


Abbildung 2.5: GNOME Planner mit einem Gantt-Diagramm.

Redmine ist ein sehr umfangreiches Projektmanagementwerkzeug, das einzelne oder mehrere große und kleine Projekte über ihren gesamten Lebenszyklus managen kann. Durch den Betrieb auf einem Webserver benötigen die Benutzer lediglich einen aktuellen Webbrowser um darauf zugreifen zu könne. Das Client-Server-Prinzip vereinfacht auch die Datendistribution innerhalb eines Projektes.

Problematisch kann der technische Aufwand sein, da der Betrieb eines Webservers nicht jedermanns Sache ist und die Daten relativ ungeschützt in einem Netzwerk liegen. Daher muss die Projektleitung Maßnahmen ergreifen um die Daten zu sichern und um die Server vor Angriffen zu schützen.

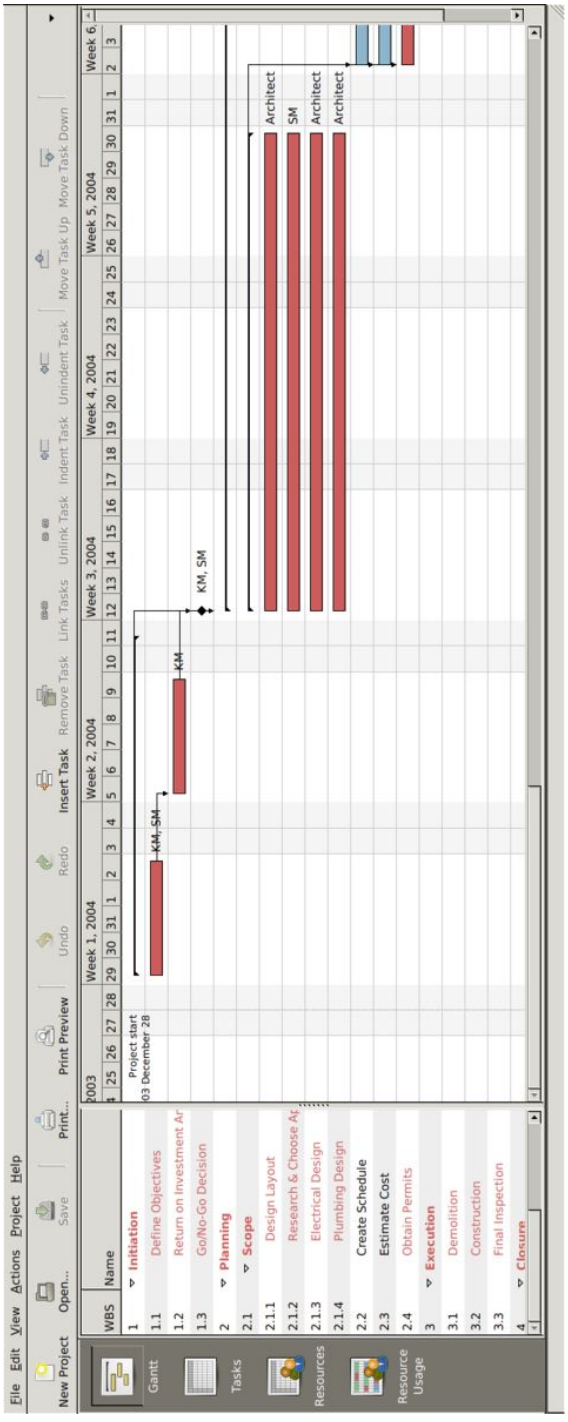


Abbildung 2.6: Die Startseite eines Redmine-Projektes.

2.3.3 *dotProject*

DotProject[4] ist ein webbasiertes Projektmanagementwerkzeug, das verschiedene Funktionen bietet:

- Projektplanung
- benutzerbasierte Terminplanung
- priorisierte zu-erledigen-Listen
- Trouble-Tickets zur Verwaltung von Anfragen
- Benutzerverwaltung mit Rechtefreigaben und Rollen
- ein eingebautes Forum zur Kommunikation
- Generierung von Berichten inkl. Filterfunktion

2.3.4 *TaskJuggler*

TaskJuggler ist ein Projektmanagementpaket, das 2001 von einigen SuSE-Entwicklern gestartet wurde. 2006 wurde das Programm in Ruby neu geschrieben und mit einer graphischen Benutzerschnittstelle (GUI) versehen. versehen. Anders als die meisten Projektmanagementprogramme verwendet TaskJuggler eine Beschreibungssprache, in der das Projekt deklariert und definiert wird. TaskJuggler berechnet daraus selbständig unter anderem Zeitintervalle in der Aufgabenplanung sowie die Ressourcenzuteilung, außerdem bietet es Funktionen für Zeitplanung, Kostenrechnung, Risikoanalyse, Statusverfolgung und Kommunikationsmanagement (Schläger & Rührschneck, 2006).

Benutzung des Programms

TaskJugglerUI[5] ist die graphische Benutzerschnittstelle zu den TaskJuggler-Programmen. Startet der Benutzer TaskJugglerUI, kann er mit dem eingebauten Assistenten ein neues Projekt einrichten lassen. Danach öffnet TaskJugglerUI automatisch den Editor, in dem die Projektbeschreibung bearbeitet werden kann (siehe Abb. 2.7). TaskJuggler verwendet dazu eine eigene Beschreibungssprache[6] die von TaskJuggler interpretiert und aus denen die Berichte und Diagramme generiert werden.

TaskJuggler unterstützt durch sogenannte Szenarios die Möglichkeit, verschiedene Ablaufmöglichkeiten im Projekt durchzuspielen. Ein Szenario kann mittels `scenario` definiert werden. Einem beliebigen Szenario können Sie verschiedene Projektdaten zuweisen, beispielsweise verschiedene Kostenstrukturen, unterschiedliche Mitarbeiter oder unterschiedliche Zeitpuffer und Termine. Damit haben Sie zum Einen die Möglichkeit das Projekt unter verschiedenen Bedingungen durchrechnen zu lassen und zum anderen können Sie über 2 Szenarios den ursprünglichen Plan für das Projekt sowie den richtigen Ablauf des Projektes verfolgen und vergleichen.

Um ein Projekt zu definieren, benötigen Sie zuerst eine Projektbeschreibung (siehe Abb. 2.9), in der Sie den Projektnamen, die Version, den Zeitraum sowie Zeit- und Währungsformat angeben können.

Anschließend definieren (Abb. 2.10) Sie die Mitarbeiter und deren Tagessatz in Euro (`rate`) sowie ggf. Abwesenheitszeiten (`vacation`). Abb. 2.11 zeigt, wie Sie Konten für die Kostenrechnung definieren, `cost` gibt Kosten an, `revenue` Einnahmen, mit `credit` können Sie Buchungen auf ein Konto vornehmen, dazu geben Sie das Datum, eine Beschreibung und den Betrag an.

4 http://dotproject.net/

5 Mit TaskJugglerUI bezeichne ich nur die graphische Benutzerschnittstelle, mit TaskJuggler hingegen das gesamte Programm, das noch aus weiteren Programmen besteht.

6 Eine Beschreibungssprache beschreibt die Syntax und Semantik der verwendeten Daten, sie weist ihnen also eine Form und eine Bedeutung zu.

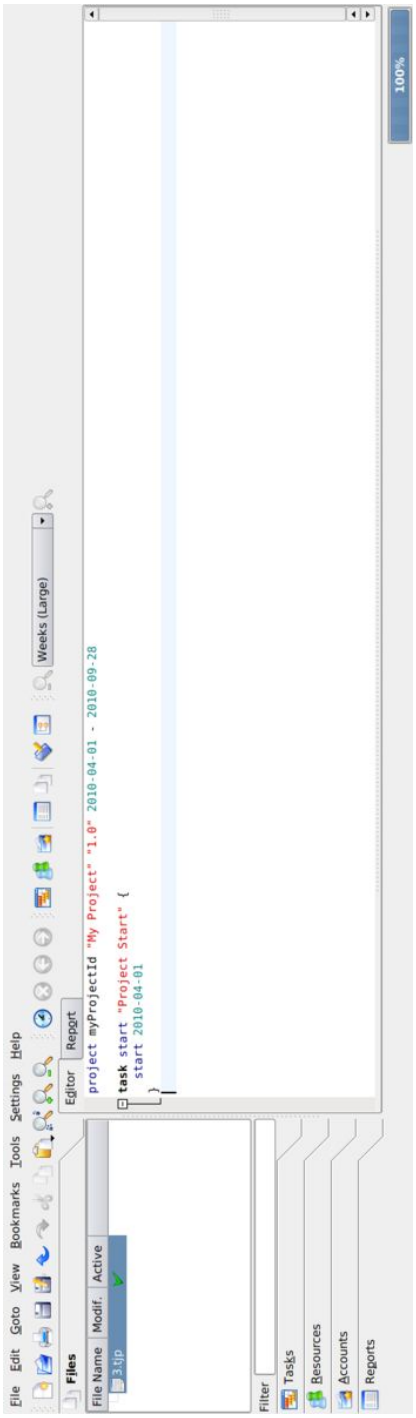


Abbildung 2.7: Der Editor von TaskJugglerUI mit einem leeren Projekt

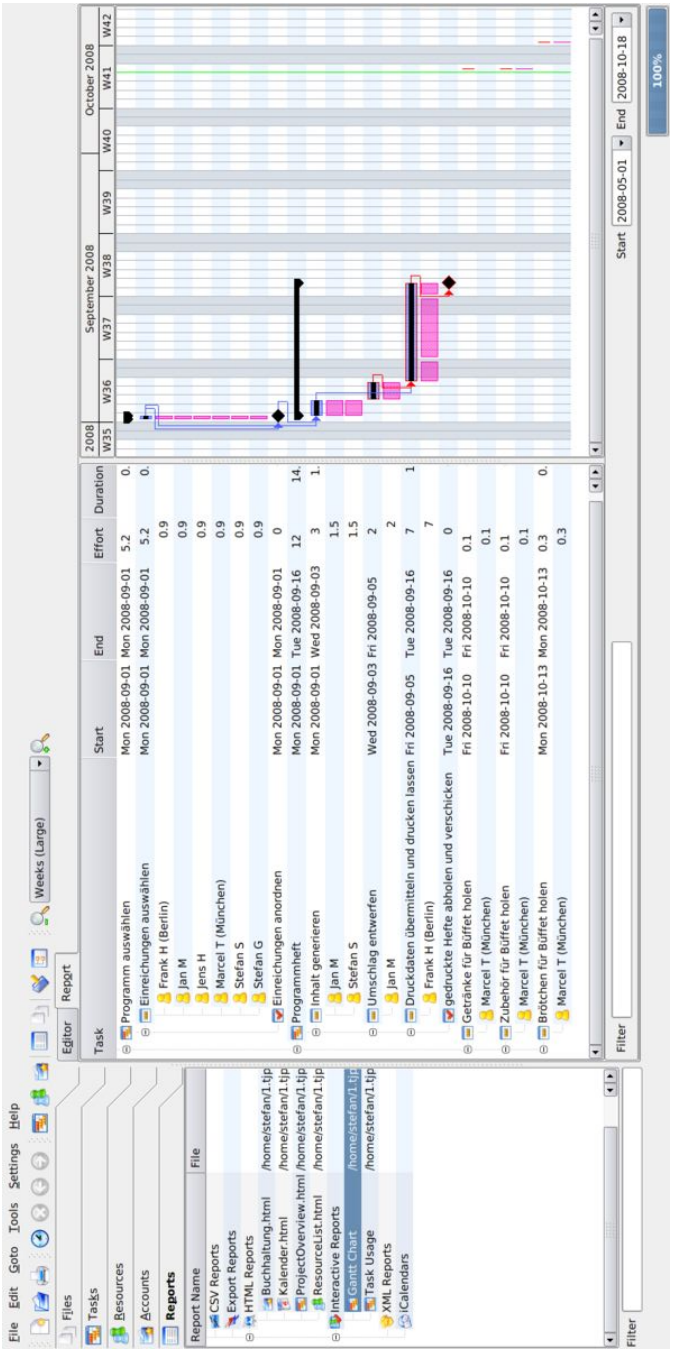


Abbildung 2.8: Der GanttReport zeigt die einzelnen Aufgaben als Baum und als Gantt-Diagramm an.

```
project bare "Magdeburger Open-Source-Tag" "2008"\
2007-03-11 2008-10-12 {
    now 2010-03-11
    timeformat "%d.%m.%Y"
    currency "EUR"
    scenario plan "Plan" {
        scenario delayed
        "Delayed"
    }
}
```

Abbildung 2.9: Projektdefinition

```
macro allocate_workers [
  allocate Stefan
  allocate Marcel
  allocate StefanG
  allocate Frank
]

resource ma "Mitarbeiter" {
  resource Stefan "Stefan S"
  resource Marcel "Marcel T (München)"
  resource StefanG "Stefan G"
  resource Frank "Frank H (Berlin)"
}

resource Jan "Jan M"{
  rate 120.0
  vacation 2008-09-10 - 2008-09-20
}
```

Abbildung 2.10: Mitarbeiterdefinition

```
account Druck "Druck" cost
account Essen "Verpflegung, Bewirtung" cost

account Workshop "Workshopeinnahmen" revenue{
   credit 2008-09-28 "VVK für Workshop 1" 75.0
}

account Sponsor "Sponsoring" revenue{
  credit 2008-06-08 "Sponsoring von ABC" 500.0
  credit 2008-07-18 "Sponsoring von XYZ" 450.0
}

```

Abbildung 2.11: Kontendefinition

Aufgaben werden wie in Abb. 2.12 gezeigt definiert. Verschiedene Aufgaben können über die Klammerung mit » « zusammengefasst werden, so besteht im Beispiel die Aufgabe »Programm auswählen« aus den Teilsaufgaben »Programm auswählen« und »Einreichungen anordnen«. Jede Aufgabe wird mit `task Kurzname "Langname"` definiert und erfordert entweder ein explizites Startdatum (`start Datum`) oder eine Abhängigkeit von einem oder mehreren Vorgängern (`depends Aufgabenblock.Vorgänger1 !Vorgänger2`). Neben der ASAP-Planung mit `start` oder `depends` kann TaskJuggler auch im ALAP-Modus planen, in dem den Aufgaben mit `end` ein Enddatum oder mit `precedes` der Nachfolger angegeben wird. Im ASAP-Modus berechnet TaskJuggler die Termine der Aufgaben vom Anfang zum Ende und führt Aufgaben zu ihrem frühesten Zeitpunkt aus, im ALAP-Modus rechnet TaskJuggler vom Ende zum Anfang und plant Aufgaben mit ihrem Enddatum. Beide Modi dürfen nicht gemischt werden!

Das Ausrufezeichen vor dem Aufgabennamen bedeutet, dass das Verweisziel (`auswahl`) zum selben Aufgabenblock (`Programm`) gehört. Stattdessen können Sie auch mit `Aufgabenblock1.Teilaufgabe2` direkt auf Teilaufgaben verweisen. Neben den Abhängigkeiten kann TaskJuggler auch direkt mit Daten und Dauern rechnen. In der Aufgabe `auswahl` wird das Startdatum mit `2008-08-31` definiert und die Dauer mit `effort` festgelegt sowie die Ressource `Orga` zugewiesen, da das Organisationsteam für die Auswahl zuständig ist. TaskJuggler verfügt über einen eingebauten Detektor für kritische Pfade, der automatisch die Pfade zwischen mindestens 2 Aufgaben analysiert. Da die Anzahl der Pfade exponentiell wächst, berechnet TaskJuggler lediglich 10 000 000 Pfade. Mittels `maxpaths` können Sie eine maximale Anzahl an zu berechnenden Pfaden festlegen, der Wert 0 lässt TaskJuggler alle Pfade des Projektes berechnen. Dies kann bei größeren Projekten unter Umständen mehrere Stunden bis Tage dauern!

In diesem Beispiel beginnt die Programmauswahl durch das Orga-Team am 2008-08-31 und dauert planmäßig 5 Tage. Ist die Aufgabe abgeschlossen, beginnt automatisch die Aufgabe `anordnen` durch `Stefan`, die einen Tag dauert. Endet die Programmauswahl früher oder später, kann das neue Enddatum angepasst werden. TaskJuggler schleift diese Änderungen dann automatisch an alle abhängigen Teilaufgaben weiter. Somit ist es zur Laufzeit des Projektes möglich, aktuelle Änderung einzuspeisen und neue Pläne zu berechnen. Dazu können Sie das Schlüsselwort `complete` verwenden, um anzugeben wieviel Prozent der Aufgabe bereits erledigt sind. Diese Angabe dient aber nur für die generierten Berichte, die Planung des Projektes wird damit nicht beeinflusst.

Weiterhin können Sie Aufgaben mittels `flag Markierung` mit einer beliebigen Markierung versehen. Mithilfe dieser Markierung können Sie Vorgänge in Berichten filtern lassen, in dem Sie beispielsweise nur Aufgaben anzeigen lassen, die mit »Orga« geflagt sind. Mit `milestone` können spezielle Meilensteine deklariert werden. Diese sind aber keine gewöhnlichen Aufgaben und dürfen daher keine Dauer enthalten. Das heißt, dass das Start- und Enddatum gleich sein muss, wenn Sie es angeben möchten. Ansonsten reicht es auch, den Meilenstein über Abhängigkeiten in den Aufgabenfluss einzubinden. Im Beispiel ist die Aufgabe `druck2` von `druck1` abhängig und als Meilenstein deklariert. In der Aufgabe `vortr` werden mit `note` und `journalentry` eine Notiz bzw. ein Tagebucheintrag erfasst. Diese können in TaskJuggler angezeigt werden und bei der Bearbeitung der Aufgabe helfen bzw. bei der Projektverfolgung hilfreich sein, in dem diese Funktionen eine einfache Form des Wissensmanagments bieten.

Mit `account Konto` weisen Sie der Aufgabe ein Konto für die Buchhaltung zu und mit `startcredit` bzw. `endcredit` die Kosten die zu Beginn bzw. Ende der Aufgabe anfallen. Direkte Buchungen auf ein Konto müssen Sie über `credit` im jeweiligen Konto vornehmen (siehe S. 16). TaskJuggler berechnet dann bei der Berichterstellung automatisch die Salden der Kontos zum jeweiligen Monatsende.

```
1
2  task Programm "Programm auswählen"{
3    task auswahl "Einreichungen auswählen"{
4      start 2008-08-31
5      effort 5d
6      allocate Orga
7      complete 80
8    }
9    task anordnen "Einreichungen anordnen"{
10     effort 1d
11     allocate Stefan
12     depends !auswahl
13   }
14 }
15
16 task Heft "Programmheft" {
17   task vortr "Inhalt generieren"{
18     effort 3d
19     allocate Stefan, Jan
20     depends Programm.auswahl, Programm.anordnen
21     statusnote "Erstellung des A5-Heftes könnte
22                 problematisch sein wegen der Druckerausrichtung"
23     journalentry 2008-09-03 "Orientierungsproblem gelöst,
24                              A5 mit psnup generiert"
25   }
26   task u14 "Umschlag entwerfen"{
27     start 2008-08-31
28     effort 2d
29     allocate Jan
30   }
31
32   task druck1 "Druckdaten übermitteln und drucken lassen"{
33     effort 7d
34     allocate Frank
35     depends !vortr, !u14
36     account Druck
37     flags kostet
38     startcredit 150.00
39     }
40
41   task druck2 "gedruckte Hefte abholen und verschicken"{
42     milestone
43     allocate Frank
44     depends !druck1
45     account Druck
46     flags kostet
47     endcredit 50.00
48   }
49 }
50
```

Abbildung 2.12: Definition der Aufgaben

TaskJuggler kann 2 verschiedene Arten von Berichten über das Projekt erstellen, interaktive und HTML-Seiten. Die interaktiven Berichte können nur in TaskJugglerUI betrachtet werden, die HTML-Seiten hingegen in jedem Webbrowser und auch in eine Webseite eingebunden werden.

Um Berichte zu erzeugen, müssen diese ebenfalls im Editor definiert werden. Abb. 2.13 zeigt die Definitionen von 5 Berichten.

Mit `taskreport` und `resourcereport` werden interaktive Berichte erstellt. Die ersten beiden Berichte, `Gantt1` und `Gantt2`, definieren einen Bericht über die anfallenden Aufgaben. Dazu wird ein Name und eine Überschrift (`headline`) sowie das Zeitformat (`timeformat`) und die Zeiteinheit Tage (`loadunit`) festgelegt.

Über das Schlüsselwort `columns` werden die anzuzeigenden Spalten bzw. Daten definiert. Mit `chart` wird das Gantt-Diagramm aktiviert, mit `hierarchindex`, `name`, `start`, `end`, `effort`, `duration`, werden die Nummerierung der Aufgaben sowie Start- und Enddatum, Aufwand und Dauer aktiviert. Der zweite Bericht zeigt zusätzlich mit `completed` an, wieviel Prozent der Aufgaben bereits erledigt sind. Damit lässt sich das Projekt im laufenden Betriebe verfolgen.

Der folgende Bericht listet die erstellten Ressourcen (Mitarbeiter und Material) auf. Die Definitionen entsprechen den anderen Berichten, lediglich die Eigenschaften der Ressourcen sind unterschiedlich, so werden die Ressourcen mit `columns` nummeriert, benannt, das Verhältnis von momentaner Auslastung zu möglicher Auslastung (`utilization`) sowie die momentane Auslastung (`freeload`) und die Kosten angezeigt.

Die letzten beiden Berichte erzeugen HTML-Seiten als Ausgaben, die sowohl im TaskJugglerUI als auch in einem beliebigen Webbrowser angezeigt werden können. Im Beispiel wird die Buchhaltung (`htmlaccountreport`) sowie ein Kalender (`htmlweeklycalendar`) erzeugt. Mittels `accumulate` werden die Kontosalden berechnet und angezeigt, im Kalender werden das Startdatum und die Dauer von 3 Wochen festgelegt (`period`).

Abb. 2.8 zeigt einen Gantt-Bericht an, links im TaskJugglerUI werden die Schaltflächen für die verschiedenen Teilbereicht angezeigt, in der Mitte die Liste mit den Aufgaben und zugewiesenen Ressourcen, rechts das Gantt-Diagramm.

Nachdem der Plan für das Projekt erstellt wurde, beginnt die Projektverfolgungsphase. Über das Schlüsselwort `complete` können Sie die bereits erledigten Prozent einer Aufgabe erfassen. Diese Prozentzahl wird aber nur in den Berichten verarbeitet und nicht zur Planung weiterer Aufgaben herangezogen. Möchten Sie den Anteil erledigter Arbeiten mit in die Planung einbeziehen, müssen Sie die genutzten Ressourcen mit `booking` verbuchen. TaskJuggler berechnet dann die bereits abgearbeiteten Zeitintervalle und vergleicht sie mit dem geplanten Aufwand für die Aufgabe.

Abb. 2.14 zeigt die Definition einer Aufgabe, der mit `effort` ein Aufwand von 10 Manntagen zugewiesen wird. In der Ressourcendefinition werden zuerst die maximalen Arbeitszeiten von `Stefan` festgelegt und anschließend insgesamt 4 Arbeitseinsätze verbucht, zuerst 8 Stunden, danach zweimal 4 Stunden. In der letzten Buchung werden insgesamt 11 Stunden verbucht, dabei fallen 3 Überstunden an, die mit `overtime` angegeben werden. Im Beispiel würde das Projekt $10d * 8h = 80h$ benötigen, davon wären durch `Stefan` bereits $8h + 4h + 4h + 11h = 27h$ abgearbeitet, also $\frac{27h}{80h} * 100 = 33,75\%$ erledigt. Da `Stefan` 8 Stunden pro Tag arbeiten darf, würde TaskJuggler das Aufgabenende auf $\frac{80h-27h}{8h} = 6,625d + 2008 - 10 - 11$ auf 2008-10-17, 13 Uhr berechnen.

Soll TaskJuggler bei der Berichterstellung die Buchungen bzw. bereits geleisteten Stunden bei der Berechnung der Enddaten berücksichtigen, müssen Sie den `projection`-Modus im Szenario aktivieren. Im Beispiel geschieht dies über `scenario` und `projection`.

```
taskreport "Gantt1" {
  headline "Gantt-Plan"
  columns hierarchindex, name, start, end, effort, duration, chart
  timeformat "%a %Y-%m-%d"
  loadunit days
}

taskreport "Gantt2" {
  headline "Gantt-Verfolgung"
  columns hierarchindex, name, start, end, effort { title "erledigt" }, duration,
    completed, chart
  timeformat "%a %Y-%m-%d"
  loadunit days
}

resourcereport "Ressourcenverbrauch" {
  headline "Ressourcenverbrauch"
  columns no, name, utilization, freeload, cost
  loadunit days
  hidetask 0
}

htmlaccountreport "Buchhaltung.html" {
  headline "Buchhaltung"
  timeformat "%d-%m-%y"
  accumulate
  columns index, name, monthly
}
htmlweeklycalendar "Kalender.html"{
  headline "Kalender"
  timeformat "%d-%m-%y"
  period 2008-09-01 +3w

}
```

Abbildung 2.13: Definition der Berichte

```
project prj "Project" "1.0" 2008-10-01 - 2008-10-31 {
    now 2008-10-11
  scenario plan "Plan" {
    projection
  }
}

task aufgabe1 "Eine Aufgabe" {
  start 2008-10-05
  effort 10d
  allocate Stefan
}

resource Stefan "Stefan S"{
    limits {
        dailymax 8h
        weeklymax 40h
        monthlymax 20d
        yearlymax 200d
    }
}


supplement resource Stefan {
  booking aufgabe1 2008-10-06 +8h
  booking aufgabe1 2008-10-08 +4h,
                2008-10-09 +4h
  booking aufgabe1 2008-10-11-08:00 +11h {overtime 1}
}
```

Abbildung 2.14: Ressourcenverbrauch buchen

Fazit

TaskJuggler ist eine ausgereifte und mächtige Projektmanagementsoftware. Anfangs erscheint die Bedienung über eine Beschreibungssprache gewöhnungsbedürftig, allerdings ist die Grammatik recht einfach zu erlernen und sehr mächtig.

Da die Projektbeschreibung in einzelnen Dateien gespeichert wird, können diese auch mit anderen Programmen bearbeitet werden, so können beispielsweise die Projektmitarbeiter ihre Arbeitszeiten über ein Webformular in eine Datenbank oder eine Datei eintragen, die dann automatisch von TaskJuggler eingelesen wird. Eine Anbindung an automatische Zeiterfassungssysteme (Stechuhren o. ä.) ist damit auch möglich.

Ebenso kann TaskJuggler Projekte bzw. Teile davon in Dateien exportieren und somit für andere Programme nutzbar machen. Damit ist es auch möglich, über ein Revisionswerkzeug[7] wie CVS oder Subversion, Projektdaten für mehrere Mitarbeiter vorzuhalten.

TaskJuggler kann der Projektleitung sehr viel automatisierbare Arbeit abnehmen und dank seiner Szenario-Technik verschiedene Projektszenarien durchspielen. Außerdem dient es nicht nur der Projektplanung, sondern auch der Verfolgung und Steuerung im laufenden Projekt bis hin zu dessen Abschluss. Es ist also ein richtiges Projektmanagementwerkzeug und nicht nur ein Planungstool.

Nachteilig ist der vergleichsweise hohe Einarbeitungsaufwand, der den Einsatz von TaskJuggler erst ab einer gewissen Projektgröße sinnvoll macht. Daher sollte man vor Projektbeginn abwägen, inwiefern der Einsatz von TaskJuggler ökonomisch vertretbar ist.

Neben Planner, Redmine, dotProject und TaskJuggler gibt es noch verschiedene weitere Open-Source-Projektmanagementpakete wie trac, OpenWorkbench, OpenProj usw. Oftmals unterscheiden sich die Programm von der Funktionalität her nicht wesentlich. Lediglich die Implementierung (verwendete Programmiersprache, Datenbank etc.) sollte zur Umgebung des Projektes passen. Außerdem unterscheiden sich die graphischen Benutzerschnittstellen etwas, aber diese sind wie so vieles Geschmackssache.

Daher ist es nicht möglich, eine Empfehlung für die einzige wahre Projektmanagementsoftware zu geben. Wie im Kap. 3.1 noch näher ausgeführt, muss die Projektleitung basierend auf den eigenen Gegebenheiten die mehr oder minder passende Software auswählen.

7 Revisionswerkzeuge dienen der zentralen Verwaltung von Quellcodedateien, an denen mehrere Entwickler arbeiten.

3

CODA

Es ist nicht genug, zu wissen, man muß auch anwenden;
es ist nicht genug, zu wollen, man muß auch tun.

(Johann Wolfgang von Goethe)

3.1 KRITIK AN UND NACHTEILE VON PROJEKTMANAGEMENTSOFTWARE

- Computervermittelte Kommunikation kann keine Kommunikation von Angesicht zu Angesicht ersetzen. Durch den Einsatz von Software als Kommunikationsmedium werden wichtige Informationskanäle weggelassen und so die Entropie der Nachrichten verringert.

- Präsentations- und Visualisierungssoftware verführt den Autor oftmals dazu, Daten vereinfacht, verkürzt oder falsch darzustellen. Edward Tufte untersuchte den Einsatz von Microsoft Powerpoint[1] in der NASA und kam zu dem Schluss, das die Aufbereitung der Daten in Powerpoint kognitiv ungünstig sei. Das selbe Problem kann auch auf Projektmanagementsoftware zutreffen, wenn Daten weggelassen oder unverständlich präsentiert werden.

- Die Hauptaufgabe von Computern bzw. Computerprogrammen ist die Berechnung von mathematischen Aufgaben, daher (ver)führt ihr Einsatz in der Regel dazu, allerlei Vorgänge über Kennziffern zu erfassen und zu berechnen. Dies bedeutet zum Einen weiteren Overhead im Projektmanagement und damit Kosten und Zeitverlust (vgl. v. a. DeMarco, 2004), zum Anderen lässt sich aber nicht jedes Problem in der Welt für ein Computerprogramm berechenbar machen (vgl. von Foerster, 1993a, 1993b, 1993c, 1993d, 1993e; von Foerster & Pörksen, 2006; Watzlawick, 2007). Wäre dem so, bräuchte man kein Projektmanagement mehr, sondern könnte *jedes* beliebige Projekt genau vorausplanen und ohne Abweichungen durchführen.

- Softwarepakete bilden oftmals keine komplette Projektmanagementstrategie ab, sondern nur einzelne Werkzeuge oder Aspekte. Dies kann dazu führen, das sich ein Benutzer auf eines dieser Werkzeuge konzentriert und andere Methoden vernachlässigt oder ignoriert[2].

- Wenn die Software einer Projektmanagementstrategie abbildet, kann dies eine andere als die im Projekt eingesetzte sein. Ein Programm für agiles Softwaremanagement lässt sich nicht unbedingt für die Wasserfallmethode einsetzen.

- Größere Softwarepakete bieten eine Funktionsfülle (»bloat«), die der Endanwender oft nicht benötigt. Daher muss die Projektleitung *vor* dem Einsatz der Anwendungssoftware deren Eignung zu überprüfen und ggf. die Anwender zu schulen.

1 http://www.edwardtufte.com/bboard/q-and-a-fetch-msg?msg_id=0001yB
2 »Wer als Werkzeug nur einen Hammer hat, sieht in jedem Problem einen Nagel.« – Paul Watzlawick

- Computerprogramme vereinfachen nicht jede Tätigkeit. So kann es insbesondere in der Anfangsphase sinnvoll sein, Skizzen oder Pläne auf Papier zu erstellen. Hacker (2005, S. 571ff) diskutiert im Kap. »Intellektuelle Regulation« den Einsatz von Computerprogrammen, insbesondere von CAD-Systemen bei Ingenieuren und Architekten, und kommt zu dem Schluss, das dem händischen Entwerfen (skizzieren, modellieren) eine hohe Bedeutung zukommt, es sei »Denken mit der Hand«.

- Vernetzte Rechner vereinfachen zwar die Kommunikation in Gruppen, ermöglichen aber auch Datenverluste. So muss die Projektleitung eine geeignete Datensicherungsstrategie (vgl. Schumacher, 2006a, 2006b, 2006c, 2007a, 2007b, 2007c) etablieren, um Datenverluste zu vermeiden. Außerdem muss sie die eingesetzten Systeme technisch und sozial absichern, um Einbrüche und Diebstähle der sensiblen Daten zu verhindern (**schumacher-grundschutz-soceng1**; **schumacher-grundschutz-soceng2**; **schumacher-grundschutz-soceng3**; **schumacher-crypt**; vgl. Kehrer, 2008; Schumacher, 2006d, 2007a, 2007d, 2007e, 2007f, 2008a, 2008b, 2009).

- Die Daten die innerhalb der Projektmanagementsoftware anfallen müssen unter Umständen für mehrere Jahre archiviert werden. Geschäftsbriefe müssen beispielsweise 10 Jahre revisionssicher aufbewahrt werden[3], einige Bauunterlagen 30 Jahre nach Projektende und länger. Die Daten müssen also regelmäßig auf neue Medien rotiert werden, außerdem müssen die Dateiformate noch lesbar sein. Zum Vergleich: der erste IBM PC XT kam am 12. August 1981 auf den Markt, die gesamte PC-Technik ist also noch nicht einmal 30 Jahre alt. Die Langzeitarchivierung stellt somit einen nicht vernachlässigbaren Kosten- und Zeitfaktor dar.

- Oftmals unterscheiden die Programm die einzelnen Phasen des Projektmanagements nicht oder nur undeutlich voneinander. Dies kann zu Problemen im Management führen, wenn während der Planungsphase schon Ausführungsprobleme bearbeitet werden, oder umgekehrt. Manche Programme konzentrieren sich zu sehr auf die Planungsphase und bieten keinerlei Managementfunktionalität für die Ausführungsphase. So kann es beispielsweise vorkommen, das ein Projektmitarbeiter abgezogen wird oder hinzukommt und somit die Aufgabenverteilung aktualisiert werden muss. Fehlt diese Möglichkeit, verkommt die Software zu einer Art statischem Snapshot eines bestimmten Zeitpunktes (eben dem Ende der Planungsphase) und nicht zu einem Managementwerkzeug, das über einen Zeitraum hinweg arbeitet.

- Größere Projektmanagementpakete verfügen über sehr viele Konfigurations- und Nutzungsmöglichkeiten, die oftmals nur zu einem Bruchteil genutzt werden. Trotzdem muss sich der Benutzer der Software mit der so entstehenden Komplexität befassen und mit ihr umgehen lernen. Dies erfordert einen erhöhten, teils unnötigen, Lernaufwand und stellt auch eine mögliche Fehlerquelle dar.

- Der Einsatz von Projektmanagementsoftware erfordert einen gewissen Overhead der sich in Zeit- und Geldeinheiten abschätzen lässt. So müssen beispielsweise genügend leistungsfähige Computer, Netzwerk- und Internetanbindungen bereitgestellt werden und die Mitarbeiter den Umgang mit der Software erst erlernen. Die anfallenden Kosten und

3 siehe §239 Abs. 4 Satz 2 HGB *»Die Handelsbücher und die sonst erforderlichen Aufzeichnungen können auch in der geordneten Ablage von Belegen bestehen oder auf Datenträgern geführt werden, soweit diese Formen der Buchführung einschließlich des dabei angewandten Verfahrens den Grundsätzen ordnungsmäßiger Buchführung entsprechen. Bei der Führung der Handelsbücher und der sonst erforderlichen Aufzeichnungen auf Datenträgern muß insbesondere sichergestellt sein, daß die Daten während der Dauer der Aufbewahrungsfrist verfügbar sind und jederzeit innerhalb angemessener Frist lesbar gemacht werden können. Absätze 1 bis 3 gelten sinngemäß.«*

Arbeitsstunden verteuern das gesamte Projektmanagement und können so den Einsatz von Projektmanagementpaketen wirtschaftlich unsinnig erscheinen lassen. Daher ist es sinnvoll, den notwendigen Overhead vor Beginn der Projektmanagementmaßnahmen abzuschätzen und gegebenenfalls aus wirtschaftlichen Gründen auf Softwarelösungen zu verzichten.

- Wenn ein Individuum oder eine Organisation Computer einsetzt, muss sich das Individuum bzw. die Organisation dem Computer anpassen und nicht umgekehrt. Dies kann dazu führen, das irgend jemand »Projektmanagement macht«, weil er eine bestimmte Software einsetzt und dies als Projektmanagement betrachtet, ohne sich aber je formal mit Projektmanagementtheorie befasst zu haben. Inzwischen integrieren immer mehr Anwendungen wie Groupware oder Email-Programme »Projektmanagementfunktionen«, was schließlich dazu führen kann, das sich die Definition von Projektmanagement verändern muss.

- Schlussendlich sind Computer und Programme immer nur Werkzeuge die von einem Menschen benutzt und gesteuert werden müssen. Ein Computerprogramm kann kein neues Wissen generieren, nicht lernen, keine Entscheidungen treffen und damit auch kein Projekt managen. Er kann lediglich Daten aufbereiten und darstellen, wie er dies zu tun hat, muss ihm aber jemand mitteilen.

3.2 FAZIT

Der Einsatz von Projektmanagementsoftware ist unter bestimmten Bedingungen sinnvoll, beispielsweise wenn die Overheadkosten für Einführung, Administration und Gebrauch der Technik und Software wirtschaftlich vertretbar sind, weil sie sich innerhalb des Projektes amortisieren.

Die Projektleitung muss sorgfältig die Software auswählen und evaluieren, da die Migration auf andere Programme im laufenden Projekt oftmals problematisch und aufwendig ist. Dabei sind neben technischen Vorgaben (z. B. verfügbare EDV-Technik, verfügbares Know-How) auch soziale entscheidend. So müssen sich die Benutzer an die Programme anpassen und nicht (oder zumindest sehr selten) umgekehrt. Daher muss die Projektleitung bei der Auswahl der Software besonderen Wert auf die Usability der Programme legen.

Ein weiteres wichtiges Auswahlkriterium ist die zugrundeliegende Managementmethode. Die Software muss diese unterstützen bzw. innerhalb der Managementmethode sinnvoll einsetzbar sein.

Unter Umständen kann es für ein Projekt sinnvoll sein, eigene Software zu entwickeln bzw. entwickeln zu lassen. Diese Entscheidung kann im Rahmen einer allgemeinen Evaluation gefällt werden und hängt auch wieder von den Kosten und der Verfügbarkeit brauchbarer Standardsoftware ab.

4

LITERATUR

DeMarco, T. (2004). *Was man nicht messen kann … …kann man nicht kontrollieren* (2. Aufl.). mitp.

Döring, N. (2003). *Sozialpsychologie des Internet* (2., vollst. überarb. u. erw. A.). Hogrefe.

Hacker, W. (2005). *Allgemeine Arbeitspsychologie* (2., vollst. überarb. u. erg. A.). Verlag Hans Huber.

Kehrer, A. (2008). IT-Security psychologisch implementieren [Beim Frühjahrsfachgespräch der German Unix User Group (GUUG) behandelte Stefan Schumacher die psychologischen Faktoren, mit denen Berater und Admins IT-Security zur Sache der Firmenmitarbeiter machen.]. *Linux-Magazin*. Verfügbar 22. Oktober 2009 unter http://www.linux-magazin.de/NEWS/GUUG-Fachgespraech-IT-Security-psychologisch-implementieren

Schläger, C., & Rührschneck, M. (2006). The TaskJuggler (http://www.taskjuggler.org) Manual. Verfügbar 7. März 2010 unter http://taskjuggler.org/download/manual-git.pdf

Schumacher, S. (2006a). Methoden zur Datensicherung – Strategien und Techniken für NetBSD. http://www.net-tex.de/backup.pdf

Schumacher, S. (2006b). Methoden zur Datensicherung – Strategien und Techniken für NetBSD. In German Unix User Group (Hrsg.), *Proceedings des GUUG Frühjahrsfachgespräches 2006* (S. 79–98). Lehmanns Media.

Schumacher, S. (2006c). Sicherung verteilter Systeme mit Bacula. *UpTimes*, 14–20. Verfügbar 7. November 2009 unter http://kaishakunin.com/publ/guug-uptimes-bacula.pdf

Schumacher, S. (2006d). Verschlüsselte Dateisysteme für NetBSD. *UpTimes*, 25–31. Verfügbar 7. November 2009 unter http://kaishakunin.com/publ/guug-uptimes-cgd_cfs.pdf

Schumacher, S. (2007a). Daten sicher löschen. *UpTimes*, 7–16. Verfügbar 7. November 2009 unter http://kaishakunin.com/publ/guug-uptimes-loeschen.pdf

Schumacher, S. (2007b). PostgreSQLs Datenbestände sichern. *UpTimes*, 4–7. Verfügbar 7. November 2009 unter http://kaishakunin.com/publ/guug-uptimes-postgresql.pdf

Schumacher, S. (2007c). Sicherung verteilter Systeme mit Bacula. *LinuxTag 2007 Konferenz-DVD*. Verfügbar 29. Juni 2007 unter http://www.LinuxTag.de/

Schumacher, S. (2007d). Systemaufrufe mit Systrace steuern. *UpTimes*, 12–19. Verfügbar 7. November 2009 unter http://kaishakunin.com/publ/guug-uptimes-systrace.pdf

Schumacher, S. (2007e). Systeme mit Systrace härten. In German Unix User Group (Hrsg.), *Proceedings des GUUG Frühjahrsfachgespräches 2007* (S. 35–44). Lehmanns Media.

Schumacher, S. (2007f). Systeme mit Systrace härten (Chaos Computer Club, Hrsg.). *Die Datenschleuder: Das wissenschaftliche Fachblatt für den Datenreisenden*, #91, 40–48. Verfügbar 3. Januar 2008 unter http://ds.ccc.de/pdfs/ds091.pdf

Schumacher, S. (2008a). *Kevin Mitnick: Die Kunst der Täuschung* [*Buchbesprechung*]. *searchnetworking.de/Sicherheit*. Verfügbar 3. März 2008 unter

http://www.searchnetworking.de/index.cfm?pid=3787&pk=110793

Schumacher, S. (2008b). *Kevin Mitnick: Die Kunst des Einbruchs* [*Buchbesprechung*]. *searchnetworking.de/Sicherheit*. Verfügbar 3. März 2008 unter http://www.searchnetworking.de/themenbereiche/sicherheit/bedrohungen-attacken/articles/110802/

Schumacher, S. (2009). Psychologische Grundlagen des Social-Engineering. In German Unix User Group (Hrsg.), *Proceedings des GUUG Frühjahrsfachgespräches 2009* (S. 77–98). Lehmanns Media.

von Foerster, H. (1993a). Prinzipien der Selbstorganisation im sozialen und betriebswirtschaftlichen Bereich. In S. J. Schmidt (Hrsg.), *Wissen und Gewissen: Versuch einer Brücke* (1. Aufl., S. 233–268). Suhrkamp.

von Foerster, H. (1993b). über selbst-organisierende Systeme und ihre Umwelten. In S. J. Schmidt (Hrsg.), *Wissen und Gewissen: Versuch einer Brücke* (1. Aufl., S. 211–232). Suhrkamp.

von Foerster, H. (1993c). Verstehen verstehen. In S. J. Schmidt (Hrsg.), *Wissen und Gewissen: Versuch einer Brücke* (1. Aufl., S. 282–299). Suhrkamp.

von Foerster, H. (1993d). Was ist Gedächtnis, daß es Rückschau *und* Vorschau ermöglicht? In S. J. Schmidt (Hrsg.), *Wissen und Gewissen: Versuch einer Brücke* (1. Aufl., S. 299–336). Suhrkamp.

von Foerster, H. (1993e). *Wissen und Gewissen: Versuch einer Brücke* (S. J. Schmidt, Hrsg.; 1. Aufl.). Suhrkamp.

von Foerster, H., & Pörksen, B. (2006). *Wahrheit ist die Erfindung eines Lügners: Gespräche für Skeptiker* (7. Aufl.). Carl-Auer-Systeme.

Watzlawick, P. (2007). *Wie wirklich ist die Wirklichkeit? Wahn, Täuschung, Verstehen* (6. Aufl.). Piper.