|  |
| --- |
|  |
| Bachelorarbeit |
| Elektronische Urlaubsverwaltung in einer mittelständischen IT-Firma – „Make or Buy“ ? |
|  |
|  |
| Vorgelegt im: Wintersemester 2014 / 15  Vorgelegt von: Christine Zoglmeier |

Erstkorrektor: Prof. Dr. Gerd Beneken

Zweitkorrektor: Prof. Dr. Reiner Hüttl

**ERKLÄRUNG**

Ich versichere, dass ich diese Arbeit selbständig angefertigt, nicht anderweitig für Prüfungszwecke vorgelegt, keine anderen als die angegebenen Quellen oder Hilfsmittel benützt sowie wörtliche und sinngemäße Zitate als solche gekennzeichnet habe.

Rosenheim den 05.03.2015

Christine Zoglmeier

**Kurzfassung**

Am Beispiel der Einrichtung eines neuen elektronischen Urlaubsverwaltungssystems für eine mittelständische IT-Firma wird aufgezeigt, wie ein umfassender und strategisch durchgeführter „Make-or-Buy?“- Entscheidungsprozess aussehen kann.

Nach einer eingehenden Analyse der Ausgangssituation sowie firmenspezifischer Anforderungen und Ziele wurde eine zielgerichtete Marktrecherche durchgeführt und eine Aufwandsschätzung für eine mögliche Eigenimplementierung erhoben. Dem folgte eine systematische Abwägung der Vor- und Nachteile einer eigenen Implementierung bzw. der Inanspruchnahme der Dienstleistung eines Drittanbieters. In Folge dessen konnte eine fundierte Entscheidung für die Nutzung eines geeigneten kommerziellen Systems gefällt werden.

Ergänzend zu der getroffenen „Buy“-Entscheidung wurde die Urlaubsverwaltung jedoch um eigene Zusatzfunktionen ergänzt. Es wurde eine Anbindung an das bestehende Zeiterfassungssystem hergestellt, eine automatische, unabhängige Datensicherung implementiert und ein Berichtsversand an die Personalbuchhaltung eingerichtet. Dadurch konnten mit überschaubarem Mehraufwand die wichtigsten Nachteile, die mit der Entscheidung für einen Drittanbieter verbunden waren, entkräftet werden.

Am Ende des Prozesses stand der Firma mit vergleichsweise geringem finanziellem und personellen Aufwand ein den Anforderungen entsprechendes, funktionelles und mit der Systemlandschaft vernetztes Urlaubsverwaltungssystem zur Verfügung.

Schlagworte: „Make-or-Buy“, Eigenfertigung oder Fremdbezug, Urlaubsverwaltung

**Inhalt**

[1 Einleitung 3](#_Toc413184499)

[2 Ausgangssituation 4](#_Toc413184500)

[2.1 Firmenentwicklung 4](#_Toc413184501)

[2.2 Systemumgebung 5](#_Toc413184502)

[2.2.1 Urlaubsverwaltung 6](#_Toc413184503)

[2.2.2 Zeiterfassung 7](#_Toc413184504)

[2.2.3 Personalbuchhaltung 8](#_Toc413184505)

[2.2.4 Systemlandkarte 8](#_Toc413184506)

[2.3 Bedarfsanalyse und Zielsetzung 9](#_Toc413184507)

[2.3.1 Anforderungen 9](#_Toc413184508)

[2.3.2 Gewünschte Systemumgebung 10](#_Toc413184509)

[3 „Make-or-Buy“ - Entscheidungsprozess 11](#_Toc413184510)

[3.1 Aufwandsschätzung für Eigenentwicklung 11](#_Toc413184511)

[3.2 Analyse der Urlaubsplanungssoftware von Drittanbietern 15](#_Toc413184512)

[3.2.1 Produktvorauswahl 15](#_Toc413184513)

[*3.2.2* *Personal-Planer* 17](#_Toc413184514)

[*3.2.3* *SaaS.de* 18](#_Toc413184515)

[*3.2.4* *Urlaub-Verwalten* 19](#_Toc413184516)

[*3.2.5* *Timetape* 21](#_Toc413184517)

[3.3 „Make-or-Buy“ –Entscheidung 23](#_Toc413184518)

[*4* Einrichtung von *Timetape* 27](#_Toc413184519)

[4.1 Konfiguration und Anlegen der Stammdaten 27](#_Toc413184520)

[4.2 Mitarbeiterschulung 29](#_Toc413184521)

[5 Implementierung eigener Zusatzfunktionen 30](#_Toc413184522)

[5.1 Technologien und Entwicklungsumgebung 30](#_Toc413184523)

[5.2 Anbindung an das Zeiterfassungssystem *Mite* 31](#_Toc413184524)

[5.2.1 Implementierungsansatz 31](#_Toc413184525)

[*5.2.2* Einlesen und Validieren der Daten aus *Timetape* 32](#_Toc413184526)

[5.2.3 Durchlaufen des XML und Schreiben der Zeiteinträge nach Mite 34](#_Toc413184527)

[5.2.4 Benachrichtigungen im Erfolgs- und Fehlerfall 35](#_Toc413184528)

[5.3 Datensicherung 36](#_Toc413184529)

[5.3.1 Zeitgesteuerte Datensicherung 36](#_Toc413184530)

[5.3.2 Web-Ansichten für Urlaubsanträge und Urlaubsansprüche 37](#_Toc413184531)

[5.3.3 Autorisierung und Authentifizierung 38](#_Toc413184532)

[5.4 Anbindung an das Buchhaltungssystem 39](#_Toc413184533)

[5.5 Konfiguration der Anwendung 41](#_Toc413184534)

[5.6 Deployment der Anwendung auf den Firmenserver 42](#_Toc413184535)

[6 Evaluation 43](#_Toc413184536)

[6.1 Erfahrungen mit *Timetape* 43](#_Toc413184537)

[6.2 Erfahrungen mit den eigenen Zusatzfunktionen 44](#_Toc413184538)

[6.3 Bewertung des Gesamtsystems und der „Buy-and-Make“-Entscheidung 45](#_Toc413184539)

[7 Abkürzungsverzeichnis 48](#_Toc413184540)

[8 Abbildungsverzeichnis 49](#_Toc413184541)

[9 Tabellenverzeichnis 50](#_Toc413184542)

[10 Literaturverzeichnis 51](#_Toc413184543)

# Einleitung

Jede Firma, die den Sprung von einem kleinen Start-up zu einer mittelständischen Firma geschafft hat, steht zunehmenden organisatorischen Herausforderungen gegenüber. Dazu gehört unter anderem die Erfassung und Auswertung von Arbeitszeiten sowie die Verwaltung von Urlaubsanträgen. Wenn einfache Lösungen, wie Formulare in Papierform oder eine Verwaltung mit Excel-Tabellen, nicht mehr ausreichen, ist der nächste Schritt in vielen Fällen die Einführung eines elektronischen Verwaltungssystems.

Firmen ohne internes IT-Fachwissen müssen dazu in der Regel auf kommerzielle Standardprodukte zurückgreifen, von denen der Markt eine große Zahl an einfachen bis professionellen Systemen in unterschiedlichen Preisklassen anbietet. Eine IT-Firma hat dagegen alternativ die Möglichkeit, sich ein eigenes System selbst zu entwickeln.

Vor diesen Optionen stand auch die IT-Firma in2code GmbH aus Rosenheim im Jahr 2014, nachdem steigende Projekt- und Mitarbeiterzahlen eine Professionalisierung ihres Urlaubsverwaltungssystems erforderlich machten. Das bisher verwendete, selbst implementierte Werkzeug war an seine Grenzen gestoßen und sollte durch ein neues, leistungsfähigeres System ersetzt werden, das zudem in die Systemlandschaft der Firma integriert werden sollte. Dabei war zunächst offen, ob ein System eines Drittanbieters eingesetzt, ein maßgeschneidertes System selbst implementiert oder eine Kombination aus beidem verwendet werden sollte.

Der Entscheidungsprozess für eine Eigenentwicklung versus den Erwerb einer Standardsoftware enthält viele Parallelen zu dem in der Industrie als „Make-or-Buy“-Decision bzw. „Eigenfertigung oder Fremdbezug?“bezeichneten, oft beschriebenen und analysierten Prozess [1, 2, 3, 4, 5]. In manchen Branchen, wie der Automobilindustrie, sind „Make-or-Buy“-Überlegungen sehr häufig und dazu noch von besonders großer Tragweite. Fehlentscheidungen und Abhängigkeiten von ungeeigneten Zulieferern oder Dienstleistern können hier dramatische finanzielle Folgen haben. Die Praxis zeigt, dass Entscheidungen vielfach zu einseitig mit Fokus auf unmittelbarer Kostenersparnis getroffen werden, wohingegen langfristig wirksame oder strategische Argumente tendenziell vernachlässigt werden [2]. Deshalb stammen auch aus diesem Bereich zahlreiche Denkansätze und Vorschläge für Frameworks, um den Entscheidern schrittweise, systematische Entscheidungshilfen an die Hand zu geben, die zu fundierteren, reproduzierbareren Entscheidungen führen sollen [1, 2, 4, 5]. Oft empfohlen wird eine Besinnung und Fokussierung auf die eigenen Kernkompetenzen und eine besonders sorgfältige Auswahl des Zulieferers bzw. Dienstleisters, die weit mehr als kurzfristige finanzielle Argumente mit berücksichtigt.

Natürlich ist die „Make-or-Buy“-Überlegung für ein Urlaubsverwaltungssystem in einer mittelständischen Firma ein deutlich leichtgewichtiger und weniger risikoreicherer Prozess als eine Zuliefererentscheidung in der Automobilindustrie. Dennoch werden auch hier Fehlentscheidungen getroffen, die zu organisatorischen Problemen oder finanziellen Nachteilen für die Firma führen können. Eine Nutzung von in anderen Branchen gewonnenen Erkenntnissen zu einem erfolgreichen Entscheidungsprozesses wäre mit Sicherheit auch hier von Vorteil.

Im Rahmen dieser Bachelorarbeit sollte deshalb der verfügbare Zeitrahmen zunächst für einen fundierten Entscheidungsfindungsprozess „Neue Urlaubsverwaltung - Make-or-Buy?“ genutzt werden. Daran sollte sich im Falle einer „Make“-Entscheidung die Implementierung eines geeigneten Systems anschließen, im Fall einer „Buy“-Entscheidung die Inbetriebnahme und ggf. eine Anbindung des gewählten Produkts in die Systemlandschaft der Firma. Als Ergebnis der Arbeit sollte der Firma ein funktionierendes, den individuellen Anforderungen genügendes, mit den wichtigsten Systemen vernetztes Urlaubsverwaltungssystem zur Verfügung stehen.

# Ausgangssituation

## Firmenentwicklung

Die IT-Firma in2code GmbH wurde im Jahr 2010 von drei TYPO3-Spezialisten gegründet. Seitdem ist die Firma kontinuierlich gewachsen und zählt mittlerweile 20 Mitarbeiter in Festanstellung (Abbildung 1). Insgesamt wurden bis Ende 2014 ca. 280 unterschiedliche Kunden betreut, die Zahl der durchgeführten Projekte liegt noch deutlich darüber. Parallel zur Zahl der Mitarbeiter und der gleichzeitig betreuten Projekte und stiegen auch die organisatorischen Herausforderungen, unter anderem auch rund um den Bereich Urlaubsplanung.

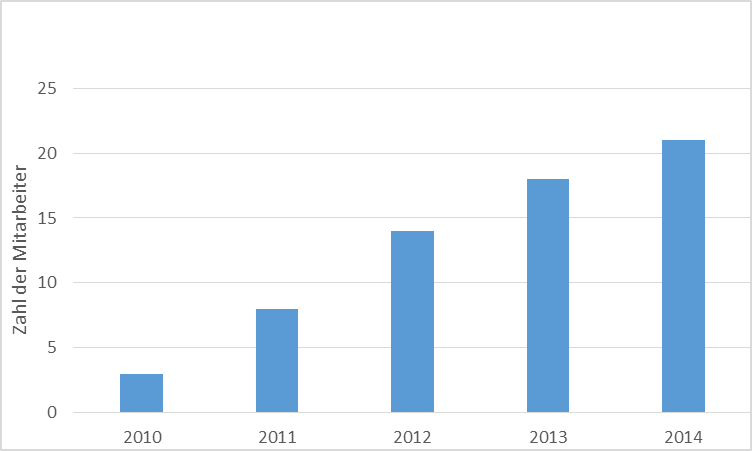


Abbildung 1 Entwicklung der Mitarbeiterzahlen seit Firmengründung

Die Firma hat sich im Jahr 2014 strukturell neu organisiert. Zuvor gab es eine Matrix-Struktur, bei der Entwickler an unterschiedlichen Projekten arbeiteten, die jeweils von einem Projektleiter betreut wurden. Um die Reibungsverluste zwischen Projektleitung und Entwicklung zu verringern und die Zufriedenheit und Effizienz der Entwickler durch Fokussierung auf ein einziges Projekt zu erhöhen, wurde eine Organisation in Projektteams eingeführt. Jedes Projektteam arbeitet dabei möglichst nur an einem einzigen Projekt gleichzeitig, die Teams organisieren sich weitgehend selbständig und sind gemeinsam ist für den Projekterfolg verantwortlich. Daneben gibt es noch einen Mitarbeiter, der ausschließlich für das Servermanagement zuständig ist sowie eine Teamassistentin. Die einzigen Vorgesetzten und somit zuständig für die Genehmigung der Urlaubsanträge aller Mitarbeiter sind die beiden Geschäftsführer.

Abbildung 2 zeigt schematisch die Organisationsstruktur der in2code GmbH; die tatsächliche Zahl der Projektteams und zugehörigen Mitarbeiter kann dabei variieren.

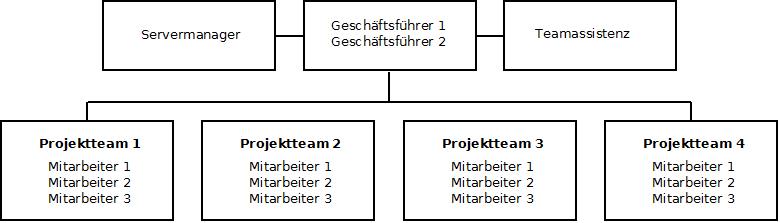


Abbildung 2 Organigramm der in2code GmbH

Diese Organisationsform bedeutet für die Urlaubsplanung, dass zu jeder Zeit mindestens ein Mitglied jedes Projektteams anwesend sein sollte, damit alle Projekte auch in Urlaubszeiten sinnvoll betreut bzw. fortgeführt werden können. In einem Urlaubsverwaltungstool sollte deshalb ersichtlich sein, welchem Projektteam ein Mitarbeiter angehört und zu welchen Zeiten seine Teamkollegen Urlaub beantragt haben, damit Überschneidungen vermieden werden können.

## Systemumgebung

Ein Urlaubsverwaltungssystem ist stets auch inhaltlich verknüpft mit einem ggf. genutzten Zeiterfassungssystem, da für jeden genommenen Urlaubstag und Mitarbeiter entgangene Arbeitsstunden anfallen, die bei der Zeiterfassung mit berücksichtigt werden müssen. Deshalb findet man bei kommerziellen Produkten häufig eine Kombination von Urlaubsverwaltung und Zeiterfassung (z.B. *Timepunch*, *REXX VacationManager*, *Saas.de*, siehe Tabelle 3, Seite 12). Doch auch der Bereich der Personalbuchhaltung wird von der Urlaubsverwaltung berührt, da Urlaubstage und aktuelle Urlaubsansprüche in der Regel in den monatlichen Lohnabrechnungen ausgewiesen werden, um den Mitarbeitern eine Möglichkeit zum Datenabgleich bieten zu können.

So waren auch bei der in2code GmbH die Bereiche Urlaubsplanung, Zeiterfassung und Personalbuchhaltung durch das neue Urlaubsverwaltungssystem betroffen. Im Folgenden ist deshalb die Ausgangssituation für diese drei Systeme bei der in2code GmbH beschrieben.

### Urlaubsverwaltung

In den ersten Monaten nach der Firmengründung konnten Urlaube durch einfache Absprachen der Kollegen untereinander geregelt werden, ein Genehmigungsprozess war nicht notwendig. Mit der zunehmenden Zahl der Mitarbeiter und der durchgeführten Projekte im Kundenauftrag mehrten sich jedoch die Urlaubsengpässe und es entstand der Wunsch nach einer systematischen Urlaubsverwaltung. Deshalb entwickelte im Jahr 2012 einer der Auszubilden der Firma auf Basis von *Typo3 4.2* [6] ein einfaches Urlaubsverwaltungs-Tool, das bis zum Jahr 2014 in der Firma im Einsatz war (Abbildung 3).

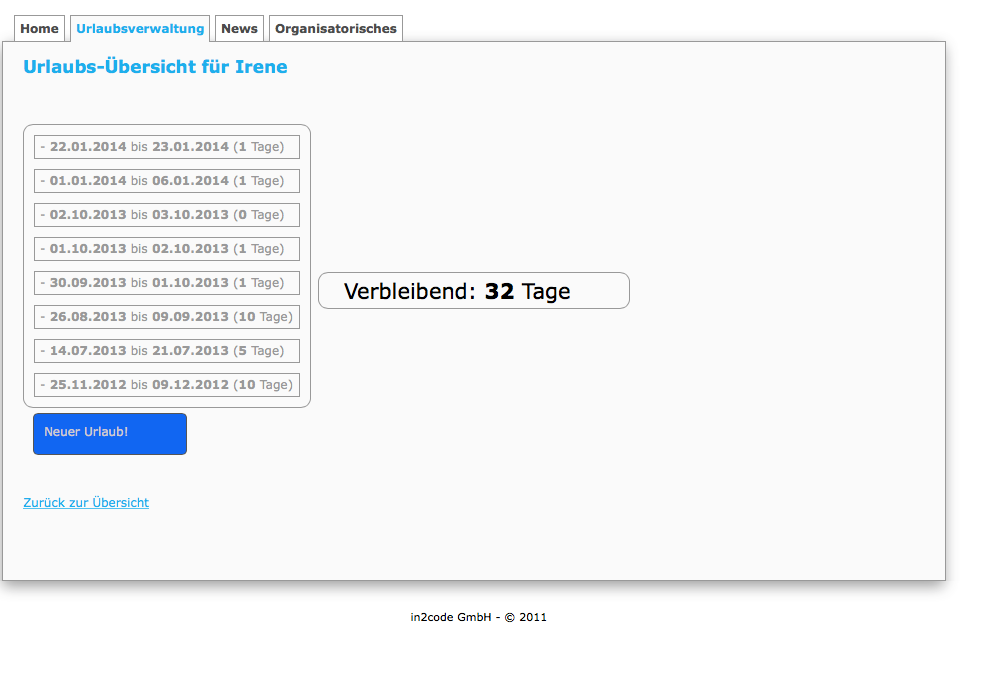
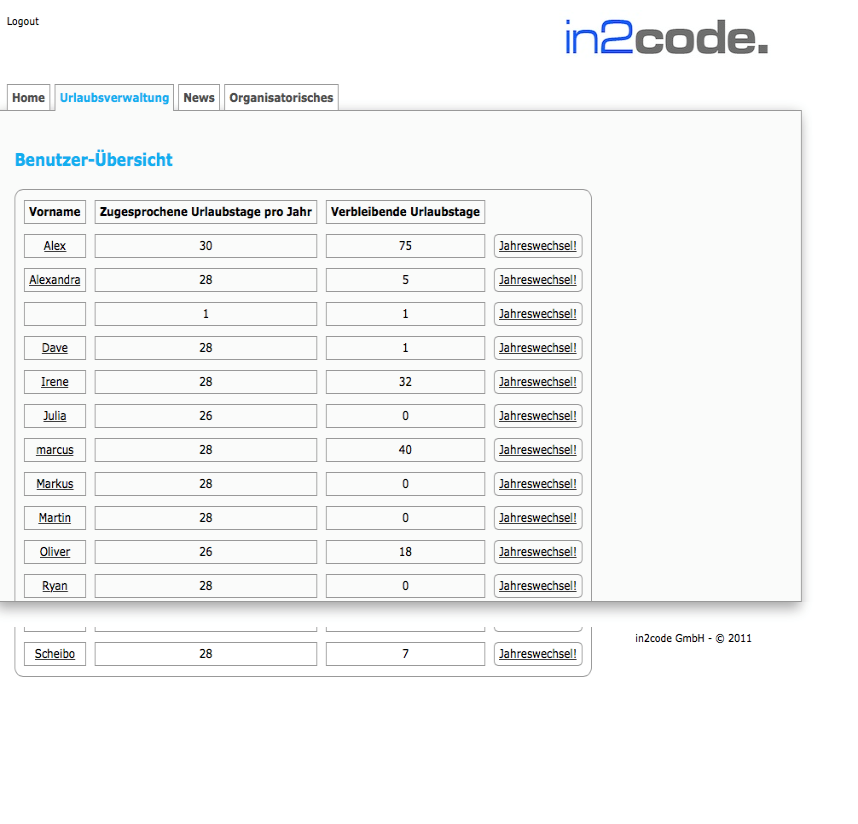


Abbildung 3 Ansichten des bisherigen Urlaubsverwaltungstools

Das Urlaubsverwaltungs-Tool ermöglichte den Mitarbeitern einen Überblick über ihren aktuellen Urlaubanspruch und ihre bisherigen Urlaube. Außerdem konnten eigene Urlaube direkt vom Mitarbeiter für sich selbst eingetragen werden. Allerdings fehlten wichtige Funktionen, wie die Möglichkeit zur Berücksichtigung verschiedener Arbeitszeitmodelle und eine Benachrichtigung der Vorgesetzten oder Teammitglieder beim Eintrag von neuen Urlauben. Besonders problematisch erwies sich auch, dass die Eintragung der Urlaube durch den Mitarbeiter keine Freigabe erforderte und die Urlaube anderer Kollegen nicht eingesehen werden konnten. Somit gab es immer wieder den Fall, dass mehrere Mitarbeiter sich zur selben Zeit Urlaub eintrugen und der Engpass erst zu spät bemerkt wurde. Deshalb sollte das neue System unbedingt einen Freigabe-Workflow unterstützen und durch entsprechende Kalenderansichten drohende Engpässe veranschaulichen.

Zudem war die verwendete Technologie mittlerweile überholt: Das Urlaubsverwaltungs-Tool war als *TYPO3*-Erweiterung auf Basis der in den früheren *TYPO3*-Versionen verwendeten Basisklasse *tslib\_piBase* [7] implementiert worden, die ab Version *TYPO3 4.2* durch das modernere Framework *Extbase* [8] ersetzt worden war. Bei einer Neukonzeption des Urlaubsverwaltungssystems sollte deshalb auf jeden Fall auch ein Technologiewechsel auf eine aktuellere, besser wartbarere Technologie, wie *Extbase* oder das MVC-basierte PHP-Framework *TYPO3 Flow* [9], erfolgen.

### Zeiterfassung

Schon bald nach Firmengründung hatte man sich bei der in2code GmbH für die Verwendung eines kommerziellen, leistungsfähigen Zeiterfassungssystems entschieden. Hintergrund dafür war die Notwendigkeit, Zeitaufwände verschiedenen Projekten exakt zuordnen zu können. Nur so konnten bei aufwandsabhängigen Projekten dem Kunden korrekte Beträge in Rechnung gestellt werden und bei Festpreisprojekten zuverlässige Daten für künftige Kalkulationen und Zeitpläne erhoben werden. Ausgewählt wurde das von der Berliner Firma Yolk entwickelte Online-System *Mite* [10]. Mit diesem Tool können die Mitarbeiter Zeitaufwände für jeden Tag rasch und übersichtlich einem Projekt bzw. nicht abrechenbaren Tätigkeiten zuordnen.

Auch die zu berücksichtigenden Zeiten an Urlaubs- und Feiertagen sollen von den Mitarbeitern in *Mite* eingetragen werden, damit die Kosten hierfür bei der Auswertung der Arbeitszeiten mit berücksichtigt werden können. Abbildung 4 zeigt bespielhaft die Zeiterfassung für einen Urlaubstag eines Mitarbeiters in *Mite*. Die Erfahrung zeigte jedoch, dass die manuellen Einträge von Urlauben und Feiertagen immer wieder vergessen wurden.

Der Funktionsumfang von *Mite* ist nahezu ideal auf die Bedürfnisse der in2code GmbH zugeschnitten und die Kosten bleiben mit 5.- € pro Monat und Mitarbeiter im vertretbaren Rahmen. Zusätzlich zu seinen regulären Funktionen bietet *Mite* eine umfassende Programmierschnittstelle auf Basis von RESTful Webservices [11] an, über die das System gut an andere Anwendungen angebunden werden kann [12]. Mit ausgezeichnetem Support unterstützt *Mite* dabei die Nutzer bei der Entwicklung von Anbindungen. Von einem Auszubildenden der in2code GmbH wurde mit Hilfe der REST-Schnittstelle bereits erfolgreich eine Verknüpfung zwischen dem verwendeten Ticket-System *Redmine* [13]und *Mite* hergestellt. Bei der Bearbeitung eines Tickets in *Redmine* kann nun durch Bedienung einer Stoppuhr der Zeitaufwand automatisch dem zugehörigen Projekt in *Mite* zugerechnet werden.

Aufgrund der guten Erfahrungen mit *Mite* wollte die in2code GmbH an diesem System der Zeiterfassung festhalten. Eine neue Urlaubsverwaltung sollte, wenn möglich, mit *Mite* verknüpft werden. Eine automatische Übernahme der Urlaubszeiten nach *Mite* würde eine Entlastung der Mitarbeiter von manuellen Zeiteinträgen bedeuten und dabei helfen, die Zahl der vergessenen Einträge zu reduzieren.

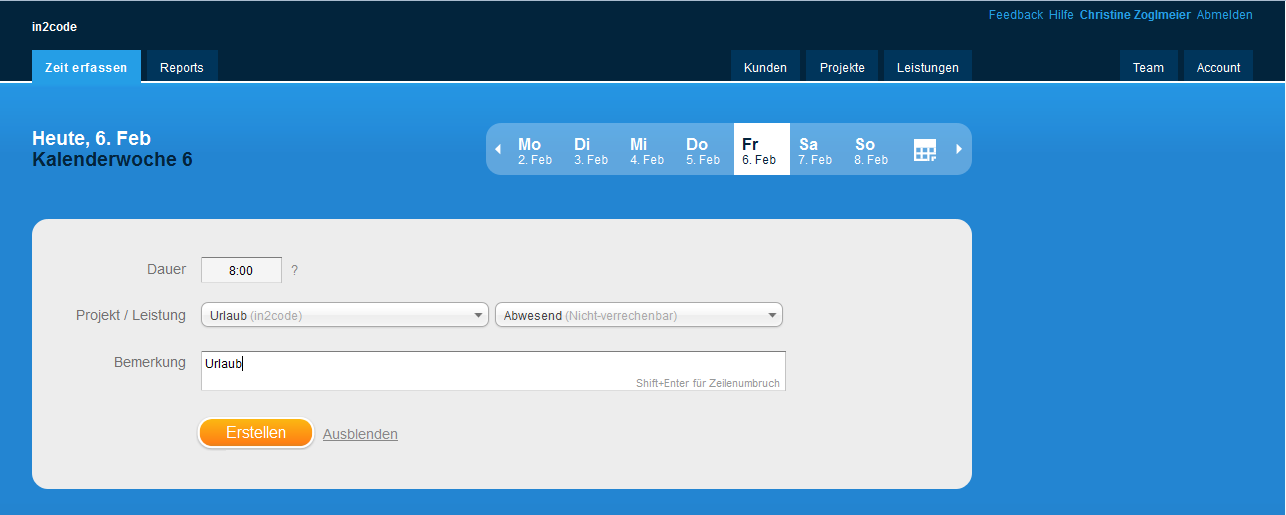


Abbildung 4 Beispiel einer Eingabemaske für Zeiten in Mite

### Personalbuchhaltung

Ein weiteres System, das mit der Urlaubsverwaltung zusammen hängt, ist die Personalbuchhaltung. Dieser Aufgabenbereich ist bei der in2code GmbH an einen externen Dienstleister, die Steuerkanzlei Muggenthaler in Prien, vergeben. Dort werden die Lohnabrechnungen für die Mitarbeiter der in2code GmbH erstellt. In den meisten Firmen werden auf den monatlichen Lohnabrechnungen auch die Urlaubstage und aktuellen Urlaubsrestansprüche für jeden Mitarbeiter ausgewiesen. Diese Information war auf den Abrechnungen für die in2code GmbH wegen des damit verbundenen organisatorischen Mehraufwands bisher noch nicht enthalten. Es bestand deshalb der Wunsch, im Zuge der Implementierung eines neuen Urlaubsverwaltungssystems die Urlaubsinformationen automatisch an den Personaldienstleister zu übermitteln. Dadurch könnten die Lohnabrechnungen vervollständigt werden und die Mitarbeiter ihren Urlaubsanspruch regelmäßig nachvollziehen.

### Systemlandkarte

Abbildung 5 zeigt schematisch die drei Systeme Zeiterfassung, Urlaubsverwaltung und Personalbuchhaltung, so wie sie zu Beginn der Arbeit bei der in2code GmbH zusammen hingen. Die Mitarbeiter tragen ihre Urlaube und Zeiten manuell in das Urlaubsverwaltungs- bzw. Zeiterfassungssystem ein. Die Geschäftsführung wertet die Einträge aus beiden Systemen für die Bilanzierung und Analyse aus. Die Buchhaltung arbeitet mit ihrem eigenen Personalbuchhaltungssystem und erhält die benötigten Personaldaten per E-Mail von der Geschäftsführung. Da die so übermittelten Personaldaten noch nicht die für die Urlaubsverwaltung relevanten Urlaubstage enthielten, sind die Linien von und zur Personalbuchhaltung gestrichelt dargestellt. Hier sollte der Informationsfluss noch um die Urlaubstage ergänzt werden.

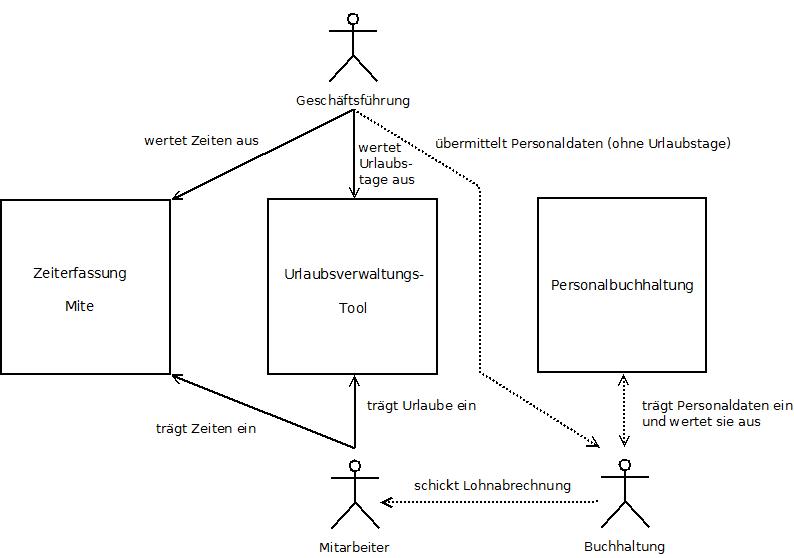


Abbildung 5 Systemlandkarte vor Implementierung des neuen Urlaubsverwaltungssystems

## Bedarfsanalyse und Zielsetzung

### Anforderungen

In Gesprächen mit der Geschäftsführung wurden die wichtigsten Anforderungen an ein neues Urlaubsverwaltungssystem gesammelt und priorisiert:

1. **Hohe Priorität:**
   1. Der Resturlaub soll unter Berücksichtigung von Feiertagen, Wochenenden und gewähltem Arbeitszeitmodell individuell für jeden Mitarbeiter automatisch berechnet werden.
   2. Es sollen verschiedene Arbeitszeitmodelle und Urlaubsansprüche für die Mitarbeiter verwaltet werden können.
   3. Es soll ein Freigabeworkflow unterstützt werden: Urlaubsanträge müssen von mindestens einem der beiden Geschäftsführer oder einem Antragsverwalter aus dem jeweiligen Projektteam genehmigt oder abgelehnt werden. In diesem Workflow sollen E-Mail-Benachrichtigungen bei neuen Urlaubsanträgen bzw. bei Genehmigung / Ablehnung eines Antrags verschickt werden.
   4. Dem Antragsverwalter sollen Kalenderansichten für die Urlaube aller Mitarbeiter der Firma zur Verfügung stehen, damit Kollisionen rechtzeitig erkannt werden können.
   5. Den einzelnen Mitarbeitern sollen Kalenderansichten für ihren bisher genommenen bzw. beantragten Urlaub zur Verfügung stehen.
   6. Die einzelnen Mitarbeiter sollen eine Übersicht über ihren aktuellen Resturlaub einsehen können.
   7. Das System soll webbasiert sein, damit die Urlaubsanträge und Urlaubsgenehmigungen ortsunabhängig vorgenommen werden können.
   8. Das System soll durch ein Login vor unberechtigtem Zugriff geschützt werden.
   9. Es soll möglich sein, die Daten über genehmigte Urlaube in den firmeneigenen Google-Kalender zu übernehmen.
2. **Mittlere Priorität:**
   1. Den Mitgliedern sollen Kalenderansichten der Urlaube ihrer Teammitglieder zur Verfügung stehen, damit auch untereinander Kollisionen frühzeitig erkannt werden.
   2. Beim Stellen eines Urlaubsantrags soll ein Urlaubsvertreter ausgewählt werden können.
   3. Der Geschäftsführung bzw. dem Antragsverwalter soll auch eine Übersicht der abgelehnten Urlaubsanträge zur Verfügung stehen, damit Ungleichgewichte bei der Ablehnung von Urlaubsanträgen vermieden werden können.
   4. Falls ein Drittanbieter mit externer Datenhaltung ausgewählt wird, sollen die wichtigsten Daten zur Urlaubsverwaltung auf dem firmeneigenen Server gespeichert werden, um die Abhängigkeit vom Drittanbieter zu reduzieren. In jedem Fall sollen dabei die Daten über die genehmigten Urlaubsanträge für jeden Mitarbeiter gespeichert werden.
   5. Die an jedem Urlaubs- bzw. Feiertag anfallenden Arbeitsstunden für jeden Mitarbeiter sollen automatisch in das Zeiterfassungssystem *Mite* übernommen werden.
3. **Niedrige Priorität:**
   1. Eine Übersicht über die aktuellen Urlaubstage jedes Mitarbeiters soll automatisch an die Personalbuchhaltung übermittelt werden.
   2. Kalender für jeden einzelnen Mitarbeiter sollen in seinen persönlichen Google-Kalender exportiert werden können.
   3. In den Google-Kalendern soll bei jedem Urlaubstag der benannte Vertreter mit angezeigt werden.

### Gewünschte Systemumgebung

Bei der Anforderungsanalyse zeigte sich, dass das Urlaubsverwaltungssystem, unabhängig davon ob es selbst implementiert oder das Produkt eines Drittanbieters ausgewählt würde, in die Systemlandschaft der Firma eingebunden werden müsste. Laut Anforderung Nummer B5 sollte eine automatische Datenübernahme aus der Urlaubsverwaltung in das Zeiterfassungssystem *Mite* erfolgen. Nach Anforderung Nummer C1 sollte auch die Information über die Urlaubstage der Mitarbeiter automatisch an die Buchhaltung übermittelt werden, damit die Lohnabrechnungen entsprechend ergänzt werden könnten. Laut Anforderung B4 sollte zudem bei Auswahl eines Drittanbieters mit externer Datenspeicherung eine unabhängige Datensicherung auf einem Server der in2code GmbH durchgeführt werden.

Abbildung 6 zeigt eine schematische Übersicht über die die Urlaubsverwaltung betreffenden Systeme mit ihren gewünschten Verknüpfungen nach Inbetriebnahme des neuen Urlaubsverwaltungssystems. Die neuen Anbindungen sind dabei rot eingezeichnet.

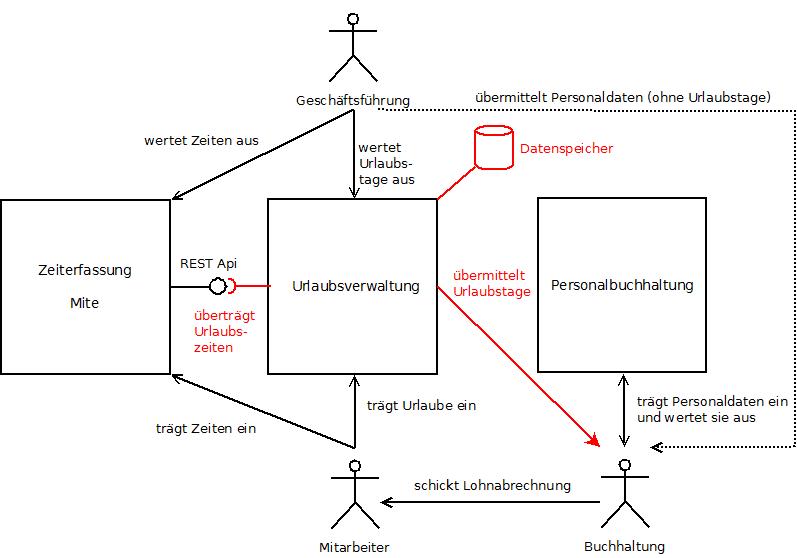


Abbildung 6 Gewünschte Systembeziehungen nach Inbetriebnahme des Urlaubsverwaltungssystems

# „Make-or-Buy“ - Entscheidungsprozess

## Aufwandsschätzung für Eigenentwicklung

Einer der schwierigsten Punkte bei einer „Make-or-Buy“-Entscheidung ist eine realistische Aufwandsschätzung für die Eigenentwicklung eines wirklich funktionsfähigen, alle Anforderungen erfüllenden Systems. Während die Kosten für den Kauf eines Fremdsystems sehr genau beziffert werden können, ist die Aufwandschätzung für Eigenentwicklung recht unsicher und wird oft zu gering angesetzt. Weniger offensichtliche Posten, wie Bugfixing, Wartung und Dokumentation werden gerne übersehen und Teile der Implementierung stellen sich als deutlich problematischer und langwieriger heraus, als angenommen.

Vor der Vergabe des Themas für diese Arbeit gab es bei der in2code GmbH bereits Pläne, das neue Urlaubsverwaltungssystem durch zwei Auszubildende der Firma entwickeln zu lassen. Diese fertigten darauf hin im September 2014 ein Lastenheft für die geplante Anwendung „in2Holiday“ an, in dem sie versuchten, ihren Implementierungsaufwand abzuschätzen [14]. In Tabelle 1 ist das Ergebnis ihrer Vorüberlegungen, dargestellt. Zum besseren Verständnis sei darauf hingewiesen, dass zu diesem Zeitpunkt geplant war, die Anwendung auf Basis einer bereits bestehenden Typo3-Flow-Anwendung [9] zu entwickeln, die im Lastenheft als „CRM“ (Customer-Relations-Management - System) bezeichnet wird. Dort existiert bereits ein Datenmodell für das Objekt „Employee“, das für die neue Anwendung erweitert werden sollte.

Tabelle 1 Aufwandsschätzung aus dem Lastenheft "in2Holiday"

|  |  |
| --- | --- |
| **Arbeitspaket** | **Aufwand in Manntagen** |
| **Anpassung des aktuellen CRM** |  |
| Ansichten Mobile/Desktop anpassen | 1 |
| Login hinzufügen | 1 |
| Objekt "Employee" erweitern | 0,5 |
| **Neue Objekte** |  |
| Kalender | 0,25 |
| Urlaubsantrag | 0,25 |
| **Views und Actions** |  |
| Hauptansicht | 6 |
| Ansicht "Antrag erstellen" | 6 |
| Einspruchsansicht | 4 |
| **Weitere Anbindungen** |  |
| Google Calendar | 0,5 |
| Mite | 0,5 |
| **SUMME** | **20** |

Die beiden Auszubildenden hatten versucht, ihren Aufwand möglichst realistisch einzuschätzen, hatten sich jedoch dabei nach eigener Aussage sehr schwer getan, weil Ihnen bei vielen Punkten die Erfahrung mit dem verbundenen Implementierungsaufwand fehlte. Insgesamt erschien der geschätzte Aufwand mit 20 Manntagen für die Entwicklung eines funktionsfähigen Urlaubsplaners sehr gering angesetzt. Es fiel auf, dass für Aufgaben, die die Programmlogik betreffen – wie die korrekte Berechnung von Urlaubsansprüchen und Resturlauben sowie die Überprüfung auf Kollisionen – im Vergleich zu Aufgaben, die das Erstellen von Ansichten/Views betreffen, deutlich weniger Zeitbedarf angesetzt worden war.

Ganz ausgelassen worden waren übergreifende Aspekte wie Testen, Dokumentation, Besprechungen, Deployment etc. Deshalb wurde die Aufwandsschätzung aus Tabelle 1 noch einmal überarbeitet und die Zeitschätzungen in den meisten Fällen erhöht. Das Ergebnis dieser Überlegungen ist in Tabelle 2 dargestellt und diente als Vergleichsgrundlage für die spätere „Make-or-Buy“-Überlegung. In dieser Tabelle ist zusätzlich, soweit sie eindeutig zugeordnet werden konnte, die Nummer der Anforderung aus Abschnitt 2.3.1 eingetragen.

Es sei darauf hingewiesen, dass der Aufwand für die Anforderung C1, „Datenübermittlung an die Personalbuchhaltung“, in beiden Schätzungen noch nicht aufgeführt ist, da diese Anforderung erst später hinzugekommen ist.

Tabelle 2 Überarbeitete Aufwandsschätzung für die Implementierung des Urlaubsplaners, Stand 19.11.2014

| **Arbeitspaket** | **Aufwand in Manntagen** | **Anforderung** |
| --- | --- | --- |
| Anpassung des aktuellen CRM |  | übergreifend |
| Ansichten Mobile/Desktop | 1 | übergreifend |
| Autorisierung/Authentifizierung | 1 | A8 |
| **Neue Datenmodelle / Objekte** |  |  |
| Abteilung | 0,5 | B1 |
| Urlaubsantrag | 0,5 | übergreifend |
| Arbeitszeitmodell | 1 | A1 |
| Vorgesetzter / Urlaubsverwalter (erweitert Employee) | 0,5 | A3 |
| Kalender | 0,5 | A4, A5 |
| **Views** |  |  |
| Hauptansicht | 5 | A6 |
| Abteilung CRUD[[1]](#footnote-1) | 0,5 | B1 |
| Urlaubsantrag CRUD | 2 | übergreifend |
| Arbeitszeitmodell CRUD | 1 | A1 |
| Mitarbeiter CRUD | 0,5 | übergreifend |
| Vorgesetzter CRUD | 0,5 | übergreifend |
| Passwort zurücksetzen | 0,5 | A8 |
| Ansicht eigenes Profil | 1 | A8 |
| Editierung Feiertage/freie Tage/Firmenurlaub | 1 | A1 |
| Firmenkalender | 3 | A4 |
| Abteilungskalender | 1 | A4 |
| persönlicher Kalender | 1 | A5 |
| Urlaubsantrag | 2 | A1 |
| Urlaubsgenehmigung | 2 | A3 |
| E-Mail Urlaubsantrag | 1 | A3 |
| E-Mail Urlaubsgenehmigung/-ablehnung | 1 | A3 |
| **Actions** |  |  |
| Abteilung CRUD | 0,5 | B1 |
| Urlaubsantrag CRUD | 0,5 | übergreifend |
| Arbeitszeitmodell CRUD | 1 | A1 |
| Mitarbeiter CRUD | 0,5 | übergreifend |
| Vorgesetzter CRUD | 0,5 | A3 |
| Vergessenes Passwort, Link versenden | 1 | A8 |
| Editierung Feiertage/freie Tage/Firmenurlaub | 1 | A1 |
| Urlaubsantrag stellen | 2 | A1 |
| Urlaubsantrag genehmigen/ablehnen | 2 | A1, A3 |
| E-Mail Urlaubsantrag / -genehmigung /-ablehnung | 2 | A3 |
| Link zur Antragsseite | 0,5 | A4 |
| Berechnung des Resturlaubs nach Arbeitszeitmodell | 3 | A1,A2,A6 |
| Prüfung von Überschneidungen | 2 | A3 |
| **Anbindung an andere Systeme** |  |  |
| Google Calendar | 1 | A5 |
| Mite | 1 | B5 |
| **Sonstiges** |  |  |
| Planung/Besprechungen | 10 | übergreifend |
| Bugfixing | 3 | übergreifend |
| Testen | 2 | übergreifend |
| Deployment/Infrastruktur | 3 | übergreifend |
| Dokumentation | 2 | übergreifend |
| **SUMME** | **67** |  |

Diese zweite Aufwandschätzung lag mit 67 Manntagen deutlich über der ersten Schätzung und ist voraussichtlich etwas realistischer. Aber auch diese Prognose war mit großen Unsicherheiten behaftet und es sollte sich später herausstellen, dass der Arbeitsaufwand für einige Arbeitspakete, wie z.B. die Anbindung an *Mite*, tatsächlich noch erheblich darüber liegen sollte (siehe auch Abschnitt 6.2). Dennoch wurden unter der Annahme eines Aufwands von 67 Manntagen im Anschluss die geschätzten Kosten einer Eigenentwicklung für die Firma berechnet. Der mittlere Bruttoarbeitslohn für einen Auszubildenden beträgt ca. 2000.- € brutto pro Monat, also ca. 100.- € pro Werktag / Manntag. Die Kosten für die Entwicklung des Urlaubsplaners bei hauptsächlicher Implementierung durch die Auszubildende lägen deshalb bei ca. 67 \* 100.- €, also ca. 6700.- Euro. Mit diesen, nach wie vor eher optimistisch geschätzten Kosten, war also der Preis für das System eines Drittanbieters zu vergleichen.

In einem nächsten Schritt wurde eine Marktanalyse durchgeführt, um herauszufinden, ob es ein preisgünstigeres, allen Anforderungen genügendes kommerzielles System gibt, das eine sinnvolle Alternative zur Eigenentwicklung darstellen könnte.

## Analyse der Urlaubsplanungssoftware von Drittanbietern

### Produktvorauswahl

Zunächst wurde eine Internet-Recherche nach elektronischen Urlaubsverwaltungssystemen durchgeführt. Dazu wurde nach den Stichworten *Urlaubsplaner*, *Urlaubsverwaltung*, *holiday planner* oder *vacation planner* in Kombination mit den Begriffen *online*, *elektronisch* oder *electronic* gesucht. Insgesamt wurden 17 gefundene Produkte und auf ihre Eignung zum Einsatz bei der in2code GmbH untersucht.

In Tabelle 3 sind die untersuchten Produkte und der im November 2014 gültige Preis (jeweils berechnet für eine Firma mit 20 Mitarbeitern) aufgelistet. Die Preisinformationen wurden, soweit vorhanden, direkt von der Website übernommen. Bei den mit \* markierten Systemen mussten individuelle Angebote vom Anbieter eingeholt werden.

Die gefundenen Produkte ließen sich pauschal in 4 Gruppen unterteilen:

* Gruppe 1: Professionelle, leistungsfähige und teure Systeme, konzipiert für Firmen mit großen Mitarbeiterzahlen. Die Urlaubsplanung ist hier oft nur ein Teilmodul eines großen, modular aufgebauten Gesamtsystems (ERP oder CRM-System).
* Gruppe 2: Systeme mit teilweise geeignetem Funktionsumfang, die jedoch ausschließlich für den Betrieb auf Windows-Systemen (meist Windows-Server) konzipiert sind.
* Gruppe 3: kleine, günstige oder sogar kostenlose Tools für die lokale Installation. Die Systeme waren jedoch meist unausgereift bzw. der Funktionsumfang unzureichend.
* Gruppe 4: Webbasierte Systeme mit geeignetem Funktionsumfang

Tabelle 3 Liste der verglichenen elektronischen Urlaubsplaner

| **Nr.** | **Name** | **Website** | **Kosten** | **Gruppe** |
| --- | --- | --- | --- | --- |
| 1 | *Tecart Urlaubsplanung* | http://www.tecart.de/crm-urlaubsplanung-software | SaaS: 5400 € p.a.  Kauflizenz: 3880 € | 1 |
| 2 | *EDV Systemhaus* | http://fehlzeitenplaner-online.um-edv.de/ | Keine Preisinformation verfügbar | 1 |
| 3 | *Confluence von Atlassian* | http://www.atlassian.com/de/ software/confluence/team-calendars | SaaS: ca. 1600 € p.a.  Betrieb auf eigenem Server: ca. 1400 € p.a. | 1 |
| 4 | *REXX Vacation Manager* | http://www.rexx-systems.com/urlaubsverwaltung.php?cp=Vacation&kw=firmen%20urlaubsplaner | \*Mietlizenz: 1200 € p.a. zzgl. Einmalige Einrichtungskosten von 2475 €  Kauflizenz: 2400 € zzgl. Einmalige Einrichtungskosten von 5000 €, zzgl. Support von 480 € p.a.  Tagessatz für weitere Leistungen (z.B. Schnittstellen: 990 €) | 1 |
| 5 | *Timepunch* | http://www.timepunch.de | Kauflizenz: 1960 €  Mietlizenz: 780 € p.a. | 1 |
| 6 | *Figgo Urlaubsverwaltung* | http://figgo.ch/de/ | \*960.- pro Jahr zzgl. Einmalige Einrichtungskosten von 1575 € | 1 |
| 7 | *Addware Urlaubsmanager* | http://www.addware.de/ urlaubsmanager.html | \*Kauflizenz 721€ | 2 |
| 8 | *Selbstdenker AG: Gruppenkalender* | http://www.selbstdenker.ag/ gruppenkalender-home/ | Keine Preisinformation verfügbar | 2 |
| 9 | *Teamsoftware Urlaubsplaner* | http://www.teamsoftware.net | 24,95 einmalig | 2, 3 |
| 10 | *EXO Urlaubsplaner und Fehlzeitenanalyse* | http://www.datacontrol-online.de/exo.php | 400.- einmalig | 2 |
| 11 | *Urlaubsplaner für Excel und OpenOffice* | http://urlaubsplaner.jgm-software.com | 39 € einmalig | 3 |
| 12 | *Projekt der TU-Berlin* | http://www.freeware.de/ download/urlaubsplaner\_32042.html | kostenlos | 3 |
| 13 | *Mitarbeiterbereich* | http://www.mitarbeiterbereich.de/#plans | 120 € p.a. | 4 |
| 14 | *Saas.de Urlaubsverwaltung* | http://www.saas.de/Produkte/ Urlaubsverwaltung.html | 200 € p.a. | 4 |
| 15 | *Urlaub-Verwalten* | https://www.urlaub-verwalten.de/ | Auf externem Server: kostenlos  Auf eigenem Server, 200 € p.a. Mietlizenz, einmalig 480 € bei Kauflizenz | 4 |
| 16 | *Personal-Planer*  *Urlaubsplaner* | http://www.personal-planer.de/urlaubsplaner.html | 396.- pro Jahr | 1,4 |
| 17 | *Timetape* | http://www.timetape.de | 300 € p.a. | 4 |

Gruppe 1 schied wegen des vergleichsweise hohen Preises und des zu komplexen und teilweise sehr eng mit anderen Modulen des Anbieters gekoppelten Funktionsumfangs rasch aus. Auch die Produkte der Gruppe 2, die Windows bzw. Windows-Server-basierten Systeme, mussten ausgeschlossen werden, da bei der in2code GmbH fast alle Rechner auf Linux- oder OS-X –Betriebssystemen beruhen und die Einrichtung eines Windows-Servers keine Option war. Die Produkte aus Gruppe 3 waren zwar günstig, aber leider nur für den lokalen Betrieb konzipiert, also nicht webbasiert, und der Funktionsumfang in keinem der betrachteten Systeme zufrieden stellend.

Es verblieben demzufolge noch 5 Produkte aus der Gruppe 4 (webbasierte Systeme mit geeignetem Funktionsumfang), die eingehender untersucht wurden:

1. *Mitarbeiterbereich*
2. *Personal-Planer*
3. *Saas.de Urlaubsverwaltung*
4. *Urlaub-Verwalten*
5. *Timetape*

Für alle 5 Produkte wurden Testzugänge angelegt, notfalls auf Antrag beim Anbieter, um den Funktionsumfang und die Bedienungsfreundlichkeit (Usability) der Anwendung eingehender testen zu können.

Dabei zeigte sich, dass das günstigste System in der Gruppe, *Mitarbeiterbereich*, eine der wichtigsten Anforderungen nicht erfüllte, da hier kein elektronischer Genehmigungsprozess unterstützt wurde. Es konnten lediglich Antragsformulare im pdf-Format per E-Mail verschickt werden, die dann ausgedruckt und unterzeichnet werden mussten. Da auch noch Fehler bei der Berechnung der Urlaubstage im Antragsformular auftraten, wurde das System ebenfalls ausgeschlossen.

Die anderen vier Systeme dagegen blieben interessant. Ihr Funktionsumfang und die Usability wurden im Detail getestet und sind im Folgenden näher beschrieben.

### *Personal-Planer*

Leistungsumfang:

* Krankheitstage mit Details erfassen
* Auswertungen der Krankheitstage
* Urlaubsbeantragung
* Mehrstufiger Genehmigungsprozess möglich
* Buchungsüberwachung
* Qualifikationsprüfung (d.h. sind immer genügend Mitarbeiter mit entsprechender Qualifikation vorhanden?)
* Urlaubssperren
* Feiertage und Schulferien auf Bundeslandebene je Mitarbeiter
* Umfassendes Berechtigungskonzept
* Brückentagsverteilung
* Ausgleichstage
* Jubiläen und Geburtstage
* PDF-Auswertungen
* Notizfunktion für Führungskräfte
* Statusmeldungen per E-Mail bzw. SMS
* Automatische Anpassung des Urlaubsanspruchs
* Revisionssichere Chronologie
* Eintrag beliebiger Termine möglich
* Dynamische Arbeitstagregelungen

Bewertung:

Das System machte einen sehr professionellen Eindruck. Der Urlaubsplaner ist eines von mehreren Modulen des Herstellers, der auch Module für Digitale Personalakten, Zeiterfassung, Personalkapazitätsplanung usw. anbietet. Laut Herstellerauskunft wurde der *Personal-Planer* zum Testzeitpunkt von rund 750 Firmen genutzt. Das Urlaubsplaner-Modul bot rund um die Bereiche Verwaltung von Abwesenheitszeiten, Genehmigungsprozess, Vertreterregelung sehr umfangreiche Funktionen. Nicht gegeben war allerdings die Möglichkeit, die Urlaubsdaten nach Google Calendar zu exportieren. Der Preis war mit 400.- pro Jahr akzeptabel. Der größte Nachteil war jedoch, dass das System durch den weit über die benötigten Funktionen hinausgehenden Umfang teilweise recht unübersichtlich wirkte. Hier wären eine sehr ausführliche Einarbeitung der Mitarbeiter und ein längerer Abstimmungsprozess darüber, welche der vom System angebotenen Funktionen von den Mitarbeitern bzw. der Geschäftsführung tatsächlich genützt werden und welche nicht, erforderlich gewesen. Auch war die etwas unmodern wirkende, in sehr kleiner Schriftgröße gehaltene Desktop-Optik nicht besonders ansprechend und überfrachtet.

### *SaaS.de*

Leistungsumfang:

* Abteilungshierarchie
* Genehmigungsprozess mit Abteilungsleiter / Chef konfigurierbar
* E-Mail-Benachrichtigungen
* Aufgabenübersicht für Vorgesetzten
* Individueller Urlaubsanspruch / Arbeitszeiten im Mitarbeiterprofil
* Urlaub stundenweise abrechenbar
* Kalenderintegration z.B. in Google Calendar, iCal
* Urlaubsantrag mit Bemerkung
* Feiertage / Schulferien voreingestellt, konfigurierbar
* Darstellung der Schulferien
* Individuelle Abwesenheitsgründe
* Berichte als Excel-Export
* Ansicht für einzelne Mitarbeiter / Abteilungen / gesamte Firma

Bewertung:

Mit 200.- € pro Jahr war das von der Egotec GmbH entwickelte System vergleichsweise günstig. Auch hier war der Urlaubsplaner wieder eines von mehreren Modulen des Herstellers, der auch die Bereiche Zeiterfassung, Projektzeiterfassung und Personaleinsatzplanung anbietet. Die Systeme der Egotec GmbH wurden laut Auskunft auf der Website zum Testzeitpunkt von über 1000 Firmen genutzt. Der Funktionsumfang des Systems war gut, auch ein Export für den Google-Kalender und eine Exportfunktion für Excel war bereits enthalten. Den Punkt Verfügbarkeit betreffend sicherte der Anbieter eine 99%ige Verfügbarkeit des Rechenzentrums zu.

Allerdings war der Prozess für die Genehmigung der Urlaube für die Zwecke der in2code GmbH zu starr. Er beruhte auf einer streng hierarchischen Abteilungsstruktur mit Genehmigung durch den jeweiligen direkten Vorgesetzten. Die Möglichkeit, einen Mitarbeiter als Antragsverwalter einzusetzen, der nicht als direkter Vorgesetzter der jeweiligen Abteilung registriert ist, war nicht vorgesehen. Auch war die beabsichtigte Kopplung mit den anderen Modulen des Anbieters nicht zu übersehen. So wurden im Testbetrieb täglich E-Mail-Benachrichtigungen wegen fehlender Arbeitszeitbuchungen erhalten, da eine parallele Nutzung des Zeiterfassungssystems vorgesehen war. Die Windows-Server-Optik war ebenfalls nicht besonders ansprechend.

### *Urlaub-Verwalten*

Leistungsumfang:

* Urlaubsverwaltung: Unterstützung des Genehmigungsprozesses, beliebige Zuordnung von Vorgesetzten zu Mitarbeitern, Übersicht über Urlaubskonten/Krankmeldungen, Überprüfung von Überschneidungen, Einträge durch Vorgesetzte für Mitarbeiter möglich
* Urlaubssperren
* Überstundenverwaltung
* Verwaltung sonstiger Abwesenheitszeiten: Übersicht der eigenen Abwesenheitszeiten, Übersicht der Abwesenheitszeiten aller Mitarbeiter eines Vorgesetzten
* Krankheitsverwaltung
* Vertreterauswahl
* Feiertage voreingestellt, konfigurierbar
* Diverse Einstellungsoptionen zu Benutzersteuerung / Berechtigungen
* Ansichten: für eigene Abwesenheiten / für Vorgesetzte Ansicht für mehrere Mitarbeiter
* Individuelle Urlaubskontingente verwaltbar
* Corporate Design möglich (allerdings sehr eingeschränkt!)
* Export der eigenen Urlaube nach Google Calendar
* E-Mail-Kopien an andere Mitarbeiter möglich

Bewertung:

*Urlaub-Verwalten* und *Timetape* stellten sich als die beiden interessantesten Kandidaten für die Bedürfnisse der in2code GmbH heraus und wurden deshalb besonders detailliert untersucht.

*Urlaub-Verwalten* wird bereits seit 2003 angeboten und hatte zum Testzeitpunkt laut eigenen Angaben über 40.000 Nutzer. Dabei gab es die Wahl zwischen drei verschiedenen Lizenzmodellen:

1. Kostenlose Variante bei Betrieb auf externem Server, Finanzierung durch Werbung
2. Mietlizenz bei Betrieb auf externem Server, 200.- € pro Jahr (für 20 Mitarbeiter)
3. Kauflizenz für Betrieb auf eigenem Server: ca. 500.- € (für 20 Mitarbeiter)

Diese flexiblen Zahlungsoptionen waren ein großer Pluspunkt für *Urlaub-Verwalten* und die kostenlose Variante vermutlich einer der Hauptgründe für die vergleichsweise hohen Nutzerzahlen. *Urlaub-Verwalten* kann so im kostenlosen Betrieb erprobt werden und später, falls eine Unabhängigkeit vom Anbieter gewünscht wird, eine Kauflizenz für den Betrieb auf dem eigenen Server erworben werden. Für diesen Fall wird ein 3-monatiger, kostenloser Probebetrieb auf dem eigenen Server gestattet, bevor die Software bezahlt werden muss.

Der Funktionsumfang des Systems war sehr umfangreich, die Bedienbarkeit intuitiv und die Nutzung stärker als die meisten anderen untersuchten Systeme auf die Urlaubsverwaltung fokussiert. Die Dokumentation auf der Website war sehr gut und ausführlich. Eine Exportfunktion nach Google Calendar war möglich, jedoch nur für Urlaube einzelner Mitarbeiter und nicht für mehrere Mitarbeiter gleichzeitig.

Ein größerer Kritikpunkt war jedoch, dass das System nur eingeschränkt mit verschiedenen Arbeitszeitmodellen umgehen konnte. Es konnten zwar beliebige Kombinationen von Wochenarbeitstagen ausgewählt werden, jedoch keine halben Arbeitstage erfasst werden. Auch konnte bei der Ablehnung eines Urlaubsantrags kein Ablehnungsgrund eingetragen werden, was sehr wünschenswert gewesen wäre. Bei der Festlegung der Zuständigkeit für Urlaubsanträge zeigte sich das Problem, dass jedem Mitarbeiter nur ein einziger Vorgesetzter zugewiesen werden konnte, der diese Urlaubsanträge bearbeitet. Dadurch war es nicht möglich, zwei oder mehr Personen über die Urlaubsanträge eines Mitarbeiters entscheiden zu lassen. Dies wäre allerdings von Seiten der in2code GmbH sehr erwünscht, um bei zeitweiser Abwesenheit eines Antragsverwalters einen Antragsstau zu verhindern.

Gerade bei einem kostenlos angebotenen System drängen sich Fragen zu Datenschutz und Datensicherheit auf. Laut Auskunft auf der Website gibt es keine Weitergabe der Daten an Dritte und es werden tägliche Datensicherungen durchgeführt. Allerdings gab es keine weiteren Informationen über Sitz, Infrastruktur und Verfügbarkeit des verwendeten Rechenzentrums / Servers, so dass hier im Bedarfsfall noch genauere Informationen eingeholt werden müssten.

Wie auch bei den vorigen Systemen war die verwendete Desktop-Optik nicht besonders ansprechend. Zwar wurde damit geworben, dass ein Corporate Design eingerichtet werden könnte, was sich jedoch auf das Hochladen eines kleinen Logos und die Auswahl der Farbe für den Titelbalken beschränkte. Es muss allerdings ergänzt werden, dass das Layout des Systems im Januar 2015, also nach Abschluss der Produkt-Vergleichsphase, grundlegend modernisiert und verbessert wurde.

### *Timetape*

Leistungsumfang:

* individuelle Arbeitszeitmodelle möglich
* Urlaubsberechnung auf halbe Tage genau
* Jahresübergreifende Berechnung des Resturlaubs
* Urlaubsanspruch für versch. Jahre konfigurierbar
* Urlaubsanträge mit intuitivem Freigabeworkflow
* Mehrere Antragsverwalter für Urlaubsanträge pro Abteilung möglich
* Verwaltung von Abteilungen / Standorten
* Kommentare zu Urlaubsantrag können zwischen Antragsteller / Antragsverwalter hin und her geschickt werden
* E-Mail-Benachrichtigungen
* Urlaubseintrag direkt durch Verwalter oder berechtigten Mitarbeiter möglich
* automatische Kollisionsprüfung mit angegebenem Vertreter
* Kollisionsprüfung bei sich überschneidenden eigenen Urlauben
* Urlaubssperren möglich
* Übersichtliche Kalender für einzelne Mitarbeiter / Abteilungen
* Exportfunktion für Google Calendar
* Berücksichtigung bundeslandspezifischer Feiertage
* Feiertage/freie Tage importier-/konfigurierbar
* 3 Berechtigungsstufen für Mitarbeiter (Administration, Antragsverwalter, Mitarbeiter)
* Excel-Exportfunktion
* Verwaltung von anderen Abwesenheitsarten (z.B. Fortbildung, HomeOffice,…) möglich
* Datensicherheit: SSL, Rechenzentrum in Deutschland (Gunzenhausen)

Bewertung:

*Timetape* hob sich positiv von den Mitbewerbern durch seine gelungene, moderne Optik und ausgezeichnete, intuitive Benutzbarkeit ab. Hier war offensichtlich von Anfang an der Fokus auf ein System gelegt worden, das ohne große Einweisung von den Mitarbeitern sofort genutzt werden kann und auch gerne genutzt wird. Dies zeigte sich z.B. auch an der Möglichkeit, ein eigenes Profilbild hochzuladen, was der Oberfläche eine persönliche Note verlieh (Abbildung 7).

Auch die klare Fokussierung auf die Verwaltung von Urlaub (und anderen Abwesenheiten) war in unserem Fall sehr positiv, es gab keine Kompromisse aufgrund der Integration mit anderen Modulen, wie einem Zeiterfassungssystem.

Der Funktionsumfang und das Genehmigungskonzept war dabei nahezu perfekt auf die Anforderungen der in2code GmbH zugeschnitten: die Teilzeitmodelle der Mitarbeiter konnten genau abgebildet werden und der Resturlaub wurde, wie gewünscht, auf halbe Tage genau berechnet. Sehr hilfreich waren auch die Möglichkeit, einen Vertreter für die Urlaubszeit benennen zu können, sowie die automatische Kollisionsprüfung mit dem Vertreter bzw. sich überschneidenden eigenen Urlaubseinträgen. Insbesondere die Benennung eines Urlaubsvertreters war eine sinnvolle Zusatzfunktion. Sie könnte dabei helfen, die Abläufe in der Firma zu verbessern, indem die Zuständigkeiten für ein Projekt vor Urlaubsantritt geklärt werden und das Projekt rechtzeitig übergeben werden kann. Wünschenswert wäre hier noch eine E-Mail-Benachrichtigung an den Urlaubsvertreter gewesen. Zum Testzeitpunkt wies das System lediglich darauf hin, den Vertreter über seine Aufgabe zu informieren. Vorausgreifend sei hier erwähnt, dass diese zusätzliche Funktionalität nach Mitteilung unseres Wunsches vom Entwickler innerhalb kurzer Zeit umgesetzt wurde (siehe Abschnitt 6.1).

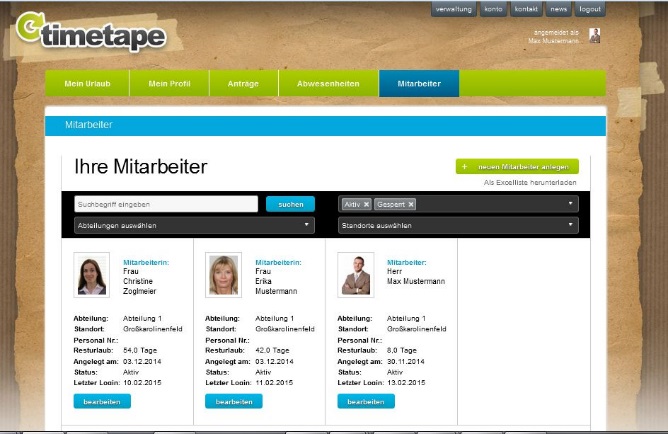


Abbildung 7 Mitarbeiteransicht in Timetape

Sehr positiv war auch, dass mehr als ein Antragsverwalter für jede Abteilung benannt werden konnte. Da somit mehrere Antragsverwalter die Urlaube der Mitarbeiter genehmigen können wurde das Risiko für Flaschenhälse und ein Anstauen unbearbeiteter Anträge vermindert.

Die Kalenderansichten für einzelne Mitarbeiter und die Abteilungen waren sehr übersichtlich gestaltet (Abbildung 8). Urlaubsüberschneidungen wären darauf unmittelbar ersichtlich.

Auch die Anbindung des Abteilungs- und Firmenkalenders an Google Calendar wurde bereits zur Verfügung gestellt. Zusätzlich gab es für die Administratoren eine Excel-Exportfunktion für Listen der Mitarbeiter mit Zusammenfassungen ihrer Urlaubs- bzw. Abwesenheitstage und Urlaubsanträge.

Die bereit gestellten Informationen zu den Themen Datensicherheit und Datenschutz überzeugten ebenfalls [15]: der Dienst wird in einem professionellen deutschen Rechenzentrum (Hetzner online AG, Nürnberg/Falkenstein) betrieben, täglich werden mehrere Backups auf externen Servern durchgeführt, die Daten werden gespiegelt und die Ausfallssicherheit durch vielfach redundante Upstreams erhöht. Passwörter werden verschlüsselt gespeichert und die Daten nur SSL-verschlüsselt übertragen. Der Inhalt der Datenschutzerklärung war abgedruckt [16], worin die Einhaltung aller gesetzlichen Richtlinien zugesichert wurde. Darin wurde sogar auf die Problematik von eingebundenen Facebook-Buttons hingewiesen und es wurden Vorschläge gemacht, wie der Anwender Datenerhebungen durch Facebook verhindern könnte was doch auf ein erhöhtes Maß an Verantwortungsbewusstsein des Anbieters in Sachen Datenschutz schließen ließ.

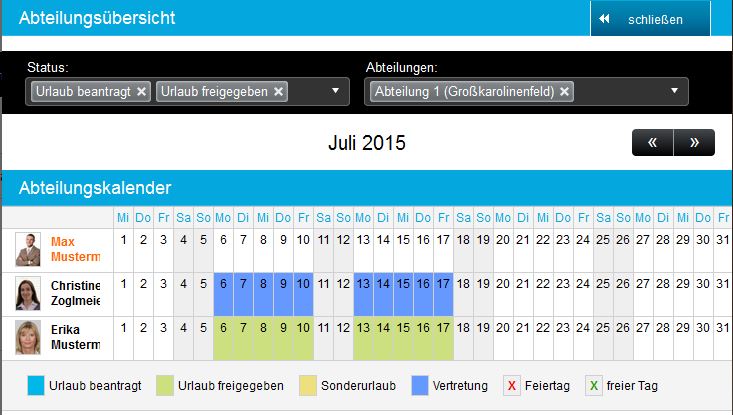


Abbildung 8 Abteilungskalenderansicht in Timetape

Insgesamt stellten sowohl *Urlaub-Verwalten* als auch *Timetape* sehr interessante Alternativen zur Eigenentwicklung eines Urlaubsplaners dar. Zusammen mit der Geschäftsführung wurde deshalb im Anschluss an die Marktanalysephase eine „Make-or-Buy“-Überlegung mit besonderem Fokus auf diesen Drittanbietern durchgeführt.

## „Make-or-Buy“ –Entscheidung

Bei der Entscheidung zwischen dem Kauf einer Standardsoftware und der Eigenentwicklung eines entsprechenden Systems, gab es unter anderem folgende Vor- bzw. Nachteile zu überdenken [17]:

Eine Eigenentwicklung ermöglicht den Zuschnitt auf firmenspezifische Bedürfnisse, Hoheit über die eigenen Daten, Unabhängigkeit von externen Dienstleistern, meist bessere Wartbarkeit und Adaptierbarkeit sowie einen möglichen Zugewinn an Technologiewissen für die eigenen Entwickler. Auch kann das ursprünglich für eigene Zwecke entwickelte Produkt ggf. kommerzialisiert oder zumindest das erworbene Technologiewissen in Drittprojekten eingesetzt werden.

Die Entscheidung für eine eigene Implementierung birgt jedoch die Gefahr, dass deutlich mehr Zeit als ursprünglich vorgesehen für die Entwicklung eines funktionierenden Systems aufgewendet wird und die Entwicklungskosten, die in der Anfangsphase eines Projekts nur sehr schwer abschätzbar sind, unerwartet hoch ausfallen. Da jede Software Fehler enthält und unvollständig ist, muss vor allem in der Anfangszeit mit erheblichem Zeitaufwand der Mitarbeiter für Fehlerbehebung und Anpassungen gerechnet werden. In einem markteingeführten Standardprodukt sind dagegen schon viele Anfangsfehler behoben, das Produkt ist meist ausgereifter, von höherer Qualität und kann sofort, ohne Zeitverlust eingesetzt werden.

Diese Aspekte wurden im Oktober 2014 in einem Treffen mit der Geschäftsführung der in2code GmbH und den Auszubildenden, die das Lastenheft [14] für die Eigenimplementierung vorbereitet hatten, diskutiert. Als Grundlage für die Diskussion wurden die Ergebnisse der Aufwandsschätzung für die Eigenentwicklung und der Analyse der Drittanbieter der in2code GmbH in einer Präsentation vorgestellt. Für die vorausgewählten Systeme wurde eine Live-Demonstration vorgeführt.

Ein Vergleich der vier Systeme *Personal-Planer*, *Saas.de* *Urlaubsverwaltung*, *Urlaub-Verwalten* und *Timetape* ist in Tabelle 4 zusammen gefasst. Darin ist angegeben, in wie weit die vier Systeme die in Abschnitt 2.3.1 gestellten Anforderungen erfüllen konnten. Zusätzlich zu den Anforderungen wurden die Kriterien Usability und Optik bewertet.

Tabelle 4 Vergleich der ausgewählten Urlaubsverwaltungssysteme

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Anforderung /Kriterium | *Personal-Planer* | *Saas.de* | *Urlaub-Verwalten* | *Timetape* |
| A1 Resturlaubsberechnung | + | + | + | + |
| A2 Arbeitszeitmodelle | + | + | o | + |
| A3 Freigabeworkflow | + | o | o | + |
| A4 Kalenderansichten Firma | + | + | + | + |
| A5 Kalenderansichten einzelner  Mitarbeiter | + | + | + | + |
| A6 Webapplikation | + | + | + | + |
| A7 Übersicht Resturlaub | + | + | + | + |
| A8 Login | + | + | + | + |
| A9 Google Calendar Firma | - | + | - | + |
| B1 Teamkalender | + | + | + | + |
| B2 Vertreter | + | - | + | + |
| B3 Abgelehnte Anträge | + | - | + | + |
| B4 Datensicherung | - | - | - | - |
| B5 Anbindung Mite | - | - | - | - |
| C1 Bericht Personalbuchhaltung | - | - | - | - |
| C2 Google Calendar Mitarbeiter | - | + | + | - |
| C3 Google Calendar Vertreter | - | - | - | - |
| Usability | Zu komplex | Gut | Gut | Sehr gut |
| Optik | Gut | Akzeptabel | Akzeptabel | Sehr gut |
| Preis | 400.- | 200.- | 0 bzw. 200.- | 200.- |

Zeichenerklärung: **-** nicht erfüllt **o** teilweise erfüllt **+** erfüllt

*Timetape* kristallisierte sich dabei als klarer Favorit heraus. Als einziges System konnte es alle Anforderungen mit hoher Priorität erfüllen. Der Preis, die moderne Optik, gute Bedienbarkeit und der nahezu ideal zugeschnittene Funktionsumfang überzeugten ebenfalls.

Beim folgenden Entscheidungsprozess, der sich nur noch auf die beiden Optionen „Kauf von *Timetape*“ oder „Eigenimplementierung“ konzentrierte, wurden folgende Argumente diskutiert und abgewogen:

1. **Anforderungen/Funktionsumfang:** *Timetape* erfüllte alle Anforderungen (siehe Abschnitt 2.3.1) mit hoher Priorität. Zusätzlich bot es eine Reihe von weiteren Funktionen und eine ausgereiftere Logik und Bedienbarkeit, als vermutlich in einer mit knappem Zeitbudget entstandenen Eigenentwicklung hätte erreicht werden können. Offen blieben von den Anforderungen mit mittlerer Priorität lediglich B5, die Anbindung an das Zeiterfassungssystem *Mite*, sowie B4, die unabhängige Datensicherung. Beide Zusatzfunktionen sind für eine funktionierende Urlaubsverwaltung jedoch zunächst nicht zwingend notwendig und müssten in jedem Fall selbst implementiert werden. Dasselbe galt auch für die niedrig priorisierte Anforderung C1, den automatischen Bericht an die Personalbuchhaltung. Auf die Anforderungen C2 und C3, angepasste Google-Kalenderformate für die einzelnen Mitarbeiter, könnte auf jeden Fall vorerst verzichtet werden.
2. **Erweiterbarkeit / Anpassbarkeit:** Bei der Auswahl des Systems eines Drittanbieters ist es nur mit längerer Vorlaufzeit und in geringerem Umfang möglich, zusätzliche Funktionen bzw. Anpassungen zu erreichen, als bei einem eigenen System. Insgesamt waren jedoch die aktuell angebotenen Funktionen von *Timetape* vollkommen ausreichend. Die Mitteilungen im „News“-Bereich zeigten, dass der Entwickler das System ständig weiter entwickelt und regelmäßig neue Funktionalitäten zur Verfügung stellt. So wird z.B. bereits daran gearbeitet, Apps für Android und iOS zur Verfügung zu stellen. Die spätere Erfahrung zeigte dann auch, dass Anpassungswünsche bereitwillig und rasch vom Entwickler umgesetzt wurden (siehe Abschnitt 6.1) und das Produkt kontinuierlich optimiert wurde.
3. **Zeitpunkt der Inbetriebnahme:** Bis eine Eigenentwicklung in Betrieb gehen könnte, würden voraussichtlich mehrere Monate vergehen. *Timetape* könnte dagegen sofort eingerichtet und genutzt werden, so dass z.B. die kritische Zeit der Weihnachtsurlaube bereits durch das System abgedeckt werden könnte.
4. **Datensicherheit:** Bei den Mitarbeiterdaten handelt es sich um persönlichkeitsbezogene Daten, bei den Daten zu Urlauben/Abwesenheiten sogar um Informationen, die theoretisch missbraucht werden könnten, z.B. um urlaubsbedingte Abwesenheiten heraus zu finden. Deshalb war das Vertrauen in eine sichere Handhabung der Daten durch den Anbieter ein wichtiger Gesichtspunkt. Der Anbieter konnte jedoch durch seine Präsentation des Themas Datenschutz auf der Website von seiner Sorgfalt überzeugen. Spätere persönliche Gespräche mit dem Anbieter bestätigten den Eindruck dieses Verantwortungsbewusstseins. Auch bei einer Eigenimplementierung wären die persönlichkeitsbezogenen Daten im Übrigen extern gespeichert, da die in2code GmbH keine eigenen Server betreibt, sondern Cloud-basierte Serverdienste nutzt (z.B. Domain Factory [18]). Auch hier müsste darauf vertraut werden, dass der Dienstleister die Datenschutzrichtlinien lückenlos einhält. Zuletzt sei noch darauf hingewiesen, dass die Mitarbeiter sich damit einverstanden erklärt haben, dass ihre Urlaube bereits jetzt im firmeninternen Google-Kalender erfasst werden, mit allen Risiken, die eine Nutzung dieses Google-Dienstes beinhaltet [19].
5. **Verfügbarkeit:** Es werden zwar durch den Anbieter keine konkreten Zusagen zur Verfügbarkeit des Dienstes gemacht, das betreibende Rechenzentrum [20] arbeitet jedoch nach modernsten Standards. Es gibt keinen Grund zur Annahme, dass der Dienst hier seltener verfügbar wäre, als bei einer Eigenimplementierung und Hosting durch den eigenen Anbieter [18]. Eine zeitweise Nicht-Verfügbarkeit der Urlaubsverwaltung hätte im Übrigen keine unmittelbaren Auswirkungen auf das eigentliche Kerngeschäft der Firma, dieses Risiko war also in Kauf zu nehmen.
6. **Urlaubsplaner als eigenes Produkt:** ein Grund für die Entscheidung zur Eigenentwicklung ist bei einigen Firmen das Kalkül, das dabei entstandene Produkt selbst vermarkten zu können. Diese Möglichkeit wäre bei der in2code GmbH kaum gegeben, da es sich bei den Kunden hauptsächlich um Firmen aus dem Enterprise-Sektor bzw. öffentliche Einrichtungen handelt, die bereits die entsprechenden professionellen Lösungen bzw. Module aus ERP-Systemen nutzen. Zwar würde die Eigenentwicklung mit Sicherheit einen zusätzlichen Wissens- und Technologiegewinn für die beteiligten Entwickler bedeuten, allerdings ist auch zu befürchten, dass ein solches, vergleichsweise wenig unternehmensbedeutsames Projekt wichtige Entwicklerkapazitäten bindet und die Erledigung von Aufgaben mit höherer Priorität behindert.
7. **Kosten/Aufwand:** die optimistisch geschätzten Kosten für eine Eigenentwicklung beliefen sich auf ca. 7000.- € Euro. Nicht eingerechnet waren hier längerfristige Kosten wie Wartungsaufwand, Kosten für den Serverbetrieb etc. Dem gegenüber standen jährliche Kosten von knapp 300.- Euro für die Nutzung von *Timetape* bei 20 Mitarbeitern. Pro zusätzlichem Mitarbeiter wäre mit ca. 20 € Mehrkosten pro Jahr zu rechnen. Da die Firma nicht beabsichtigt, ihre Mitarbeiterzahlen in Zukunft stark zu erhöhen, waren hier keine explodierenden Kosten zu erwarten. In den jährlichen Nutzungskosten von *Timetape* wären bereits alle Nebenkosten, wie Wartung, Support und Erweiterung des Systems durch den Anbieter, enthalten, so dass im Gegensatz zur Eigenentwicklung keine versteckten oder unerwarteten Mehrkosten zu erwarten sind.

Aufgrund der aufgeführten Überlegungen fiel für alle Anwesenden eindeutig die „Make-or-Buy“-Entscheidung zu Gunsten des Erwerbs von *Timetape* als zeit- und kostengünstigere Alternative zur Entwicklung eines eigenen Systems. Alle dringend benötigten Funktionen könnten damit sofort abgedeckt werden. Weitere gewünschte, aber nicht unbedingt notwendige Funktionen sollten, soweit es der zeitliche Rahmen zuließ, im Rahmen dieser Bachelorarbeit implementiert werden, so dass schlussendlich eine „Buy-and-Make“-Lösung zum Tragen kommen sollte.

Vorab sollte jedoch geklärt werden, ob *Timetape* eine Schnittstelle anbieten kann, über die Mitarbeiter- und Urlaubsdaten abgerufen werden können. Eine solche Schnittstelle wäre Voraussetzung für die Anbindung von *Timetape* an *Mite* und für alle weiteren Zusatzfunktionen wie die unabhängige Datensicherung oder den Bericht an die Personalbuchhaltung. Erfreulicher Weise wurde diese Anfrage unverzüglich positiv beantwortet. Obwohl die Schnittstelle sich noch im Teststadium befand, richtete der Entwickler für den Zugang der in2code GmbH und für das in der Entwicklungsphase genutzte Testsystem (siehe Abschnitt 5.1) Schnittstellen mit API-Keys ein, über die die wichtigsten Daten des Systems im XML-Format abgerufen werden konnten. Damit konnte die Entscheidung für *Timetape,* ergänzt durch selbst implementierte Zusatzfunktionen, endgültig getroffen werden.

# Einrichtung von *Timetape*

## Konfiguration und Anlegen der Stammdaten

Nach der Entscheidung für *Timetape* Ende Oktober 2014 wurde sofort mit der Einrichtung des Systems begonnen. Dazu wurden zunächst im Verwaltungsbereich die benötigten Konfigurationen vorgenommen und Stammdaten angelegt. In Abbildung 9 ist die Verwaltungsansicht für eingeloggte Administratoren in *Timetape* dargestellt. Da das System schlank und übersichtlich aufgebaut war, konnten die erforderlichen Einstellungen rasch vorgenommen werden.

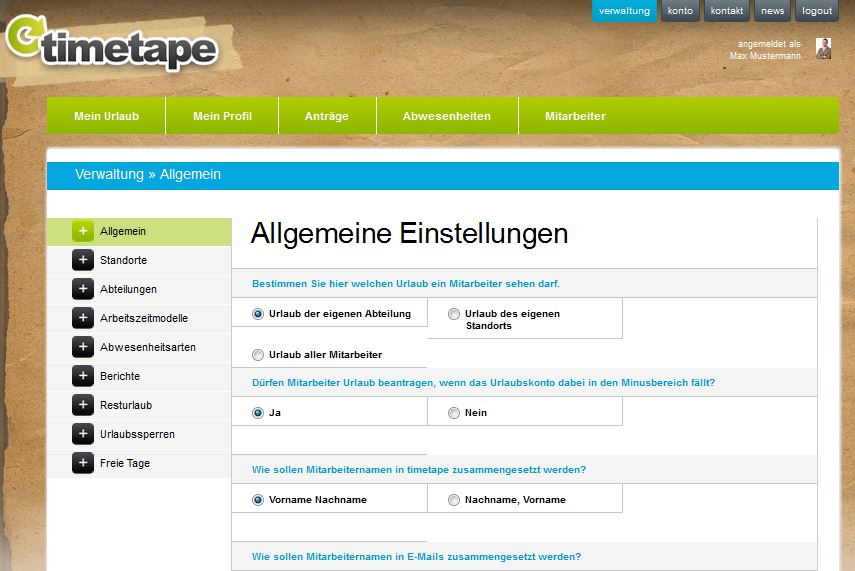


Abbildung 9 Verwaltungsbereich von *Timetape*

Die wichtigsten vorgenommenen Einstellungen sind im Folgenden zusammen gefasst:

**Allgemein**: Mitarbeiter dürfen die Urlaubseinträge der anderen Mitarbeiter sehen

**Standorte:** Unter diesem Menüpunkt können verschiedene Firmenstandorte mit ggf. abweichenden Feiertagsregelungen verwaltet werden. Die in2code GmbH besitzt nur einen Standort in Rosenheim. Für diesen Standort wurde der bayerische Feiertagskalender ausgewählt, der bei Bedarf auch noch editiert werden kann.

**Abteilungen:** Die *Timetape* vorgesehenen Abteilungen entsprechen bei der in2code GmbH den Projektteams. Hier wurden die 5 aktuell bestehenden Projektteams angelegt. Jedem Projektteam wurden als Antragsverwalter die beiden Geschäftsführer zugeordnet.

**Arbeitszeitmodelle:** In diesem wichtigen Bereich wurden die aktuell genutzten Arbeitszeitmodelle der Mitarbeiter der in2code GmbH angelegt. Hier können die Arbeitstage, die Arbeitsstunden für jeden Tag und eine etwaige Mittagspause eingetragen werden. Die täglich vorgesehenen Arbeitszeiten für jeden Mitarbeiter werden für die Berechnung des Resturlaubs berücksichtigt, wobei der Urlaub jedoch auf 0,5 Tage genau auf- bzw. abgerundet wird, was auch den Wünschen der Geschäftsführung nach einer handhabbaren Abrechnung entspricht. Abbildung 10 zeigt eine Ansicht mit drei beispielhaften Arbeitszeitmodellen. Für die Mittagspause wurde bei allen Arbeitszeitmodellen 0 Stunden eingetragen, da eine eingetragene Mittagspause vom System nicht bei den Resturlaubs- und Arbeitszeitberechnungen abgezogen wird und die Erfassung der Pausenzeiten für die Firma an dieser Stelle nicht relevant ist.

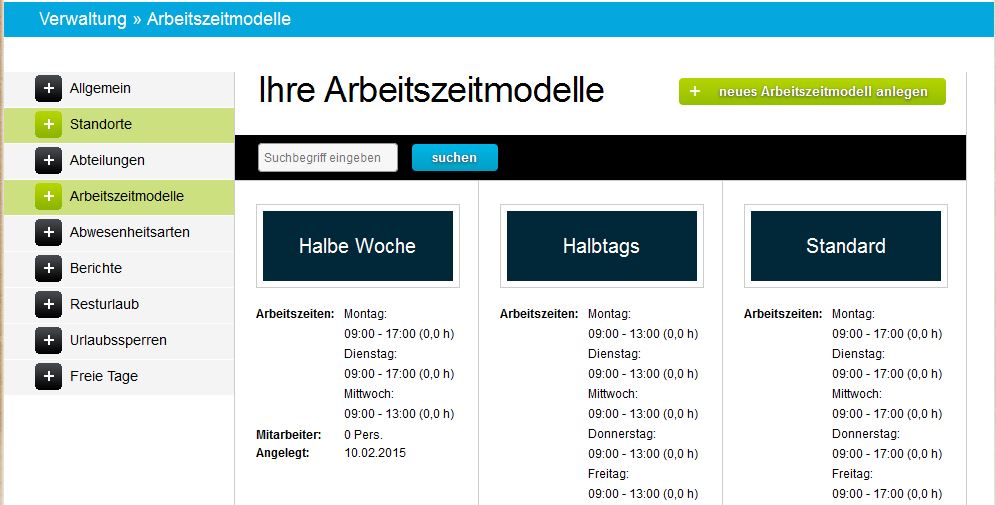


Abbildung 10 Anlegen von Arbeitszeitmodellen in Timetape

**Abwesenheitsarten:** *Timetape* bietet neben der Urlaubsverwaltung auch die Möglichkeit, eine große Zahl von weiteren Abwesenheitsgründe anzugeben, die in den Kalenderansichten angezeigt werden können. Im Verwaltungsbereich kann konfiguriert werden, welche der Abwesenheitsarten den Mitarbeitern zur Erfassung angeboten werden. Hier wurde von den 28 angebotenen Abwesenheitsarten vorerst nur die Gründe „Dienstreise“, „Homeoffice“, „Berufsschule“ und „Überstundenabbau“ aktiviert.

**Berichte:** Unter diesem Menüpunkt kann eine nach Monaten gruppierte Übersicht der genehmigten Urlaubstage aller Mitarbeiter im Excel-Format generiert werden.

**Resturlaub:** Laut aktueller Regelung verfällt der Resturlaub der Vorjahre nicht. Unter diesem Punkt könnten anders lautende Regelungen, z.B. ein Verfall des Resturlaubs zum 31.03. jedes Jahres, festgelegt werden.

**Urlaubssperren:** Hier können auf einzelne Abteilungen beschränkte Urlaubssperren angelegt werden. Vorerst wurde keine Urlaubssperre angelegt.

**Freie Tage:** Zusätzlich zu den gesetzlichen Feiertagen können hier von der Firma weitere freie Tage festgelegt werden, für die kein Urlaubstag genommen werden muss. Hier wurde für Sylvester und Heilig Abend jeweils ein halber freier Tag angelegt.

Nachdem alle Konfigurationen vorgenommen waren, wurden alle aktiven Mitarbeiter angelegt und ihnen die zutreffenden Arbeitszeitmodelle, Projektgruppen und Standorte zugeordnet. Vom System erhielt jeder Mitarbeiter daraufhin seine Zugangsinformationen und einen Link zum Ändern des Passworts.

## Mitarbeiterschulung

*Timetape* wurde unter anderem wegen seiner intuitiven Bedienbarkeit und seines selbsterklärenden Antrags-Workflows ausgewählt. Deshalb wäre eigentlich keine zusätzliche Einweisung und Schulung der Mitarbeiter notwendig gewesen. Zudem sind die Mitarbeiter der in2code GmbH durchwegs versiert in der Nutzung von elektronischen Systemen und können sich rasch in neue Werkzeuge einarbeiten. Dennoch wurde ein Schulungstermin für alle Mitarbeiter angesetzt, um alle über das neue System zu informieren und firmeninterne „Best Practices“ zur Nutzung des Tools zu anzusprechen.

Insbesondere wurde hier die von *Timetape* gebotene Möglichkeit besprochen, beim Stellen eines Urlaubsantrags einen Urlaubsvertreter festzulegen. Dieses Feld muss bei Antragstellung vom System aus nicht verpflichtend ausgefüllt werden, ein Mitarbeiter sollte jedoch nach Möglichkeit stets einen Vertreter benennen, dem er seine laufenden Projekte vor Urlaubsantritt angemessen übergeben kann. Wenn ein Vertreter nicht während des gesamten Urlaubszeitraums verfügbar wäre, sollte der in geeignete Teilanträge mit unterschiedlichen Vertretern aufgeteilt werden. Die Vertreterregelung wurde am längsten diskutiert, da es hier z.T. Vorbehalte gegenüber der praktischen Durchsetzbarkeit gab. Es blieb nur der Appell an die Mitarbeiter, die von *Timetape* gebotene Unterstützung zu nutzen und sich rechtzeitig abzusprechen, um die Projekte bei Abwesenheit gut betreut zu wissen.

Mit der Umstellung auf *Timetape* änderte sich für die Mitarbeiter auch der Stichtag für die Abrechnung ihres Resturlaubs. Während bisher die Urlaubsansprüche aufgrund des abweichenden Geschäftsjahres der Firma zum 30. Juni berechnet wurden, wurde nun auf die vom Tool unterstützte, allgemein übliche Abrechnung nach Kalenderjahren umgestellt. Auch über diesen Sachverhalt mussten die Mitarbeiter informiert werden.

Eine Zusammenfassung der Schulung wurde an alle Mitarbeiter verschickt, damit auch abwesende Mitarbeiter informiert werden und alle die wichtigsten Punkte nachgelesen können.

Ab Ende November 2014 war *Timetape* bei der in2code GmbH im laufenden Betrieb und alle Urlaubsanträge wurden nur noch auf diesem Wege gestellt.

# Implementierung eigener Zusatzfunktionen

*Timetape* erfüllte bereits alle Anforderungen, die für eine reibungslose Verwaltung der Urlaube benötigt waren. Deshalb war der Bereich Urlaubsverwaltung mit der „Buy“-Entscheidung vollständig abgedeckt. Weniger dringende, aber dennoch wünschenswerte Funktionen waren die Anbindung an das Zeiterfassungssystem, eine unabhängige Datensicherung und ein automatischer Monatsbericht an die Personalbuchhaltung. Diese Funktionen wurden im Rahmen der Bachelorarbeit zusätzlich selbst implementiert, also das gekaufte System um eine „Make“-Komponente ergänzt. Die entwickelte Anwendung mit ihrer Sammlung von Zusatzfunktionen zu *Timetape* wurde *in2timetape* benannt.

## Technologien und Entwicklungsumgebung

Technologieentscheidungen wurden meist im Hinblick darauf getroffen, dass die Anwendung im Anschluss von der in2code GmbH möglichst gut weiter entwickelt werden kann. Daher sollte, wo möglich, auf vorhandenes Technologiewissen aufgebaut werden und bei der Auswahl neuer Technologien auf gut wartbare Standard-Technologien gesetzt werden.

Basistechnologie für die Programmierung der Zusatzfunktionen war das PHP-Framework *Typo3 Flow* [9], das zum Technologiestandard der in2code GmbH gehört. Die neue Anwendung *in2timetape* wurde demzufolge als ein Package entwickelt, das der vorgegebenen MVC-Ordnerstruktur und den Namenskonventionen von *Typo3 Flow* entspricht und sich im Ordner Packages/Application des Frameworks befindet.

Als Entwicklungsumgebung wurde die ebenfalls von allen Firmenmitarbeitern genutzte IDE *PHPStorm* [21] verwendet. Zur Code-Versionierung wurde *Git* [22] eingesetzt und als Code-Basis ein privates Repository auf der kostenlosen Plattform *Bitbucket* [23] angelegt. Dieses Repository kann später direkt in die Code-Bibliothek der in2code GmbH kloniert werden.

Zur Verwaltung der Packages wurde der Paketmanager *Composer* [24] eingesetzt. Zum Datentransfer auf den Server der in2code GmbH wurde der kostenlose FTP-Client *FileZilla* [25] benutzt.

Für die Mail-Benachrichtigungen wurde das Paket *Typo3 Flow.Swiftmailer* [26] eingesetzt.

Für die Funktionen Autorisierung und Authentifizierung wurden Klassen aus den Packages *Typo3 Flow.Login* [27] und *Typo3 Flow.AccountManagement* [28] verwendet. Da beide Packages jedoch seit 2013 nicht weiter entwickelt wurden und nicht ganz ausgereift sind, wurden sie nicht über *Composer* eingebunden. Stattdessen wurden die benötigten Klassen ausgewählt, modifiziert und direkt in das eigene Anwendungspaket eingebunden. Komfortfunktionen für die REST-Aufrufe an *Mite* lieferte die PHP-Bibliothek *Mite.php* von Thomas Klein [29], die auf der API-Dokumentationsseite von *Mite* empfohlen wurde [30].

Die Validierung von Benutzereingaben im Front-End wurde mit Hilfe der Javascript-Bibliothek *Parsley.js* [31] durchgeführt.

Die XSD-Schemadateien wurden mit einem Online-Tool von *Freeformatter.com* [32] erzeugt und anschließend überarbeitet. Für die Validierung von XML-Dateien gegen die Schemadatei wurde das Validierungstool von *online-utilities* [33] verwendet.

Da die Anbindung von *Timetape* an *Mite* Datenzugriffe auf zwei Live-Systeme erforderte, durch die im schlimmsten Fall Daten gelöscht oder manipuliert werden könnten, wurden in der Entwicklungsphase für beide Tools Test-Systeme eingesetzt. Für *Timetape* konnte ohne Beantragung direkt ein kostenloses System mit drei fiktiven Mitarbeitern angelegt werden, für *Mite* wurde vom Anbieter ein Testsystem auf Anfrage kostenlos zur Verfügung gestellt.

Die Anwendung wurde zunächst lokal unter Verwendung von XAMPP 1.8.3 [34] entwickelt und erst nach Fertigstellung auf einen Server der in2code GmbH deployed (siehe Abschnitt 5.6).

## Anbindung an das Zeiterfassungssystem *Mite*

### Implementierungsansatz

Die Anbindung von *Timetape* an *Mite* war die wichtigste, selbst zu implementierende Zusatzfunktion. Dabei sollten die in *Timetape* eingetragenen, genehmigten Urlaubstage sowie erfassten Feiertage mit der für den jeweiligen Mitarbeiter an diesem Tag gültigen Stundenzahl in das Zeiterfassungssystem *Mite* übertragen werden. Da sich der Status eines Urlaubsantrags noch bis zum letzten Tag ändern kann, sollte die Übertragung nur einmalig am Abend des jeweiligen Tages vorgenommen werden, wenn der Urlaubstag also sicher angetreten wurde. Dafür bot sich ein zeitgesteuerter Aufruf der Anwendung (Cronjob) auf einem Server der in2code GmbH an. *Typo3 Flow* bietet für die Erstellung von automatisierbaren Befehlen die Verwendung eines CommandControllers [35] an. Innerhalb dieses Controllers kann eine beliebige Anwendungslogik, wie in einem normalen ActionController, implementiert werden, mit dem Unterschied, dass die Funktionen nicht über einen HTTP-Request im Browser, sondern über einen Kommandozeilenaufruf gestartet werden. Dadurch können einfach eigene Befehle mit beliebig komplexer Logik entwickelt werden, die anschließend manuell oder automatisiert aufgerufen werden können.

Den Benennungsregeln von *Typo3 Flow* folgend wurde Anwendungslogik deshalb in eines CommandControllers mit der Bezeichnung TimetapeCommandController und dort in einer Funktion writeToMiteCommand() umgesetzt. Die Anwendung kann so durch folgenden einfachen Aufruf gestartet werden:

./flow timetape:writetomite

Durch diesen Aufruf ohne weiteren Parameter werden die Daten für den aktuellen Tag aus *Timetape* ausgelesen und in *Mite* eingetragen. Damit die Datenübernahme auch für einen anderen als den aktuellen Tag möglich ist, wurde die Funktion so ergänzt, dass der Befehl optional auch mit einem Datumsparameter aufgerufen werden kann.

./flow timetape:writetomite --date 2014-12-24

Die Anwendung hatte dabei zumindest folgende Aufgaben erfüllen:

1. Einlesen und Validieren des Timetape-XML
2. Durchsuchen des Timetape-XML nach Urlaubseinträgen für einen bestimmten Tag
3. Zuordnung des *Timetape*-Mitarbeiters zu dem entsprechenden *Mite*-Mitarbeiter über die gemeinsame E-Mail-Adresse
4. Schreiben des Urlaubs- bzw. Feiertags-Eintrags mit der richtigen Minutenzahl für diesen Mitarbeiter in *Mite* mit Überschreiben vorhergehender Einträge aus *Timetape.*
5. Benachrichtigungen im Erfolgs- und Fehlerfall

Diese Aufgaben wurden in der Funktion writeToMiteCommand(), die allein den Programmablauf steuerte, hintereinander durch Aufruf verschiedener Unterfunktionen abgearbeitet. Auf die wichtigsten Schritte wird im Folgenden näher eingegangen.

### Einlesen und Validieren der Daten aus *Timetape*

Der erste Schritt und Voraussetzung für alle weiteren Zusatzfunktionen war das Einlesen der Daten aus *Timetape* über die gegebene XML-Schnittstelle. Dabei handelte es sich nicht um einen kompletten REST-Webservice, bei dem über verschiedene GET-Aufrufe mit den entsprechenden Parametern gezielt Daten abgefragt werden konnten, sondern nur um eine einzige URL, über die ein großes XML-File mit einer Zusammenfassung aller Daten abgerufen werden konnte. Eine benötigte Auswahl oder Sortierung der Daten musste im Anschluss an das Laden des XML selbst implementiert werden.

Da die Schnittstelle sich noch in der Entwicklungsphase befand und von *Timetape* noch nicht offiziell angeboten wurde, gab es noch keine ausführliche Dokumentation oder zugehörige Schemadatei, gegen die validiert werden könnte. Allerdings stellte der Entwickler von *Timetape* ein Dokument zur Verfügung [36], das die Grundstruktur der leeren XML-Datei und die zulässigen Werte für bestimmte Felder beschrieb, z.B. für den möglichen Status eines Urlaubsantrags oder zulässige Abwesenheitsarten.

Das XML enthielt eine Liste aller Mitarbeiter (mitarbeiterlist) mit ihren jeweiligen Personaldaten (siehe auch die Schemadatei Abbildung 11 ). Außerdem gab es für jeden Mitarbeiter eine Liste der nach Jahren gruppierten Urlaubsansprüche (urlaubsanspruchlist), eine Liste aller bisher gestellten Urlaubsanträge (urlaubsantraegelist), sowie Listen der Krankheitstage (krankheitstagelist) und erfassten Abwesenheiten (abwesenheitenlist). Besonders wichtig war die letzte bei jedem Mitarbeiter aufgeführte Liste, die Liste der Abwesenheitstage (tagelist). Hier waren für den aktuellen Monat alle einzelnen Abwesenheitstage des Mitarbeiters mit dem jeweiligen Abwesenheitsgrund (z.B. Feiertag, Wochenende, Urlaub,…) und der Uhrzeit der Abwesenheit aufgelistet. An dieser Stelle konnten stundengenau die anzurechnenden Urlaubs- bzw. Feiertagszeiten für jeden Mitarbeiter ausgelesen werden. Die Tageliste konnte übrigens als einziges Element in dem XML durch die Angabe der Parameter „von“ und „bis“ auf einen bestimmten Zeitraum eingegrenzt werden.

Das korrekte Einlesen der Daten war die Grundlage für alle weiteren Implementierungsschritte. Da die Logik der Datenverarbeitung untrennbar mit der vorgegebenen Syntax und Semantik der XML-Schnittstelle verknüpft werden musste, war es sehr wichtig, das eingelesene XML vor der Interpretation gegen ein geeignetes Schema zu validieren. Ansonsten könnte eine unbemerkte Änderung an der Schnittstelle, z.B. das Weglassen oder Umbenennen eines ausgelesenen Elements, zu Programmfehlern führen.

Da keine XSD-Schemadatei vorgegeben war, wurde mit Hilfe eines Online-Tools von *Formatter.com* selbst eine Schemadatei erzeugt (Abbildung 11).

Abbildung 11 Ausschnitt aus dem für die Timetape-Api erzeugten Schema, der Datei timetape.xsd.

Eine über die Schnittstelle abgerufene XML-Datei aus dem Live-Betrieb des Timetape-Accounts der in2code GmbH diente hierzu als Ausgangsbasis. Das automatisch erzeugte XSD musste jedoch noch an einigen Stellen manuell angepasst werden. So hatte z.B. der *XSD-Formatter* bei allen Elementen, bei denen im XML kleine ganzzahlige Werte auftraten, in der Schemadatei der Datentyp „byte“ eingesetzt. Da der für diesen Datentyp gültige Bereich von -128 bis 127 bei abweichenden Daten nicht ausreichen würde, wurde der Typ überall durch „integer“ ersetzt. Überall, wo im Ausgangs-XML leere Listen vorgekommen waren, war das erzeugte Schema unvollständig und musste noch um die inneren Elemente erweitert werden. Wo die Liste nur einen Eintrag enthalten hatte, mussten eingefügte Beschränkungen für die zulässige Zahl von Aufzählungselementen aufgehoben werden.

Außerdem wurden noch eigene Validierungs-Restriktionen eingebaut: an den Stellen, wo im Programmablauf aus dem XML Einträge mit einer bestimmten Bezeichnung ausgelesen werden, sollte sichergestellt werden, dass diese Bezeichnung nicht zwischenzeitlich verändert worden war. Deshalb wurde hier in der Schemadatei zusätzlich eine Liste aller erlaubten Einträge definiert (siehe Datei timetape.xsd im Quellcode).

Nach dem Laden des Timetape-XML wurde im Programmablauf gegen das erzeugte Schema validiert. Falls die Struktur des XML oder die Benennung von wichtigen Feldern vom Schema abwich, schlägt die Validierung fehl, das Programm wird beendet und eine Fehlermeldung per E-Mail verschickt. Ebenso wird ein Fehler gemeldet, wenn das Laden des XML nicht innerhalb einer vorgegebenen Timeout-Zeit erfolgreich abgeschlossen werden konnte, z.B. weil sich die Zugangsdaten geändert habe oder der Dienst nicht erreichbar ist.

### Durchlaufen des XML und Schreiben der Zeiteinträge nach Mite

Die weiteren, auf die wesentlichen Funktionen reduzierten Programmschritte für die Verknüpfung von *Timetape* und *Mite* sind in Abbildung 12 in Pseudocode beschrieben.

Nach der Validierung wurde das XML in ein Objekt der Hilfsklasse SimpleXMLElement [37] umgewandelt, die komfortabel in Schleifen durchlaufen und auf bestimmte Attributwerte hin geprüft werden kann.

Zunächst wurde die Mitarbeiterliste durchlaufen und überprüft, ob für den angegebenen Tag bereits Einträge in *Mite* aus *in2timetape* vorliegen, was anhand der eingetragenen Kommentare erkannt wird. So können fehlerhafte oder veraltete Einträge überschrieben werden bzw. die Funktion zu Testzwecken mehrmals hintereinander aufgerufen werden, ohne dass in *Mite* redundante Einträge produziert werden.

Anschließend wurde die Zeitenliste für jeden Mitarbeiter durchlaufen. Wurden dabei geeignete Urlaub- bzw. Feiertagseinträge gefunden, wurden sie mit Hilfe der vorab für mit den Benutzerdaten des *Mite*-Accounts der in2code initialisierten Hilfsklasse *Mite.php* [29] nach *Mite* geschrieben. In *Mite* werden Zeiten stets für gegebene Projekte erfasst und die Zeiten bestimmten Service-Tätigkeiten zugeordnet. Entsprechend dieser Logik werden auch Abwesenheitszeiten erfasst. Dafür musste die ID bekannt sein, mit der im *Mite*-Account der in2code GmbH Projekte mit dem Namen „Urlaub“ bzw. „Feiertag“ und der Service mit der Bezeichnung „Abwesenheit“ verwaltet wird. Um die Anwendung flexibel und wartbar zu halten, wurden die Projekt- und Service-IDs von *Mite*, zusammen mit anderen veränderlichen Parametern, in eine zentrale Konfigurationsdatei (Settings.yaml) ausgelagert (siehe Abbildung 20, Seite 41).

Abbildung 12 Vereinfachte Ablauflogik der Funktion writeToMiteCommand() in Pseudocode

**writeToMiteCommand**(date = null) {

initMite() //Die Hilfsklasse Mite.php initialisieren

validateDate() //Datum validieren

//Bei fehlendem Datum heutiges Datum setzen

parseTimetapeData() //Timetape-XML laden und validieren

for (Mitarbeiterliste:Mitarbeiter){

findMiteUserByMail() //Über die Email-Adresse den

//zugehörigen Mite-Nutzer finden

deleteOldTimetapeEntries() //Veraltete Einträge finden und entfernen

for (Tageliste:Tag) {

if (date = Tag->datum) {

for (Zeitenlist:Zeit) {

calculateMinutes() //Minuten für den Mite-Eintrag berechnen

if (zeittyp == URLAUB) {

postHolidayToMite()} //Urlaubs-Zeitentrag nach Mite schreiben

if (zeittyp == FEIERTAG) {

postBankHolidayToMite()} //Feiertags-Zeitentrag nach Mite schreiben

} } } } } }

### Benachrichtigungen im Erfolgs- und Fehlerfall

Eine besonders große Bedeutung kam in diesem System einer sorgfältigen Fehlerbehandlung zu. Sobald ein Fehler auftrat, der einen korrekten Zeitübertrag beeinträchtigen könnte, sollte der Programmablauf sofort beendet werden. Auf den Fehler sollte so rasch wie möglich aufmerksam gemacht werden, damit die Ursache beseitigt und eine manuelle Korrektur der Zeiteinträge vorgenommen werden kann. Da Fehlerausgaben auf der Kommandozeile bzw. Einträge in Error-Log-Files vermutlich nur selten kontrolliert werden, wurde bei jedem Fehler eine E-Mail-Benachrichtigung an die Service-Mailadresse der in2code GmbH und an die Entwicklerin geschickt.

Damit sichergestellt werden kann, dass der Cronjob auch zur erwarteten Zeit durchlaufen wurde, werden auch im Erfolgsfall unabhängig davon ob tatsächlich Einträge vorgenommen wurden oder nicht, E-Mails mit entsprechenden Informationen verschickt. Falls zu einem späteren Zeitpunkt Benachrichtigungen nur noch im Fehlerfall gewünscht werden, lässt sich dieser Dienst in der Konfigurationsdatei jederzeit ausschalten (siehe Abschnitt 5.5).

## Datensicherung

### Zeitgesteuerte Datensicherung

Ein häufig angeführter Einwand gegen die Nutzung des Datenverarbeitungssystems eines Drittanbieters ist das damit verbundene Abhängigkeitsverhältnis. Bei Wechsel auf ein anderes System besteht zunächst keine Möglichkeit, auf die beim Anbieter gespeicherten Daten zurückzugreifen. Auch muss auf die Zusagen des Anbieters bezüglich der Verfügbarkeit der Dienste vertraut werden und zumindest zeitweise mit Nicht-Verfügbarkeit der Daten gerechnet werden. Ferner ist es ohne direkten Datenzugriff nur über Umwege möglich, andere als die vom Anbieter angebotenen Auswertungen und Darstellungen der Daten vorzunehmen. Um hier eine gewisse Unabhängigkeit zu erreichen, wurde eine automatische Sicherung der wichtigsten mit der Urlaubsverwaltung verbundenen Daten auf dem Firmenserver eingerichtet.

Die Sicherungslogik wurde in einer Funktion mit dem Namen saveTimetapeDataCommand() des TimetapeCommandControllers implementiert. Der Aufruf erfolgt über folgenden Kommandozeilenbefehl, der in einem täglich laufenden Cronjob angestoßen wird.

./flow timetape:savetimetapedata

Der grundsätzliche Ablauf der Datensicherung ist in Abbildung 13 in Form von Pseudocode beschrieben.

Abbildung 13 Ablauflogik der Funktion saveTimetapeDataCommand()in Pseudocode

**saveTimetapeDataCommand()** {

parseTimetapeData()

for (Mitarbeiterliste:Mitarbeiter){

//Mitarbeiter anlegen und speichern,

saveEmployeeIfNotExists() //sofern er noch nicht existiert

for (Urlaubsantragsliste:Urlaubsantrag { //Jeden Urlaubsantrag speichern,

//früheren Urlaubsantrag mit

saveHolidayRequest()} //derselben ID überschreiben

for (Urlaubsanspruchsliste:Urlaubsanspruch) {

//Aktuellen Urlaubsanspruch speichern,

saveHolidayEntitlement()} //früheren Urlaubsanspruch überschreiben

}}

Beim Durchlaufen der Mitarbeiterliste werden für jeden Mitarbeiter (Employee) die gesuchten Daten für die Urlaubsanträge (HolidayRequests) und Urlaubsansprüche (HolidayEntitlements) eingelesen und die Objekte für die Speicherung in der Datenbank erstellt.

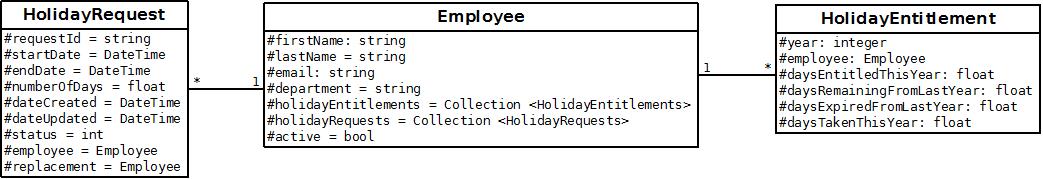
Abbildung 14 zeigt das Datenmodell der für die Datensicherung erstellten und abgespeicherten Objekte.

Abbildung 14 Datenmodell der gesicherten Objekte

Wie bereits bei der Anbindung an *Mite* wurde auch hier versucht, alle möglichen Fehlerfälle abzufangen. Bei Auftreten eines Fehlers wird die Sicherung sofort beendet und eine E-Mail-Benachrichtigung verschickt. Im Erfolgsfall wird eine Nachricht über die erfolgte Datensicherung verschickt. Details und zusätzliche Optionen zum Mailversand lassen sich wiederum in der zentralen Konfigurationsdatei Settings.yaml einstellen (Siehe Abschnitt 5.5).

### Web-Ansichten für Urlaubsanträge und Urlaubsansprüche

Die auf dem Server der in2code GmbH gesicherten Daten über Urlaubsansprüche und Urlaubsanträge werden zunächst nur in der Datenbank abgelegt und müssen über ein Datenbankmanagement-Tool abgefragt werden. Damit die Daten übersichtlicher präsentiert und leichter überprüft werden können, wurden zusätzlich Web-Ansichten zur Auflistung der Daten zur Verfügung gestellt (Abbildung 15 und Abbildung 16).

In der Ansicht der Urlaubsansprüche (Abbildung 15) werden dabei die derzeit letzten Einträge für Urlaubsansprüche des aktuellen Jahres angezeigt. In der Ansicht der Urlaubsanträge (Abbildung 16) werden alle gespeicherten Urlaubseinträge, alphabetisch sortiert nach den Mitarbeitern, und der aktuelle Genehmigungsstatus angezeigt. In der letzten Spalte ist ein Link zu diesem Urlaubsantrag in *Timetape* gegeben, über den eingeloggte Timetape-Administratoren den Antrag einsehen oder verwalten können.

Falls zu einem späteren Zeitpunkt gewünscht, können diese Web-Ansichten recht einfach erweitert und um beliebige Such- , Filter- oder Auswertungsfunktionen ergänzt werden.

Die Namen der Mitarbeiter wurden aus Datenschutzgründen in den beiden Abbildungen entfernt.

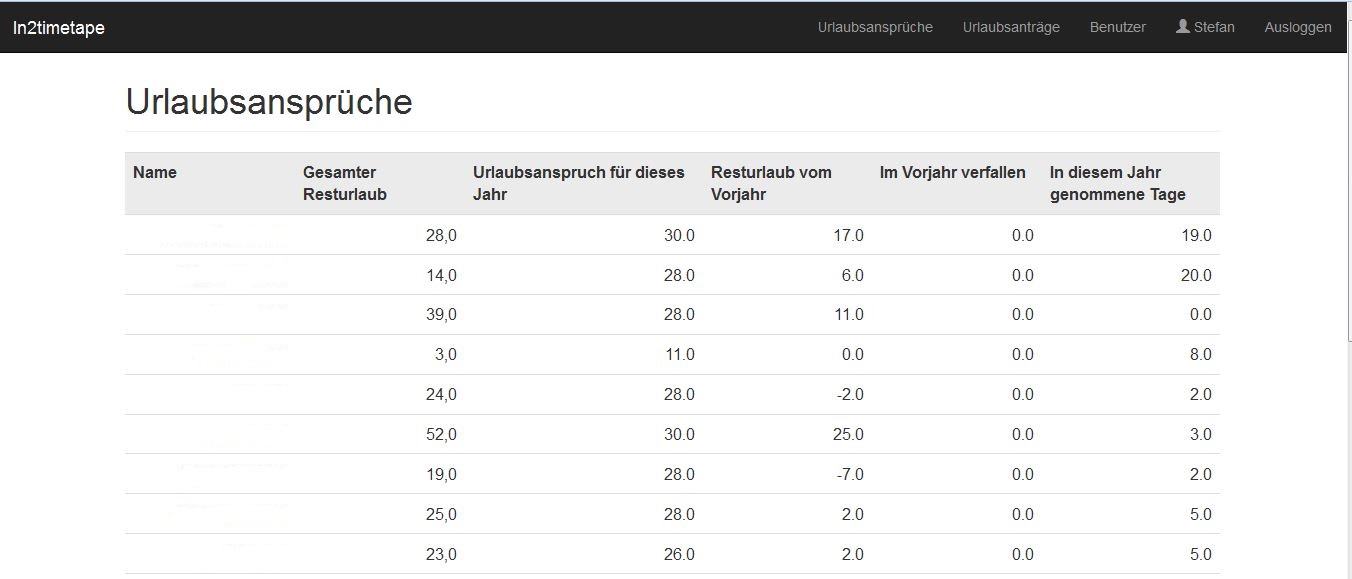


Abbildung 15 Liste der Urlaubsansprüche in der Web-Ansicht von *in2timetape*

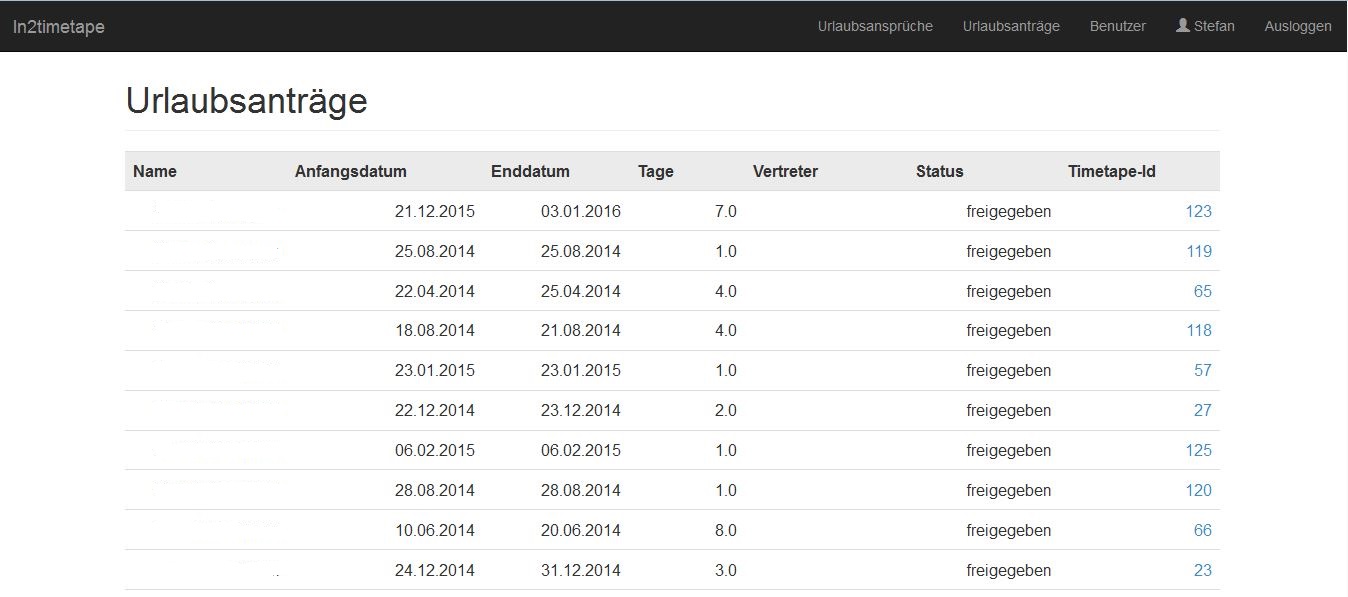


Abbildung 16 Liste der Urlaubsanträge in der Webansicht von *in2timetape*

### Autorisierung und Authentifizierung

Zunächst war geplant, den Zugriff auf die Web-Ansichten nur über einen entsprechenden Eintrag in der Konfigurationsdatei (.htaccess) des Webservers einzuschränken, so dass die Anwendung ausschließlich von Rechnern innerhalb des Firmennetzwerks aufgerufen werden kann. Die Ansichten enthielten jedoch persönliche Daten aller Mitarbeiter, die nicht uneingeschränkt von allen Firmenbeschäftigten eingesehen werden sollten. Auch kann es vorkommen, dass externe Gäste das Firmennetzwerk nutzen, z.B. bei Schulungen im Haus. Deshalb wurde die Anwendung durch ein Login vor unberechtigtem Zugriff geschützt.

Neue Nutzer der Anwendung können über entsprechende Eingabe-Masken auf der Web-Oberfläche oder durch den Aufruf eines Kommandozeilenbefehls nach folgendem Muster angelegt werden:

./flow account:create --username *newUsername* --password *newPassword* --first-name *MyFirstName* --last-name *MyLastName* --roles *In2code.In2timetape:Administrator*

Das verwendete Framework Typo3 Flow liefert für rollenbasierte Berechtigungskonzepte gute Unterstützung bei der Implementierung [38]. So können neue Rollen einfach durch einen Eintrag in der zuständigen Konfigurationsdatei Policy.yaml angelegt werden (Abbildung 17). Hier kann auch festgelegt werden, auf welche Methoden bzw. Ressourcen ein Nutzer in der jeweiligen Rolle Zugriff hat. Bei der Anwendung *in2timetape*, die vorerst nur von der Geschäftsführung genutzt werden soll, gibt es deshalb aktuell auch nur die Rolle des Administrators, der die Methoden aller Controller nutzen darf. Falls künftig noch weitere Rollen, z.B. Nutzer ohne Berechtigung zum Anlegen neuer Benutzer, aufgenommen werden sollen, kann das in der Datei Policy.yaml durch Hinzufügen weniger Zeilen ergänzt werden.

Abbildung 17 Konfiguration der Autorisierung in der Datei Policy.yaml

**resources:  
 methods:  
 In2code\_In2timetape\_AccountControllerMethods:**

**'method(In2code\In2timetape\Controller\AccountController->.\*())'  
 In2code\_In2timetape\_RequestControllerMethods:**

**'method(In2code\In2timetape\Controller\RequestController->.\*())'  
 In2code\_In2timetape\_EntitlementControllerMethods:**

**'method(In2code\In2timetape\Controller\EntitlementController->.\*())'  
roles:  
 Administrator:** []  
**acls:  
 Administrator:  
 methods:  
 In2code\_In2timetape\_AccountControllerMethods:** GRANT  
 **In2code\_In2timetape\_RequestControllerMethods:** GRANT  
 **In2code\_In2timetape\_EntitlementControllerMethods:** GRANT

## Anbindung an das Buchhaltungssystem

Die letzte Zusatzfunktion, die im Rahmen dieser Arbeit implementiert wurde, war die automatische Benachrichtigung der Personalbuchhaltung über den Urlaubsstatus jeden Mitarbeiters. Diese Daten sollten zum 15. jedes Monats per E-Mail an die verantwortliche Ansprechpartnerin in der Personalbuchhaltung übermittelt werden, damit die Informationen in die Lohnabrechnungen aufgenommen werden können. Dazu wurde ein HTML-Template mit einer Tabelle vorbereitet, das im Verlauf der Anwendung mit den aktuell in der Datenbank vorliegenden Daten zu den Urlaubsansprüchen für das aktuelle Jahr befüllt, gerendert und als E-Mail verschickt wurde. Abbildung 18 zeigt beispielhaft den Inhalt einer E-Mail an die Buchhaltung mit fiktiven Mitarbeiterdaten. Die ersten 6 Spalten stammen dabei direkt aus der Datenbank, die letzte Spalte wird bei der Ausgabe dynamisch berechnet.

Urlaubstage für die Mitarbeiter der in2code GmbH für das Jahr 2015

Stand: 15.02.2015 17:53:38

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Vorname** | **Nachname** | **Urlaubsanspruch für dieses Jahr** | **Resturlaub vom Vorjahr** | **Im Vorjahr verfallen** | **In diesem Jahr genommene Tage** | **Gesamter Resturlaub** |
| Max | Mustermann | 28.0 | 2.0 | 0.0 | 10.0 | 20,0 |
| Erika | Meier | 28.0 | 1.0 | 0.0 | 2.0 | 27,0 |
| John | Doe | 26.0 | 5.0 | 0.0 | 13.0 | 18,0 |

Abbildung 18 Beispiel einer Benachrichtigungs-Mail über die Urlaubstage an die Buchhaltung

Der Benachrichtigungsdienst wurde wie die vorigen Zusatzfunktionen als eine Command-Funktion mit dem Namen sendAccountantMailCommand() im TimetapeCommandController implementiert. Sie wird automatisch durch einen Cronjob am 15. jedes Monats durch folgenden Kommandozeilenaufruf gestartet:

./flow timetape:sendaccountantmail

Wichtig ist hierbei, dass unmittelbar vor dem Mailversand nochmals eine Datensicherung durchgeführt wird, damit auch die aktuellen Daten verschickt werden.

Die Ablauflogik der Funktion sendAccountantMailCommand() ist in Abbildung 19 wiederum in Pseudocode erläutert.

**sendAccountantMailCommand()** {

getEmailTemplate(); //Email-Template holen und konfigurieren

getAllEmployeesFromDatabase(); //Alle Mitarbeiter aus der DB holen

for (Employees:Employees){ //Für jeden Mitarbeiter den Urlaubsanspruch

findHolidayEntitlementForThisYear()} //für das aktuelle Jahr finden

assignEntitlementsToView() //Die gefundenen Urlaubsansprüche an die //View übergeben

renderView() //In der View die gewünschten Felder in der

//Tabelle befüllen und Ausgabe rendern

sendMailToAccountant() //Mail an Buchhaltung verschicken

}

Abbildung 19 Ablauflogik der Funktion sendAccountantMailCommand() in Pseudocode

## Konfiguration der Anwendung

*#In2code Settings***TYPO3:***#Einstellungen für den Mailversand* **SwiftMailer:  
 transport:  
 type: 'Swift\_SmtpTransport'  
 arguments:** ~  
 **options:  
 host: 'smtp.example'  
 port: '25'  
 username: 'timetape@example'  
 password: 'xyz'  
In2code:  
 In2timetape:  
 timetape:  
 api: 'https://api.timetape.de/de/export/mitarbeiter.xml?apikey=xyz'***# Pfad zur Schemadatei, mit der das Timetape-XML validiert wird* **pathToTimetapeXsd: 'path/to/timetape.xsd'  
 mite:  
 url: 'in2code'  
 apiKey: 'xyz'***# Dieser User Agent wird beim Eintrag in Mite mit angezeigt* **userAgent: 'In2timetape'***# Diese Projekt- und Service-IDs kommen aus dem Mite-Account* **urlaubProjectId: '1234'  
 feiertagProjectId: '4567'  
 abwesenheitServiceId: '2345'***# Diese Kommentare werden beim Eintrag in Mite mit angegeben* **noteUrlaub: 'Urlaub aus Timetape'  
 noteFeiertag: 'Feiertag aus Timetape'  
 noteFreiertag: 'Freier Tag aus Timetape'***# Durch dieses Flag können die Einträge in Mite als "locked" gesetzt werden* **lockTimeEntries:** 1  
*# Durch dieses Flags kann ausgewählt werden, ob die Urlaube, Feiertage bzw. freien Tage*

*# nach Mite geschrieben werden sollen* **writeUrlaubEntries:** 1  
 **writeFeiertagEntries:** 1  
 **writeFreiertagEntries:** 1  
 **mail:  
 recipientMail: 'service@in2code.de'  
 recipientName: 'In2code Service'  
 senderMail: 'timetape2mite@in2code.info'  
 senderName: 'Timetape2Mite'  
 accountantMail: 'Mustermann@Buchhaltung.de'  
 accountantName: 'Frau Mustermann'  
 ccRecipientMail: 'Christine.Zoglmeier@in2code.de'  
 ccRecipientName: 'Christine'***# Durch diese Flags kann ausgewählt werden, ob Mails verschickt werden sollen:  
# [sendNoEntryMail] Bei erfolgreichem Durchlauf des Cronjobs ohne neuen Eintrag  
# [sendNewEntryMail] Beim Erstellen mindestens eines neuen Eintrags  
# [sendErrorMails] Im Fehlerfall  
# [sendSavedMail] Nach erfolgreicher Datensicherung / -aktualisierung* **sendNoEntryMail:** 1  
 **sendNewEntryMail:** 1  
 **sendErrorMails:** 1  
 **sendSavedMail:** 1  
*# Pfad für das Email-Template für die Mail an die Buchhaltung* **emailTemplate: 'path/to/MailTemplate.html'***# Dieser "Betreff" wird beim Versand der Mails mit angezeigt* **errorsSubject: 'Fehler bei Timetape2Mite'  
 newEntriesSubject: 'Neue Einträge in Mite'  
 noEntriesSubject: 'Keine Einträge in Mite'  
 noEntriesMessage: 'Für den heutigen Tag liegen keine Einträge in Timetape vor'  
 savedDataMessage: 'Die Datensicherung wurde erfolgreich durchgeführt'  
 savedDataSubject: 'Datensicherung erfolgreich'  
 accountantMailSubject: 'Urlaubstage In2code'**

Abbildung 20 Konfigurationsdatei Settings.yaml das Packages *in2timetape*

*Typo3 Flow* bietet komfortable Möglichkeiten, die eigene Anwendung deskriptiv durch Einträge in entsprechende Konfigurationsdateien im YAML-Format [39] zu konfigurieren [40]. Die Dateien werden durch den ConfigurationManager des Frameworks ausgelesen und zusammen geführt. In der Anwendung kann dann auf die konfigurierten Parameter zugegriffen werden. Abbildung 20 zeigt den Inhalt der Datei Settings.yaml des Packages *in2timetape* (die Einträge für Zugangsdaten wurden dabei verändert). In dieser Datei können alle konfigurierbaren Parameter der Anwendung eingetragen werden, z.B. Zugangsdaten, Pfade zu verwendeten Dateien, E-Mail-Adressen für die Benachrichtigungen und in welchen Fällen diese Benachrichtigungen verschickt werden sollen.

## Deployment der Anwendung auf den Firmenserver

Die Anwendung *in2timetape* wurde zunächst lokal mit einem XAMPP-System unter Verwendung von Testsystemen für *Timetape* und *Mite* entwickelt. Im Anschluss daran wurden die Testsysteme durch die *Timetape* und *Mite*-Systeme der in2code GmbH ausgetauscht und die Funktionalität mit den Echtdaten getestet. Als auch diese Tests erfolgreich verlaufen waren, konnte die Anwendung auf den Server der in2code GmbH deployed werden. Für solche Zwecke nutzt die in2code GmbH sogenannte Managed-Server Instanzen des Cloud-Anbieters Domain-Factory [18]. Auf einer dieser Server-Instanzen wurde durch einen Mitarbeiter der in2code GmbH eine Domäne für die Anwendung und eine Datenbank eingerichtet. In das Webverzeichnis dieser Domäne wurde die Konfigurationsdatei für den Paketmanager *Composer*, die Datei composer.json, per SFTP mit Hilfe des FTP-Clients *FileZilla* übertragen. Nach Installation von *Composer* wurden konnten alle erforderlichen Framework-Dateien und Packages durch den Aufruf

composer.phar update

installiert werden. Das Anwendungs-Package *in2timetape* wurde aus dem lokalen Code-Repository per SFTP in den Ordner Packages/Application übertragen. Auf dem Server musste noch die globale Konfigurationsdatei Settings.yaml für die verwendete Datenbankverbindung angepasst werden. Die Erstellung der benötigten Datenbanktabellen für Framework und Anwendung wurde mit Hilfe der eingebundenen Datenbank-Abstraktions-Bibliothek *Doctrine* [41] bewerkstelligt. Dazu wird folgender Befehl aufgerufen, mit dem alle vorhandenen Datenbank-Migrationsdateien ausgeführt werden.

./flow doctrine:migrate

Zuletzt musste nur noch die Konfigurationsdatei Settings.yaml des Pakets an den Stellen angepasst werden, wo von der lokalen Testumgebung auf die Serverkonfiguration umgeschaltet werden muss. Wichtig war hier der Pfad zu verwendeten Ressourcen, der bei einem Aufruf der Anwendung durch den Cronjob absolut und vollständig angegeben werden muss. Eine Besonderheit des Managed Servers von Domain Factory war, dass aufgrund der darauf vorhandenen, modular einsetzbaren PHP-Versionen jedem PHP-Aufruf ein Präfix vorangestellt werden musste, um auf der Kommandozeile die aktuellere CLI-Version 5.5 nutzen zu können. Das auf dem Server verfügbare Management-Tool zur Einrichtung der Cronjobs erwartete außerdem anstatt eines Befehlsaufrufs ein ausführbares Shell-Skript. Deshalb wurden die drei zeitgesteuert auszuführenden Befehle mit dem notwendigen PHP-Präfix und den notwendigen absoluten Pfadangaben in folgende Shell-Skripten verpackt, die dann zu den gewünschten Zeitpunkten aufgerufen werden:

1. Skript zur Übertragung der Daten von Timetape nach Mite, Ausführung werktags, 17 Uhr

#!/bin/bash  
'php5-54LATEST-CLI' /path/on/server/flow timetape:writetomite

1. Skript zur täglichen Datensicherung, Ausführung täglich, 9 Uhr

#!/bin/bash  
'php5-54LATEST-CLI' /path/on/server/flow timetape:savetimetapedata

1. Skript für den Bericht an die Buchhaltung, Ausführung am 15. Jedes Monats, 9:15 Uhr

#!/bin/bash  
'php5-54LATEST-CLI' /path/on/server/flow timetape:sendaccountantmail

# Evaluation

## Erfahrungen mit *Timetape*

Zum Evaluationszeitpunkt war *Timetape* seit mehr als zwei Monaten bei der in2code GmbH in laufendem Betrieb. Die Mitarbeiter stellten alle Urlaubsanträge nur noch über dieses System. Dabei waren die Rückmeldungen durchwegs positiv. Es gab keinerlei Unklarheiten bezüglich der Bedienung des Systems, alle Urlaubsanträge der Mitarbeiter oder Verwaltungsaufgaben der Geschäftsführung konnten rasch und problemlos ohne Rückfragen selbständig umgesetzt werden. Dies bestätigte die durchdachte und ausgereifte Bedienbarkeit von *Timetape*. Zahlreiche Mitarbeiter nutzten die nun zur Verfügung stehende Möglichkeit, um frühzeitig ihren gesamten Jahresurlaub im Voraus zu planen und einzutragen. Diese Informationen können der Geschäftsführung dabei helfen, die Aufgabenverteilung und Projektplanung besser an die Verfügbarkeit der Mitarbeiter anzupassen.

Laut eigener Aussage bedeutete die neue Urlaubsverwaltung für die Geschäftsführung vom ersten Tag an eine spürbare Entlastung. Es gab nun klare Abläufe im Antragsprozess, die erhöhte Transparenz und die Einführung der Vertreterregelung verbesserten die Kommunikation unter den Mitarbeitern und verringerten das Risiko für Urlaubskollisionen. Leider wurde jedoch die vorgesehene Benennung eines Urlaubsvertreters bisher noch nicht von allen Mitarbeitern umgesetzt, worauf in Zukunft noch stärker hingewiesen werden muss.

Als außerordentlich positiv stellte sich die Unterstützung durch den Entwickler von *Timetape,* Herr Walther, heraus. Wichtige Wünsche für Zusatzfunktionen oder Verbesserungsvorschläge wurden von ihm bereitwillig, zügig und ohne Mehrkosten umgesetzt. So hatten wir vorgeschlagen, dass der beim Stellen des Urlaubsantrags benannte Vertreter per E-Mail informiert werden sollte, damit er noch einmal schriftlich an seine Aufgabe erinnert wird. Damit sollte verhindert werden, dass der Vertreter eine mündlich gegebene Zusage vergisst oder über seine Aufgabe gar nicht erst informiert wurde. Dieser Wunsch wurde vom Entwickler innerhalb kurzer Zeit umgesetzt, ergänzt durch die sinnvolle Möglichkeit, den benannten Vertreter durch den Antragsverwalter nachträglich ändern zu können , z.B. bei krankheitsbedingten Ausfällen oder Projektumstrukturierungen. Von uns gemeldete kleinere Probleme, wie ein vom aktuellen Google Calendar nicht mehr korrekt interpretiertes Kalenderformat, wurden von Entwickler innerhalb kürzester Zeit behoben.

Auch die eigentlich noch nicht zum offiziellen Produktangebot gehörende XML-Schnittstelle passte der Entwickler extra für unsere Zwecke so an, dass stundengenaue Einträge für alle Urlaube und Feiertage ausgelesen werden konnten. Zum Vergleich sei erwähnt, dass allein die Einrichtung einer Schnittstelle bei einem professionellen System wie dem *REXX-Vacation Manager* Mehrkosten von 990.- € bedeutet hätte (siehe Tabelle 1Tabelle 3).

In unserem Fall erwies es sich als großer Vorteil, dass das noch nicht allzu lange auf dem Markt befindliche Produkt eines kleinen Anbieters ausgewählt worden war, dessen Entwickler eine hohe Produktidentifikation und Motivation zur Produktverbesserung mitbrachte. Bei einem größeren Anbieter wären diese Anpassungsprozesse vermutlich deutlich schwerfälliger bzw. kostspieliger gewesen. Natürlich birgt die Entscheidung für einen sehr kleinen Dienstleister auch immer das Risiko einer zeitweisen Nicht-Verfügbarkeit des Ansprechpartners oder gar einer Einstellung des Dienstes. Da eine allzeit verfügbare Urlaubsverwaltung jedoch für die Geschäftsprozesse der in2code GmbH nicht zwingend erforderlich ist, sondern nur zu den Rahmenbedingungen gehört, konnte dieses Risiko ohne größere Bedenken in Kauf genommen werden.

Zusammenfassend lässt sich sagen, dass sich durch *Timetape* der gesamte Ablauf der Urlaubsverwaltung reibungsloser, effizienter und konfliktärmer gestaltete, was auch einen positiven Effekt auf die Kernprozesse der Firma haben dürfte.

## Erfahrungen mit den eigenen Zusatzfunktionen

Die Anbindung an das Zeiterfassungssystem *Mite* konnte Ende Januar 2015 in Betrieb genommen werden, so dass hier über die Erfahrung der ersten Wochen berichtet werden kann. Der zeitgesteuerte Aufruf der Anwendung funktionierte zuverlässig und es wurden an jedem Werktag die korrekten Benachrichtigungen über ggf. vorgenommene Einträge verschickt. Probleme gab es kurzzeitig in der dritten Woche, als die Anwendung um eine neue Prüffunktion ergänzt wurde, die alte Einträge aus *Timetape* vor dem Schreiben neuer Einträge wieder entfernen sollte. Hier wurde aufgrund eines Fehlers, der im Testsystem nicht aufgefallen war, auch der gültige Eintrag des aufgerufenen Tages entfernt. Nachdem der betroffene Mitarbeiter sich gemeldet hatte, wurde der Fehler sogleich behoben und seitdem erfolgt der Zeitenübertrag wieder korrekt.

Gerade in diesen ersten Wochen kam es immer wieder vor, dass Mitarbeiter ihren Urlaub wie bisher gewohnt manuell eintrugen, obwohl anschließend durch *in2timetape* zusätzliche Urlaubszeiten gebucht wurden. Eine automatische Überprüfung ist hier kaum möglich, da doppelte Urlaubseinträge prinzipiell möglich und erlaubt sind, und wurde auch durch die Geschäftsführung nicht gewünscht.

Die Mitarbeiter wurden deshalb noch einmal auf den automatischen Zeitübertrag durch *in2timetape* hingewiesen. Nach einer entsprechenden Eingewöhnungszeit dürfte es dann keine Doppelbuchungen mehr geben.

Über die Praxiserfahrung mit der automatischen Datensicherung und dem Bericht an die Personalbuchhaltung kann hier nur kurz berichtet werden, da beides zu Abschluss dieser Arbeit erst wenige Tage in Betrieb war. Zumindest wurden in dieser Zeit alle Funktionen planmäßig ausgeführt. Zum 15.02.2015 wurde der erste Bericht an die Personalbuchhaltung verschickt, der für die darauf folgenden Monatsabrechnungen verwendet werden soll. Beim Abgleich mit den zu diesem Zeitpunkt in *Timetape* erfassten Daten gab es keine Differenzen.

Rückblickend gesehen besonders auffällig ist die große Diskrepanz zwischen dem ursprünglich geschätzten und dem tatsächlich benötigten Implementierungsaufwand für die Anbindungen. So wurde z.B. für die Anbindung von Mite in den ersten Schätzungen nur ein halber (Tabelle 1, Seite 12) bzw. ein Manntag (Tabelle 2, Seite 13) veranschlagt. Erst nach gründlicherer Einarbeitung in die Problematik wurde klar, dass hier nicht nur zwei einfache API-Aufrufe miteinander verknüpft werden müssen, sondern umfassende Validierungen, eine komplexe Datenverarbeitungslogik und eine sorgfältige Fehlerbehandlung mit automatischem Benachrichtigungssystem umgesetzt werden müssen. Daher lag der tatsächliche Zeitaufwand mit ca. 12 Manntagen deutlich über dem geschätzten einzelnen Tag. Für Datensicherung und –präsentation sowie den Bericht an die Personalbuchhaltung kamen weitere 8 Manntage dazu. Eine Unterschätzung des Implementierungsaufwands für unbekannte Aufgaben ist eine häufige Ursache für das Scheitern von IT-Projekten [42]. In unserem Fall, wo es um vergleichsweise unbedeutende Zusatzfunktionen und geringe Zeitaufwände ging, war der entstandene Mehraufwand verschmerzbar. Bei einer Entscheidung für die Eigenimplementierung des Gesamtsystems hätten vergleichbare Fehleinschätzungen jedoch möglicherweise zum Scheitern des Projekts geführt.

## Bewertung des Gesamtsystems und der „Buy-and-Make“-Entscheidung

*Timetape* erfüllte alle eingangs an eine Urlaubsverwaltung gestellten Anforderungen mit hoher Priorität und fast alle Anforderungen mittlerer Priorität (siehe Abschnitt 2.3.1). Noch fehlende spezielle Anforderungen, wie die Anbindung an das bestehende Zeiterfassungssystem und an die Personalbuchhaltung, sowie eine unabhängige Datensicherung, konnten durch eine zusätzliche Eigenimplementierung ergänzt werden. Zwar entstand dadurch ein Mehraufwand von ca. 20 Manntagen, es konnten so aber auch mögliche Bedenken gegen eine Entscheidung für einen Drittanbieter entkräftet werden. Oft bedeutet die Entscheidung für einen Standard-System demnach den Verzicht auf die Möglichkeit zu eigenen, maßgeschneiderten Zusatzfunktionen. Die selbst implementierte *Typo3-Flow* Anwendung *in2timetape* schaffte jedoch die Voraussetzung für beliebige Erweiterungen. Falls in Zukunft weitere Zusatzfunktionen benötigt werden, wie maßgeschneiderte Berichte, Auswertungen oder individualisierte Kalendereinträge, deren Umsetzung vom Anbieter nicht erwartet werden kann, ist dies auf Grundlage der bereits eingelesenen Daten und vorimplementierten Funktionalitäten problemlos möglich. Ferner lockert die eingerichtete Datensicherung das Abhängigkeitsverhältnis vom gewählten Anbieter: auf wichtige Daten kann nun auch bei dessen Nicht-Verfügbarkeit zugegriffen werden und bei einem Anbieterwechsel wäre eine Datenmigration möglich. Schließlich zeigt die erfolgreiche Anbindung an Personalbuchhaltung und Zeiterfassung, dass die Entscheidung für Drittanbieter nicht notwendigerweise Systeminseln produzieren muss, sondern bei entsprechender Auswahl des Anbieters dennoch mit der eigenen Systemlandschaft vernetzt werden kann. Mit vergleichbar geringem Zusatzaufwand konnten also viele der mit einer „Buy“-Entscheidung verbundenen Nachteile aufgehoben werden.

Zum Ende dieser Arbeit stand der Firma demnach ein laufendes, auf ihre Bedürfnisse zugeschnittenes, in die Systemlandschaft eingebundenes Urlaubsverwaltungssystem zur Verfügung. Die Gesamtkosten dafür errechneten sich aus der Nutzungsgebühr für *Timetape* von ca. 300.- Euro pro Jahr und dem zusätzlichen Eigenaufwand von ca. 20 Manntagen. Letzter hätte die Firma bei Vergütung durch ein Auszubildenden-Gehalt ca. 2000.- Euro gekostet. Daraus ergeben sich Kosten von 2300.- Euro für das erste Jahr bzw. 3500.- Euro für 5 Jahre. Eine vollständige Eigenentwicklung wäre der in2code GmbH hier mit Sicherheit erheblich teurer gekommen. Die eigenen, optimistischen Schätzungen beliefen sich auf ca. 7000.- Euro. Es ist jedoch davon auszugehen, dass der tatsächliche Aufwand für ein sinnvoll verwendbares, ausgereiftes System noch deutlich darüber gelegen hätte. So hat die Nachfrage beim Anbieter von *Timetape* ergeben, dass für die Entwicklung von *Timetape*, inklusiver aller Feinschliffe, Umstrukturierungen und laufenden Erweiterungen, bisher ein gesamter Implementierungsaufwand von ca. einem Jahr zusammen gekommen ist.

Wie bereits in der Einleitung angeführt, wird von Experten eine stärkere strategische Ausrichtung von „Make-or-Buy“-Entscheidungen gefordert [1, 2]. Im Falle der in2code GmbH wurde deshalb berücksichtigt, dass die Entwicklung eines mittelständischen Urlaubsverwaltungssystems nicht zum Kerngeschäft und den Kernkompetenzen der Firma gehört und mögliche Käufer eines solchen Systems nicht zu ihrem angestrebten Kundenstamm gehören. Stattdessen würde eine „Make“-Entscheidung das eigentliche Kerngeschäft, die - zeitlich stets knapp budgetierte - Entwicklung von Großkundenprojekten, durch die Bindung von Entwicklerkapazitäten behindern. Ein interessantes Argument bei einer strategischen „Make-or-Buy“-Entscheidung ist auch, dass ein richtig ausgewählter Lieferant oder Dienstleister auch als wertvoller Innovator für die eigenen Produkte oder Abläufe dienen kann [3]. Durch die Fokussierung auf ein eigenes (Teil-)produkt kann der Dienstleister oft höhere Qualität und Identifikation mit seinem Produkt erreichen. Bei der Entwicklung können so neue Konzepte und Ideen umgesetzt werden, für die bei einem aus einer Hand stammenden Gesamtprodukt kein Freiraum gewesen wäre. Diesen positiven Aspekt kann man auch bei der Entscheidung der in2code GmbH für einen Dienstleisteranführen: *Timetape* bot eine ganze Reihe von zusätzlichen Funktionalitäten, an die z.T. in der eigenen Planungsphase gar nicht gedacht worden war. Manche davon, wie z.B. die Benennung eines Urlaubsvertreters mit automatischer Kollisionsprüfung, gaben wertvollen Anstoß zur Verbesserung der eigenen Abläufe.

Bisher wurde noch nicht näher darauf eingegangen, dass natürlich auch der „Make-or-Buy“-Entscheidungsprozess selbst mit erheblichem Zeitaufwand verbunden sein kann. Je sorgfältiger die individuelle Ausgangslage und die eigenen Ziele analysiert und je mehr potentielle Drittanbieter im Detail verglichen werden, desto mehr Ressourcen müssen für den Prozess eingeplant werden. Hier wird hier oft nur ungern Zeit investiert, vor allem wenn zeitliche Ressourcen im Tagesgeschäft knapp sind und das betroffene System für die Kernprozesse der Firma weniger relevant ist. Mögliche Folgen können der Kauf eines überteuerten, in diesem Umfang nicht benötigten oder eines billigen, aber unzulänglichen Systems eines Drittanbieters sein. Möglich ist auch ein zu langes Festhalten an einer bestehenden, schlechten Lösung, die vermeidbare Arbeit verursacht, Abläufe behindert und somit vermeidbare, langfristige Kosten verursacht.

Natürlich wird es für künftige „Make-or-Buy“-Entscheidungen der Firma nur in den seltensten Fällen möglich sein, einen vergleichbaren Zeitaufwand zu investieren, wie er im Rahmen dieser Bachelorarbeit zur Verfügung stand. Idealerweise können künftige Entscheidungsprozesse jedoch anhand der gesammelten Erfahrungen und mit der eigenen Firmenstrategie abgeglichenen Sammlung von Argumenten rascher und systematischer geführt werden.

# Abkürzungsverzeichnis

API Application Programming Interface

FTP File Transfer Protocol

IDE Integrated Development Environment (Integrierte Entwicklungsumgebung)

MVC Model-View-Controller (Trennung von Datenhaltungs-, Präsentations- und Anwendungslogik)

PHP PHP: Hypertext Preprocessor

SFTP Secure File Transfer Protocol

YAML YAML Ain’t Markup Language [39]

# Abbildungsverzeichnis

[Abbildung 1 Entwicklung der Mitarbeiterzahlen seit Firmengründung 4](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312236)

[Abbildung 2 Organigramm der in2code GmbH 5](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312237)

[Abbildung 3 Ansichten des bisherigen Urlaubsverwaltungstools 6](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312238)

[Abbildung 4 Beispiel einer Eingabemaske für Zeiten in Mite 8](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312239)

[Abbildung 5 Systemlandkarte vor Implementierung des neuen Urlaubsverwaltungssystems 9](#_Toc413312240)

[Abbildung 6 Gewünschte Systembeziehungen nach Inbetriebnahme des Urlaubsverwaltungssystems 11](#_Toc413312241)

[Abbildung 7 Mitarbeiteransicht in Timetape 22](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312242)

[Abbildung 8 Abteilungskalenderansicht in Timetape 23](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312243)

[Abbildung 9 Verwaltungsbereich von *Timetape* 27](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312244)

[Abbildung 10 Anlegen von Arbeitszeitmodellen in Timetape 28](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312245)

[Abbildung 11 Ausschnitt aus dem für die Timetape-Api erzeugten Schema, der Datei timetape.xsd. 33](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312246)

[Abbildung 12 Vereinfachte Ablauflogik der Funktion writeToMiteCommand() in Pseudocode 35](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312247)

[Abbildung 13 Ablauflogik der Funktion saveTimetapeDataCommand()in Pseudocode 36](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312248)

[Abbildung 14 Datenmodell der gesicherten Objekte 37](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312249)

[Abbildung 15 Liste der Urlaubsansprüche in der Web-Ansicht von *in2timetape* 38](#_Toc413312250)

[Abbildung 16 Liste der Urlaubsanträge in der Webansicht von *in2timetape* 38](#_Toc413312251)

[Abbildung 17 Konfiguration der Autorisierung in der Datei Policy.yaml 39](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312252)

[Abbildung 18 Beispiel einer Benachrichtigungs-Mail über die Urlaubstage an die Buchhaltung 40](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312253)

[Abbildung 19 Ablauflogik der Funktion sendAccountantMailCommand() in Pseudocode 40](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312254)

[Abbildung 20 Konfigurationsdatei Settings.yaml das Packages *in2timetape* 41](file:///C:\in2timetapeRepo\Bachelorarbeit\Bachelorarbeit.docx#_Toc413312255)

# Tabellenverzeichnis

[Tabelle 1 Aufwandsschätzung aus dem Lastenheft "in2Holiday" 12](#_Toc413183377)

[Tabelle 2 Überarbeitete Aufwandsschätzung für die Implementierung des Urlaubsplaners 13](#_Toc413183378)

[Tabelle 3 Liste der verglichenen elektronischen Urlaubsplaner 15](#_Toc413183379)

[Tabelle 4 Vergleich der ausgewählten Urlaubsverwaltungssysteme 24](#_Toc413183380)

# Literaturverzeichnis

|  |  |
| --- | --- |
| [1] | C. Fill und E. Visser, „The outsourcing dilemma: a composite approach to the make or buy decision,“ *Management Decision,* Bd. 38, pp. 43-50, 01 02 2000. |
| [2] | J. A. Welch und P. R. Nayak, „Strategic sourcing: a progressive approach to the make-or-buy decision,“ *ACAD MANAGE PERSPECT,* Bd. 6, Nr. 1, pp. 23-31, 1 02 1992. |
| [3] | F. Veloso und S. Fixson, „Make–Buy Decisions in the Auto Industry: New Perspectives on the Role of the Supplier as an Innovator,“ *Technological Forecasting and Social Change,* Bd. 67, pp. 239-257, 07 06 2001. |
| [4] | „University of Cambridge, IfM, Strategy and Performance - Industrial Make-or-Buy Decisions,“ [Online]. Available: http://www.ifm.eng.cam.ac.uk/research/csp/mvb/. [Zugriff am 09 01 2015]. |
| [5] | M. Jentsch, Eigenfertigung oder Fremdbezug?: Herangehensweise an die Make-or-Buy-Fragestellung in der Praxis, Saarbrücken: VDM Verlag Dr. Müller, 2010. |
| [6] | „TYPO3 - The Enterprise Open Source CMS,“ [Online]. Available: http://typo3.org/. [Zugriff am 20 11 2014]. |
| [7] | „TYPO3 4.1.0: tslib\_pibase Class Reference,“ [Online]. Available: http://doc-typo3.ameos.com/4.1.0/classtslib\_\_pibase.html . [Zugriff am 07 02 2015]. |
| [8] | „Official TYPO3 Extbase and Fluid Guide,“ [Online]. Available: http://docs.typo3.org/typo3cms/ExtbaseGuide/. [Zugriff am 20 11 2014]. |
| [9] | „TYPO3 Flow – the Enterprise PHP Application Framework,“ [Online]. Available: http://flow.typo3.org/home. [Zugriff am 13 02 2015]. |
| [10] | „mite. Schlanke Zeiterfassung für Teams,“ [Online]. Available: http://mite.yo.lk. [Zugriff am 10 01 2015]. |
| [11] | R. T. Fielding, „Architectural Styles and,“ 2000. [Online]. Available: http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm. [Zugriff am 11 01 2015]. |
| [12] | „API. Dokumentation der offenen Datenschnittstelle von mite.,“ [Online]. Available: http://mite.yo.lk/api. [Zugriff am 10 01 2015]. |
| [13] | „Redmine,“ [Online]. Available: http://www.redmine.org/. [Zugriff am 09 01 2015]. |
| [14] | S. Loghin und S. Stein, „Lastenheft für Urlaubsplaner „in2Holiday“ V1.1,“ 2014. |
| [15] | „Timetape.de -Sicherheit,“ [Online]. Available: https://www.timetape.de/de/sicherheit.html. [Zugriff am 19 11 2014]. |
| [16] | „Timetape.de - Datenschutz,“ [Online]. Available: https://www.timetape.de/de/datenschutz.html. [Zugriff am 19 11 2015]. |
| [17] | C. Beier, „CBeier.net - Eigenentwicklung vs Standardsoftware,“ 17 03 2010. [Online]. Available: http://www.cbeier.net/blog/weblog/eigenentwicklung-vs-standardsoftware. [Zugriff am 09 02 2015]. |
| [18] | „Domain Factory - Managed Server,“ [Online]. Available: http://www.df.eu/de/server-hosting/managed-server/. [Zugriff am 09 01 2015]. |
| [19] | T. Rossberg, „Google-Online-Kalender: Datenschutzproblematik und Lösung,“ 12 06 2012. [Online]. Available: http://kola.opus.hbz-nrw.de/volltexte/2012/791/. [Zugriff am 09 02 2015]. |
| [20] | „Hetzner Online AG: Rechenzentren,“ [Online]. Available: http://www.hetzner.de/hosting/unternehmen/rechenzentrum. [Zugriff am 19 11 2014]. |
| [21] | „PHP IDE :: JetBrains PhpStorm,“ [Online]. Available: https://www.jetbrains.com/phpstorm/. [Zugriff am 13 02 2015]. |
| [22] | „git -- everything is local,“ [Online]. Available: http://git-scm.com/. [Zugriff am 13 02 2015]. |
| [23] | „Free source code hosting for Git and Mercurial by Bitbucket,“ [Online]. Available: https://bitbucket.org/. [Zugriff am 13 02 2015]. |
| [24] | „Composer,“ [Online]. Available: https://getcomposer.org/. [Zugriff am 03 02 2015]. |
| [25] | „FileZilla - The free FTP solution,“ [Online]. Available: https://filezilla-project.org/. [Zugriff am 19 11 2014]. |
| [26] | „typo3/swiftmailer - Packagist,“ [Online]. Available: https://packagist.org/packages/typo3/swiftmailer. [Zugriff am 19 01 2015]. |
| [27] | „svparijs/Flow.Login · GitHub,“ [Online]. Available: https://github.com/svparijs/Flow.Login. [Zugriff am 09 02 2015]. |
| [28] | „svparijs/TYPO3.AccountManagement · GitHub,“ [Online]. Available: https://github.com/svparijs/TYPO3.AccountManagement. [Zugriff am 09 02 2015]. |
| [29] | „thomasklein/mite.php · GitHub,“ [Online]. Available: https://github.com/thomasklein/mite.php. [Zugriff am 09 01 2015]. |
| [30] | „API. Dokumentation der offenen Datenschnittstelle von mite.,“ [Online]. Available: http://mite.yo.lk/api/. [Zugriff am 09 01 2015]. |
| [31] | „Parsley - The ultimate JavaScript form validation library,“ [Online]. Available: http://parsleyjs.org/. [Zugriff am 10 02 2015]. |
| [32] | „Free Online XSD/XML Schema Generator From XML - FreeFormatter.com,“ [Online]. Available: https://www.freeformatter.com/xsd-generator.html. [Zugriff am 09 01 2015]. |
| [33] | „W3C XML Schema (XSD) Validation online,“ [Online]. Available: http://www.utilities-online.info/xsdvalidation. [Zugriff am 09 01 2015]. |
| [34] | „Download XAMPP,“ [Online]. Available: https://www.apachefriends.org/de/download.html. [Zugriff am 30 11 2014]. |
| [35] | „Command Line,“ [Online]. Available: http://docs.typo3.org/flow/TYPO3FlowDocumentation/TheDefinitiveGuide/PartIII/CommandLine.html. [Zugriff am 10 01 2015]. |
| [36] | S. Walther, „timetape-XML-Format-Mitarbeiter-V1.3,“ 2014. |
| [37] | „PHP: SimpleXMLElement - Manual,“ [Online]. Available: http://php.net/manual/de/class.simplexmlelement.php. [Zugriff am 10 01 2015]. |
| [38] | „TYPO3FlowDocumentation/TheDefinitiveGuide/PartIII/Security,“ [Online]. Available: http://docs.typo3.org/flow/TYPO3FlowDocumentation/TheDefinitiveGuide/PartIII/Security.html#policies-aka-access-control-lists-acls. [Zugriff am 10 02 2015]. |
| [39] | „YAML – Wikipedia,“ [Online]. Available: http://de.wikipedia.org/wiki/YAML. [Zugriff am 12 12 2014]. |
| [40] | „TYPO3FlowDocumentation/TheDefinitiveGuide/PartIII/Configuration,“ [Online]. Available: http://docs.typo3.org/flow/TYPO3FlowDocumentation/TheDefinitiveGuide/PartIII/Configuration.html. [Zugriff am 08 02 2015]. |
| [41] | „Doctrine - PHP Database Libraries,“ [Online]. Available: http://www.doctrine-project.org/index.html. [Zugriff am 19 11 2014]. |
| [42] | R. N. Charette, „Why Software Fails - IEEE Spectrum,“ 2005. [Online]. Available: http://www.rose-hulman.edu/Users/faculty/young/OldFiles/CS-Classes/csse372/201310/Readings/WhySWFails-Charette.pdf. [Zugriff am 10 02 2015]. |

1. CRUD: create, read, update, delete

   Ansichten (Views) bzw. Programmierlogik (Actions) zum Erstellen, Anzeigen, Ändern und Löschen von Datensätzen [↑](#footnote-ref-1)