

Time Tracking Tool

Dokumentinformationen

Auftraggeber	4teamwork GmbH
Projekt	Time Tracking Tool
Projektkürzel	ftw.timetracker
Autor	Julian Infanger
Version	1.0
Verteiler	Julian Infanger Pascal Habegger Mike Lawson Manuel J. Schaffner
Druckdatum	16. März 2010
Status	Abgeschlossen

Änderungskontrolle

Nr.	Datum	Version	Kommentar	Autor
1	23.02.2010	1.0	Initialversion	Julian Infanger
2	02.03.2010	1.1	Voranalyse abschliessen	Julian Infanger
3	05.03.2010	1.2	Konzeptphase abschliessen	Julian Infanger
4	15.03.2010	1.3	Realisierungsphase abschliessen	Julian Infanger
5	15.03.2010	1.4	Dokumentation mit Quellcode ergänzen	Julian Infanger
6	16.03.2010	2.0	Dokument überarbeiten, korrigieren und abschliessen	Julian Infanger

Inhaltsverzeichnis

Dokumentinformationen	1
Inhaltsverzeichnis	2
Abbildungsverzeichnis	4
Tabellenverzeichnis	4
Management Summary	5
I. Ablauf und Umfeld	8
1. Aufgabenstellung	8
1.1. Ausgangslage	8
1.2. Auftragsformulierung	8
1.3. Mittel und Methoden	8
1.4. Projektorganisation	9
1.5. Projektrollen	9
2. Vorkenntnisse	9
3. Vorarbeiten	10
4. Firmenstandarts	10
5. Meilensteine	10
6. Arbeitsplan	11
7. Zeitplan	12
8. Arbeitsjournal	13
9. Projektjournal	24
10. Schlussbericht	28
10.1. Vergleich Soll / Ist	28
10.2. Persönliches Fazit	29
11. Unterschriften	29

II. Projektdokumentation	30
12. Voranalyse	30
12.1. Analyse Ist Zustand / Soll-Zustand	30
12.2. Pflichtenheft / Systemanforderungen	31
12.3. Systemziele	32
12.4. Varianten	34
12.4.1. Variante 1: Speichern als Archetypes-Objekte	34
12.4.2. Variante 2: Speichern in Annotations	35
12.4.3. Variante 3: Speichern in relationale Datenbank	35
12.5. Variantenentscheid	36
12.6. Wirtschaftlichkeit	36
12.7. Risikoanalyse	36
12.8. Informationssicherheit und Datenschutz	37
12.9. Lösungen suchen und Freigabe Phase Konzept	37
13. Konzept	37
13.1. Konzept entwickeln	37
13.1.1. Systemanforderungen	37
13.1.2. Systemarchitektur	39
13.1.3. Wirtschaftlichkeit	41
13.2. Fertigprodukte evaluieren	41
13.3. Schutzmassnahmen erarbeiten	41
13.4. Testkonzept, -prozedur	41
14. Realisierung	44
14.1. Klassen	44
14.2. GUI	45
14.3. Funktionen	45
14.3.1. Stopp	45
14.3.2. Rec	46
14.3.3. Tickets aus der Übersicht entfernen	46
14.3.4. Übertragen in Arbeitsrapporte	46
14.3.5. Runden der geleisteten Zeiten	46
14.4. Testprotokoll	47
15. Einführung	48
15.1. System einführen	48
16. Quellenverzeichnis	49
17. Glossar	49
18. Unterschriften	50

Abbildungsverzeichnis

1.	Die Projektorganisation	9
2.	Ein normales Ticket vom Extranet	30
3.	Die Rapportierung eines Aufwandes	31
4.	Zustandsdiagramm	33
5.	Die Bedienelemente	39
6.	Die Übersicht über alle gestoppten Zeiten	39
7.	Klassendiagramm ftw.timetracker	44
8.	Das Bedienungspanel	45
9.	Die Übersicht über alle gestoppten Zeiten	45

Tabellenverzeichnis

1.	Bedienelemente	5
2.	Projekttrollen	9
3.	Vorkenntnisse	10
4.	Meilensteine	10
5.	Bedienung auf einem Ticket	33
6.	Tabelle für den Variantenentscheid	36
7.	Definition der Testfälle	43
8.	Testprotokoll	47

Management Summary

Aufgabenstellung

Im Rahmen der IPA wird eine Erweiterung für das Extranet der Firma 4teamwork (basierend auf dem Opensource-CMS Plone) entwickelt. Momentan geschieht die Reportierung der Zeiten manuell. Dazu muss der Mitarbeiter das Ticket auswählen, für welches er gearbeitet hat. Dann trägt er Startzeit, Endzeit und einen Kommentar ein und speichert diese Daten in den zugehörigen Arbeitsrapport.

Um dem Mitarbeiter diesen Ablauf zu erleichtern, wird eine webbasierte Stoppuhr entwickelt, welche sich die Start- und Endzeiten pro Ticket merkt und diese zentral abspeichert. Somit kann der Mitarbeiter am Ende des Tages die Auflistung seiner Zeiten öffnen, um Korrekturen einzutragen, Kommentare anzupassen und die gestoppten Zeiten in die jeweiligen Arbeitsrapporte zu speichern.

Dieses Produkt besteht aus folgenden zwei Komponenten:

1. Bedienungselemente

Die Bedienungselemente werden unterhalb des Bannerbildes auf der rechten Seite des Browser-Fensters angezeigt. Somit hat man, egal wo man im Extranet ist, direkten Zugriff auf diese Elemente und kann so die Stoppuhr bedienen.






	öffnet die Auflistung der gestoppten Zeiten
	stoppt die Stoppuhr
	startet die Stoppuhr
	pausiert die Stoppuhr
	fortfahren der pausierten Stoppuhr
display	zeigt Informationen über das laufende Ticket

Tabelle 1: Bedienelemente

2. Auflistung der Zeiten

Diese Ansicht zeigt dem Mitarbeiter alle gestoppten Zeiten an. Diese werden in einem Eintrag pro Ticket zusammengefasst. Die aufgezeichneten Zeiten können dann vom Mitarbeiter betrachtet, aber nicht bearbeitet oder gelöscht werden. Er kann nur die zusammengefassten Einträge löschen oder vor dem Übertragen in die Arbeitsrapporte bearbeiten. Das Übertragen erfordert einen Kommentar zwingend, der die geleisteten Arbeiten beschreibt. Falls der Mitarbeiter bei den einzelnen gestoppten Zeiten einen Kommentar vergeben hat, werden diese, mit Komma getrennt, im Haupteintrag als Kommentar vorgeschlagen.

Varianten

Für das Projekt werden Informationen zum Ticket, sowie Start- und Endzeit und der Kommentar gespeichert. Um diese Daten zu speichern gibt es verschiedene Möglich-

keiten.

Es werden folgende drei Möglichkeiten miteinander verglichen:

- **Speichern als Archetypes-Objekt:** Die Daten werden als neue Archetypes-Objekte angelegt.
- **Speichern in Annotations:** Die Daten werden per Adapter in Annotations gespeichert.
- **Speichern in eine MySQL-Datenbank:** Die Daten werden mittels Statements in eine relationale Datenbank geschrieben.

Es wurde das Speichern in Annotations ausgewählt, da diese Variante für die besten Eigenschaften für dieses Projekt hat.

Konzept

Den ausführlichen Konzeptbericht ist in Abschnitt 13 zu finden.

Funktionen

- **Rec:** Starten der Stoppuhr
- **Stop:** Stoppen der Stoppuhr
- **Pause:** Pausieren der Messung
- **Play:** Aufheben der Pausierung
- **list_times:** Alle gestoppten Zeiten anzeigen
- **transfer:** Einträge in die Arbeitsrapporte übertragen
- **delete:** Einträge aus der Liste löschen

Speicherung der Daten Die Daten werden in Dictionaries gespeichert. Die genaue Struktur ist im Konzeptbericht beschrieben.

Testbericht

Die in der Konzeptphase definierten Tests habe ich in der Realisierung durchgeführt. Da ich die Testfälle als Kriterien für die Realisierung betrachtet habe, waren alle durchgeführten Tests erfolgreich.

Das Testprotokoll mit den beobachteten Resultaten ist in Abschnitt 14.4 zu finden.

Time Tracking Tool

Mittelbedarf

- **Hardware:** MacBook Pro
- **Programmiersprachen:** Python, HTML, JavaScript, CSS, L^AT_EX
- **Software:** TextMate, TexShop
- **Sicherung / Backup:** Subversion

Die Testumgebung ist eine Kopie von unserem Extranet (basierend auf Plone). Dieses gilt während der IPA gleichzeitig als produktive Umgebung.

Fazit

Ich bin mit dem Time Tracking Tool grundsätzlich sehr zufrieden. Da das Produkt recht einfach in der Bedienung ist und meiner Meinung nach für den Mitarbeiter auch eine Vereinfachung einer Tätigkeit ist, wird diese Erweiterung für das Extranet seinen Zweck erfüllen.

Zukunftsauaussichten

Ich könnte das Produkt noch mit einigen Funktionen erweitern, welche nicht im Rahmen der IPA definiert wurden.

- **Anwesenheitszeit berechnen:** Die Anwesenheitszeit der Mitarbeiter wird automatisch aus den gestoppten Zeiten berechnet.
- **Effektive / geschätzte Zeit:** Die geleistete Zeit eines Tickets wird mit der geschätzten Zeit des Tickets verglichen. Sind 80% der geschätzten Zeit überschritten, färbt sich das Display orange. Bei der Überschreitung von 100% der geschätzten Zeit wird das Display rot gefärbt.

Teil I.

Ablauf und Umfeld

1. Aufgabenstellung

1.1. Ausgangslage

Ticket-System Im webbasierten Extranet (basierend auf Plone) der Firma 4teamwork gibt es interne und externe Projekte. Jedes dieser Projekte besitzt einen Arbeitsrapport und einen Issue-Tracker. Im Issue-Tracker werden Tickets mit Aufgaben erfasst, welche dann den Mitarbeitern zugewiesen werden können. Jedem Ticket ist ein Arbeitsrapport zugewiesen (normalerweise der Arbeitsrapport des Projektes).

Erfassen der Arbeitszeit Wenn ein Mitarbeiter mit der Arbeit für ein Ticket beginnt, dann merkt er sich die Startzeit. Sobald er das Ticket erledigt hat, muss er sich die Endzeit merken. Diese beiden Zeiten kann er dann im verknüpften Arbeitsrapport eintragen und mit einem Kommentar über die erledigte Tätigkeit abspeichern. Das Problem ist, dass oftmals entweder die Startzeit, die Endzeit oder die gesamte Reportierung vergessen wird.

1.2. Auftragsformulierung

Zur Erleichterung dieses Ablaufs soll eine webbasierte Stoppuhr entwickelt werden, welche es dem Mitarbeiter ermöglicht, die Zeit, welche er an einem Ticket arbeitet, zu stoppen.

1.3. Mittel und Methoden

Ich arbeite auf einem MacBook Pro. Für dieses Projekt arbeite ich vor allem mit Python, HTML, JavaScript und CSS, welche ich mit dem TextMate Editor programmiere. Die Dokumentation schreibe ich mit \LaTeX , wozu ich den Editor TexShop verwende. Als Testumgebung installiere ich eine Kopie von unserem Extranet auf meinem Computer, welches während der IPA gleichzeitig als produktive Umgebung zählt. Damit die Daten (Produkt und Dokumentation) sicher gespeichert sind, speichere ich die neusten Versionen immer in der Quelltextverwaltung Subversion. Somit habe ich im Falle eines Datenverlusts nicht alle Daten verloren.

1.4. Projektorganisation

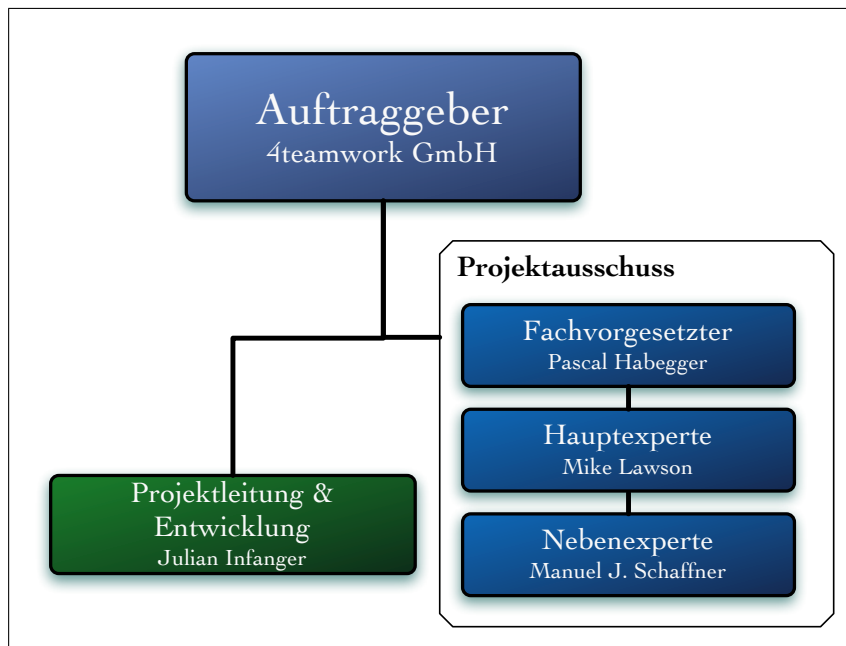


Abbildung 1: Die Projektorganisation

1.5. Projektrollen

Rolle	Person
Auftraggeber	4teamwork GmbH
Projektleitung	Julian Infanger
Fachvorgesetzter	Pascal Habegger
Hauptexperte	Mike Lawson
Nebenexperte	Manuel J. Schaffner

Tabelle 2: Projektrollen

2. Vorkenntnisse

In diesem Abschnitt ist eine kurze Übersicht über die für dieses Projekt relevanten Vorkenntnisse.

Time Tracking Tool

Python	6 Monate
HTML, CSS	3.5 Jahre
JavaScript	3 Jahre (wenig eingesetzt)
LaTeX	6 Monate (wenig eingesetzt)
Plone	6 Monate
Zope, Zope3-Komponentenarchitektur	6 Monate

Tabelle 3: Vorkenntnisse

3. Vorarbeiten

- **jQuery:** Auseinandersetzung mit jQuery mit Hilfe der jQuery-Dokumentation, Beispielen und Tutorials (<http://docs.jquery.com>).
- **LaTeX:** Diverse Dokumentationen und Beispiele studieren, um eine Dokumentvorlage zu erstellen.

4. Firmenstandarts

Bei der Firma 4teamwork GmbH gibt es keine Standarts betreffend Vorlagen für solche Dokumente, daher benutze ich die Vorlage, welche von pkorg zur Verfügung gestellt ist.

5. Meilensteine

Folgende Meilensteine habe ich in meinem Projekt definiert:

Datum	Beschreibung	erreicht
25.02.2010	Start der IPA	Ja
02.03.2010	Ende der Phase Voranalyse	Ja
05.03.2010	Ende der Phase Konzept	Ja
15.03.2010	Ende der Phase Realisierung	Ja
16.03.2010	Projektabschluss	Ja
29.03.2010	Präsentation der IPA	

Tabelle 4: Meilensteine

6. Arbeitsplan

Projektstart	5 h
Dokument eröffnen	
Arbeitsplan erstellen	
Zeitplan erstellen	
Voranalyse	6 h
Analyse Ist- / Soll-Zustand	
Pflichtenheft	
Systemziele	
Varianten / Variantenentscheid	
Wirtschaftlichkeit	
Risikoanalyse	
Informationssicherheit / Datenschutz	
Konzept	16 h
Konzept entwickeln	
Prototyp erstellen	
Testkonzept / Testprozedur	
Realisierung	24 h
Systemdesign erstellen	
System erstellen	
Testprotokoll	
Einführung vorbereiten	
Schutzmassnahmen umsetzen	
Dokumentation	22 h
Gesammelte Erfahrungen und Ergebnisse dokumentieren	
Grafiken und Diagramme erstellen	
Tagesjournale ausfüllen	
Dokumentation drucken	
sonstiges / Reserve	7 h
Sitzungen / Besprechungen / Expertenbesuche	
Reserve bei Problemen	
Total	80 h

8. Arbeitsjournal

Gemäss Art. 5 Absatz 2 der Wegleitung über die individuelle praktische Arbeit (IPA) an Lehrabschlussprüfungen des BBT vom 27. August 2001 gilt: „Die zu prüfende Person führt ein Arbeitsjournal. Sie dokumentiert darin täglich das Vorgehen, den Stand der Prüfungsarbeit, sämtliche fremde Hilfestellungen und besondere Vorkommnisse wie z.B. Änderungen der Aufgabenstellung, Arbeitsunterbrüche, organisatorische Probleme, Abweichungen von der Soll-Planung.“

Donnerstag, 25.02.2010 (halber Tag)

Tätigkeit	Soll	Ist
Paketstruktur erstellen	1	0.5
Initialversion des Dokuments erstellen	1	1
Aufgabenstellung formulieren	1.5	2
Buildout-Umgebung einrichten	0.5	0.5
Vorkenntnisse, Vorarbeiten und Firmenstandards definiert	1	1
Schwierigkeiten, Bemerkungen und Reflexion		
Heute habe ich mit dem Erstellen einer ersten Version der Dokumentation begonnen. Da ich die Dokumentation mit \LaTeX schreibe, musste ich zum Teil kleinere Sachen nachschlagen, da ich noch nicht alle grundlegenden \LaTeX -Befehle auswendig kenne. Ich habe die Aufgabenstellung formuliert, was keine grossen Probleme mit sich brachte. Ab und zu musste ich etwas studieren, was genau mit den einzelnen Punkten gemeint war. Auch beim Einrichten der Buildout-Umgebung und dem Erstellen der Projektstruktur traten keine Probleme auf.		
Nächste Schritte		
Als nächstes werde ich mir einen Arbeitsplan und einen Zeitplan erstellen. Ausserdem werde ich mit der Voranalyse weiterfahren und je nach Zeit mit einem kleinen Prototyp beginnen. Zudem werde ich mich auf den ersten Besuch des Hauptexperten (Mike Lawson) vorbereiten, indem ich Fragen und Unklarheiten aufschreibe und die nötigen Dokumente vorbereite.		

Time Tracking Tool

Freitag, 26.02.2010 (ganzer Tag)

Tätigkeit	Soll	Ist
Erstellen des Arbeitsplan	1	0.5
Erstellen des Zeitplanes	1	1.5
Analyse Ist- / Soll-Zustand	1	1
Besprechung mit Pascal Habegger: Auflistung der gestoppten Zeiten	0	0.5
Erster Besuch von Hauptexperte	1	1
Projektjournal erstellen und führen	1.5	2
Management Summary erstellen	1	1
Systemziele definieren	1	1
Schwierigkeiten, Bemerkungen und Reflexion		
<p>Für das Erstellen des Arbeitsplanes habe ich mich an die Projektphasen und die Überschriften der Dokumentation gehalten. Daher war ich etwas schneller als ich zuerst gedacht habe. Dafür habe ich mich beim Zeitplan etwas unterschätzt. Den Aufwand für die jeweiligen Phasen zu schätzen war nicht einfach, aber ich hoffe, dass die Schätzungen realistisch sind. Mit der Analyse des Ist- und des Soll-Zustands hatte ich keine Probleme, ich konnte mich da an die Definition aus der Aufgabenstellung halten.</p> <p>Ich hatte heute gleich zwei Sitzungen. In der ersten habe ich mich mit Pascal Habegger über die Auflistung der gestoppten Zeiten unterhalten. Die zweite war der Besuch des Hauptexperten Mike Lawson. Weitere Informationen zu diesen beiden Sitzungen sind im Projektjournal festgehalten.</p> <p>Ich habe zudem die Systemziele definiert und mit dem Management Summary begonnen.</p>		
Nächste Schritte		
<p>Am Montag werde ich das Management Summary weiter ergänzen und mit der Implementation eines ersten Prototyps beginnen. Zudem werde ich mir drei verschiedene Varianten für das Speichern der gestoppten Zeiten überlegen, beschreiben und diese im Variantenvergleich auswerten, um so zu entscheiden, welche ich für mein Projekt gebrauchen werde.</p>		

Time Tracking Tool

Montag, 01.03.2010 (ganzer Tag)

Tätigkeit	Soll	Ist
Mittel und Methoden	1	0.5
Informationssicherheit und Datenschutz	0.5	0.5
verschiedene Varianten auflisten	0.5	1
Variantenentscheid	0.5	0.5
Erstellen eines Prototyps	5	5.5
Schwierigkeiten, Bemerkungen und Reflexion		
<p>Heute Morgen habe ich zuerst die benötigten Mittel und Methoden festgehalten. Ich konnte diese vom Beschrieb auf pkorg übernehmen und anpassen, daher hat dies nicht so lange gedauert. Auch der Punkt Informationssicherheit und Datenschutz gab nicht extrem viel zu schreiben.</p> <p>Dann hatte ich den restlichen Tag Zeit um einen Prototypen zu erstellen. Eigentlich hatte ich gedacht, dass ich schneller bin, aber ich bin an kleinere Probleme mit dem Dictionary geraten. Ich wusste nicht genau, wie ich die Daten verschachteln soll, damit der Zugriff auf die nötigen Daten möglichst klein gehalten werden kann. Nachdem ich mir in Ruhe Gedanken gemacht hatte und zwei verschiedene Varianten getestet hatte, habe ich gemerkt, dass es sinnvoll ist, die jeweiligen Unterobjekte immer in neue Listen zu speichern (siehe Anforderungen an die Daten 13.1.1).</p>		
Nächste Schritte		
<p>Als nächstes werde ich beginnen die Funktionen für den Prototyp zu programmieren, da dieser momentan recht statisch ist. Ich hoffe, die Funktionen für das Speichern und Holen der Daten, sowie das Löschen aus dem Speicher erstellen zu können. Die Resultate werde ich im Konzept festhalten. Wenn noch genügend Zeit bleibt, werde ich beginnen, die Testfälle zu spezifizieren.</p>		

Time Tracking Tool

Dienstag, 02.03.2010 (ganzer Tag)

Tätigkeit	Soll	Ist
Abschluss der Phase Voranalyse	0.5	0
Prototyp erstellen (Funktionen)	3	4
Konzept: Beschreiben des Produktes und dessen Elemente	1.5	2
Dokumentation anpassen und Platzierung von Bildern / Tabellen	1	1
Erstellen von Bildern für die Dokumentation	1.5	0.5
Schwierigkeiten, Bemerkungen und Reflexion		
<p>Zuerst habe ich die Voranalyse-Phase abgeschlossen. Ansonsten habe ich mich weiter mit dem Prototyp beschäftigt. Ich habe viel Zeit in die verschiedenen Funktionen investiert. Die Funktionen Rec und Stop funktionieren. Auch die Auflistung der gestoppten Zeiten klappt nun und der Prototyp erfüllt somit seinen Zweck als erste Idee für das Projekt. Somit konnte ich unter anderem die Bedienung bestimmen und die Verschachtelung der Auflistung (siehe Abbildung 6) definieren.</p> <p>Ich habe dann gemerkt, dass ich noch viel Zeit in diesen Prototyp investieren könnte, aber dies wäre nicht der Sinn des Prototyps, und so habe ich am Nachmittag mit dem Konzept angefangen.</p> <p>Im Konzept brauchte ich einige Bilder und Tabellen. Ich habe also die Bilder erstellt. Mir ist aufgefallen, dass ich bis noch keine Captions (Bild- und Tabellenbeschreibungen) gebraucht hatte, also habe ich diese noch bei den bestehenden Tabellen ergänzt. Dabei habe ich festgestellt, dass \LaTeX die Tabellen und Bilder (welche in einer bestimmten Umgebung sein müssen, damit die Caption richtig gemacht wird) selbst positioniert. Dies muss ich unter Umständen vor dem Abschluss der Dokumentation noch anpassen, damit das Dokument nicht unübersichtlich wird.</p> <p>Zudem habe ich für das Konzept Mockups erstellt, welche das Aussehen des Time Tracking Tool beschreiben.</p>		
Nächste Schritte		
<p>Als nächstes werde ich das Konzept weiter bearbeiten. Ich denke nicht, dass mir der Donnerstag dafür reicht, da ich nur den halben Tag an der IPA arbeiten kann und der restliche Tag in der Berufsschule bin. Ich hoffe dass ich am Freitag die Konzeptphase abschliessen kann. Somit wäre ich im Zeitplan (siehe Meilensteine in Abschnitt 5).</p> <p>Die Testfälle werde ich auch am Freitag definieren, da ich heute keine Zeit dazu gefunden habe.</p>		

Time Tracking Tool

Donnerstag, 04.03.2010 (halber Tag)

Tätigkeit	Soll	Ist
Anforderungen an die Funktionalität	1	1.5
Anforderungen an die Daten	1	1.5
Anforderungen an die Informationssicherheit / Datenschutz	1	0.5
Testfälle definieren	1	0.5
Schwierigkeiten, Bemerkungen und Reflexion		
<p>Heute habe ich vor allem die Anforderungen an das System beschrieben. Dies hat mich zum Teil mehr Zeit gekostet, da ich an die Daten und die Funktionalitäten recht viel Anforderungen zu beschreiben hatte.</p> <p>Die restliche Zeit vom Morgen habe ich die Testfälle definiert. Da ich noch nicht alle Testfälle definieren konnte, werde ich die restlichen morgen machen. Ich denke, dass ich bis jetzt gut im Zeitplan liege und morgen die Konzeptphase abschliessen kann.</p>		
Nächste Schritte		
<p>Als nächstes werde ich die Testfälle fertig definieren und das Konzept nochmals durchschauen nach Fehler.</p> <p>Wenn dies alles gemacht ist, werde ich die Konzeptphase abschliessen.</p>		

Time Tracking Tool

Freitag, 05.03.2010 (ganzer Tag)

Tätigkeit	Soll	Ist
Projektorganisation (Diagramm) erstellen	0.5	0.5
Testfälle definieren	0.5	0.5
Projektsitzung mit dem Fachvorgesetzten	0.5	0.5
Journal anpassen	0.5	0.5
Dokumentation anpassen	1.5	2
Abschluss Konzeptphase	0.5	0.0
Realisierung: Programmieren der Funktionen	4	4
Schwierigkeiten, Bemerkungen und Reflexion		
<p>Am Morgen habe ich die Testfälle fertig definiert, sowie das Diagramm für die Projektorganisation erstellt. Dabei hatte ich keine Probleme. Danach hatte ich eine Besprechung mit Pascal Habegger, meinem Fachvorgesetzten. Dabei ging es vor allem darum, den Stand der Dinge zu besprechen und ich hatte noch einige Fragen. Ich habe die Besprechung im Projektjournal (in Abschnitt 9) dokumentiert.</p> <p>Nach der Besprechung hatte ich einige Anpassungen vorgenommen, welche ich in der Dokumentation nachführen musste.</p> <p>Vor dem Mittag konnte ich die Konzeptphase frühzeitig abschliessen und somit mit der Realisation beginnen. Ich habe gesamthaft 1,5 Stunden mehr als geplant für die Konzeptphase gebraucht. Der Grund dafür war, dass ich zu viel Zeit in die Erstellung eines Prototyps investiert habe. Ich habe mit der Erstellung der Funktionen begonnen. Dabei habe ich gemerkt, dass ich mit JavaScript nicht so schnell wie gedacht vorwärts komme. Ich musste viele Dinge im Internet nachschlagen.</p>		
Nächste Schritte		
<p>Ich hoffe, dass ich am Montag die grundlegenden Funktionen (Rec, Stopp, Play, Pause) abschliessen kann. Vor allem nehme ich mir vor, die Funktionen so abzuschliessen, dass die Anzeige der Knöpfe richtig aktualisiert wird, ohne die Seite neu zu laden. Dies ist über jQuery möglich. Dazu werde ich wieder die Dokumentation (http://api.jquery.com/jquery.ajax) zur Hand nehmen.</p>		

Time Tracking Tool

Montag, 08.03.2010 (ganzer Tag)

Tätigkeit	Soll	Ist
Fertigstellen der Funktion zum Löschen von gestoppten Tickets	1	0.5
Erstellen der Funktionen für Rec und Stop	1.5	2.5
Funktionen Play und Pause programmieren	1.5	2
Bedienungs-Panel aktualisieren mit jQuery	1.5	2
Verschiedene Ausbesserungen	0.5	0.5
Dokumentation nachführen	1.5	0.5
Schwierigkeiten, Bemerkungen und Reflexion		
<p>Am heutigen Tag habe ich praktisch nur am Erstellen des Systems gearbeitet. Ich konnte fast alle Grundfunktionen abschliessen. Mit der <code>delete_marked</code> Funktion können gestoppte Tickets aus der Auflistung gelöscht werden. Das Löschen der Tickets aus den Annotations geschieht im Hintergrund, und die Einträge werden per jQuery aus der Liste gelöscht. Dies bewirkt, dass man nicht warten muss, und die Seite nicht neu geladen wird. Die Funktionen Rec, Stop, Play und Pause sind auch fertiggestellt und funktionieren. Ich habe die Funktionen aber noch nicht getestet.</p> <p>Ich hatte Probleme, das Bedienpanel zu aktualisieren. Da ich die Funktionen mit jQuery im Hintergrund aufrufe, wird die Seite nicht neu geladen. Dies macht das ganze Tool massiv schneller. Aber dafür wird das Panel nicht aktualisiert. Zuerst wollte ich bei jedem Funktionsaufruf mit jQuery alle Knöpfe anpassen. Ich habe dann aber nach kurzer Zeit festgestellt, dass dies viel zu viel Aufwand ist. Also habe ich eine Funktion programmiert, welche das Bedienungs-Panel neu lädt. Somit muss nicht die ganze Seite neu geladen werden, sondern nur das Panel.</p> <p>Auch einige kleinere Ausbesserungen habe ich gemacht. Ich habe zum Beispiel die Tickets und Projekte verlinkt, sowie eine Funktion erstellt, welche das Total der Zeiten der Projekte zusammengerechnet.</p>		
Nächste Schritte		
<p>Morgen werde ich die Funktion zum Übertragen der gestoppten Zeiten in die Arbeitsrapporte programmieren.</p> <p>Zudem werde ich die Funktionen in der Dokumentation beschreiben. Wenn ich dann noch genügend Zeit habe, werde ich mir ein farbliches Design überlegen, und das Time Tracking Tool mittels CSS zu stylen. Ausserdem werde ich mit dem Fachvorgesetzten kurz besprechen, wie die gestoppte Zeit gerundet werden soll.</p>		

Time Tracking Tool

Dienstag, 09.03.2010 (ganzer Tag)

Tätigkeit	Soll	Ist
Funktionen dokumentieren	2.5	2
Übersicht anpassen, damit sie mit CSS gestylt werden kann	1	1
Funktionen anpassen	2	3
In Annotations PersistentDict anstelle von normalen Dictionaries speichern	1.5	1
Installationsanleitung schreiben	1	0.5
Schwierigkeiten, Bemerkungen und Reflexion		
<p>Zuerst habe ich die bestehenden Grundfunktionen im Realisierungsteil der Dokumentation beschrieben. Des Weiteren habe ich die Übersicht der gestoppten Zeiten so angepasst, dass diese mit CSS gestylt werden kann. So muss ich für das Stylen nur die CSS-Datei anpassen. Ausserdem habe ich eine kleine Funktion beim Speichern der Annotations eingebaut, welche die Dictionaries in persistente Dictionaries und die Listen in persistente Listen umwandelt. Dies hat zur Folge, dass alle Daten persistent gespeichert.</p> <p>Zudem habe ich eine kurze Installationsanleitung für die Einleitung geschrieben.</p> <p>Ich habe vernachlässigt, dass ich die Tickets von verschiedenen Tagen verwalten können muss. Daher musste ich mir die Views und Funktionen anpassen. Mit diesem Teil bin ich noch nicht ganz fertig geworden.</p>		
Nächste Schritte		
<p>Als nächstes werde ich die Funktionen und Views weiter anpassen, sodass die Tickets je nach Tag angezeigt und verwaltet werden können. Ich hoffe, dass ich fertig damit werde, da ich am Donnerstag nur den halben Tag Zeit dafür habe.</p>		

Time Tracking Tool

Donnerstag, 11.03.2010 (halber Tag)

Tätigkeit	Soll	Ist
Funktion zum Löschen der Tickets ohne Zeitmessungen	1	0.5
Funktion zum Löschen der Projekte ohne Tickets mit Zeitmessungen	1	0.5
View list_times anpassen	2	3
Schwierigkeiten, Bemerkungen und Reflexion		
<p>Heute habe ich zuerst die Funktionen erstellt, welche ein Ticket ohne Einträge, bzw. ein Projekt ohne Tickets aus dem Dictionary löscht.</p> <p>Danach habe ich die Übersicht (list_times) angepasst. Jetzt können die Tickets von einzelnen Tagen angezeigt werden, und es werden nicht alle Einträge auf einer Ansicht dargestellt.</p> <p>Das Erstellen dieser Funktionen hat mehr Zeit gebraucht als ich geplant hatte. Dies hatte damit zu tun, dass ich nicht gedacht hatte, dass das Navigieren in den Dictionaries so mühsam wird.</p>		
Nächste Schritte		
<p>Als nächstes werde ich die delete-Funktion so umbauen, dass nicht das ganze Ticket gelöscht wird, sondern nur die Einträge des heutigen Tages. Zudem werde ich mich auf den zweiten Besuch der Experten vorbereiten.</p> <p>Am Nachmittag werde ich die transfer-Funktion erstellen, damit die gestoppten Zeiten in die Arbeitsrapporte gespeichert werden können.</p>		

Time Tracking Tool

Freitag, 12.03.2010 (ganzer Tag)

Tätigkeit	Soll	Ist
Zweiter Besuch des Hauptexperten	1	1
Projektjournal nachführen	1	0.5
Runden-Funktion (plone.app.registry)	1.5	1
Delete-Funktion anpassen	1	1.5
Transfer-Funtion erstellen	3	4
Schwierigkeiten, Bemerkungen und Reflexion		
<p>Am Morgen hat der zweite Besuch des Hauptexperten (Mike Lawson) wie geplant stattgefunden. Danach habe ich die besprochenen Punkte im Projektjournal festgehalten (Abschnitt 9).</p> <p>Den restlichen Morgen habe ich damit verbracht, die delete-Funktion anzupassen, damit nur die Tickets des jeweiligen Tages gelöscht werden. Zudem habe ich mit plone.app.registry einen Wert in als Einstellung festgelegt. Dieser kann nach Belieben angepasst werden. Mit dem erwähnten Wert wird bestimmt, auf wie viele Minuten die zusammengerechnete Zeit für ein Ticket aufgerundet wird. Der Standartwert ist 15 Minuten.</p> <p>Den Nachmittag habe ich damit verbracht, die Funktion zu schreiben, mit welcher der Mitarbeiter seine gestoppten Zeiten in die Arbeitsrapporte übertragen lassen kann.</p> <p>Des Weiteren habe ich mir Gedanken gemacht, wie ich den Quellcode kommentieren werde. Ich habe mich entschieden, dies nach dem Standart für epydoc machen werde, da man so automatisch eine Dokumentation der Klassen, Funktionen, etc. generieren lassen kann. Da dies aber nicht Teil meiner IPA ist, werde ich dies nur machen, falls ich am Ende der IPA noch genügend Zeit dafür habe.</p>		
Nächste Schritte		
<p>Als nächstes werde ich die Dokumentation aktualisieren, damit diese auf dem neusten Stand ist. Ausserdem werde ich beginnen, den Quellcode zu dokumentieren. Wenn ich genügend Zeit habe, werde ich die definierten Testfälle testen und die Ergebnisse im Protokoll festhalten.</p>		

Time Tracking Tool

Montag, 15.03.2010 (ganzer Tag)

Tätigkeit	Soll	Ist
Testfälle durchführen und Ergebnisse festhalten	1	1
Eingabe verschiedener Zeitformaten berücksichtigen	1.5	1
Dokumentation anpassen	1.5	2
Code kommentieren	5	4
Code in Dokumentation einbinden	0	1
Schwierigkeiten, Bemerkungen und Reflexion		
<p>Zuerst habe ich alle Testfälle durchgeführt und die Ergebnisse dokumentiert. Danach habe ich in der Funktion zum Übertragen der gestoppten Zeiten in die Arbeitsrapporte eingebaut, dass die verschiedenen Zeitformaten berücksichtigt werden. Ausserdem habe ich den ganzen Quellcode kommentiert. Es wäre für ein anderes Mal sinnvoller, diesen immer laufend zu kommentieren, da ich jetzt am Schluss recht viel Zeit aufbringen musste, um dies nachzuführen. Den Quellcode habe ich dann auch in die Dokumentation eingebunden.</p> <p>Die Dokumentation habe ich dann noch ein bisschen angepasst und die Deckblätter ausgedruckt.</p>		
Nächste Schritte		
<p>Morgen werde ich noch die Dokumentation abschliessen und drucken. Danach werde ich mich für die Präsentation vorbereiten.</p>		

Dienstag, 16.03.2010 (halber Tag)

Tätigkeit	Soll	Ist
Schlussbericht schreiben	1	0.5
Management Summary fertig ergänzen	0.5	0.5
Reflexion Arbeitszeit total	0.5	0.5
Kontrolle und Überarbeitung Dokumentation	1.5	1
Dokumentation drucken und binden	1	2
Schwierigkeiten, Bemerkungen und Reflexion		
<p>Heute, am letzten Projekttag, habe ich den Schlussbericht mit dem Soll-/Ist-Vergleich und dem persönlichen Fazit geschrieben. Danach habe ich das Management Summary fertig gestellt und die Reflexion der Arbeitsjournale gemacht.</p> <p>Nachdem ich das Dokument nochmals durchgelesen, kontrolliert und korrigiert hatte, konnte ich es ausdrucken und binden.</p>		
Nächste Schritte		
<p>Vorbereitung auf die Präsentation.</p>		

Arbeitszeit total

Soll	80h
Ist	82h
Reflexion	Ich denke dass ich grundsätzlich die geschätzte Zeit gut eingehalten habe. Ich habe zum Teil etwas mehr Zeit aufgewendet als ich eingeplant hatte. Zuerst habe ich mir gedacht, dass ich noch einige zusätzlichen Ziele erfüllen könnte, welche nicht in der IPA definiert waren, aber ich habe dann festgestellt, dass ich nicht zu viel Zeit in optionale (mir selbst gestellte) Ziele investieren sollte, und mich stattdessen besser auf die Ziele, welche in Rahmen der IPA definiert worden waren, konzentriere.

9. Projektjournal

Im Projektjournal werden Informationen chronologisch gesammelt, welche im Verlauf der Arbeit eine Rolle spielten. Besprechungs-Protokolle mit Entscheiden und Abmachungen sind besonders wichtig.

25.02.2010: Start der IPA

26.02.2010: Besprechung mit dem Fachvorgesetzten

Themen:

- Diverse Fragen zur Auflistung der gestoppten Zeiten

Resultate:

- Beim Stoppen der Stoppuhr erscheint ein Dialog, wo der Mitarbeiter einen Kommentar eingeben kann (optional). Dies ist, damit der Mitarbeiter sich den Kommentar nicht bis zur Übertragung in die Arbeitsrapporte merken muss.
- Die gestoppten Zeiten von gleichen Tickets werden in einem Ticket zusammengefasst. Die unterstehenden Zeiten können ausgeklappt, aber nicht bearbeitet oder übertragen werden. Es wird die Start- und Endzeit und der Kommentar angezeigt. Die Kommentare aller „Untereinträge“ werden mit Komma getrennt im zusammengefassten Eintrag angezeigt. Dieser zusammengefasste Eintrag kann dann in die Arbeitsrapporte übertragen werden.
- Die Kostenstelle kann nicht ausgewählt werden, diese ist auf dem Ticket definiert. Die Kostenstelle (sofern vorhanden) wird beim Speichern in den Arbeitsrapport von dem Ticket übernommen.

Time Tracking Tool

- Die Modifikation der Einträge ist erst unmittelbar vor dem Übertragen in die Arbeitsrapporte möglich. Änderungen können nicht zwischengespeichert werden.
- Der Mitarbeiter hat ein zusätzliches Feld in der Übersicht, in welchem die verrechenbare Zeit angezeigt wird. Je nach Ticket ist diese die Arbeitszeit oder nichts. Diese kann auch vor dem Übertragen verändert werden.
- Es gibt zwei mögliche erlaubte Formate, um die Zeit zu speichern.
 hh:mm (Bsp: 1:30 = 1h 30min)
 hh.mm (Bsp: 1.30 = 1h 18min)
- Der Mitarbeiter wird informiert, falls er Einträge hat, welche älter als einen Tag sind und noch nicht in die Arbeitsrapporte übertragen wurden.

26.02.2010: Erster Besuch des Hauptexperten (Mike Lawson)

Themen:

- Vorstellen
- Aufgabenstellung besprechen
- Unterschied Projektjournal / Arbeitsjournal
- Ablauf der IPA besprechen
- Besprechen der Dokumentation
- Fragen beantworten

Resultate:

- **Quellcode in der Dokumentation:** Nur relevante Codeausschnitte in die Dokumentation kopieren. Der gesamte Code kann auf Anfrage geliefert werden.
- **Arbeitsjournal:** bis ca. eine Seite pro Tag mit Reflexionen bezogen auf die IPA.
- **Projektjournal:** Beinhaltet Besprechungsprotokolle, Abmachungen und relevante Entscheidungen (mit Begründung) welche sich auf das Projekt beziehen.
- **Web Summary:** Wird bei der Präsentation an die Experten abgegeben. Der Umfang ist etwa 1.5 Seiten. Der Inhalt ist eine Beschreibung des Projekts und eine persönliche Reflexion.
- **Kriterium 119 (Brauchbarkeit):** Während der IPA und bei der Demonstration ist eine Test-Instanz des Extranet (Kopie) die produktive Umgebung. Das Time Tracking Tool wird nach der IPA auf dem produktiven Extranet installiert (neuer Release).

02.03.2010: Ende der Phase Voranalyse

05.03.2010: Projektsitzung mit dem Fachvorgesetzten

Themen:

- Stand der Dinge
- Präsentation Prototyp
- Diverse Fragen

Resultate:

- Das automatische Ausrechnen der Anwesenheitszeit ist kein Muss-Ziel und rückt daher in den Hintergrund. Dies wird als Aussicht im Schlussbericht aufgelistet.
- Im Dictionary `all_times` soll die UID¹ anstelle der URL gespeichert werden. Ich habe mich so entschieden, dass ich die UID für die Erkennung verwende, da diese eindeutig ist.
- Der Knopf „alle übertragen“ kann weggelassen werden, da der Mitarbeiter alle Tickets markieren und diese dann übertragen kann.

05.03.2010: Ende der Phase Konzept

09.03.2010: Runden von geleisteter Arbeit

Themen:

- Wie soll die geleistete Zeit auf einem Ticket als Vorschlag für die Rapportierung gerundet werden?

Resultate:

- Die Zeit soll auf eine bestimmte Minutenzahl aufgerundet werden (z.B: 10 oder 15 Minuten). Diese Zahl kann in `plone.registry` definiert werden.

12.03.2010: Zweiter Besuch des Hauptexperten

Themen:

- Kontrolle Zeitplan, Arbeits- und Projektjournal
- Stand der Arbeit und der Dokumentation besprechen
- Ablauf der Abgabe und Präsentation

¹Eindeutige Erkennungs-ID von Plone-Objekten.

Resultate:

- Zeitplan, Arbeitsplan und Projektjournal entsprechen den Anforderungen.
- In der Dokumentation muss ein gutes Mass zwischen Bildern / Diagrammen und Text gefunden werden.
Es muss **jeder** selbst geschriebene Codeteil in der Dokumentation enthalten sein. Weitere nur dann, wenn diese für das Verständnis notwendig sind.
- **Abgabe:** Die Dokumentation muss am Dienstag, 16.03.2010, bis spätestens 12.00 Uhr auf pkorg.ch hochgeladen werden. Die Zeit des Poststempels ist nicht relevant, muss aber noch am selben Tag sein. Es ist zu beachten, dass die Dokumentation am Mittwoch eintreffen sollte. Daher wird die Dokumentation mit A-Post geschickt.
Zu beachten ist ausserdem, dass die Dokumentation eine Maximalgrösse von 20MB nicht überschreiten darf, da diese sonst nicht mehr hochgeladen werden kann.
- Der Ablauf der Präsentation wurde besprochen. Anwesend sind nur die Experten, der Fachvorgesetzte und unter Umständen der Lehrmeister. Für andere Interessenten kann eine weitere Präsentation ausserhalb der IPA gehalten werden.

15.03.2010: Ende der Phase Realisierung

16.03.2010: Projektabschluss

10. Schlussbericht

10.1. Vergleich Soll / Ist

Kriterium	Aktueller Status	Ok
Einfaches Starten und Stoppen der Zeit auf einem Ticket	Die Bedienung der Stoppuhr ermöglicht einfaches Starten und Stoppen der Zeit auf einem Ticket.	Ok
Die Bedienung wird angepasst, wenn sich nicht auf einem Ticket befindet	Die Bedienung wird je nach Zustand individuell angepasst. Befindet man sich nicht auf einem Ticket, sind diverse Aktionen deaktiviert.	Ok
Es kann immer nur die Zeit von einem Ticket gestoppt werden	Wenn man die Zeit eines anderen Tickets messen will, muss die laufende Zeitmessung abgebrochen werden.	Ok
Auflistung der gestoppten Zeiten	Es kann eine Übersicht über alle gestoppten Zeiten zu Tickets angezeigt werden.	Ok
Bearbeiten / Löschen der gestoppten Zeiten	Tickets können gelöscht werden. Ausserdem können sie vor dem Übertragen in die Arbeitsrapporte angepasst werden.	Ok
Gestoppte Zeiten einzeln oder als Auswahl in die Arbeitsrapporte übertragen	Die gestoppten Zeiten werden pro Ticket gesammelt und können so in die Arbeitsrapporte übertragen werden. Einzelne und mehrere Tickets möglich.	Ok
Kommentar ist zum Übertragen in die Arbeitsrapporte zwingend nötig	Es erscheint eine Fehlermeldung, ein Ticket ohne Kommentar wird nicht übertragen.	Ok

10.2. Persönliches Fazit

Ich habe die gewünschten Kriterien erfüllt und das Projekt rechtzeitig beendet. Das Time Tracking Tool kann und wird im Extranet der Firma 4teamwork GmbH produktiv eingesetzt werden.

Ich denke, dass das Produkt den Mitarbeitern die Arbeitszeitrapportierung erleichtern wird und ebenfalls die Qualität der rapportierten Zeiten steigern.

Mit diesem Projekt habe ich vor allem im Bereich jQuery viele neue Erfahrungen gesammelt. Zudem konnte ich mein Wissen im Umgang mit Plone, Zope und Python erweitern. Da ich die ganze Dokumentation mit L^AT_EX geschrieben habe, habe ich viel Neues über L^AT_EX gelernt.

Im Grossen und Ganzen hat das Projekt mir persönlich viel neue Erfahrungen gebracht, und ich hoffe, dass die Firma 4teamwork GmbH einen ähnlich grossen Nutzen davon haben wird.

11. Unterschriften

Datum	Name	Unterschrift
	Julian Infanger, Lernender	
	Pascal Habegger, Fachvorgesetzter	

Teil II.

Projektdokumentation

Für die Projektdokumentation meiner IPA verwende ich die Projektmethode HERMES, auf welcher die Vorlage von pkorg.ch aufbaut. Im Rahmen meines Projektes habe ich diese Vorlage an meine Bedürfnissen angepasst.

12. Voranalyse

12.1. Analyse Ist Zustand / Soll-Zustand

Ist: Die Rapportierung der geleisteten Arbeiten geschieht bei der Firma 4teamwork im webbasierten Extranet, welches auf dem Opensource Web-CMS Plone aufbaut. Für jede anstehende Arbeit gibt es Tickets (Abbildung 2) welche an einen Arbeitsrapport gebunden sind. Alle Projekte (interne und externe) haben einen „Ticket Tracker“, in welchem diese Tickets erfasst und verwaltet werden. Wenn ein Mitarbeiter an einem solchen Ticket arbeitet, merkt er sich die Start- und Endzeit, bzw. die geleistete Zeit und trägt diese im verknüpften Arbeitsrapport mit einem Kommentar über die Tätigkeiten ein (Abbildung 3).

Zurück zum Tracker
Letzte Änderung durch Infanger Julian am 19.01.2010 14:42
ftw.booking übersetzen

Antwort hinzufügen

Ticket bearbeiten

Nummer	#11
Status	offen
Bereich	
Typ	
Einstufung	Mittel
Eingereicht durch	Infanger Julian
Eingereicht am	19.01.2010
Fortschritt	20 %
Geschätzte Stunden	3.0
Geleistete Stunden	0.92
Zuständigkeit	Infanger Julian
Verknüpfungen	⊕

Arbeitszeit rapportieren

Datum	Title	h / verr.	Autor	Rapport	Kostenstelle	Status
19.01.2010	Tracker ftwBooking #11 Übersetzungen:	0.5 / 0.0	julian.infanger	Arbeitsrapport ftwBooking		aktiv

Abbildung 2: Ein normales Ticket vom Extranet

Time Tracking Tool

Arbeitszeit rapportieren
Arbeitsrapport ftwBooking : ftwBooking

05 . 03 . 2010 09:00 - 09:00 oder Dauer 00:00

Tracker ftwBooking #11 Übersetzungen:

Keine Kostenstelle

Add

Abbildung 3: Die Rapportierung eines Aufwandes

Soll: Neu soll eine Stoppuhr entwickelt werden, mit welcher dieser Vorgang erleichtert wird. So kann der Mitarbeiter, beim Beginn der Arbeit für ein Ticket, die Stoppuhr starten, welche sich dann die Startzeit merkt.

Hat der Mitarbeiter seine Arbeit beendet, dann kann er die Uhr stoppen, und diese rechnet dann die Dauer der Arbeit aus und speichert diese.

Am Ende des Tages kann der Mitarbeiter eine Auflistung seiner gestoppten Zeiten anzeigen lassen. Diese kann er dann kontrollieren, anpassen und mit einem Kommentar versehen in den dazugehörigen Arbeitsrapport übertragen.

12.2. Pflichtenheft / Systemanforderungen

Kriterien

- Einfaches Starten und Stoppen der Zeit auf einem Ticket
- Die Bedienung wird angepasst, wenn sich nicht auf einem Ticket befindet
- Es kann immer nur die Zeit von einem Ticket gestoppt werden
- Auflistung der gestoppten Zeiten
- Bearbeiten / Löschen der gestoppten Zeiten
- Gestoppte Zeiten einzeln oder als Auswahl in die Arbeitsrapporte übertragen
- Kommentar ist zum Übertragen in die Arbeitsrapporte zwingend nötig

Anwendungsbereiche Der Timetracker wird von der Firma 4teamwork eingesetzt. Dafür wird das auf dem Opensource CMS Plone basierende Extranet benötigt.

Produktübersicht Der Timetracker ermöglicht das Starten und Stoppen einer Zeit auf einem Ticket.

Dazu wird eine Bedienung mit den Knöpfen „zur Übersicht“, „Stopp“, „Rec“ und „Play / Pause“ oben rechts erstellt. Ausserdem befindet sich unterhalb dieser Bedienung ein

Display, welches anzeigt zu welchem Ticket man die Zeit stoppt und wie lange man schon an diesem Ticket arbeitet. Die Übersicht ist eine Auflistung der gestoppten Zeiten, welche noch nicht in den Arbeitsrapport übertragen wurden. Diese Zeiten sind nach Datum und Projekt gegliedert. Die Einträge setzen sich aus allen gestoppten Zeiten zu einem Ticket zusammen. Hier hat der Mitarbeiter die Möglichkeit die Einträge zu bearbeiten oder zu löschen. Er kann dann einzelne, alle oder eine Auswahl von Zeiten in die Arbeitsrapporte übertragen. Dazu ist zwingend ein Kommentar nötig, welcher zusammen mit dem Titel des Tickets im Arbeitsrapport gespeichert wird.

Zeiten, welche unvollständig sind (z.B: wurde die Zeit gestartet, aber noch nicht gestoppt) werden in der Übersicht als solche markiert und können auch nicht in die Rapporte übertragen werden.

Produktfunktionen

- **Rec:** Es wird ein neuer Eintrag mit Informationen zu Benutzer, Startzeit und Ticket erstellt
- **Stopp:** Der Eintrag wird mit der Endzeit ergänzt und in die Liste der gestoppten Zeiten übertragen.
- **Pause:** Die Stoppuhr wird pausiert.
- **Play:** Die Pausierung wird beendet.
- **get_timelist:** Gibt eine Liste mit den gestoppten Zeiten zurück
- **get_current_time:** Gibt Informationen zu dem Ticket zurück, zu welchem gerade die Zeit gestoppt wird (wenn vorhanden).
- **delete_entry:** Die gestoppte Zeit wird gelöscht.
- **transfer_entries:** Diese Funktion überträgt eine oder mehrere gestoppte Zeiten in die jeweiligen Arbeitsrapporte.

Produktleistungen / Qualitätsanforderungen Die Daten dürfen nicht gelöscht werden (auch nicht unvollständige) wenn sich der Benutzer abmeldet oder der Browser geschlossen wird. Ebenfalls sollen die Daten noch vorhanden sein, wenn das Extranet neu gestartet wird.

Die erfassten Zeiten sind in Stunden angegeben.

12.3. Systemziele

Einfaches Starten und Stoppen der Uhr Der Mitarbeiter kann auf einem Ticket die Zeit stoppen, welche er an diesem arbeitet. Dafür hat das Produkt verschiedene Knöpfe, welche unterhalb der Navigation gut sichtbar platziert werden. Somit kann der Mitarbeiter einfach den „record“-Knopf drücken und es wird die Startzeit zu diesem Ticket

Time Tracking Tool

gespeichert. Wenn der Mitarbeiter den „Stopp“-Knopf drückt, dann wird eine Endzeit gespeichert.

Individuelle Bedienung Die Bedienungs-Knöpfe ändern sich je nach Zustand des Timetrackers. Ausserdem wird unterschieden, ob man sich im Browser auf einem Ticket oder einem anderen Inhaltstyp befindet.

Folgendes Diagramm beschreibt die möglichen Zustände.

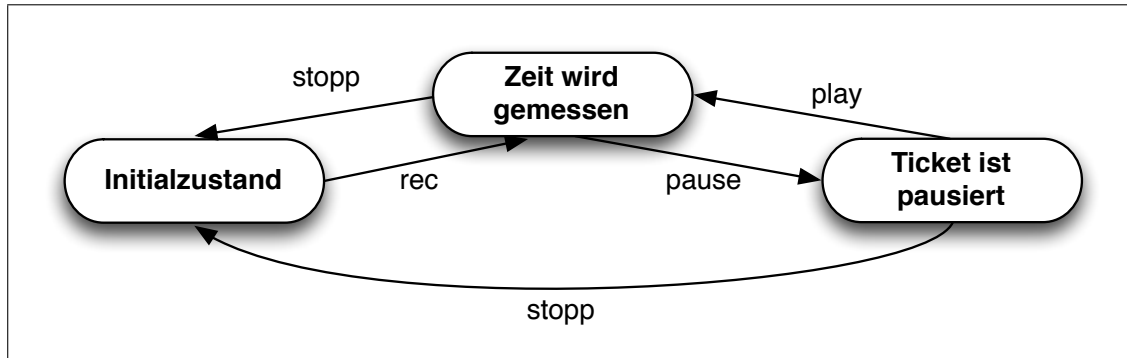


Abbildung 4: Zustandsdiagramm

	Initialzustand	Zeit wird gemessen	Ticket ist pausiert
	öffnet die Übersicht der gestoppten Zeiten		
	deaktiviert	setzt Endzeit	
	setzt Startzeit	deaktiviert	
	unsichtbar		hebt Pause auf
	deaktiviert	pausiert Zeit	unsichtbar
display	leer	Ticketnummer und gestoppte Zeit	

Tabelle 5: Bedienung auf einem Ticket

Auflistung der gestoppten Zeiten Um dem Mitarbeiter am Ende des Tages eine Übersicht über seine geleisteten Arbeiten zu geben, werden alle gestoppten Zeiten (Einträge) angezeigt. Diese sind nach Datum, Projekt und ID sortiert um eine einfache Struktur zu erstellen. Es werden alle Einträge zu einem Ticket zusammengerechnet und als Eintrag angezeigt.

Anzeigen von genauen Zeiten Die Einträge werden wie bereits beschrieben aus allen gestoppten Zeiten zu einem Ticket zusammengerechnet. Der Mitarbeiter hat die

Möglichkeit, diese einzelnen Zeiten anzusehen, indem er diese ausklappt. Er kann diese Einträge aber weder bearbeiten noch löschen. Diese Zeiten dienen nur zur Übersicht und können daher auch nicht in die Rapporte übertragen werden. Angezeigt wird ihm die Startzeit, die Endzeit und der Kommentar (falls eingegeben). Die eingegebenen Kommentare werden, mit Komma getrennt, als Kommentar für den zusammengefassten Eintrag vorgeschlagen.

Übertragung in Arbeitsrapporte Der Mitarbeiter kann seine gestoppten Zeiten in den Arbeitsrapport übertragen. Dazu hat er in der Auflistung seiner Zeiten folgende Möglichkeiten:

- einzelne Einträge in den Arbeitsrapport übertragen
- mehrere markierte Einträge in den Arbeitsrapport übertragen

Um die Einträge zu übertragen ist ein Kommentar, welcher die getätigten Arbeiten kurz beschreibt zwingend nötig.

12.4. Varianten

Hier unterscheide ich drei Varianten, um die Daten (Infos zum Ticket, Kommentar, Start-, Endzeit und Status) zu speichern.

Es werden folgende Kriterien unterschieden:

- **Speicherplatzverbrauch:** Wie viel Speicherplatz braucht die Variante? Gewichtung: 20%.
- **Geschwindigkeit:** Wie schnell ist die Variante? Wie ist die Geschwindigkeit bei vielen Einträgen? Gewichtung: 50%.
- **Flexibilität:** Wie einfach ist es mit der Variante, das Produkt zu erweitern? Gewichtung: 10%.
- **Entwicklungsaufwand:** Wie gross ist der Aufwand, diese Variante zu verwenden? Müssen spezielle Vorkehrungen getroffen werden? Gewichtung: 20%.

Für jedes dieser Kriterien werden maximal fünf Punkte vergeben, wobei 1 bedeutet, dass das Kriterium nicht erfüllt wird und 5, dass das Kriterium bestens erfüllt wird. Diese werden mit dem Faktor (Gewichtung / 10) multipliziert. Die Variante mit den meisten Punkten eignet sich am besten für mein Projekt.

12.4.1. Variante 1: Speichern als Archetypes-Objekte

Vorgehen Die Daten werden als neues Archetypes-Objekt² gespeichert. So wird für jeden Eintrag ein Objekt erstellt, welches die nötigen Felder hat, um die Eigenschaften zu speichern. Archetypes-Objekte speichern sehr viele Informationen und haben viele Attribute, welche für dieses Projekt nicht benötigt werden.

²Methode für die Erstellung von Inhalts-Objekten (<http://plone.org/products/archetypes>)

Vorteile / Nachteile Die erstellten Archetype-Objekte besitzen viele Attribute. Daher ist der Entwicklungsaufwand relativ klein, aber da nicht alle dieser Attribute gebraucht werden und jedes mal ein Archetype-Objekt erstellt wird, benötigt diese Variante viel Speicherplatz und kann bei vielen Objekten langsam werden.

Diese Objekte sind im ZMI sichtbar, was ihnen noch zusätzliche Flexibilität gibt.

- + Geringer Entwicklungsaufwand
- + Im ZMI sichtbar
- Viel Speicherplatz, da Overhead an Informationen / Attributen
- Geschwindigkeit

12.4.2. Variante 2: Speichern in Annotations

Vorgehen Die Einträge werden als Dictionary in Annotations gespeichert. Dafür wird ein Adapter erstellt, welcher die benötigten get- und set-Methoden enthält.

Vorteile / Nachteile Da diese Annotations jedem beliebigen Inhaltstyp angehängt werden können, bietet diese Variante grosse Flexibilität. So kann man die Annotations einfach an das benutzereigene Verzeichnis hängen, und somit hat man für jeden Benutzer einen eindeutigen Speicherplatz für die Annotations.

Da ein Dictionary mit Text gespeichert wird, braucht es weniger Speicherplatz und ist schneller als die Variante 1. Der Entwicklungsaufwand ist etwas höher als bei Variante 1 aber immer noch gering.

Da die Annotations nicht im ZMI sichtbar sind, verlieren sie etwas an Flexibilität.

- + Flexibilität
- + weniger Speicherplatz
- + Geschwindigkeit
- Nicht im ZMI sichtbar.

12.4.3. Variante 3: Speichern in relationale Datenbank

Vorgehen Die Einträge werden in eine MySQL-Datenbank gespeichert. Dabei wird pro gestoppte Zeit ein neuer Eintrag in der Datenbank gemacht.

Vorteile / Nachteile Der grosse Vorteil von einer MySQL Datenbank ist, dass diese sehr schnell ist, vor allem wenn viel Daten gespeichert werden. Aber man muss alle SQL-Statements selber implementieren. Da die im Extranet installierten Produkte keine SQL-Datenbank benötigen, muss extra ein MySQL-Server und SQL-Adapter installiert

Time Tracking Tool

werden. Zudem muss man sich speziell Gedanken um die Sicherheitsaspekte machen, um unter anderem SQL-Injection³ zu vermeiden.

- + Geschwindigkeit (auch bei vielen Daten)
- Statements implementieren
- Höherer Aufwand bzgl. Security
- MySQL-Server benötigt

12.5. Variantenentscheid

Aufgrund des Vergleiches der verschiedenen Varianten (siehe Tabelle 6), entscheide ich mich für Variante 2: Speichern in Annotations.

Kriterium	Variante 1	Variante 2	Variante 3	Gewichtung
Speicherplatzverbrauch	1	5	3	20% (x2)
Geschwindigkeit	1	4	5	50% (x5)
Flexibilität	4	4	4	10% (x1)
Entwicklungsaufwand	5	4	1	20% (x2)
Total:	21	42	37	100%

Tabelle 6: Tabelle für den Variantenentscheid

12.6. Wirtschaftlichkeit

Kosten Die Kosten des Projektes betreffen den Aufwand (10 Tage). Ansonsten entstehen keine Kosten, da das Projekt nach Abschluss funktioniert. Die Bedienung ist selbsterklärend, daher braucht es auch keine Einführung.

Nutzen Durch die Benutzung des Timetrackers wird die Zeit, welche zum Rapportieren gebraucht wird reduziert. Ausserdem wird durch die Notwendigkeit eines Kommentars die Erstellung leerer Einträge verhindert.

Dieses Projekt verspricht grosse Zeiteinsparungen bei der Arbeitszeiterfassung durch die Mitarbeiter. Ausserdem wird die Quantität und die Qualität der Rapporte gesteigert.

12.7. Risikoanalyse

Da die Rapporte jederzeit auch manuell erstellt werden können, besteht kein grosses Risiko, falls das Projekt scheitern sollte.

³Eingabe von schädlichem Code im Statement (<http://de.wikipedia.org/wiki/SQL-Injection>)

12.8. Informationssicherheit und Datenschutz

Zugriffsrechte auf die gestoppten Zeiten Jeder Mitarbeiter hat seinen eigenen Login, um sich im Extranet einzuloggen. Jeder Mitarbeiter hat so nur Zugriff auf seine gestoppten Zeiten. Es gibt keine speziellen Benutzer, welche die gestoppten Zeiten von allen Benutzern ansehen können.

Benutzer Da das CMS Plone schon über ein Anmeldeverfahren verfügt, muss ich dies in meinem Projekt nicht speziell beachten.

12.9. Lösungen suchen und Freigabe Phase Konzept

Für das in der Voranalyse (Abschnitt 12) definierte Projekt wird die Phase Konzept freigegeben:

Datum	Name	Unterschrift
	4teamwork GmbH, Auftraggeber	

13. Konzept

In der Konzepterarbeitung werden die Grundlagen für die Realisierung meines Projektes entwickelt.

13.1. Konzept entwickeln

13.1.1. Systemanforderungen

Folgende Anforderungen stellen sich an die Lösung:

Anforderungen an die Funktionalität

- **Rec:** Starten der Stoppuhr
Das Starten der Stoppuhr ist nur möglich, wenn man sich auf einem Ticket befindet. Beim Starten werden Daten zum Ticket sowie die momentane Zeit (als Startzeit) in das Dictionary⁴ `current_times` (beschrieben in Abschnitt **Anforderungen an die Daten**).
- **Stop:** Stoppen der Stoppuhr
Dieser Knopf ist nur aktiv, wenn eine Zeit gestoppt wird. Beim Stoppen wird der

⁴Datentyp um Werte zu speichern. Bestehen immer aus key : value Paaren

Mitarbeiter gefragt, was er an diesem Ticket gearbeitet hat. Er kann einen Kommentar eingeben. Dieser Kommentar ist aber nicht notwendig, erst beim Übertragen in den Arbeitsrapport muss der Mitarbeiter eine Beschreibung der Arbeitstätigkeit als Kommentar eingeben. Der optionale Kommentar wird dann mit den bereits im `current_list` Dictionary gespeicherten Daten und der momentanen Zeit (als Endzeit) in das Dictionary `all_times` (beschrieben in Abschnitt **Anforderungen an die Daten**) gespeichert. Der bestehende Eintrag im Dictionary `current_list` wird dann gelöscht.

- **Pause:** Pausieren der Messung
Das Pausieren einer Zeitmessung ist nur möglich, wenn eine Zeit gestoppt wird. Es geschieht dann das Gleiche wie wenn man den Stopp-Knopf drückt, abgesehen davon, dass das Dictionary `current_list` nicht komplett gelöscht wird, sondern es wird nur die Start- und Endzeit gelöscht. Somit ist noch bekannt, an welchem Ticket man gearbeitet hat.
- **Play:** Aufheben der Pausierung
Dies ist nur möglich, wenn man ein Ticket pausiert hat. Dadurch wird dem Dictionary `current_list` die momentane Zeit (als Startzeit) gesetzt.
- **list_times:** Alle gestoppten Zeiten anzeigen
Dieser Knopf blendet über JavaScript (jQuery⁵) eine Übersicht über alle gestoppten Zeiten ein. Diese Liste ist nach Projekten gegliedert. Jedes Projekt beinhaltet die Tickets, welche zu diesem Projekt gehören. Jedes Ticket beinhaltet die einzelnen gestoppten Zeiten. Diese können aufgeklappt werden und dienen zur Information. Daher können sie nicht bearbeitet, gelöscht oder übertragen werden. Diese Übersicht ist in Abbildung 6 ersichtlich.
- **transfer:** Einträge in die Arbeitsrapporte übertragen
Mit dieser Funktion können die gestoppten Zeiten in die jeweiligen Arbeitsrapporte übertragen werden. Diese können einzeln, mehrere (Auswahl) oder alle zusammen übertragen werden. Dazu ist ein Kommentar nötig; dieser beinhaltet im Normalenfall einige Stichworte zu der geleisteten Arbeit. Der default-Wert wird von den einzelnen Zeiten zusammengetragen und mit Komma getrennt. Der Kommentar, die geleistete und die verrechenbare Zeit kann vor dem Übertragen bearbeitet werden. Dazu kann man die Stunden und Minuten mit einem Doppelpunkt trennen, oder die Stunden als Dezimalzahl angeben, welche mit Komma getrennt werden, angeben.
- **delete:** Einträge aus der Liste löschen
Um einen Eintrag zu löschen muss man diesen (oder mehrere) mit den Checkboxes auswählen. Diese Auswahl wird nach einer verlangter Bestätigung des Mitarbeiters aus dem Dictionary `all_times` gelöscht.

⁵JavaScript-Framework (<http://jquery.com>)

Time Tracking Tool

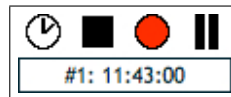


Abbildung 5: Die Bedienelemente

Meine offenen Zeiten			
heute			
<input checked="" type="checkbox"/> Titel	Kommentar	Zeit	
<input checked="" type="checkbox"/> Anwesenheitszeit		7.5	▶
OpenGever		4.5	
<input type="checkbox"/> #348 Tests für Inhalte	Integrationstests erstellt	2.5	▶
<input type="checkbox"/> #335 OpenGever.4teamwork.ch: Dokumente auschecken funktioniert nicht		2.0	▶
emtkva		3	
<input checked="" type="checkbox"/> #34 Download als Zip-File	Entwicklung	0.8	▶
<input type="checkbox"/> #71 Klassifizierung funktioniert nicht mehr richtig		1.5	▶
<input checked="" type="checkbox"/> #57 E-Mail Text bei Einreichen	Mailtemplate angepasst	0.7	▶
		markierte rapportieren	alle rapportieren
02.02.2010			
<input checked="" type="checkbox"/> Titel	Kommentar	Zeit	
OpenGever		0.5	
<input type="checkbox"/> #335 OpenGever.4teamwork.ch: Dokumente auschecken funktioniert nicht	Telefongespräch	0.5	▶
		markierte rapportieren	alle rapportieren

Abbildung 6: Die Übersicht über alle gestoppten Zeiten

Anforderungen an die Informationssicherheit und den Datenschutz Jeder Mitarbeiter hat nur Zugriff auf seine eigenen gestoppten Zeiten. Jeder Benutzer hat ein Ordner, welcher sein Home-Folder ist. Da die Annotations immer an dieses Verzeichnis gebunden werden, ist dies gewährleistet. Somit hat jeder Benutzer seine eigenen (eindeutigen) Annotations.

Jeder Mitarbeiter einen eigenen Login (von Plone). Daher kann dieser nur auf seine Annotations zugreifen.

13.1.2. Systemarchitektur

Speicherung der Daten Im Variantenentscheid (Abschnitt 12.5) habe ich mich für das Speichern in Annotations entschieden.

Beim Erstellen eines Prototyps habe ich mich entschieden, die Daten in zwei Dictionaries zu speichern. Wenn der Mitarbeiter die Messung der Zeit abbricht (mit dem Pause-

Time Tracking Tool

Stop- oder Rec-Knopf) dann werden diese Informationen zusammen mit der Endzeit und einem Kommentar (wenn eingegeben) in das zweite Dictionary gespeichert.

- **Aktuelle Messung:**

Hier wird das Projekt, die ID des Tickets und die Startzeit gespeichert. Diese Liste zeigt an, für welches Ticket man gerade in diesem Moment die Zeit stoppt. Daher ist es auch nur möglich jeweils ein Ticket in diesem Dictionary zu speichern.

Listing 1: Aufbau dictionary current_times

```
1 {
2     'project_uid' : '5043928761ca737ea62bf070024125e5',
3     'project_title' : 'OpenGever',
4     'ticket_id' : '2',
5     'ticket_uid' : '536961353e10b78c59b468354809963b',
6     'ticket_title' : '#2 ein Test-Ticket',
7     'start' : (2010, 03, 04, 08, 30)
8 }
```

- **Alle gestoppten Zeiten:**

In diesem zweiten Dictionary befinden sich alle Zeiten, welche vom Mitarbeiter gestoppt wurden. Diese dienen der Auflistung der gestoppten Daten. Wird der Eintrag in die Arbeitsrapporte übertragen, dann wird dieses Ticket aus der Liste gelöscht. Diese Liste ist nach Projekt und Ticket gegliedert. Jedes Projekt hat einen Namen und ein Dictionary mit Tickets. Jedes von diesen Tickets hat wiederum einen Namen und eine Liste mit gestoppten Zeiten, welche eine Startzeit, eine Endzeit und einen Kommentar haben. Für die Identifikation der Projekten und den Tickets wird die eindeutige UID gebraucht.

Listing 2: Aufbau dictionary all_times

```
1 {
2     '5043928761ca737ea62bf070024125e5' :
3     { 'title' : 'OpenGever', 'items' : {
4         '7143928761ba276ea62ba071524925b0' : {
5             'title' : '#23 irgendetwas funktioniert nicht', 'items' :
6             [{
7                 'start': (2010, 3, 1, 8),
8                 'end': (2010, 3, 1, 8, 30),
9                 'comment': '',
10            },
11            {
12                'start': (2010, 3, 1, 9),
13                'end': (2010, 3, 1, 9, 30),
14                'comment': 'irgendetwas gemacht',
15            }
16        ]
17    },
18    '1f87928ab761ca77a12bf07006212a41' : {
19        'title': '#26 Test Ticket', 'items' :
20        [{
21            'start': (2010, 3, 1, 13),
22            'end': (2010, 3, 1, 13, 30),
23            'comment': 'das ist ein Test',
24        }
25    ]
26    }
```



```
23     }}
24   }
25 },
26 '1082a928761ac737e62bf00241255' : {
27   'title' : 'EMT KVA', 'items' : {}
28 }
29 }
```

13.1.3. Wirtschaftlichkeit

Das Projekt lohnt sich. Die Begründung für diese Erkenntnis im Voranalysebericht in Abschnitt 12.6 hat sich nicht verändert.

13.2. Fertigprodukte evaluieren

Im Rahmen von Plone existieren keine vergleichbaren Produkte. Dies wurde vom Fachvorgesetzten Pascal Habegger abgeklärt, und aus diesem Grund entstand die Idee für dieses Projektes.

13.3. Schutzmassnahmen erarbeiten

Für dieses Projekt sind keine speziellen Schutzmassnahmen nötig, da Plone bereits über ein sicheres Anmeldeverfahren verfügt. Ausserdem wird das Extranet auf einem sicheren Server ausgeführt.

13.4. Testkonzept, -prozedur

Testfälle spezifizieren Hier sind die einzelnen Testfälle definiert.

Nr	Beschreibung	Erwartetes Resultat
1	individuelle Bedienung	Die Bedienung entspricht den Definitionen (Abbildung 4 und Tabelle 5)
2	Starten der Stoppuhr Auf einem Ticket wird der Rec-Knopf gedrückt.	Falls noch kein Ticket gestoppt wird, wird ein neuer Eintrag in das Dictionary current_time gemacht.
3	Stoppen der Stoppuhr Bei einem laufenden Ticket wird der Stop-Knopf gedrückt.	Es wird nach einem Kommentar gefragt. Dieser wird mit der aktuellen Uhrzeit und den Informationen zum Ticket im Dictionary all_times gespeichert.
4	Pausieren des Tickets Bei einem laufenden Ticket wird der Pause-Knopf gedrückt	Das laufende Ticket wird gestoppt (wie beim Stoppen eines Tickets). Der Eintrag wird nicht aus dem Dictionary gelöscht.
5	Aufheben der Pausierung Bei einem pausierten Ticket wird der Play-Knopf gedrückt.	Der Eintrag, welcher sich im Dictionary all_times befindet (ohne Startzeit) wird gestartet.

Time Tracking Tool

6	Anzeigen der gestoppten Zeiten Es wird der Auflistung-Knopf gedrückt.	Alle gestoppten Zeiten werden korrekt gegliedert mittels JavaScript (jQuery) eingebundet. Die angezeigte Seite muss nicht verlassen werden. Die Tickets sind aufklappbar, und man sieht die einzelnen Aufwände. Die Projekte und Tickets sind korrekt verlinkt.
7	Übertragen einer gestoppten Zeit Eine einzelne Zeit wird mit dem Übertragen-Knopf in den Arbeitsrapport übertragen	Die gewählte Zeit wird mit dem Kommentar und der Zeit (geleistete und verrechenbare) in den richtigen Arbeitsrapport übertragen und aus der Liste aller gestoppten Zeiten entfernt.
8	Mehrere gestoppte Zeiten übertragen Es werden mehrere Tickets per Checkbox ausgewählt und in die Rapporte übertragen.	Alle gewählten Zeiten werden mit den jeweiligen Kommentaren und der Zeiten (geleistete und verrechenbare) in die richtigen Arbeitsrapporte übertragen und aus der Liste der gestoppten Zeiten entfernt.
9	Löschen von mehreren Einträgen Es werden mehrere Einträge per Checkbox ausgewählt und gelöscht.	Der Mitarbeiter wird zu einer Bestätigung aufgefordert. Je nach Antwort werden die gewählten Ticket aus dem Dictionary all_times gelöscht.
10	Übertragen ohne Kommentar Es wird beim Übertragen mindestens ein Kommentarfeld leer gelassen	Es erscheint eine Fehlermeldung, welche den Mitarbeiter darauf hinweist, dass an einer Stelle keinen Kommentar eingegeben wurde. Die Tickets werden nicht übertragen. Das betreffende Kommentarfeld wird markiert.
11	Fehlerhafte Eingabe geleistete Zeit / verrechenbare Zeit Es wird an mindestens einer Stelle ein falsches Format für die Zeit eingegeben.	Eine Fehlermeldung, welche auf die fehlerhafte Eingabe hinweist, erscheint. Die Tickets werden nicht übertragen. Die betreffenden Felder werden hervorgehoben.
12	Eingabe von verschiedenen Zeitformaten Es werden beide möglichen Zeitformaten (hh:mm / hh.mm) eingegeben.	Es sind zwei Möglichkeiten erlaubt (Stunden und Minuten, Dezimalzahl von Stunden). Die gewählte Variante wird erkannt und die Zeit korrekt gespeichert.
13	Browser wird unerwartet beendet Der Browser wird neu gestartet und der Benutzer meldet sich neu an.	Wird das Extranet erneut im Browser geöffnet, sind die Daten nach der Anmeldung unverändert.

Time Tracking Tool

14	Extranet wird mit anderem Browser geöffnet Der Browser wird ganz beendet und das Extranet wird mit einem anderen Browser aufgerufen.	Die gestoppten Zeiten sind noch vorhanden und unverändert. Falls ein Ticket gestartet / pausiert war, ist keine Änderung vorhanden.
15	Extranet wird neu gestartet Das Extranet wird beendet und neu gestartet.	Die gestoppten Zeiten sind noch vorhanden und unverändert. Falls ein Ticket gestartet / pausiert war, ist keine Änderung vorhanden.
16	Benutzung von mehreren Benutzern Mehrere Benutzer benützen gleichzeitig das Time Tracking Tool	Jeder Benutzer hat nur Zugriff auf seine Daten. Die verschiedenen Benutzer beeinflussen sich gegenseitig nicht.
17	Ticket ohne Arbeitsrapport Es wird ein Ticket übertragen, welches keinen verknüpften Arbeitsrapport hat.	Es erscheint eine Fehlermeldung, welche den Benutzer informiert, welches Ticket keinen verknüpften Arbeitsrapport hat.

Tabelle 7: Definition der Testfälle

14. Realisierung

14.1. Klassen

Folgendes Diagramm beschreibt die erstellten Python Klassen:

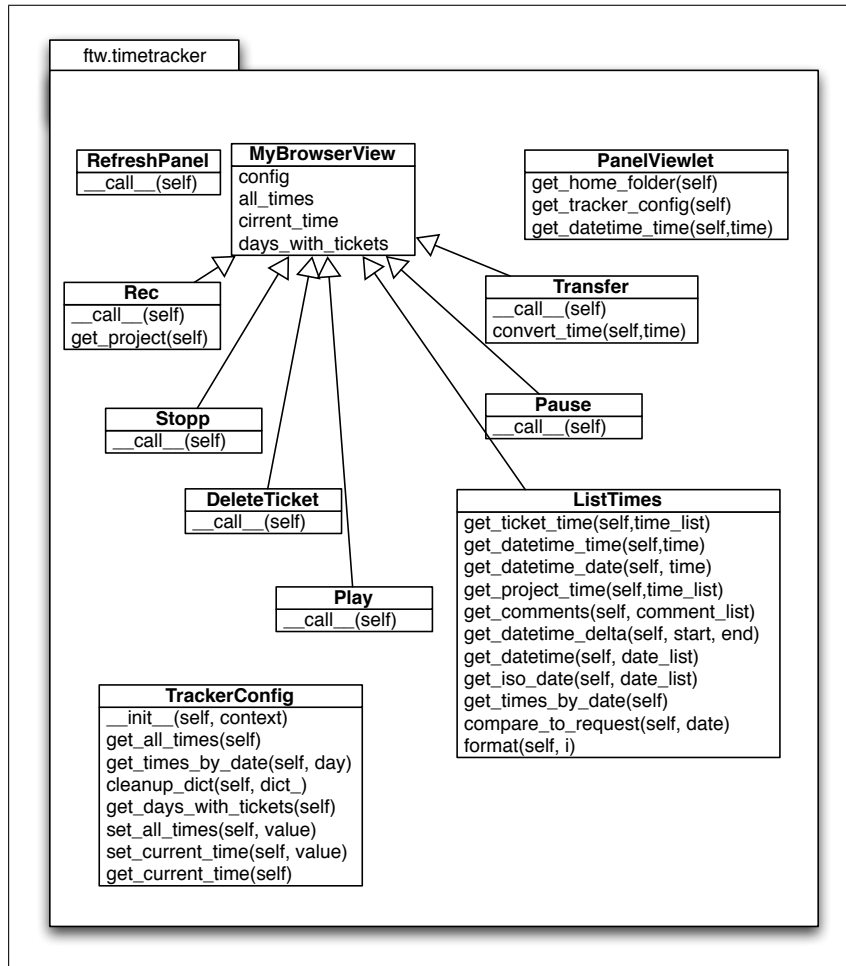


Abbildung 7: Klassendiagramm ftw.timetracker

Time Tracking Tool

14.2. GUI

Die grafische Benutzeroberflächen des Time Tracking Tools sehen wie folgt aus:

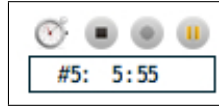


Abbildung 8: Das Bedienungspanel

Sie haben gestoppte Tickets, welche Sie noch nicht verrechnet haben in der Vergangenheit.					
15.03.2010					
15.03.2010					
<input type="checkbox"/>	Titel	Kommentar	Zeit	verr.	
Zweites Projekt			0.5	0.5	
<input type="checkbox"/>	#4 ein weiteres ticket ohne sinn		0.25	0.25	➡
<input type="checkbox"/>	#5 ein weiteres ticket ohne sinn 2	as	0.25	0.25	➡
	15.03.2010: 08:17 - 08:17	as	0.0		
	15.03.2010: 08:20 - 08:21	-	0.01		
	15.03.2010: 08:21 - 08:21	-	0.0		
	15.03.2010: 08:21 - 08:21	-	0.0		
<input type="button" value="markierte übertragen"/> <input type="button" value="markierte löschen"/>					

Abbildung 9: Die Übersicht über alle gestoppten Zeiten

14.3. Funktionen

In diesem Abschnitt beschreibe ich einige Grundfunktionen des Time Tracking Tools. Für die Funktionen habe ich jeweils eine neue Browserview erstellt, welche ich mit jQuery im Hintergrund aufrufe. Dazu verwende ich jQuery.ajax⁶.

14.3.1. Stopp

Diese BrowserView stoppt ein laufendes Ticket. Dazu wird zuerst geprüft, ob das Dictionary current_time existiert. Wenn dieses Dictionary keinen Start-Wert hat, ist das Ticket pausiert. In dem Fall wird das Dictionary geleert und es muss nichts gespeichert werden. Ansonsten wird per JavaScript ein Eingabefeld für einen Kommentar angezeigt.

Es wird im all_times Dictionary im gewünschten Ticket ein neuer Eintrag mit Start-, Endzeit sowie Kommentar erstellt. Dazu wird zuerst geprüft ob das benötigte Projekt und Ticket schon existieren und werden nötigenfalls hinzugefügt. Ausserdem wird das

⁶<http://api.jquery.com/jquery.ajax>

Dictionary `current_times` geleert.
Der Code ist im Anhang (`browser.py`) zu sehen.

14.3.2. Rec

Die `BrowserView Rec` prüft zuerst, ob sich der Mitarbeiter auf einem Ticket befindet. Ist dies der Fall, dann werden Projekt-UID, Projekt-Titel, Ticket-UID, Ticket-Titel, Ticket-ID und die Startzeit in das Dictionary `current_time` gespeichert.
Diese `BrowserView` ist in ?? zu finden.

14.3.3. Tickets aus der Übersicht entfernen

Um gestoppte Tickets aus der Übersicht zu löschen, wurde die `BrowserView Delete-Ticket` erstellt. Dazu kann man in der Auflistung die gewünschte(n) Tickets auswählen und den Löschen-Knopf betätigen.

Es wird dann geprüft, ob diese existieren und gelöscht werden können. Wenn dies der Fall ist werden sie aus dem Dictionary `all_times` entfernt. Nach dem Funktionsaufruf, wird noch mit einer zusätzlichen Funktion geprüft, ob es leere Projekte im HTML-Code hat. Diese werden dann auch ausgeblendet, sowie die gelöschten Tickets. Dies wird so gemacht, damit man nicht die ganze Seite neu laden muss.

Siehe dazu `browser.py` im Anhang.

14.3.4. Übertragen in Arbeitsrapporte

Für das Übertragen der gestoppten Zeiten in die Arbeitsrapporte habe ich die `BrowserView Transfer` erstellt (siehe `browser.py` im Anhang).

Wird der Knopf zum Übertragen in die Arbeitsrapporte gedrückt, dann werden zuerst die nötigen Parameter (Ticket-UID, Kommentar, geleistete Zeit, verrechenbare Zeit und Datum) gesammelt und per JavaScript im Hintergrund an die `BrowserView` geschickt. Danach wird die geleistete und verrechenbare Zeit ermittelt, unter Berücksichtigung der verschiedenen Zeitformaten. Dann wird das Ticket, bzw. dessen Arbeitsrapport geholt und ein neuer Eintrag mit den Eigenschaften aus den Parametern erstellt.

Zum Abschluss wird das Ticket aus der Übersicht gelöscht.

14.3.5. Runden der geleisteten Zeiten

Die geleisteten Zeiten werden pro Ticket zusammen gerechnet. Danach werden sie auf eine bestimmte Anzahl Minuten aufgerundet.

Diese Anzahl Minuten wird mit `plone.app.registry` gespeichert. Dazu wird das Packet `plone.app.registry` in das Projekt `ftw.timetracker` aufgenommen.

Dazu wird das Property `round_step` erstellt, welches den Standartwert 15 Minuten hat.

14.4. Testprotokoll

Hier werden die in der Konzeptphase (Tabelle 7) erarbeiteten Testfälle durchgeführt.

Nr	Beobachtungen / Resultat	Ok
1	Die Bedienung entspricht den definierten Erwartungen	Ok
2	Die Startzeit wird gesetzt. Das Display und die Knöpfe werden aktualisiert.	Ok
3	Es wird eine Endzeit gesetzt und der Aufwand wird zum passenden Ticket gespeichert. Falls ein Ticket pausiert ist, wird das pausierte aus der Liste gelöscht.	Ok
4	Das Ticket wird gestoppt, aber der Eintrag wird nicht aus dem current_list dictionary gelöscht.	Ok
5	Das Ticket wird erneut gestartet, unabhängig davon, ob sich der Benutzer auf diesem Ticket befindet.	Ok
6	Die Übersicht wird korrekt eingeblendet. Die Tickets werden nach Datum angezeigt.	Ok
7	Es wird eine neue Zeit im definierten Arbeitsrapport gespeichert. Die Daten stimmen. Der Eintrag wird danach aus der Liste entfernt.	Ok
8	Es werden für alle gewählten Tickets die jeweiligen Einträge in die Arbeitsrapporte gemacht.	Ok
9	Die gewählten Tickets werden aus der Liste und aus dem Dictionary (Annotations) entfernt. Es werden nur die Tickets vom gewählten Tag gelöscht.	Ok
10	Die Kommentarfelder der Tickets ohne Kommentar werden rot markiert.	Ok
11	Das fehlerhafte Zeitfeld wird rot markiert.	Ok
12	Die Zeitformate werden richtig verarbeitet und gespeichert.	Ok
13	Die Dictionaries sind nach dem unerwarteten Beenden des Browsers unverändert.	Ok
14	Die Dictionaries sind in beiden unterschiedlichen Browsern genau gleich.	Ok
15	Die Dictionaries sind nach dem Neustart des Extranets gleich wie vorher.	Ok
16	Die Benutzer haben keinen Einfluss aufeinander.	Ok
17	Es wird eine Fehlermeldung angezeigt und die gemessene Zeit wird nicht übertragen.	Ok

Tabelle 8: Testprotokoll

15. Einführung

15.1. System einführen

Die Installation des Produktes auf dem produktiven Extranet ist nicht Gegenstand meiner IPA, da während meiner IPA die lokale Kopie des Extranets als produktives System zählt.

Nachfolgend ist eine Installationsanleitung, welche erläutert, wie das Time Tracking Tool installiert werden muss.

Installationsanleitung

1. Produkt in den src-Ordner auschecken

Um das Time Tracking Tool auszuchecken, muss man im Terminal in den buildout-Ordner wechseln und dann folgenden Code ausführen:

```
svn co https://svn.4teamwork.ch/repos/ftw/ftw.timetracker/trunk  
src/ftw.timetracker
```

Mit diesem Befehl wird das Paket ftw.timetracker aus dem Repository in den src-Ordner ausgecheckt.

2. Buildout anpassen

In der Datei buildout.cfg muss folgendermassen angepasst werden.

Listing 3: buildout.cfg

```
1  ...  
2  eggs += ftw.timetracker  
3  ...  
4  develop += src/ftw.timetracker  
5  ...  
6  zcml += ftw.timetracker  
7  ...
```

3. Buildout starten

Damit die gemachten Änderungen in der Datei buildout.cfg wirksam werden, muss man das Buildout starten. Dazu muss man sich mit dem Terminal in den Buildout-Ordner navigieren. Danach kann man `bin/buildout` ausführen. Wenn dies keinen Fehler gegeben hat, kann man die Instanz mit dem Befehl

`bin/instance fg` starten.

16. Quellenverzeichnis

Bücher

- **Der L^AT_EX Begleiter** (Frank Mittelbach, Michel Groossens)
ISBN: 978-3-8273-7166-9
- **L^AT_EX Einführung Band 1** (Helmut Kopka)
ISBN: 3-8273-1557-3

Webseiten

- <http://www.python.org>
Offizielle Webseite der dynamischen, objektorientierten Sprache Python
- <http://www.plone.org>
Offizielle Webseite des Open Source Content Management Systems Plone
- <http://docs.jquery.com>
Offizielle Webseite vom JavaScript Framework jQuery
- <http://epydoc.sourceforge.net/epydoc.html>
Epydoc Standard für API-Dokumentation in Python-Code

17. Glossar

A

Annotations Annotations sind ein Konzept zur Speicherung von Daten, die nicht direkt auf dem zugehörigen Objekt gespeichert werden, sondern als eigenständiges Objekt., S. 3.

Archetypes Archetypes ist eine Methode für das Erstellen von neuen Inhaltstypen in Plone. Dazu muss man nur ein Schema und eine Klasse definieren, die benötigten Standard-Ansichten (Erstellen, Bearbeiten, Anzeigen, Löschen, Berechtigungen Verwalten, etc) werden von Archetypes zur Verfügung gestellt. (<http://plone.org/products/archetypes>), S. 3.

B

Buildout Werkzeug für das automatisierte Installieren von Python Paketen und Applikationen. (<http://pypi.python.org/pypi/zc.buildout>), S. 13.

Time Tracking Tool

C

CMS Content Management System (Inhaltsverwaltungssystem): Ein System zum Erstellen und verwalten von Inhalten, meist für Webbasierte Anwendungen, S. 5.

I

IPA Individuelle Praktische Arbeit, S. 5.

P

persistente Bei persistent gespeicherten Daten wird sichergestellt, dass diese nicht in einem flüchtigen Speicher gespeichert werden. Das bedeutet, dass die gespeicherten Daten nicht gelöscht werden, wenn das Programm beendet wird., S. 20.

Plone Content Management System (siehe <http://www.plone.org>), S. 5.

Python Python ist eine dynamische, objektorientierte Programmiersprache (siehe <http://www.python.org>), S. 7.

Z

ZMI Zope Managment Interface. Web-Interface für die Verwaltung und Konfiguration des Zope Webapplikationsservers., S. 35.

Zope (*Z Object Publishing Environment*) ist ein freier Anwendungsserver, geschrieben in Python. (<http://www.zope.org/>), S. 10.

18. Unterschriften

Datum	Name	Unterschrift
	Julian Infanger, Lernender	
	Pascal Habegger, Fachvorgesetzter	

19. Anhang

Quelltext

Um ein besseres Verständnis zu ermöglichen, folgt hier die Auflistung aller Dateien, die erstellt wurden.

Modul `ftw.timetracker`

```
setup.py
ftw/
  __init__.py
  timetracker/
    __init__.py
    configure.zcml
    interfaces.py
    tracker_config.py
    version.txt
    browser/
      __init__.py
      browser.py
      configure.zcml
      list_times.py
      viewlets.py
      images/
        add.png
        clock.png
        pause.png
        pause_inactive.png
        play.png
        play_inactive.png
        rec.png
        rec_inactive.png
        stopp.png
        stopp_inactive.png
      javascript/
        jquery-1.4.1.min.js
        timetracker.js
      stylesheet/
        timetracker.css
      templates/
        list_times.pt
        panel.pt
    locales/
      ftw.timetracker.pot
    de/
      LC_MESSAGES/
        ftw.timetracker.mo
        ftw.timetracker.po
    profiles/
      default/
        cssregistry.xml
        jsregistry.xml
        metadata.xml
        registry.xml
        viewlets.xml
```

Time Tracking Tool

Listing 4: configure.zcml

```
1 <configure
2     xmlns="http://namespaces.zope.org/zope"
3     xmlns:genericsetup="http://namespaces.zope.org/genericsetup"
4     xmlns:browser="http://namespaces.zope.org/browser"
5     xmlns:il8n="http://namespaces.zope.org/il8n"
6     il8n_domain="ftw.timetracker">
7
8     <!-- Include configuration for dependencies listed in setup.py -->
9     <includeDependencies package="." />
10
11     <il8n:registerTranslations directory="locales" />
12
13     <include package=".browser" />
14
15     <!-- Register an extension profile to make the product installable -->
16     <genericsetup:registerProfile
17         name="default"
18         title="ftw.timetracker"
19         description=""
20         directory="profiles/default"
21         provides="Products.GenericSetup.interfaces.EXTENSION"
22     />
23
24     <adapter
25         provides=".interfaces.ITrackerConfig"
26         for="*"
27         factory=".tracker_config.TrackerConfig"
28     />
29
30 </configure>
```

Listing 5: interfaces.py

```
1 #
2 # File:      interfaces.py
3 # Author:    Julian Infanger <julian.infanger@4teamwork.ch>
4 # Modified:  16.03.2010
5 #
6 # Copyright (c) 2010 by 4teamwork.ch
7
8 # imports
9 from zope.interface import Interface
10 from zope import schema
11
12 class ITrackerSettings(Interface):
13     """
14     Class for registering properties in plone.app.registry.
15     Here the property round_step is defined.
16     """
17     round_step = schema.Int(title = u"Rundungsschritte (Minuten)",
18                             description = u"legen Sie fest, auf wie viele Minuten die
19                                         Zeiten fuer ein Ticket aufgerundet werden sollen.",
20                             default = 15)
21
22 class ITrackerConfig(Interface):
23     """
24     This is the Interface for the tracker_config.
25     """
```

Time Tracking Tool

Listing 6: tracker_config.py

```
1  #
2  # File:      tracker_config.py
3  # Author:    Julian Infanger <julian.infanger@4teamwork.ch>
4  # Modified:  16.03.2010
5  #
6  # Copyright (c) 2010 by 4teamwork.ch
7
8  # imports
9  import copy
10 import datetime
11
12 from persistent.dict import PersistentDict
13 from persistent.list import PersistentList
14
15 from ftw.timetracker.interfaces import ITrackerConfig
16
17 from zope.annotation.interfaces import IAnnotations
18 from zope.interface import implements
19
20 def persistent_aware(item):
21     """
22     This function assures the persistency of the dictionaries.
23     It gets an item and checks recursive the content, if the item is a dictionary
24     or a list/tuple it converts it to a PersistentDict or a PersistentList.
25     @param item:      a dictionary, list, tuple or any other item
26     @return:          PersistentDict, PersistentList or any other item
27     """
28     if isinstance(item, dict) or isinstance(item, PersistentDict):
29         newitem = PersistentDict()
30         for key, value in item.items():
31             newitem[key] = persistent_aware(value)
32         return newitem
33     elif isinstance(item, list) or isinstance(item, tuple) or isinstance(item,
34         PersistentList):
35         newitem = PersistentList()
36         for value in item:
37             newitem.append(persistent_aware(value))
38         return newitem
39     return item
40
41 class TrackerConfig(object):
42     """
43     This class implements the ITrackerConfig interface.
44     Its used to save and get some data from the annotations.
45     """
46     implements(ITrackerConfig)
47
48     def __init__(self, context):
49         """
50         Initializes the TrackerConfig class.
51         Sets the context and the annotations.
52         """
53         self.context = context
54         self.annotations = IAnnotations(self.context)
55
56     def get_all_times(self):
57         """
58         Gets the dictionary all_times from the annotations containing all measured
59         times.
60         @return:      PersistentDict all_times
```

Time Tracking Tool

```
59     """
60     return self.annotations.get('all_times', PersistentDict())
61
62 def get_times_by_date(self, day):
63     """
64     Gets all the measured times from a single day. It filters the all_times
65     dictionary by date.
66     Then it cleans the dictionary, means that empty tickets and projects will be
67     removed.
68     @return:    dictionary containing all times of a single day
69     """
70     def filterer(item):
71         val = datetime.datetime(*item['start']).timetuple()[3]
72         date = datetime.datetime(*day).timetuple()[3]
73         if val == date:
74             return True
75         return False
76     times = copy.deepcopy(self.get_all_times())
77     for project in times.values():
78         for ticket in project['items'].values():
79             stamps = ticket['items']
80             ticket['items'] = filter(filterer, stamps)
81     return self.cleanup_dict(times)
82
83 def cleanup_dict(self, dict_):
84     """
85     This function checks a dictionary whether it doesn't contains any empty tickets
86     or projects.
87     Empty projects or tickets will be removed.
88     @param dict_:    dictionary
89     @return:         cleaned dictionary
90     """
91     for pid, project in dict_.items():
92         for tid, ticket in project['items'].items():
93             if len(ticket['items'])==0:
94                 del project['items'][tid]
95             if len(project['items'])==0:
96                 del dict_[pid]
97     return dict_
98
99 def get_days_with_tickets(self):
100     """
101     Gets all days where are some measured tickets.
102     Checks all start dates of tickets and appends them to a list if
103     they aren't already inside there.
104     @return:    list with days containing tickets
105     """
106     result = [datetime.datetime.now().timetuple()[3]]
107     for pid, project in self.get_all_times().items():
108         for tid, ticket in project['items'].items():
109             for entry in ticket['items']:
110                 day = datetime.datetime(*entry['start']).timetuple()[3]
111                 if not day in result:
112                     result.append(day)
113     result.sort()
114     result.reverse()
115     return result
116
117 def set_all_times(self, value):
118     """
119     Function to set the dictionary all_times. It will be saved into the annotation.
120     @param value:    new value for all_times
```

Time Tracking Tool

```
118         """
119         self.annotations['all_times'] = persistent_aware(value)
120
121     def set_current_time(self, value):
122         """
123         Function to set the dictionary current_time. It will be saved into the
124         annotation.
125         @param value:    new value for current_time
126         """
127         self.annotations['current_time'] = persistent_aware(value)
128
129     def get_current_time(self):
130         """
131         Gets the dictionary current_time from the annotations containing the current
132         time measure.
133         @return:    PersistentDict current_time
134         """
135         return self.annotations.get('current_time', PersistentDict())
```

Listing 7: browser.py

```
1  #
2  # File:      browser.py
3  # Author:    Julian Infanger <julian.infanger@4teamwork.ch>
4  # Modified:  16.03.2010
5  #
6  # Copyright (c) 2010 by 4teamwork.ch
7
8  # imports
9  import datetime
10
11 from Acquisition import aq_inner, aq_parent
12 from DateTime import DateTime
13 from persistent.dict import PersistentDict
14
15 from Products.CMFCore.utils import getToolByName
16 from Products.CMFPlone.interfaces.siteroot import IPloneSiteRoot
17 from Products.Five import BrowserView
18
19 from zope.viewlet.interfaces import IViewlet
20 from zope.viewlet.interfaces import IViewletManager
21 from zope.component import getMultiAdapter
22
23 from ftw.timetracker import _
24 from ftw.timetracker.interfaces import ITrackerConfig
25
26
27 class MyBrowserView(BrowserView):
28     """
29     Class MyBrowserView
30     This is the super class which defines some helpful variables
31     used by the following BrowserViews
32     """
33     def __call__(self):
34         """
35         Defines the variables in the call function
36         self.config:      the ITrackerConfig interface (for the annotations)
37         self.all_times:    all times which have a start and a end time (self.
38                           config.get_all_times())
39         self.current_time: the ticket which is now running (self.config.
40                           get_current_time())
```

Time Tracking Tool

```
39         self.days_with_tickets: all days where are some tickets (self.config.
40             get_days_with_tickets())
41     @return:         super-method call or None
42     """
43     mt = getToolByName(self.context, 'portal_membership')
44     homefolder = mt.getHomeFolder()
45     if homefolder==None:
46         raise Exception("%s: %s" % (_(u"no homefolder"), mt.getAuthenticatedMember(
47             )))
48     self.config = ITrackerConfig(homefolder)
49     self.all_times = self.config.get_all_times()
50     self.current_time = self.config.get_current_time()
51     self.days_with_tickets = self.config.get_days_with_tickets()
52     super_call = getattr(super(MyBrowserView, self), '__call__', None)
53     return super_call and super_call() or None
54
55 class RefreshPanel(BrowserView):
56     """
57     This class is used for refreshing the panel.
58     Its used after changing the state of the stopwatch (Rec, Stop, Pause, Play)
59     """
60     def __call__(self):
61         """
62         The call function gets the ViewletManager and its viewlet ftw.timetracker.panel
63         .
64         After updating the it returns the render-Method of the ftw.timetracker.panel
65         viewlet.
66         @return:     renderer.update() the rendered viewlet ftw.timetracker.panel
67         """
68         manager = getMultiAdapter((self.context, self.request, self),
69             IViewletManager, name='plone.portaltop')
70         renderer = getMultiAdapter((self.context, self.request, self, manager),
71             IViewlet, name='ftw.timetracker.panel')
72         renderer = renderer.__of__(self.context)
73         renderer.update()
74         return renderer.render()
75
76 class Rec(MyBrowserView):
77     """
78     This class is used for recording an effort for a ticket.
79     It extends the MyBrowserView class.
80     """
81     def __call__(self):
82         """
83         This Funtion gets the following informations of a ticket:
84         project_uid:    projects uid, used for saving in the dictionary
85         project_title:  projects title, used in the ticket_overview
86         ticket_id:      tickets id, used in the display
87         ticket_uid:     tickets uid, used for saving in the dictionary
88         ticket_title:   tickets title, used in the ticket_overview
89         start:          start time, used for calculating the effort-time
90         And saves them into the current_time dictionary.
91         @return:        errormessage or nothing
92         """
93         super(Rec, self).__call__()
94         if self.context.portal_type!='PoiIssue':
95             return _(u'this is not a ticket')
96         elif self.current_time:
97             return _(u'there is already a running ticket')
98         else:
99             now = datetime.datetime.now()
100             effort = PersistentDict({
```


Time Tracking Tool

```
97         'project_uid' : self.get_project().UID(),
98         'project_title' : self.get_project().Title(),
99         'ticket_id' : self.context.id,
100         'ticket_uid' : self.context.UID(),
101         'ticket_title' : self.context.Title(),
102         'start' : datetime.datetime.timetuple(now)[:6]
103     })
104     self.config.set_current_time(effort)
105
106     def get_project(self):
107         """
108         This function gets the parent object until there is a project
109         or the SiteRoot is reached
110         @return: the project of the context-object
111         """
112         project_list = ['ftw_ProjectV1',]
113         project = aq_parent(aq_inner(self.context))
114         while project.portal_type not in project_list:
115             if IPloneSiteRoot.providedBy(project):
116                 raise Exception("no project found")
117             project = aq_parent(aq_inner(project))
118         return project
119
120     class Stopp(MyBrowserView):
121         """
122         The Stopp class is a BrowserView used for stopping a time measure ticket.
123         """
124         def __call__(self):
125             """
126             Gets the ticket from the current_time dictionary and sets the end point.
127             Then it saves the data into the dictionary all_times and cleans the
128             current_time dictionary.
129             @return: errormessage or nothhing
130             """
131             super(Stopp, self).__call__()
132             if not self.current_time:
133                 return _(u'there is not a running ticket')
134             elif not self.current_time.has_key('start'):
135                 self.config.set_current_time(None)
136             else:
137                 project_uid = self.current_time['project_uid']
138                 ticket_uid = self.current_time['ticket_uid']
139                 comment = self.context.REQUEST.get('comment', None)
140                 end = datetime.datetime.timetuple(datetime.datetime.now())[:6]
141                 # creates the project in dictionary if it doesn't exists yet
142                 if not self.all_times.has_key(project_uid):
143                     self.all_times[project_uid] = {
144                         'title' : self.current_time['project_title'],
145                         'items' : {}
146                     }
147                 # creates the ticket in dictionary if it doesn't exists yet
148                 if not self.all_times[project_uid]['items'].has_key(ticket_uid):
149                     self.all_times[project_uid]['items'][ticket_uid] = {
150                         'title' : self.current_time['ticket_title'],
151                         'items' : [],
152                     }
153                 # saves the data into the all_times dictionary
154                 self.all_times[project_uid]['items'][ticket_uid]['items'].append({
155                     'start' : self.current_time['start'],
156                     'end' : end,
157                     'comment' : comment != 'null' and comment or None,
158                 })
```

Time Tracking Tool

```
158         self.config.set_all_times(self.all_times)
159         self.config.set_current_time(None)
160
161     class Play(MyBrowserView):
162         """
163         This BrowserView is used for re-playing a paused ticket.
164         """
165         def __call__(self):
166             """
167             Checks if there is an entry in current_times dictionary and if it hasn't a
168             start time.
169             Then it sets a start time to the entry.
170             @return:    errormessage or nothhing
171             """
172             super(Play, self).__call__()
173             if not self.current_time:
174                 return _(u'there is not a paused ticket')
175             elif self.current_time.has_key('start'):
176                 return _(u'this is not a paused ticket')
177             else:
178                 now = datetime.datetime.now()
179                 self.current_time['start'] = datetime.datetime.timetuple(now)[:6]
180                 self.config.set_current_time(self.current_time)
181
182     class Pause(MyBrowserView):
183         """
184         With this class a time measure can be paused.
185         It's similar to the Stopp class but it doesn't cleans the current_times dictionary.
186         """
187         def __call__(self):
188             """
189             Gets the entry from the current_time dictionary and saves its data into the
190             all_times dictionary.
191             Then it removes the start time from the dictionary and.
192             @return:    errormessage or nothhing
193             """
194             super(Pause, self).__call__()
195             if not self.current_time:
196                 return _(u'there is not a running ticket')
197             else:
198                 project_uid = self.current_time['project_uid']
199                 ticket_uid = self.current_time['ticket_uid']
200                 comment = self.context.REQUEST.get('comment', None)
201                 end = datetime.datetime.timetuple(datetime.datetime.now())[:6]
202                 if not self.all_times.has_key(project_uid):
203                     self.all_times[project_uid] = {
204                         'title' : self.current_time['project_title'],
205                         'items' : {}
206                     }
207                 if not self.all_times[project_uid]['items'].has_key(ticket_uid):
208                     self.all_times[project_uid]['items'][ticket_uid] = {
209                         'title' : self.current_time['ticket_title'],
210                         'items' : [],
211                     }
212                 self.all_times[project_uid]['items'][ticket_uid]['items'].append({
213                     'start' : self.current_time['start'],
214                     'end' : end,
215                     'comment' : comment != 'null' and comment or None,
216                 })
217                 self.config.set_all_times(self.all_times)
218                 if self.current_time.has_key('start'):
219                     del self.current_time['start']
```

Time Tracking Tool

```
218         self.config.set_current_time(self.current_time)
219
220     class Transfer(MyBrowserView):
221         """
222         BrowserView for transferring the collected times into the timesheet.
223         """
224         def __call__(self):
225             """
226             Gets the data from the post-request. It checks the data, but normally there
227             shouldn't be some errors,
228             because it's already validated by JavaScript before calling this function.
229             Gets the timesheet of the ticket and creates a new timesheet_entry with
230             invokeFactory.
231             Then it removes the ticket from dictionary.
232             @return:    errormessage or True
233             """
234             super(Transfer, self).__call__()
235             request = self.context.REQUEST
236             catalog = getToolByName(self.context, 'portal_catalog')
237
238             uid = request.get('uid', None)
239             comment = request.get('comment', None)
240             time = request.get('time', None)
241             billable = request.get('billable', None)
242             date = request.get('date', None)
243
244             if not uid or not comment or not time or not billable or not date:
245                 # should never appear, checked with JavaScript
246                 return _(u"some arguments are missed")
247
248             # get the time, considering the different formates
249             time = self.convert_time(time)
250             billable = self.convert_time(billable)
251
252             tickets = catalog({'UID' : uid})
253             if len(tickets) == 0:
254                 return _(u'could not find ticket')
255             elif len(tickets) > 1:
256                 return _(u'multiple tickets found')
257
258             obj = tickets[0].getObject()
259             timesheet = obj.getTimesheet()
260
261             if not timesheet:
262                 return _(u'no timesheet found') + ':\n' + comment
263
264             date_parts = date.split("-")
265             args={'title' : comment,
266                  'hours' : time,
267                  'billable' : billable,
268                  'startDate' : date_parts[0] + '/' + date_parts[1] + '/' + date_parts[2]
269             }
270             now = DateTime()
271             id_ = 'ftw_TimesheetEntry' + '.' + now.strftime('%Y-%m-%d') + '.' + now.
272                 strftime('%M%S')
273             ts = timesheet.invokeFactory(type_name='ftw_TimesheetEntry', id=id_, **args)
274
275             if not ts:
276                 return _(u'could not create rapport')
277
278             date = datetime.datetime(*[int(a) for a in date_parts]).timetuple()[0:3]
279             dict_ = self.all_times
```

Time Tracking Tool

```
277     for pid, project in dict_.items():
278         for tid, ticket in project.get('items', {}).items():
279             if tid == uid:
280                 for entry in ticket.get('items', []):
281                     entry_date = datetime.datetime(*entry['start']).timetuple()[3]
282                     if date == entry_date:
283                         ticket['items'].remove(entry)
284 dict_ = self.config.cleanup_dict(dict_)
285 self.config.set_all_times(dict_)
286 return True
287
288 def convert_time(self, time):
289     """
290     Converts the time into its real value.
291     If there is an : into the time it splits the time into hours and minutes.
292     Then it gets the decimal value of the minutes and return its sum.
293     @param time:    time from the request (string)
294     @return:        formatted time as decimal value (float)
295     """
296     if '.' in time:
297         return float(time)
298     else:
299         time_parts = time.split(':')
300         hours = float(time_parts[0])
301         minutes = float(time_parts[1])
302         minutes = minutes / 60
303         return hours + minutes
304
305 class DeleteTicket(MyBrowserView):
306     """
307     This BrowserView is used to delete a list of Tickets from the ticket_overview.
308     """
309     def __call__(self):
310         """
311         First it gets the list of tickets (uids) which should be deleted.
312         Then it iterates all tickets and if the ticket-uid and the date matches,
313         it removes the ticket from the dictionary.
314         After this it will clean up the dictionary.
315         @return:    nothing
316         """
317         super(DeleteTicket, self).__call__()
318         request = self.context.REQUEST
319         ticket_uids = request.get('marked')
320         date = request.get('date', None)
321         if not date:
322             return
323         date_parts = [int(d) for d in date.split("-")]
324         date = datetime.datetime(*date_parts).timetuple()[3]
325         dict_ = self.all_times
326         for pid, project in dict_.items():
327             for tid, ticket in project.get('items', {}).items():
328                 if tid in ticket_uids:
329                     for entry in ticket.get('items', []):
330                         entry_date = datetime.datetime(*entry['start']).timetuple()[3]
331                         if date == entry_date:
332                             ticket['items'].remove(entry)
333         dict_ = self.config.cleanup_dict(dict_)
334         self.config.set_all_times(dict_)
```

Time Tracking Tool

Listing 8: configure.zcml

```
1 <configure
2   xmlns="http://namespaces.zope.org/zope"
3   xmlns:browser="http://namespaces.zope.org/browser"
4   i18n_domain="ftw.timetracker">
5
6   <browser:resourceDirectory
7     name="ftw.timetracker.images"
8     directory="images"
9   />
10
11  <browser:resourceDirectory
12    name="ftw.timetracker.javascript"
13    directory="javascript"
14  />
15
16  <browser:resourceDirectory
17    name="ftw.timetracker.stylesheet"
18    directory="stylesheet"
19  />
20
21  <browser:page
22    for="*"
23    name="list_times"
24    class=".list_times.ListTimes"
25    template="templates/list_times.pt"
26    permission="zope2.View"
27  />
28
29  <browser:page
30    for="*"
31    name="refresh_panel"
32    class=".browser.RefreshPanel"
33    permission="zope2.View"
34  />
35
36  <browser:page
37    for="*"
38    name="rec"
39    class=".browser.Rec"
40    permission="zope2.View"
41  />
42
43  <browser:page
44    for="*"
45    name="stopp"
46    class=".browser.Stopp"
47    permission="zope2.View"
48  />
49
50  <browser:page
51    for="*"
52    name="play"
53    class=".browser.Play"
54    permission="zope2.View"
55  />
56
57  <browser:page
58    for="*"
59    name="pause"
60    class=".browser.Pause"
```

Time Tracking Tool

```
61     permission="zope2.View"
62     />
63
64     <browser:page
65         for="*"
66         name="transfer"
67         class=".browser.Transfer"
68         permission="zope2.View"
69     />
70
71     <browser:page
72         for="*"
73         name="delete_ticket"
74         class=".browser.DeleteTicket"
75         permission="zope2.View"
76     />
77
78     <browser:viewlet
79         name="ftw.timetracker.panel"
80         class=".viewlets.PanelViewlet"
81         manager="plone.app.layout.viewlets.interfaces.IPortalTop"
82         template="templates/panel.pt"
83         insert-after="plone.searchbox"
84         permission="zope2.View"
85     />
86 </configure>
```

Listing 9: list_times.py

```
1  #
2  # File:      list_times.py
3  # Author:    Julian Infanger <julian.infanger@4teamwork.ch>
4  # Modified:  16.03.2010
5  #
6  # Copyright (c) 2010 by 4teamwork.ch
7
8  # imports
9  import datetime
10 from ftw.timetracker.browser.browser import MyBrowserView
11 from plone.registry.interfaces import IRegistry
12 from ftw.timetracker.interfaces import ITrackerSettings
13 from zope.component import queryUtility
14
15
16 class ListTimes(MyBrowserView):
17     """
18     This BrowserView is used to display the ticket overview.
19     It can display a list of collected times to tickets and projects.
20     """
21     def get_ticket_time(self, time_list):
22         """
23         Gets the total time of a ticket. For rounding the time it gets the registry
24         and the tracker_settings. Then it can get the property round_step.
25         @param time_list:  list of times
26         @return:           rounded sum of all the times
27         """
28         registry = queryUtility(IRegistry)
29         settings = registry.forInterface(ITrackerSettings)
30         total = 0;
31         for time in time_list:
```

Time Tracking Tool

```
32         total = total + (self.get_datetime(time['end']) - self.get_datetime(time['
33             start'])).seconds
34     minutes = total / 60
35     round_step = getattr(settings, 'round_step', 15)
36     res = int(minutes / round_step) + 1
37     return float(res) * float(round_step) / 60
38
39 def get_datetime_time(self, time):
40     """
41     Converts a timetuple into the time (hh:mm).
42     @param time:      timetuple (2010, 1, 1, 12, 30)
43     @return:          formatted time (12:30)
44     """
45     parts = datetime.datetime(*time).timetuple()
46     return "%s:%s" % (self.format(parts[3]), self.format(parts[4]))
47
48 def get_datetime_date(self, time):
49     """
50     Converts a timetuple into a date (01.01.2010).
51     @param time:      timetuple(2010, 1, 1, 12, 30)
52     @return:          formatted date (01.01.2010)
53     """
54     parts = datetime.datetime(*time).timetuple()
55     return "%s.%s.%s" % (self.format(parts[2]), self.format(parts[1]), self.format(
56         parts[0]))
57
58 def get_project_time(self, time_list):
59     """
60     Gets the time of a project.
61     @param time_list:  all ticket times
62     @return:           rounded sum of all ticket times
63     """
64     total = 0
65     for ticket in time_list.values():
66         ticket_timelist = ticket['items']
67         total += self.get_ticket_time(ticket_timelist)
68     return round(total, 2)
69
70 def get_comments(self, comment_list):
71     """
72     This function gets all the comments for a ticket and
73     separates them with comma if there is a comment.
74     @param comment_list:  list of comments
75     @return:              comments, separated by comma
76     """
77     return ', '.join([a['comment'] for a in comment_list if a['comment'] not in ['',
78         None]])
79
80 def get_datetime_delta(self, start, end):
81     """
82     Converts two timetuples into datetime objects and returns
83     the difference between them.
84     @param start:      start time (timetuple)
85     @param end:        end time (timetuple)
86     @return:           timedelta (end-start)
87     """
88     delta = datetime.datetime(*end) - datetime.datetime(*start)
89     return float(delta.seconds) / 60 / 60
90
91 def get_datetime(self, date_list):
92     """
93     Converts a timetuple into a datetime object.
```

Time Tracking Tool

```
91     @param date_list:    timetuple (2010, 1, 1, 12, 30)
92     @return:            datetime object
93     """
94     return datetime.datetime(*date_list)
95
96     def get_iso_date(self, date_list):
97         """
98         Converts a timetuple into a iso-date string.
99         @param date_list:    timetuple (2010, 1, 1, 12, 30)
100        @return:            formatted iso-date (2010-01-01)
101        """
102        date = datetime.datetime(*date_list).timetuple()[0:3]
103        return "%s-%s-%s" % (date[0], date[1], date[2])
104
105        def get_times_by_date(self):
106            """
107            This function gets all the measured times for a single day by
108            calling the get_times_by_date method of the tracer_config.
109            If there isn't a date in the request it creates a new data (today).
110            @return:        dictionary get_times_by_date of the tracer_config
111            """
112            today = datetime.datetime.now().timetuple()[0:3]
113            date = self.context.REQUEST.get('date', False)
114            if date:
115                parts = date.split("-")
116                return self.config.get_times_by_date([int(d) for d in parts])
117            return self.config.get_times_by_date(today)
118
119            def compare_to_request(self, date):
120                """
121                Compares the date in request with the date in parameter.
122                If it's the same then it returns selected, means the option in select is
123                selected.
124                @param date:    iso date (2010-01-01)
125                @return:        string 'selected' if its the same date, otherwise None
126                """
127                if date == self.context.REQUEST.get('date', False):
128                    return 'selected'
129                return None
130
131            def format(self, i):
132                """
133                This function gets number and puts a 0 in front,
134                if the number is smaller than 10 and returns it as string.
135                @param i:    number (int)
136                @return:    number or number as string with 0 in front if smaller than 10
137                """
138                if i < 10:
139                    return "0%s" % i
140                return i
```

Listing 10: viewlets.py

```
1  #
2  # File:      viewlets.py
3  # Author:    Julian Infanger <julian.infanger@4teamwork.ch>
4  # Modified:  16.03.2010
5  #
6  # Copyright (c) 2010 by 4teamwork.ch
7
8  # imports
```


Time Tracking Tool

```
9 import datetime
10 from plone.app.layout.viewlets import ViewletBase
11 from Products.CMFCore.utils import getToolByName
12 from ftw.timetracker.interfaces import ITrackerConfig
13
14 class PanelViewlet(ViewletBase):
15     """
16     This is the class for the panel viewlet.
17     It shows the panel to handle the Time Tracking Tool.
18     """
19     def get_home_folder(self):
20         """
21         Gets the authenticated users home folder
22         @return: users home folder
23         """
24         return getToolByName(self.context, 'portal_membership').getHomeFolder()
25
26     def get_tracker_config(self):
27         """
28         Gets the tracker configuration for the annotations
29         @return: tracker configuration
30         """
31         return ITrackerConfig(self.get_home_folder())
32
33     def get_datetime_time(self, time):
34         """
35         Converts a timetuple into a datetime object.
36         @param time: timetuple (2010, 1, 1)
37         @return: converted datetime object
38         """
39         return datetime.datetime(*time).time()
```

Listing 11: timetracker.js

```
1  /* document is ready */
2  jq(function() {
3      // vor & zurueck (Browser)
4      refresh_panel();
5      // list_times aufklappen
6      jq("#button_list_times").live("click", function(e) {
7          list_times(false);
8          jq("#list_times").slideToggle('fast');
9          e.preventDefault();
10     });
11
12     // rec button
13     jq("#button_rec").live("click", function(e) {
14         jq.ajax({type : 'POST',
15             url : './rec',
16             success : function(data, textStatus, XMLHttpRequest) {
17                 if (textStatus == 'success') {
18                     refresh_panel();
19                 }
20                 if (data.length > 0) {
21                     myalert(data);
22                 }
23             }
24         });
25         e.preventDefault();
26     });
27 }
```

Time Tracking Tool

```
28 // stopp button
29 jq("#button_stopp").live("click", function(e){
30     var data = {};
31     if (jq("#button_play").length == 0) {
32         data['comment'] = window.prompt('Geleistete Arbeiten an diesem Ticket?', '');
33     }
34     jq.ajax({type : 'POST',
35         url : './stopp',
36         data : data,
37         success : function(data, textStatus, XMLHttpRequest) {
38             if (textStatus == 'success') {
39                 refresh_panel();
40             }
41             if (data.length > 0) {
42                 myalert(data);
43             }
44         }
45     });
46     e.preventDefault();
47 });
48
49 // pause button
50 jq("#button_pause").live("click", function(e){
51     jq.ajax({type : 'POST',
52         url : './pause',
53         data : {
54             comment : window.prompt('Geleistete Arbeiten an diesem Ticket?', '')
55         },
56         success : function(data, textStatus, XMLHttpRequest) {
57             if (textStatus == 'success') {
58                 refresh_panel();
59             }
60             if (data.length > 0) {
61                 myalert(data);
62             }
63         }
64     });
65     e.preventDefault();
66 });
67
68 // play button
69 jq("#button_play").live("click", function(e){
70     jq.ajax({type : 'POST',
71         url : './play',
72         success : function(data, textStatus, XMLHttpRequest) {
73             if (textStatus == 'success') {
74                 refresh_panel();
75             }
76             if (data.length > 0) {
77                 myalert(data);
78             }
79         }
80     });
81     e.preventDefault();
82 });
83
84 /*
85 Function to remove the checked tickets from list
86 calls the view delete_ticket with parameter irl and data.
87 */
88 function remove_checked() {
```

Time Tracking Tool

```
89     var uids = new Array();
90     var checked = jq('.check:checked');
91     checked.each(function() {
92         uids.push(this.getAttribute('value'));
93     });
94     jq.ajax({type : 'POST',
95             url : './delete_ticket',
96             data : {
97                 'marked:list' : uids,
98                 'date' : get_selected_date()
99             },
100     success : function(data, textStatus, XMLHttpRequest) {
101         if (textStatus == 'success') {
102             checked.each(function() {
103                 jq('#list_times tr.'+this.getAttribute('value')).remove();
104                 checked.parents('tr').remove();
105                 cleanup_projects();
106             });
107         }
108     });
109 }
110
111 /*
112 This function removes all tr.projects with no tickets in it.
113 */
114 function cleanup_projects() {
115     var projects = jq("#list_times tr.tproject");
116     projects.each(function(i){
117         if (jq("#list_times tr."+this.id).length == 0) {
118             jq(this).remove();
119         }
120     });
121 }
122
123 /*
124 This function refreshs the panel by reloading the field with the id panel.
125 */
126 function refresh_panel() {
127     jq.ajax({url : './refresh_panel", success : function(data, textStatus,
128             XMLHttpRequest){
129         if (textStatus == 'success') {
130             jq("#panel").html(jq(data)[0].childNodes);
131         }
132     });
133 }
134
135 /*
136 Lists all the times. This function contains some click events for buttons.
137 */
138 function list_times(date) {
139     var param = '';
140     if (date != false) {
141         param = '?date=' + date;
142     }
143     jq('#list_times').load('./list_times'+param, function(){
144         // markierte loeschen
145         jq("#delete_ticket").click(function() {
146             if (jq('.check:checked').val() != undefined) {
147                 if (confirm("Den Eintrag wirklich loeschen?")) {
148                     remove_checked();
149                 }
150             }
151         });
152     });
153 }
```

Time Tracking Tool

```
150 // einzelne uebertragen
151 jq("a.transfer_single").click(function(e) {
152     e.preventDefault();
153     var parent = jq(this).parents("tr");
154     var title = parent.find(".ticket_title").html() + ": ";
155     var comment = parent.find(".tcomment").attr('value');
156     var time = check_time_format(parent.find(".time").attr('value'));
157     var billable = check_time_format(parent.find(".billable").attr('value')
158     );
159     var uid = parent.find('input.check').attr('value');
160     if (!comment) {
161         parent.find(".tcomment").css('background', '#FF0000');
162         myalert("Kommentar fehlt");
163     }
164     else if (!time) {
165         parent.find(".time").css('background', '#FF0000');
166         myalert("Die geleistete Zeit fehlt, oder hat ein falsches Format.");
167     }
168     else if (!billable) {
169         parent.find(".billable").css('background', '#FF0000');
170         myalert("Die verrechenbare Zeit fehlt, oder hat ein falsches Format.");
171     }
172     else {
173         jq.ajax({type : 'POST',
174             url : './transfer',
175             data : {
176                 'uid' : uid,
177                 'comment' : title + comment,
178                 'time' : time,
179                 'billable' : billable,
180                 'date' : get_selected_date()
181             },
182             success : function(data, textStatus, XMLHttpRequest) {
183                 if (data == 'True') {
184                     parent.remove();
185                     cleanup_projects();
186                 }
187                 else {
188                     myalert(data);
189                 }
190             }
191         });
192         console.info("validation bestanden :D");
193     }
194 });
195 // markierte uebertragen
196 jq("#transfer").click(function() {
197     var checked = jq('.check:checked');
198     if (checked.val() != undefined) {
199         checked.each(function() {
200             jq('#'+this.getAttribute('value')).parents('tr').find('.transfer_single').click();
201         });
202     }
203 });
204 // Fenster schliessen
205 jq("#close_list_times").click(function(e) {
206     jq('#list_times').slideToggle('fast');
207     e.preventDefault();
208 });
```

Time Tracking Tool

```
208     });
209     // toggle tickets
210     jq("img.toggle").click(function(){
211         jq("."+(jq(this).attr('id'))).toggle();
212         if (jq(this).attr("src") == 'treeCollapsed.gif') {
213             jq(this).attr("src", 'treeExpanded.gif');
214         }
215         else {
216             jq(this).attr("src", 'treeCollapsed.gif');
217         }
218     });
219     // checkboxen
220     jq(".check_all").click(function(){
221         if (jq(this).attr("checked")) {
222             jq(".check").attr("checked", "checked");
223         }
224         else {
225             jq(".check").attr("checked", "");
226         }
227     });
228     // Markierungen aufheben
229     jq("tr input.tcomment, tr input.time, tr input.billable").click(function()
230     {
231         jq(this).css('background', '');
232     });
233     // select
234     jq('#select_date').change(function() {
235         list_times(get_selected_date());
236     });
237     jq('.tcomment:first').focus();
238     jq('#times_title').html(convert_iso_date(get_selected_date()));
239 });
240
241
242 /*
243 Converts a iso-date (2010-01-01) into a german date (01.01.2010)
244 return date
245 */
246 function convert_iso_date(date) {
247     parts = date.split("-");
248     for (var i=0;i<parts.length;i++) {
249         if (parts[i]<10) {
250             parts[i] = '0'+parts[i];
251         }
252     }
253     return parts[2]+"."+parts[1]+"."+parts[0];
254 }
255
256 /*
257 Gets the selected date in option.
258 return: selected date
259 */
260 function get_selected_date() {
261     date = jq('#select_date').attr('value');
262     if (!date || date == undefined) {
263         ct = new Date();
264         var month = ct.getMonth();
265         month++;
266         date = ct.getFullYear()+"-"+month+"-"+ct.getDate();
267     }
268     return date;
```

Time Tracking Tool

```
269     }
270
271     /*
272     Checks the date format
273     allowed: hh:mm or hh.hh,
274     return: false if not allowed else the time
275     */
276     function check_time_format(time) {
277         time = time.replace(/\s*/g, '');
278         if (!time) {
279             return false;
280         }
281         else if (/^\d{1,2}(:|:[0-5][0-9]{0,1})?\.?\d+$/ .test(time) == false) {
282             return false;
283         }
284         return time;
285     }
286
287     /*
288     Function showing a errormessage.
289     @param msg: error message
290     */
291     function myalert(msg) {
292         var err_msg = jq("#error_messge");
293         err_msg.html(msg);
294         err_msg.show();
295         setTimeout('jq("#error_messge").hide();', 10000);
296     }
297 });
```

Listing 12: timetracker.css

```
1 .passed_times_info, .passed_times_info a {
2     background-color: #FF0000;
3     padding: 5px;
4     color: #fff;
5     font-weight: bold;
6 }
7
8 table.times {
9     width: 80%;
10 }
11
12 table.times td.tsmall {
13     width: 20px;
14     text-align: center;
15 }
16
17 table.times td.tttitle {
18     width: 250px;
19 }
20
21 table.times .ttime {
22     width: 30px;
23     text-align: center;
24 }
25
26 table.times tr.thead {
27     background-color: #FFCC33;
28 }
29
```

Time Tracking Tool

```
30 table.times tr.tproject, table.times tr.tproject a {
31     background-color: #002F55;
32     color: #fff;
33 }
34
35 table.times tr.tticket {
36     background-color: #EDF0F3;
37 }
38
39 table.times img.toggle {
40     cursor: pointer;
41 }
42
43 table.times input.tcomment {
44     width: 100%;
45 }
46
47 table.times td.tsmall img {
48     border: none;
49     cursor: pointer;
50 }
51
52 table.times tr.hidden {
53     display: none;
54 }
55
56 table.times td.tbutton {
57     width: 130px;
58 }
59
60 table.times td.tbutton input {
61     width: 100%;
62 }
63
64 table.times td.tbutton #delete_ticket {
65     background-color: #F6CECE;
66 }
67
68 #close_list_times {
69     display: block;
70     text-decoration: none;
71     border-top: 1px solid #ddd;
72 }
73
74 #error_messge {
75     display: block;
76     color: red;
77     text-align: center;
78 }
```

Listing 13: list_times.pt

```
1 <center i18n:domain="ftw.timetracker" tal:define="day_times python:view.
   get_times_by_date()">
2 <!-- If there are some tickets with start-date in past -->
3 <p tal:condition="python: len(view.days_with_tickets) > 1" class="passed_times_info
   ">
4     Sie haben gestoppte Tickets, welche Sie noch nicht verrechnet haben in der
       Vergangenheit.<br />
5     <select style="text-align:center;" id="select_date">
6         <tal:rep tal:repeat="day view/days_with_tickets">
```

Time Tracking Tool

```
7         <option tal:define="selected python: view.compare_to_request(view.  
            get_iso_date(day)) "   
8             tal:attributes="value python: view.get_iso_date(day); selected  
                selected"  
9             tal:content="python: view.get_datetime_date(day) " />  
10     </tal:rep>  
11 </select>  
12 </p>  
13 <h1 id="times_title"/>  
14 <!-- thead -->  
15 <table class="listing times" align="center" tal:condition="day_times">  
16     <tr class="thead">  
17         <td class="tsmall">&nbsp;</td>  
18         <td class="tsmall"><input type="checkbox" class="check_all"/></td>  
19         <td class="ttitle">Titel</td>  
20         <td>Kommentar</td>  
21         <td class="ttime">Zeit</td>  
22         <td class="ttime">verr.</td>  
23         <td class="tsmall">&nbsp;</td>  
24     </tr>  
25 <!-- projects -->  
26 <tal:block repeat="project python:day_times.keys()" tal:define="td day_times">  
27     <tr class="tproject" tal:attributes="id project">  
28         <td colspan="4">  
29             <a tal:attributes="href string:resolveuid/${project}">  
30                 <span tal:content="python: td[project]['title']"/>  
31             </a>  
32         </td>  
33         <td class="ttime" tal:content="python: view.get_project_time(td[project]['  
            items'])"/>  
34         <td class="ttime" tal:content="python: view.get_project_time(td[project]['  
            items'])"/>  
35         <td>&nbsp;</td>  
36     </tr>  
37 <!-- tickets -->  
38 <tal:rep repeat="tickets python:td[project]['items'].keys()">  
39     <tr tal:attributes="class string:tticket ${project}">  
40         <td class="tsmall">  
41               
42         </td>  
43         <td class="tsmall">  
44             <input type="checkbox" class="check" name="marked:list" tal:  
                attributes="value tickets" /></td>  
45         <td>  
46             <a tal:attributes="href string:resolveuid/${tickets}">  
47                 <span class="ticket_title" tal:content="python:td[project  
                    ]['items'][tickets]['title']" />  
48             </a>  
49         </td>  
50         <td>  
51             <input type="text" class="tcomment" tal:attributes="value  
                python: view.get_comments(td[project]['items'][tickets]['  
                    items'])" />  
52         </td>  
53         <td class="ttime">  
54             <input type="text" class="ttime time" tal:attributes="value  
                python:view.get_ticket_time(td[project]['items'][tickets  
                    ]['items'])" />  
55         </td>  
56         <td class="ttime">
```


Time Tracking Tool

```
57         <input type="text" class="ttime billable" tal:attributes="value
58             python:view.get_ticket_time(td[project]['items'][tickets
59             ]['items'])" />
60         </td>
61         <td class="tsmall">
62             <a href="#" class="transfer_single">
63                 
64             </a>
65         </td>
66     </tr>
67     <!-- efforts -->
68     <tr tal:repeat="item python:td[project]['items'][tickets]['items'" tal
69         :attributes="class string: hidden ${tickets} ${project}">
70         <td class="tsmall">&nbsp;</td>
71         <td class="tsmall">&nbsp;</td>
72         <td tal:content="python:view.get_datetime_time(item['start'])+' -
73             '+view.get_datetime_time(item['end'])"/>
74         <td tal:content="python:item['comment'] or '-'"/>
75         <td class="ttime" tal:content="python:round(view.get_datetime_delta
76             (item['start'],item['end']), 2)"/>
77         <td class="tsmall">&nbsp;</td>
78     </tr>
79 </tal:rep>
80 </tal:block>
81 </table>
82 <br />
83 <table align="center" class="times" tal:condition="day_times">
84     <tr>
85         <td class="tbutton">
86             <input type="button" class="standalone" id="transfer" value="markierte
87             uebertragen" />
88         </td>
89         <td class="tbutton">
90             <input type="button" class="standalone" id="delete_ticket" value="
91             markierte loeschen" />
92         </td>
93         <td>&nbsp;</td>
94     </tr>
95 </table>
96 <!-- no tickets -->
97 <span tal:condition="not: day_times" i18n:translate="">there aren't any stopped
98     tickets</span>
99 <br /><br />
100 <div id="error_messge" />
101 <a href="#" id="close_list_times" i18n:translate="">close</a>
102 </center>
```

Listing 14: panel.pt

```
1 <tal:block condition="view/get_home_folder">
2     <tal:define define="homefolder view/get_home_folder;config view/get_tracker_config;
3     current config/get_current_time">
4         <div style="right:50px;position:absolute;z-index:2;top: 133px; background: #fff
5         ;" id="panel" tal:condition="python:here.portal_membership.
6         getAuthenticatedMember().id!='acl_users'">
7             <table>
8                 <tr>
9                     <td>
10                         <a href="#" id="button_list_times" style="margin:2px;">
11                             
12                         </a>
```

Time Tracking Tool

```
10         </td>
11     <td>
12         <a href="#" style="margin:2px;" id="button_stopp" tal:condition
13             ="current">
14             
15         </a>
16         
18     </td>
19     <td style="margin:2px;">
20         <a href="#" id="button_rec" tal:condition="python:context.
21             portal_type=='PoiIssue' and not current">
22             
23         </a>
24         
27     </td>
28 </tr>
29 <tr>
30     <td colspan="5" style="text-align:center;border:1px solid;">
31         <tal:display condition="python: current and current.has_key('
32             start')">
33             <span tal:content="python:'#s: %s'%(current['ticket_id'],
34                 view.get_datetime_time(current['start']))" tal:
35                 attributes="title python:current['ticket_title']" id="
36                 display"/>
37         </tal:display>
38         <tal:display condition="python: current and not current.has_key
39             ('start')">
40             <span tal:content="python:'#s: pausiert' % current['
41                 ticket_id']" tal:attributes="title python:current['
42                 ticket_title']" id="display"/>
43         </tal:display>
44         <tal:display condition="not: current">
45             nichts zu tun?
46         </tal:display>
47     </td>
48 </tr>
49 </table>
50 </div>
51 <div style="display:none;border:3px solid;background:#fff;position:absolute;z-
52     index:999;top:10px;left:5%;width:90%;" id="list_times">
53 </div>
54 </tal:define>
55 </tal:block>
```

Listing 15: cssregistry.xml

Time Tracking Tool

```
1 <?xml version="1.0"?>
2 <object name="portal_css">
3
4   <stylesheet title=""
5     id="++resource++ftw.timetracker.stylesheet/timetracker.css"
6     media="screen" rel="stylesheet" rendering="import"
7     cacheable="True" compression="safe" cookable="True"
8     enabled="1" expression=""/>
9
10 </object>
```

Time Tracking Tool

Listing 16: jsregistry.xml

```
1 <?xml version="1.0"?>
2 <object name="portal_javascripts" meta_type="Javascripts Registry">
3   <javascript cacheable="True" compression="safe" cookable="True"
4     enabled="1" expression="" id="++resource++ftw.timetracker.javascript/timetracker.js
      " />
5 </object>
```

Listing 17: metadata.xml

```
1 <metadata>
2   <version>1</version>
3   <dependencies>
4     <dependency>profile-plone.app.registry:default</dependency>
5   </dependencies>
6 </metadata>
```

Listing 18: registry.xml

```
1 <registry>
2   <records interface="ftw.timetracker.interfaces.ITrackerSettings" />
3 </registry>
```

Listing 19: viewlets.xml

```
1 <?xml version="1.0"?>
2 <object>
3   <order manager="plone.portaltop">
4     <viewlet name="ftw.timetracker.panel" insert-after="plone.path_bar"/>
5   </order>
6 </object>
```

Listing 20: ftw.timetracker.po

```
1 msgid ""
2 msgstr ""
3 "Project-Id-Version: PACKAGE VERSION\n"
4 "POT-Creation-Date: 2010-03-16 10:01+0000\n"
5 "PO-Revision-Date: 2010-03-16 10:01+0000\n"
6 "Last-Translator: Infanger Julian <julian.infanger@4teamwork.ch>\n"
7 "Language-Team: de <LL@li.org>\n"
8 "MIME-Version: 1.0\n"
9 "Content-Type: text/plain; charset=utf-8\n"
10 "Content-Transfer-Encoding: 8bit\n"
11 "Plural-Forms: nplurals=1; plural=0\n"
12 "Preferred-Encodings: utf-8 latin1\n"
13 "Domain: ftw.timetracker\n"
14
15 #: ftw.timetracker/ftw/timetracker/browser/templates/list_times.pt:93
16 msgid "close"
17 msgstr "schliessen"
18
19 #: ftw.timetracker/ftw/timetracker/browser/browser.py:273
20 msgid "could not create rapport"
21 msgstr "Rapport konnte nicht erstellt werden"
```

Time Tracking Tool

```
22
23 #: ftw.timetracker/ftw/timetracker/browser/browser.py:252
24 msgid "could not find ticket"
25 msgstr "Ticket konnte nicht gefunden werden"
26
27 #: ftw.timetracker/ftw/timetracker/browser/browser.py:254
28 msgid "multiple tickets found"
29 msgstr "mehrere Tickets gefunden"
30
31 #: ftw.timetracker/ftw/timetracker/browser/browser.py:45
32 msgid "no homefolder"
33 msgstr "Kein Benutzerverzeichnis gefunden"
34
35 #: ftw.timetracker/ftw/timetracker/browser/browser.py:260
36 msgid "no timesheet found"
37 msgstr "Es wurde kein verknuepfter Arbeitsrapport gefunden"
38
39 #: ftw.timetracker/ftw/timetracker/browser/browser.py:244
40 msgid "some arguments are missed"
41 msgstr "Nicht alle Argumente gefunden"
42
43 #: ftw.timetracker/ftw/timetracker/browser/templates/list_times.pt:90
44 msgid "there aren't any stopped tickets"
45 msgstr "keine gestoppten Tickets gefunden"
46
47 #: ftw.timetracker/ftw/timetracker/browser/browser.py:93
48 msgid "there is already a running ticket"
49 msgstr "Es wird bereits ein Ticket gestoppt"
50
51 #: ftw.timetracker/ftw/timetracker/browser/browser.py:173
52 msgid "there is not a paused ticket"
53 msgstr "kein pausiertes Ticket"
54
55 #: ftw.timetracker/ftw/timetracker/browser/browser.py:132
56 msgid "there is not a running ticket"
57 msgstr "kein laufendes Ticket"
58
59 #: ftw.timetracker/ftw/timetracker/browser/browser.py:175
60 msgid "this is not a paused ticket"
61 msgstr "dieses Ticket ist nicht pausiert"
62
63 #: ftw.timetracker/ftw/timetracker/browser/browser.py:91
64 msgid "this is not a ticket"
65 msgstr "das ist kein Ticket"
```