

Leibniz Universität Hannover
Wirtschaftswissenschaftliche Fakultät
Institut für Produktionswirtschaft
Prof. Dr. Stefan Helber

Hausarbeit im Rahmen der Veranstaltung
Entwicklung von Anwendungssystemen im WiSe 2014/2015
(Veranstaltungs-Nr. 173610)

RCPSP
RCPSP

Andreas Hipp	Robert Matern
Ungerstr. 24	Plathnerstr. 49
30451 Hannover	30175 Hannover
Matr.-Nr. 3027520 ???	Matr.-Nr. 2798160

Abgabedatum: 24.03.2015

Inhaltsverzeichnis

Abbildungsverzeichnis	iii
Tabellenverzeichnis	iii
Abkürzungsverzeichnis	iv
Symbolverzeichnis	v
1 Einleitung	1
2 Grundlagen zur ressourcen-beschränkten Projektplanung und zu dem Framework Ruby on Rails	2
2.1 Kapazitätsplanung	2
2.2 Kostenplanung	4
2.3 Ruby on Rails	6
3 Implementierung des RCPSP mittels Ruby on Rails	7
3.1 Darstellung der Funktionsweise der Anwendung anhand eines Userguides . .	7
4 Kritische Würdigung des Anwendungssystems	18
5 Fazit	18
Literatur	19
A Anhang	21
A.1 GAMS-Implementierung des Beispiels	21
A.2 Ruby on Rails Programmcode	21

Abbildungsverzeichnis

1	Terminalfenster unter Apple Mac OS X	6
2	Startseite Projektplanung Applikation	8
3	Anmeldebildschirm	10
4	Fehleranzeige bei Anmeldung	11
5	Profilseite eines Users	14
6	Übersicht der Ressourcen für User	15
7	Profilseite des Administrators	17
8	Übersicht der Ressourcen aus Sicht des Administrators	17

Tabellenverzeichnis

Quellcodeverzeichnis

1	Code der Startseite (<code>app/views/static_pages/home.html.rb</code>)	8
2	Controller der statischen Seiten (<code>app/controller/static_pages_controller.rb</code>)	9
3	<code>config/routs.rb</code>	9
4	<code>app/models/users.rb</code>	11
5	<code>app/views/layouts/_header.html.erb</code>	12
6	<code>app/views/resources/index.html.erb</code>	13
7	<code>app/views/users/show.html.erb</code>	13
8	<code>app/views/resources/_signed_in.html.erb</code>	15

Abkürzungsverzeichnis

RCPS	Resource-Constrained Project Scheduling
RoR	Ruby on Rails
SGS	Schedule Generation Scheme

Symbolverzeichnis

d_i	Dauer von Vorgang i
FE_i	frühestes Ende von Vorgang i
$i, h = 1, \dots, I$	Vorgänge
k_{ir}	Kapazitätsbedarf von Vorgang i auf Ressource r
kp_r	verfügbare Kapazität von Ressource r je Periode
\mathcal{N}_i	Menge der direkten Nachfolger von Vorgang i
oc_r	Kosten einer Einheit Zusatzkapazität von Ressource r
O_{rt}	Zusatzkapazität von Ressource r in Periode t
$r = 1, \dots, R$	Ressourcen
SE_i	spätestes Ende von Vorgang i
$t, \tau = 1, \dots, T$	Perioden
\mathcal{V}_i	Menge der direkten Vorgänger von Vorgang i
$X_{jt} \in \{0, 1\}$	gleich 1, falls Vorgang j in Periode t endet, sonst 0

1 Einleitung

Bei einem Projekt handelt es sich um eine zeitlich befristete, relativ innovative und risikobehaftete Aufgabe von erheblicher Komplexität, die meist einer gesonderten Planung bedarf.¹ Dementsprechend von großer Bedeutung ist die vorhergehende und genaue Planung von Projekten.² Projektplanung ist die Planung aller Arbeitsgänge eines Projekts durch Zuweisung eines Startzeitpunktes, so dass die Zeitbeziehung zwischen den Vorgängen eingehalten und knappe Ressourcenkapazitäten nicht überschritten werden.² Durch das Zerlegen des Projekts in einzelne Arbeitsgänge wird versucht die Komplexität zu reduzieren und eine geordnete Abfolge der Arbeitsgänge zu erstellen, um das Projektziel zu erreichen.³ Projektziele können dabei unterschiedlich kategorisiert werden, z. B. in Sach-, Termin- oder Kostenziele.⁴

Nach DIN 69900 hat ein Arbeitsgang oder ein einzelner Vorgang eines Projekts einen definierten Anfang sowie ein definiertes Ende und dient für das Projekt als Ablauelement zur Beschreibung eines bestimmten Geschehens.⁵ Trotz der Zerlegung besitzen die einzelnen Arbeitsgänge des Projekts eine Beziehung, mit der die Reihenfolge der Ablauffolge bestimmbar ist.⁶ Oft wird zur Darstellung der Vorgangsbeziehung ein Vorgangsknoten-Netzplan verwendet.⁷ Ein Arbeitsgang ist i. d. R. verbunden mit dem Einsatz von Ressourcen, welche wiederum mit Kosten verbunden sind. Eine Möglichkeit, das Projektziel unter minimaler Ressourcenverwendung zu erreichen, ist die effiziente Planung der Ablauffolge der Arbeitsgänge eines Projekts.⁸ Damit ist es möglich, mehrere Projekte bei einer gegebenen Zeitvorgabe unter Einhaltung von Ressourcenrestriktionen fertigzustellen bzw. bei konstanter Ressourcenkapazität ein Projekt in kürzerer Zeit abzuschließen.

Zur Bestimmung der optimalen Ablauffolge der einzelnen Arbeitsgänge eines Projekts kann ein Optimierungsmodell verwendet werden, bei der für eine festgelegten Ablauffolge eines Projekts und unter Berücksichtigung der Ressourcenbeschränkung die Fertigstellungszeit minimiert wird. Im Kapitel 2 wird eine solche Modellformulierung für das ressourcenbeschränkte Projektplanungsproblem als sogenannte Kapazitätsplanung vorgestellt.⁹ Alternativ wird in dem Kapitel das Optimierungsmodell um die Bedienung erweitert, dass Zusatzkapazitätseinheiten gebucht werden können. Mit dieser Modellerweiterung wird von der Kostenplanung in Projekten gesprochen. Bezeichnet wird im Allgemeinen das ressourcenbeschränkte Projektplanungsproblem mit der englischen Bezeichnung des *Resource-Constrained*

¹Vgl. Voigt und Schewe (2014)

²Vgl. Zimmermann et al. (2006), S. VI

³Vgl. Zimmermann et al. (2006), S. 4

⁴Vgl. Felkai und Beiderwieden (2011), S. 52

⁵Vgl. DIN 69900 (2009), S. 15

⁶Vgl. Kellenbrink (2014), S. 6-7

⁷?????

⁸Vgl. Bartels (2009), S. 11-12

⁹????

Project Scheduling Problem (RCPSP). Bei dem RCPSP handelt es sich um eine abstrakte mathematische Modellformulierung. Ziel der vorliegenden Arbeit ist es das RCPSP in Ruby on Rails (RoR) zu implementieren. Bei RoR handelt es sich um ein Framework zur Entwicklung von Webdokumenten bzw. Internetseiten.¹⁰ Es baut auf der Programmiersprache Ruby auf und ist ursprünglich von David Heinemeier Hansson entwickelt.¹¹ Die Implementierung bedarf einer Verknüpfung von RoR und GAMS¹². Unter GAMS wird eine algebraische Modellierungssprache für mathematische Optimierungsprobleme verstanden, mit der das RCPSP gelöst wird.¹³ Im Kapitel 3 wird die Entwicklung des Anwendungssystems zum Lösen des RCPSP ausführlich beschrieben. Ergänzt wird diese Arbeit durch eine kritische Würdigung des Anwendungssystems in Kapitel 4 sowie einem Fazit in Kapitel 5.

2 Grundlagen zur ressourcen-beschränkten Projektplanung und zu dem Framework Ruby on Rails

2.1 Kapazitätsplanung

Ein Großteil an Projekten besitzt die Eigenschaft eines beschränkten Ressourcenkontingents.¹⁴ Soll demgemäß die vorgegebene Terminierung des Projektes als zuvor festgesetztes Ziel erreicht werden, muss neben der Reihenfolgerestriktion auch der Ressourcenbedarf der unterschiedlichen Arbeitsgänge sichergestellt werden. Mit der Einhaltung des Ressourcenbedarfs ist es möglich, alle zur Erfüllung des Projektes notwendigen Arbeitsgänge auszuführen und somit letztendlich das Projekt abzuschließen. Neben limitierten Ressourcen, die während des gesamten Projekts nur ein Mal zur Verfügung stehen, wie bspw. das Projektbudget, gibt es Ressourcen, die nach einer bestimmten Anzahl von Perioden erneuert werden können.¹⁵ Erneuerbare Ressourcen sind bspw. die Produktionskapazität einer Maschine oder der Personaleinsatz für ein Projekt. In dieser Arbeit wird der Fokus auf diese erneuerbaren Ressourcen gelegt.

Zur Lösung des ressourcenbeschränkten Projektplanungsproblems kann das Modell RCPSP genutzt werden. Das RCPSP legt durch Fixierung der Aktivitätsstartzeitpunkte den Projektgrundablauf zur Zielerreichung der Minimierung der Projektdauer fest. Dies geschieht unter Einhaltung der Startzeitpunkt- bzw. der Vorrangsbedingung der einzelnen Arbeitsgänge sowie der Kapazitätsbeschränkung der erneuerbaren Ressourcen.¹⁶ Die im folgenden aufgestellte Zielfunktion des RCPSP zur Minimierung der Projektdauer ist die gängige Version

¹⁰???

¹¹???

¹²General Algebraic Modeling System

¹³???

¹⁴Vgl. Kellenbrink (2014), S. 11

¹⁵Vgl. Neumann-Braun et al. (2003), S. 21-22

¹⁶Vgl. Demeulemeester und Herroelen (2011), S. 23

der Kapazitätsplanung,¹⁷ andere Variationen sind aber ebenfalls möglich.¹⁸

Nachfolgend wird das deterministische RCPSP in diskreter Zeit formuliert.¹⁹ Charakteristisch für eine mathematische Modellformulierung in diskreter Zeit sind die Zeiteinheiten, die den Perioden t, τ entsprechen.

Modell RCPSP

$$\min Z = \sum_{t=FE_I}^{SE_I} t \cdot X_{I,t} \quad (1)$$

unter Beachtung der Restriktionen

$$\sum_{t=FE_i}^{SE_i} X_{it} = 1 \quad i = 1, \dots, I \quad (2)$$

$$\sum_{t=FE_h}^{SE_h} t \cdot X_{ht} \leq \sum_{t=FE_i}^{SE_i} (t - d_i) \cdot X_{it} \quad i = 1, \dots, I; h \in \mathcal{V}_i \quad (3)$$

$$\sum_{i=1}^I \sum_{\tau=\max(t, FE_i)}^{\tau=\min(t+d_i-1, SE_i)} k_{ir} \cdot X_{i\tau} \leq kp_r \quad r = 1, \dots, R; t = 1, \dots, T \quad (4)$$

$$X_{it} \in \{0, 1\} \quad i \in \mathcal{I}; t \in \{FE_i, \dots, SE_i\} \quad (5)$$

Es wird ein Projekt betrachtet, dass aus I unterschiedlichen Arbeitsgängen besteht. Jeder Arbeitsgang i hat eine definierte Menge von zu erledigenden Vorgängerarbeitsgängen $h \in \mathcal{V}_i$. Des Weiteren ist für die Fertigstellung des Projekts die Abarbeitung der Arbeitsgänge in topologischer Reihenfolge notwendig. D. h. der Vorgänger h hat stets eine kleinere Ordnungszahl als sein Nachfolger i ($h < i$) und muss zur Fortsetzung des Projektverlaufs beendet sein. Die Bearbeitungsdauer eines Arbeitsgangs i wird mit dem Parameter d_i festgelegt. Bei dem RCPSP in diskreter Zeit wird die Annahme getroffen, dass die Dauer durch einen ganzzahligen Parameter abgebildet wird. Der Startzeitpunkt des Projekts ist $t = 0$ und erstreckt sich über einen Gesamtzeitraum von T Perioden. Um die Reihenfolgebedingungen einzuhalten, werden einem Projekt zwei Dummy-Arbeitsgänge „Beginn“ ($i = 1$) und „Ende“ ($i = I$) hin-

¹⁷Vgl. Drexel et al. (1997), S. 98

¹⁸Vgl. Talbot (1982), S. 1200

¹⁹???

zugefügt, welche mit einer Dauer von 0 Zeiteinheiten bewertet werden.²⁰ Dadurch wird der Projektbeginn und das Projektende exakt terminiert. Der Parameter k_{ir} stellt die benötigten Kapazitäten der erneuerbaren Ressource r bei der Durchführung von Arbeitsgang i dar. Die Ressourcen $r \in R$ sind in einer Periode innerhalb des Umfangs ihrer Kapazität kp_r nutzbar. Da es sich um erneuerbare Ressourcen handelt, stehen diese zu jeder neuen Periode in vollem Umfang erneut zur Verfügung. Ungenutzte Ressourcen sind jedoch nicht auf nachfolgende Arbeitsgänge und Perioden übertragbar.²¹ Um den Fertigstellungszeitpunkt der einzelnen Arbeitsgänge i festlegen zu können, wird der Modellformulierung in diskreter Zeit die binäre Entscheidungsvariable X_{it} hinzugefügt.²² Diese Binärvariable nimmt den Wert 1 an, falls der Arbeitsgang i zum Zeitpunkt t beendet wird.

Mittels der Zielfunktion (1) wird der Fertigstellungszeitpunkt des Projekts minimiert. Dafür wird der Zeitraum zwischen dem frühesten und spätesten Fertigstellungszeitpunkt FE_I und SE_I aller durchzuführenden Arbeitsgänge I betrachtet. Nebenbedingung (2) stellt sicher, dass ein Arbeitsgang i zwischen dem jeweiligen für diesen Arbeitsgang geltenden frühesten und spätesten Fertigstellungszeitpunkt nur exakt ein Mal durchgeführt wird. Die Reihenfolge restriktion wird mit der Nebenbedingung (3) eingehalten. Sie stellt sicher, dass jeder Vorgänger $h \in \mathcal{V}_i$ beendet ist, bevor der Arbeitsgang i startet. Der Term $(t - d_i)$ garantiert für den Arbeitsgang i , dass dieser erst beginnt, sobald der Vorgänger h mit der Dauer d_i abgeschlossen ist. Der Parameter kp_r spiegelt die Kapazitätsgrenze für eine erneuerbare Ressource r je Periode t wieder. In Nebenbedingung (4) findet zum einen eine formale Darstellung dieser Kapazitätsbegrenzung statt. Zum anderen wird der Ressourcenverzehr während der gesamten Bearbeitungsdauer der Fertigstellung beachtet, in dem der Kapazitätsbedarf k_{ir} aller Arbeitsgänge I summiert wird. Eben diese Summe wird schließlich durch kp_r beschränkt. Mit der Nebenbedingung (5) wird die Binärvariable X_{it} für den Zeitraum $t = \{FE_i, \dots, SE_i\}$ formal definiert. Aufgrund der Reihenfolgebeziehung (3) darf der jeweils betrachtete Arbeitsgang nur in diesem Zeitraum fertiggestellt werden. Die gemischt-ganzzahlige Modellformulierung lässt sich durch Standard-Lösungsverfahren exakt lösen.²³

2.2 Kostenplanung

Aufbauen auf der Kapazitätsplanung kann das RCPSP um die Nutzung von Zusatzkapazitäten der Ressourcen erweitert werden, damit dem Projektplanungsmodell gestattet ist den Vorgänge zusätzliche Kapazitätseinheiten der notwendigen Ressourcen bereitzustellen. Die Kapazitätsrestriktion wird dementsprechend um die Entscheidungsvariable $O_{rt} \geq 0$ erweitert. Die Variable O_{rt} beschreibt die Einheiten an Zusatzkapazitäten einer Ressource r in der Periode t . Damit steht nicht die Einhaltung der verfügbaren Kapazitäten im Vordergrund,

²⁰Vgl. Zimmermann et al. (2006), S. 66

²¹Vgl. Kellenbrink (2014), S. 12

²²Vgl. Pritsker et al. (1969), S. 94

²³z. B. mittels eines Branch-and-Bound-Verfahrens, Vgl. Kellenbrink (2014), S. 14

sonder unter Beachtung der Projektstruktur die aufgewendeten Zusatzkosten des Projekts. Dem Optimierungsmodell ist es damit gestattet durch Erhöhung der Kapazitäten der Ressourcen die anfängliche Ressourcenbeschränkung zu umgehen. Bei der Modellerweiterung der Kostenplanung wird der Parameter oc_r eingeführt, der für eine betrachtete Ressource r die Kosten einer Einheit der Zusatzkapazitäten beschreibt. Ziel des Optimierungsmodells ist es damit die Kosten des Projekts zu minimieren. Das Modell hilft damit der Entscheidung, ob durch Einführen der Möglichkeit von Zusatzkapazitäten das Projektziel verbessert erreicht wird. Es handelt sich um den Trade-off des frühzeitigen Erreichens des Projektziels durch Nutzung von Zusatzkapazitäten und der gesamten Projektkosten die für das Projekt aufgewendet werden sollen.

Nachfolgend wird das deterministische RCPSP+ in diskreter Zeit formuliert.

Modell RCPSP+

$$\min Z = \sum_{t=1}^T \sum_{r=1}^R oc_r \cdot O_{r,t} \quad (6)$$

unter Beachtung der Restriktionen (2), (3), (5) sowie

$$\sum_{i=1}^I \sum_{\tau=\max(t, FE_i)}^{\tau=\min(t+d_i-1, SE_i)} k_{ir} \cdot X_{i\tau} \leq kp_r + O_{rt} \quad r = 1, \dots, R; \quad t = 1, \dots, T \quad (7)$$

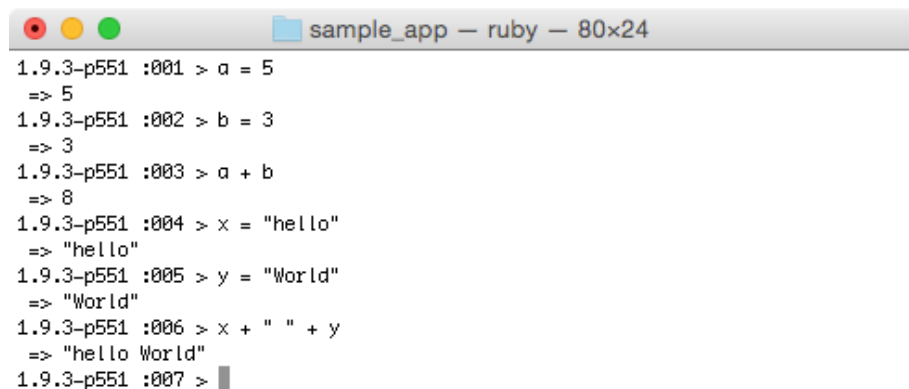
$$O_{rt} \geq 0 \quad r = 1, \dots, R; \quad t = 1, \dots, T \quad (8)$$

Bei dem RCPSP+ wird die Zielfunktion insoweit formuliert, dass über alle Perioden $t \in T$ und über alle Ressourcen $r \in R$ die Summe der Kosten oc_r für die Anzahl an notwendige Einheiten an Zusatzkapazität O_{rt} minimiert wird. Weiterhin bleibt die Nebenbedingung (2) und (3) bestehen, dass jeder Vorgang exakt einmal zwischen dem frühesten Ende (FE_i) und dem spätesten (SE_i) fertiggestellt und die Topologie der Vorgänge eingehalten wird. Weiterhin gilt die Nebenbedingung (5), dass es sich bei der Entscheidungsvariable X_{jt} um eine binäre Variable handelt. Erweitert wird das RCPSP aus der Kapazitätsplanung mit einer modifizierten Nebenbedingung zur Einhaltung Kapazitätsbeschränkung. Mit der Nebenbedingung (7) wird die Kapazitätsrestriktion für eine Ressource $r \in R$ in einer Periode $t \in T$ eingehalten, jedoch ist es dem Modell gestattet die vorhandene Ressourcenkapazität kp_r um die Ausprägung der Entscheidungsvariable O_{rt} zu erweitern. Durch Lösen des Optimierungsmodells wird der Ablaufplan des Projekts unter Beachtung der unterschiedlich zulässigen Gesamtdauern SE_I generiert. Weiter wird für jede Ressource $r \in R$ zur jeweiligen Periode $t \in T$ die notwendige Anzahl an benötigter Zusatzkapazität O_{rt} ermittelt. Die

Nebenbedingung (8) beschreibt die Eigenschaft der Entscheidungsvariable O_{rt} , dass es sich um eine positive Variable bzw. einen Nullwert handelt.

2.3 Ruby on Rails

Das Frameworks Ruby on Rails (RoR) zur Entwicklung von Web-Applikationen mit Datenbankbezug wurde von David Heinemeier Hansson im Jahre 2004 erstmals vorgestellt.²⁴ Mit dem Name von RoR wird klar, dass das Framework die Programmiersprache Ruby nutzt. Ruby wird von den den meisten gängigen Betriebssystemen unterstützt (Microsoft Windows, Apple Mac OS X, Linux, etc.) und ist bspw. dem Betriebssystem Apple Mac OS X in der Version 1.8.7 standardmäßig integriert.²⁵ Bei Ruby handelt es sich um eine objekt-orientierte Programmiersprache mit dem Grundsatz *principle of least surprise* und folgt einigen Besonderheiten, wie z. B. einer einfachen Syntax, keiner typisierten Variablen und einer reinen Objektorientierung.²⁶ Abbildung 1 zeigt das Terminal von Apple Mac OS X mit typischen Ruby Kommandobefehlen. RoR nutzt diesen einfachen Syntax zur Entwicklung von Web-Applikationen, wobei aufgrund einfacherer Bedienung auf integrierte Entwicklungsumgebung zurückgegriffen wird, wie z. B. RadRails oder RubyMine.²⁷



```
1.9.3-p551 :001 > a = 5
=> 5
1.9.3-p551 :002 > b = 3
=> 3
1.9.3-p551 :003 > a + b
=> 8
1.9.3-p551 :004 > x = "hello"
=> "hello"
1.9.3-p551 :005 > y = "World"
=> "World"
1.9.3-p551 :006 > x + " " + y
=> "hello World"
1.9.3-p551 :007 > █
```

Abbildung 1 Terminalfenster unter Apple Mac OS X

Mit Hilfe des RoR Frameworks lassen sich dadurch schnell Web-Applikationen mit Da-

²⁴Vgl. Grimmer (2006)

²⁵Vgl. Wintermeyer (o. J.)

²⁶Vgl. ?, S. 297-298

²⁷Vgl. Hartl (2012), S. 10

tenbankbezug entwickeln, wobei der wesentlichen Vorteile in der Softwarearchitektur des Model-View-Controller-Paradigmas liegt.²⁸ Das Paradigma besagt, dass eine durch einen Browser angestoßene Anfrage an den Server durch den Rails **controller** verarbeitet wird. Der **controller** verarbeitet die Anfrage und leitet die nachfolgenden Schritte ein. Bei Web-Applikationen erfolgt eine solche Verarbeitung durch anzeigen bzw. dem sogenannten *rendern* von HTML-Dokumenten der Rails **views**, die von Browsern angezeigt werden können. Der **controller** rendert die **views** und ermöglicht weitere RoR-Befehle im HTML-Dokument. Bei komplexen und dynamischen Seiten übernimmt der **controller** geforderte Daten aus den Rails **models**, die wiederum mit einer Datenbank verbunden sind. Durch diese Architektur lassen sich umfangreiche und an spezifische Anfrage angepasste Web-Applikationen entwickeln. Ein weiterer Vorteil von RoR ist die einfache Implementierung von Unterprogrammen. Ein Unterprogramm ist in Ruby/RoR ein **Gemfile**, das durch den Bundler zur bestehenden Web-Applikation hinzugefügt wird.²⁹ Im nachfolgenden Kapitel wird die Entwicklung einer Web-Applikation mittels RoR beschrieben. Dabei liegt die Besonderheit der Ausarbeitung auf Integration eines notwendigen Unterprogramms (**Gemfile**) und die Verbindung zum Programm GAMS, damit das in diesem Kapitel vorgestellte Projektplanungsproblem gelöst werden kann.

3 Implementierung des RCPSP mittels Ruby on Rails

3.1 Darstellung der Funktionsweise der Anwendung anhand eines Userguides

Die Funktionsweise der mit *RoR* programmierten Anwendung „*Projektplanung*“ zur Lösung der Kapazitäts- und Kostenplanung des *RCPSP* lässt sich am anschaulichsten mit Hilfe eines Userguides darstellen. Neben der Besonderheiten, die durch das Problem der Projektplanung auftreten, können im selben Zuge auch die Spezifika der einzelnen Benutzerrollen aufgezeigt werden. Beachtet werden muss, dass die hier vorgestellte Web-Applikation auf der Arbeit von Hartl (2012) aufbaut.

Zunächst wird die Anwendung aus der Sicht eines Anwenders betrachtet, der sich nicht in die Applikation per Benutzererkennung eingeloggt hat. Konkret kann man sich darunter einen potentiellen Mitarbeiter des entsprechenden Projektes vorstellen, der sich über die Projektplanung informieren möchte, um sich gegebenenfalls als Mitarbeiter im Projekt (User) anzumelden. Im Testbetrieb wird der Ruby-Servers gestartet und durch Eingabe der URL „*http://localhost:3000/*“ in die Adresszeile eines beliebigen modernen Browsers wird die Startseite der Projektplanung angezeigt (siehe Abbildung 2). Alternativ ist der Betrieb

²⁸Vgl. Walter (2008), S. 463

²⁹Vgl. Hartl (2012), S. 9-17

auf einem Webserver möglich, sofern die benötigte Software installiert und betriebsbereit ist. Auf der Startseite hat der User zum einen die Möglichkeit, sich anzumelden bzw. sich einzuloggen, für den Fall, dass er bereits User der Anwendung ist.

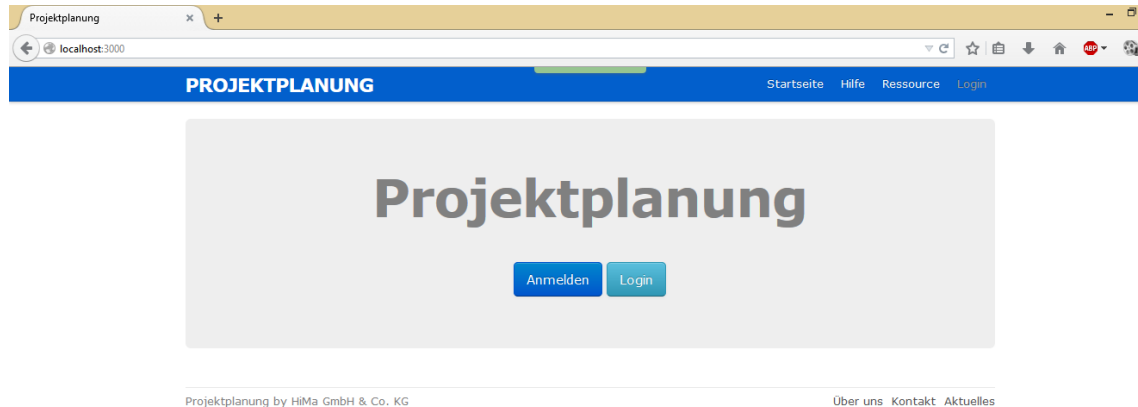


Abbildung 2 Startseite Projektplanung Applikation

Bei der Startseite (`home.html.rb`) der RoR-Applikation handelt es sich um eine statische Seite (`static_pages`) der `views`. Weiter gehören zu dieser Kategorie die HTML.RB-Dokumente die Seiten `about`, `contact`, `help` und `rcpsp`. Letztere wird zum späteren Zeitpunkt thematisiert. Ein Beispiel einer statischen Seite eines RoR `views` liefert Quellcode 1.

Quellcode 1 Code der Startseite (`app/views/static_pages/home.html.rb`)

```
<div class="center hero-unit">
  <h1>Projektplanung</h1>
  <br/>
  <br/>
  <% if signed_in? %>
    <th×%= link_to "Zur Optimierung", rcpsp_path, class: "btn btn-info btn-large" %></th>
  <%else %>
    <th×%= link_to "Anmelden", signup_path, class: "btn btn-primary btn-large" %></th>
    <th×%= link_to "Login", signin_path, class: "btn btn-info btn-large" %></th>
  <% end %>
</div>
```

Anhand der `static_pages` kann die Besonderheit von RoR deutlich gemacht werden. Durch das Model-View-Controller-Paradigma hilft der `static_pages.controller` bei der Verarbeitung von Anfragen. Es handelt sich hier um das typische ein Scaffolding (Bauprinzip) in RoR, bei dem ein `controller`, `models` und `views` erstellt werden.³⁰ Generiert werden

³⁰Vgl. Walter (2008), S. 464

können die Scaffolds durch einen Ruby-Befehl im Terminalfenster.

```
$ rails generate scaffold <name> <name:datentyp>
```

Wie der Name aber schon andeutet, bedarf es bei den statischen Seiten kaum der Verarbeitung von Datensätzen der RoR models zur Erstellung von dynamischen Seiten, wie der Quellcode 2 zeigt.

Quellcode 2 Controller der statischen Seiten
(app/controller/static_pages_controller.rb)

```
class StaticPagesController < ApplicationController
  def home
  end

  def help
  end

  def about
  end

  def contact
  end

  def rcpsp
    @project = Project.find(1)
  end
end
```

Für die `static_pages` bedarf es einen speziellen *Match*, der in `config/routes.rb` Datei hinterlegt wird (Vgl. Quellcode 3). Die `config/routes.rb` ordnet den Scaffolds und HTML-Dokumente spezifische Verzeichnisse in der Applikation zu. RoR erkennt die Unterseiten der angelegten Scaffolds und ermöglicht die Verlinkung der Seiten auch ohne spezifische Angabe.

Quellcode 3 config/routes.rb

```
class StaticPagesController < ApplicationController
SampleApp::Application.routes.draw do

  resources :projects
  resources :procedure_resources
  resources :procedures
  resources :resources
  resources :procedure_procedures
  resources :users
  resources :sessions, only: [:new, :create, :destroy]

  root to: 'static_pages#home'
```

```

match '/signup', to: 'users#new'
match '/signin', to: 'sessions#new'
match '/signout', to: 'sessions#destroy', via: :delete

match '/help', to: 'static_pages#help'
match '/about', to: 'static_pages#about'
match '/contact', to: 'static_pages#contact'
match '/rcpsp', to: 'static_pages#rcpsp'

match 'rcpsp/read_optimization_results', :to => 'rcpsps#↔
  read_optimization_results'
match 'rcpsp/optimize', :to => 'rcpsps#optimize'
match 'rcpsp/solution', to: 'rcpsps#solution'

match 'rcpsp/read_optimization_results2', :to => 'rcpsps#↔
  read_optimization_results2'
match 'rcpsp/optimize2', :to => 'rcpsps#optimize2'
match 'rcpsp/solution2', to: 'rcpsps#solution2'

```

Mit dem Link *Anmelden* erfolgt die Weiterleitung von der Startseite zur Anmeldeseite. Be-schließt sich der Besucher der Seite, sich für das Projekt anzumelden, muss er alle Felder des Anmeldeformulars befüllen.

Abbildung 3 Anmeldebildschirm

Neben dem Namen, einer Mailadresse und eines konformen Passwortes sind projektspezifische Informationen zur erfolgreichen Registrierung nötig. Im Feld *Arbeitszeit pro Tag* muss ein entsprechender Wert eingegeben werden, den der neue User bereit ist, pro Tag für das Projekt an Zeit zu investieren. Wird in diesem Feld keine ganze Zahl, sondern eine Dezimalzahl oder ein Wort eingegeben, kann die Anmeldung im System nicht stattfinden. Es wird ein Fehler angezeigt, der das Defizit aufzeigt und behoben werden muss (siehe Abbildung

4). Ausgelöst wird dieser Fehler durch einen Vermerk im zugehörigen RoR `models`, dass es sich um eine ganzzahlige Zahl handelt (*Integer*). Der Quellcode 4 zeigt dies anhand des hier betrachteten Beispiels.

Abbildung 4 Fehleranzeige bei Anmeldung

Quellcode 4 `app/models/users.rb`

```
class User < ActiveRecord::Base
  attr_accessible :email, :name, :password, :password_confirmation, :←
    capacity, :resource_id
  has_secure_password

  belongs_to :resource

  before_save { |user| user.email = email.downcase }
  before_save :create_remember_token

  validates :name, presence: true, length: { maximum: 50 }

  VALID_EMAIL_REGEX = /\A[\w+\-\.]+\@[a-z\d\-\\.]+\.[a-z]+\z/i
  validates :email, presence: true, format: { with: VALID_EMAIL_REGEX },
    uniqueness: { case_sensitive: false }
  validates :password, presence: true, length: { minimum: 6 }
  validates :password_confirmation, presence: true
  validates :capacity, :numericality => { :only_integer => true }

  private
  def create_remember_token
    self.remember_token = SecureRandom.urlsafe_base64
  end
end
```


Auf der Startseite (sowie allen anderen Seiten) sind eben Links wie *Hilfe* und *Kontakt* auch der Link zu *Ressourcen*-Übersicht, in der alle Ressourcen des Projektes gelistet sind. Hier kommt der Grundsatz von RoR zum Tragen: *Don't repeat yourself*. Der Gestaltung und der Aufbau einer jeden Seite in der Web-Applikation orientiert sich anhand der CSS-Stylesheets bzw. der Layout-Dateien. Die Layout-Dateien sind unter `app/views/` gelistet und definieren auf jeder Seite spezifische Bereiche. Die `application.html.erb` generiert für jede Seite dieses einheitliche Layout, unterstützt durch die Dateien `_footer.html.erb` und `_header.html.erb`. Im `_header.html.erb` ist der Link zur Ressourcen-Übersicht vermerkt (Vgl. Quellcode 5).

Quellcode 5 `app/views/layouts/_header.html.erb`

```
<header class="navbar navbar-fixed-top">
  <div class="navbar-inner">
    <div class="container">
      <%= link_to "Projektplanung", root_path, id: "logo" %>
      <nav>
        <ul class="nav pull-right">
          <li><%= link_to "Startseite", root_path, id: "main-links" %></li>
          <li><%= link_to "Hilfe", help_path, id: "main-links" %></li>
          <li><%= link_to "Ressource", resources_path, id: "main-links" %></li>
          <% if signed_in? %>
            <li><%= link_to "Mitarbeiter", users_path, id: "main-links" %></li>
            <li id="fat-menu" class="dropdown">
              <a href="#" class="dropdown-toggle" data-toggle="dropdown"> ←
                Menü <b class="caret"></b></a>
              <ul class="dropdown-menu">
                <li><%= link_to "Profil", current_user %></li>
                <li><%= link_to "Einstellungen", edit_user_path(←
                  current_user) %></li>
                <% if current_user.admin? %>
                  <li><%= link_to "Projektplanung", rcpsp_path %></li>
                <% end %>
                <li class="divider"></li>
                <li><%= link_to "Logout", signout_path, method: "delete" ←
                  %></li>
              </ul>
            </li>
          <% else %>
            <li><%= link_to "Login", signin_path %></li>
          <% end %>
        </ul>
      </nav>
    </div>
  </div>
</header>
```

Der `_header.html.erb` zeigt schon einige *If*-Befehle, mit denen unterschiedliche Daten anhand der Eigenschaften unangemeldeten, angemeldeten und Admin-Usern angezeigt werden. Durch folgen des Links *Ressource* wird die Index-Seite des der RoR `views/ressourcen` angezeigt (Siehe Abbildung ???). Der Quellcode 6 zeigt die notwendige Programmierung für die Seite.

Quellcode 6 `app/views/resources/index.html.erb`

```
<% provide(:title, 'Übersicht der Ressourcen') %>
<h1>Übersicht der Ressourcen</h1>

<% if signed_in? %>
<%= render 'resources/signed_in' %>
<% else %>
<%= render 'resources/free' %>
<% end %>
```

RoR durchläuft aufgrund der Aktivierung des Links die Aktion *index* des dazugehörigen Controllers `resources_controller.rb` und generiert die zugehörige HTML-Seite (`views`). Die Indexseite prüft, ob der aktuelle User angemeldet ist. Abhängig dieser Entscheidung integriert RoR einen unterschiedlichen Seiteninhalt. Sofern der aktuelle User nicht angemeldet ist, wird eine vereinfachte Ressourcen-Übersicht angezeigt (siehe Abbildung 3). In dieser Ansicht sind alle jeweilig aktuellen Ressourcen mit zugehörigen Namen aufgelistet, sowie dem Link *Bewerben*, der wiederum mit der Anmeldeseite verlinkt ist.

Findet keine Anmeldung in die Web-Applikation statt, sind keine weiterführenden Tätigkeiten möglich. Die Startseite liefert keine weiterführenden Informationen und bei der Eingabe von anderen Links in die Adresszeile des Browsers wird der aktuelle User zur *Login*-Seite geführt, da alle Daten für nicht angemeldete Anwender gesperrt sind. Um die Applikation nutzen zu können, ist demzufolge die Anmeldung als User zwingend notwendig. Findet diese entweder nach erstmaliger Registrierung über den Link *Anmelden* oder über *Login* statt, wird die eigene Profilseite angezeigt (siehe Abbildung 5).

Die Profilseite gibt einen Überblick über all die Daten, die für den User in Hinblick auf das Projekt relevant sind. Es werden die Daten dargestellt, die bei der Anmeldung angegeben wurden (Arbeitszeit, Rolle im Projekt) sowie die Vorgänge, die durch die Wahl der Ressource für diesen User relevant sind, in denen er also arbeiten muss. Der Aufbau der Seite ist im Quellcode 7 dargestellt.

Quellcode 7 `app/views/users/show.html.erb`

```
Quellcode ergänzen!!! Ist noch falsch in der App!!!
```

Zu jedem Vorgang wird die Dauer und gegebenenfalls die Zeitspanne angegeben, wann er je-

Projektplanung | Susi Sorglos

localhost:3000/users/2

PROJEKTPLANUNG Startseite Hilfe Ressource Mitarbeiter Menü

Susi Sorglos

Arbeitszeit pro Tag: 2

Rolle im Projekt: Ressource1

[Einstellung](#)

Name	Vorgangsdauer	FA*	SA*	FE*	SE*
Beispielvorgang1	2	0	0	2	2
Beispielvorgang2	5	2	2	7	7
Beispielvorgang4	3	11	11	14	14
Beispielvorgang6	4	7	10	11	14
Beispielvorgang7	5	14	14	19	19
Beispielvorgang9	2	7	17	9	19
Beispielvorgang10	3	2	11	5	14
Beispielvorgang11	5	5	14	10	19
Beispielvorgang12	2	19	19	21	21

[Zurück](#)

*FA=frühester Anfangszeitpunkt, SA=spätester Anfangszeitpunkt, FE=frühester Endzeitpunkt und SE=spätester Endzeitpunkt in Tagen nach Start des Projekts.

Abbildung 5 Profilseite eines Users

weils stattfindet. Die Grenze liegt zwischen dem frühesten Startzeitpunkt FA_i und spätesten Endzeitpunkt SE_i des Vorgangs i . Ebenfalls wird der kritische Pfad angezeigt. Dieser zeigt für die aufgeführten Vorgängen den Endzeitpunkt nach Start des Projekts unter Einhaltung der Ressourcenbeschränkung an, jeweils in Zeiteinheiten. Ob diese Tabelle mit Daten gefüllt ist, hängt davon ab, ob das Kapazitäts- bzw. Kostenplanungsproblem bereits gelöst wurde. Möchte der User seine Daten, wie z.B. die Wahl der Ressource oder die Quantität der Arbeitszeit, ändern, gelangt er über den Button *Einstellung* zu einer Seite, die äquivalent aufgebaut ist wie die Anmeldeseite, um dort seine Daten zu aktualisieren. Nach korrekter Eingabe können die Daten über den Button *Speichere Änderungen* gesichert werden. Auf der Profilseite erscheint daraufhin eine Anzeige *Profil updated* mit der Bestätigung, dass das Profil aktualisiert wurde. Im Vergleich zum fremden Anwender gelangt der angemeldete User außerdem in der Kopfzeile über den Link *Mitarbeiter* über eine Übersicht aller Mitarbeiter, die für das Projekt auf dieser Applikation angemeldet sind. Die Profilseite jedes Mitarbeiters kann betrachtet werden mit all den Informationen, die auch auf der eigenen Profilseite einzusehen sind. Es können jedoch keine Änderungen vorgenommen werden. Neben der Verlinkung zu der Übersicht der Mitarbeiter lässt sich in der Kopfzeile ein Feld *Menü* finden, dass die Unterpunkte *Profil*, *Einstellungen* und *Logout* enthält. Die Verlinkung *Profil* stellt eine Verlinkung zur Profilseite dar, unter *Einstellungen* kann das eigene Profil aktualisiert werden.

Unter *Ressourcen* kann der User, wie auch der nicht angemeldete Anwender, zur Übersicht der vorhandenen Ressourcen gelangen. Die Anzeige stellt sich für den angemeldeten User jedoch vielfältiger dar, als für den einfachen Anwender (siehe Abbildung 6), da hier eine andere Quellcode integriert wird (Vgl. Quellcode 6). Der Quellcode 8 zeigt den integrierten Inhalt.

PROJEKTPLANUNG				Startseite	Hilfe	Ressource	Mitarbeiter	Menu
Übersicht der Ressourcen								
Die Projektzusatzkosten betragen 504 Geldeinheiten. Die Optimierung erfolgte mit einem Datenstand vom 17.03.2015.								
Name	Gesamt- kapazität	Kosten/ ME	Grund- kosten	Zusatz- kosten/ ME	Zusatz- einheiten	Zusatzkosten Gesamt	Gesamtkosten	
Ressource1	10	2	62	8	63	504	566	Anzeigen
Ressource2	10	3	42	5	0	0	42	Anzeigen

Projektplanung by HiMa GmbH & Co. KG

Über uns Kontakt Aktuelles

Abbildung 6 Übersicht der Ressourcen für User

Quellcode 8 app/views/resources/_signed_in.html.erb

```

<% if @project.zwc!=nil %>
  Die Projektzusatzkosten betragen <strong>%= @project.zwc %</strong> ←
  Geldeinheiten. Die Optimierung erfolgte mit einem Datenstand vom <←
  strong>%= @project.updated_at.strftime("%d.%m.%Y") %>.</strong><br>←
  <br>
<% end %>

<table id="resources" class="table table-hover">
  <thead>
    <tr>
      <th>Name</th>
      <th>Gesamt- kapazität</th>
      <th>Kosten/ ME</th>
      <th>Grund- kosten</th>
      <th>Zusatz- kosten/ ME</th>
      <th>Zusatz- einheiten</th>
      <th>Zusatzkosten Gesamt</th>
      <th>Gesamtkosten</th>
      <th></th>
      <th></th>
      <th></th>
    </tr>
  </thead>
  <tbody>

  <% @resources.each do |resource| %>
    <tr>
      <td>%= resource.name %</td>
      <td>%= User.sum(:capacity, :conditions => {:resource_id => resource←
      }) %</td>
      <td>%= resource.cost %</td>
      <th>%= resource.cost * resource.procedures.sum(&:prot) %</th>
      <td>%= resource.ocr %</td>

```

```

        <td>=<%= resource.oce %></td>
        <th>=<%= resource.ocr * resource.oce %></th>
        <th>=<%= resource.ocr * resource.oce + (resource.cost * resource.←
            procedures.sum(&:prot) ) %></th>
        <td>=<%= link_to 'Anzeigen', resource %></td>
        <% if signed_in? && current_user.admin? %>
            <td>=<%= link_to 'Ändern', edit_resource_path(resource) %></td>
            <td>=<%= link_to 'Löschen', resource, method: :delete, data: { ←
                confirm: 'Sind sie sicher?' } %></td>
        <% end %>
    </tr>
<% end %>
</tbody>
</table>

<br />

<% if current_user.admin? %>
    <div class="btn-group btn-lg">
        <%= link_to 'Neue Ressource anlegen', new_resource_path, :class => "←
            btn" %>
        <%= link_to 'Zurück zur Projektplanung', rcpsp_path, :class => "btn" ←
            %>
    </div>
<% end %>

```

Für den User sind alle Eigenschaften der verschiedenen Ressourcen einsehbar. Es werden die Gesamtkapazität, Kosten pro ME, Grundkosten und Zusatzkosten pro ME angezeigt. Wurde bereits eine Lösung für das Problem der Kostenplanung ermittelt, werden die kalkulierten Werte für die Zusatzeinheiten, gesamten Zusatzkosten und die Gesamtkosten pro Ressource dargestellt. Zudem wird der Zielfunktionswert, bei der Kostenplanung die gesamten anfallenden Zusatzkosten, in Verbindung mit dem Zeitpunkt der Optimierung über der Tabelle dargestellt. Die Darstellung der Tabellen in dieser Web-Applikation orientiert sich an dem Bootstrap-Framework³¹. Alternativ bietet die App die Anzeige der Tabellen anhand einer JavaScript-Tabelle.³² Über den Button *Anzeigen* in der hier betrachteten Tabelle sind die Eigenschaften einer Ressource separat einsehbar. Da der User bzw. Mitarbeiter in diesem Modell durch die Planung innerhalb des Projektes eingeteilt wird und seine Rechte nicht über die Organisation der eigenen Daten hinaus reicht, hat er keine weiteren Kompetenzen bei der Nutzung dieser Applikation.

Die Verwaltung der Mitarbeiter und die Organisation sowie Durchführung der Projektplanung kann ausschließlich nach der Anmeldung als Administrator erfolgen. Der Admin gilt in

³¹<http://getbootstrap.com>

³²Auf Implementierung wurde jedoch aufgrund der möglichen Inkompatibilität zu bestimmten Browser und aufgrund der Laufzeitverbesserung der Web-Applikation verzichtet.

dieser Anwendung als durchführende Gewalt, er trägt den Namen *Example User*. Nach der erfolgreichen Anmeldung erscheint zunächst erneut die Profilseite, aufgebaut wie beim einfachen User. Im Gegensatz zu normalen Usern bietet die Seite dem Admin jedoch zusätzliche Handlungsspielräume neben der einfachen Auflistung der Vorgänge (siehe Abbildung 7). Er hat die Möglichkeit, die Dauer der Vorgänge abzuändern oder Vorgänge aus dem Projekt zu löschen. In gleicher Weise stellt sich die Ausweitung der Kompetenzen bei den Ressourcen dar. Über die Verlinkung *Ressourcen* können nun die Ressourcen ebenfalls gelöscht oder die Eigenschaften (Kosten und Zusatzkosten je ME) verändert werden. Desweiteren kann über den Button *Neue Ressource anlegen* eben dies vollzogen werden.


PROJEKTPLANUNG									
<div>Startseite Hilfe Ressource Mitarbeiter Menü</div> <div>  Example User </div> <div>Arbeitszeit pro Tag: 2</div> <div>Rolle im Projekt: Ressource1</div> <div>Einstellung</div>									
Name	Vorgangsdauer	FA*	SA*	FE*	SE*				
Beispielvorgang1	2	0	0	2	2	Anzeigen	Ändern	Löschen	
Beispielvorgang2	5	2	2	7	7	Anzeigen	Ändern	Löschen	
Beispielvorgang4	3	11	11	14	14	Anzeigen	Ändern	Löschen	
Beispielvorgang6	4	7	10	11	14	Anzeigen	Ändern	Löschen	
Beispielvorgang7	5	14	14	19	19	Anzeigen	Ändern	Löschen	
Beispielvorgang9	2	7	17	9	19	Anzeigen	Ändern	Löschen	
Beispielvorgang10	3	2	11	5	14	Anzeigen	Ändern	Löschen	
Beispielvorgang11	5	5	14	10	19	Anzeigen	Ändern	Löschen	
Beispielvorgang12	2	19	19	21	21	Anzeigen	Ändern	Löschen	
<div>Zurück</div> <div>*FA=frühester Anfangszeitpunkt, SA=spätester Anfangszeitpunkt, FE=frühester Endzeitpunkt und SE=spätester Endzeitpunkt in Tagen nach Start des Projekts.</div>									

Abbildung 7 Profilseite des Administrators

PROJEKTPLANUNG									
<div>Startseite Hilfe Ressource Mitarbeiter Menü</div> <div>Übersicht der Ressourcen</div> <div>Die Projektzusatzkosten betragen 504 Geldeinheiten. Die Optimierung erfolgte mit einem Datenstand vom 17.03.2015.</div>									
Name	Gesamt-kapazität	Kosten/ME	Grund-kosten	Zusatz-kosten/ME	Zusatz-einheiten	Zusatzkosten Gesamt	Gesamtkosten		
Ressource1	10	2	62	8	63	504	566	Anzeigen	Ändern
Ressource2	10	3	42	5	0	0	42	Anzeigen	Ändern
<div>Neue Ressource anlegen</div> <div>Zurück zur Projektplanung</div>									
<div>Projektplanung by HiMa GmbH & Co. KG</div> <div>Über uns Kontakt Aktuelles</div>									

Abbildung 8 Übersicht der Ressourcen aus Sicht des Administrators

Um nun zur Kernaufgabe der Applikation, der Projektplanung, zu gelangen, kann entweder der Button *Zurück zur Projektplanung* getätigt werden, oder in der Kopfzeile wird unter *Menü* der Unterpunkt *Projektplanung* ausgewählt. Wie bereits erwähnt, hat nur der Admin

die Berächtigung, dieses Menü aufzurufen. Im oberen Bereich der Seite sind die Verlinkungen zur Verwaltung der nötigen Inputs zur Lösung beider Planungsproblematiken angesiedelt. Das Problem der Projektplanung, das *RCPSP*, besteht aus der Kapazitätsplanung und Kostenplanung (siehe Kapitel ..).

4 Kritische Würdigung des Anwendungssystems

5 Fazit

Literatur

- Bartels, J.H. (2009): Projektplanung–Grundlagen und Anwendungsbeispiele. In: Anwendung von Methoden der ressourcenbeschränkten Projektplanung mit multiplen Ausführungsmodi in der betriebswirtschaftlichen Praxis. Springer, S. 7–42.
- Demeulemeester, E. und Herroelen, W. (2011): Robust project scheduling. Bd. 3. Now Publishers Inc.
- DIN 69900 (2009): Projektmanagement - Netzplantechnik; Beschreibung und Begriffe. In: Berlin: Beuth.
- Drexl, A.; Kolisch, R. und Sprecher, A. (1997): Neuere Entwicklungen in der Projektplanung. In: Schmalenbachs Zeitschrift für betriebswirtschaftliche Forschung, S. 95–120.
- Felkai, R. und Beiderwieden, A. (2011): Analysieren und Formulieren von Projektzielen. In: Projektmanagement für technische Projekte. Springer, S. 45–64.
- Grimmer, L. (2006): Interview with David Heinemeier Hansson from Ruby on Rails. <https://web.archive.org/web/20130225091835/http://dev.mysql.com/tech-resources/interviews/david-heinemeier-hansson-rails.html>.
- Hartl, M. (2012): Ruby on Rails Tutorial: Learn Web Development with Rails. Pearson Education.
- Kellenbrink, C. (2014): Einführung in die ressourcenbeschränkte Projektplanung. In: Ressourcenbeschränkte Projektplanung für flexible Projekte. Springer, S. 5–18.
- Neumann-Braun, K.; Schwindt, C. und Zimmermann, J. (2003): Project scheduling with time windows and scarce resources: temporal and resource-constrained project scheduling with regular and nonregular objective functions. Springer.
- Pritsker, A.A.B.; Waiters, L.J. und Wolfe, P.M. (1969): Multiproject scheduling with limited resources: A zero-one programming approach. In: Management science. Bd. 16, Nr. 1, S. 93–108.
- Talbot, F.B. (1982): Resource-constrained project scheduling with time-resource tradeoffs: The nonpreemptive case. In: Management Science. Bd. 28, Nr. 10, S. 1197–1210.
- Voigt, K.I. und Schewe, G. (2014): Definition Projekt Version 7 - Gabler Wirtschaftslexikon.
- Walter, T. (2008): Das Ruby-Framework Ruby on Rails. In: Kompendium der Web-Programmierung. Springer, S. 463–491.
- Wintermeyer, S. (o. J.): Installation von Ruby on Rails 3.2 mit RVM. <http://ruby-auf-schienen.de>. <http://ruby-auf-schienen.de/3.2/rails3-install-osx.html>.

Zimmermann, J.; Stark, C.; Rieck, J. et al. (2006): Projektmanagement. In: Projektplanung. Springer Berlin Heidelberg, S. 1–113.

A Anhang

A.1 GAMS-Implementierung des Beispiels

A.2 Ruby on Rails Programmcode