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

$\mathbf{A}$	bbildungsverzeichnis	iii
$\mathbf{A}$	bkürzungsverzeichnis	iii
Sy	ymbolverzeichnis	vi
1	Einleitung	1
2	Grundlagen zur ressourcen-beschränkten Projektplanung und zu dem Framework Ruby on Rails  2.1 Kapazitätsplanung	2 2 4 6
3	Implementierung des RCPSP mittels Ruby on Rails 3.1 Darstellung der Funktionsweise der Anwendung anhand eines Userguides	<b>7</b> 7
4	Kritische Würdigung des Anwendungssystems	18
5	Fazit	20
6	Fazit	20
Li	teratur	21
$\mathbf{A}$	Anhang A.1 Datenbankschema	23 23 23 30

# Abbildungsverzeichnis

1	Terminalfenster unter Apple Mac OS X	(
2	Startseite Projektplanung Applikation	8
3	Anmeldebildschirm	Ć
4	Profilseite eines Users	10
5	Übersicht der Ressourcen für User	11
6	Profilseite des Administrators	13
7	Übersicht der Ressourcen aus Sicht des Administrators	13
8	Projektplanung mit dem RCPSP - Übersicht	14
9	Fehler aufgrund eines Zyklus in der topologischen Reihenfolge	15
10	Einstellung des Starttermins anhand eines Kalendermenüs	16
11	Ergebnis der Kapazitätsplanung	16
12	Ergebnis der Kapazitätsplanung	18
13	Datenbankschema der Web-Applikation Projektplanung	23

# ${\bf Abk\"{u}rzungsverzeichnis}$

Admin	Administrator
GAMS	General Algebraic Modeling System
RCPSP	Resource-Constrained Project Scheduling
RoR	Ruby on Rails
SGS	Schedule Generation Scheme

# Quellcodeverzeichnis

1	Ausschnitt aus dem RoR-Controller für das RCPSP	7
2	GAMS-Code zur Kapazitätsplanung	3
3	GAMS-Code zur Kostenplanung	6
4	Gemfile der Web-Applikation Projektplanung	0
5	Routes-Datei der Web-Applikation Projektplanung	1
6	RoR-Controller für die Vorgangsrelationen	2
7	RoR-Controller für die Vorgangs-Ressourcen-Kombinationen	4
8	RoR-Controller für die Vorgänge	6
9	RoR-Controller für das Projekt	9
10	RoR-Controller für das RCPSP	1
11	RoR-Controller für die Ressourcen	8
12	RoR-Controller für die statischen Seiten	1
13	RoR-Controller für die Users	1
14	RoR-Modell für die Vorgangsrelationen	3
15	RoR-Modell für die Vorgangs-Ressourcen-Kombinationen	3
16	RoR-Modell für die Vorgänge	3
17	RoR-Modell für das Projekt	3
18	RoR-Modell für die Ressourcen	4
19	RoR-Modell für die Users	4
20	RoR-Seite für die Vorgangsrelationen - Formular	5
21	RoR-Seite für die Vorgangsrelationen - Übersicht	5
22	RoR-Seite für die Vorgangsrelationen - Erstellung	6
23	RoR-Seite für die Vorgangsrelationen - Grafische Darstellung	6
24	RoR-Seite für die Vorgangs-Ressourcen-Kombinationen - Formular 5	7
25	RoR-Seite für die Vorgangs-Ressourcen-Kombinationen - Bearbeitung 56	8
26	RoR-Seite für die Vorgangs-Ressourcen-Kombinationen - Übersicht 5	8
27	RoR-Seite für die Vorgangs-Ressourcen-Kombinationen - Erstellung 59	9
28	RoR-Seite für die Vorgangs-Ressourcen-Kombinationen - Anzeige 59	9
29	RoR-Seite für die Vorgänge - Formular	0
30	RoR-Seite für die Vorgänge - Bearbeitung	0
31	RoR-Seite für die Vorgänge - Übersicht	0
32	RoR-Seite für die Vorgänge - Erstellung	2
33	RoR-Seite für die Vorgänge - Anzeige	2
34	RoR-Seite für die Ressourcen - Formular	4
35	RoR-Seite für die Ressourcen - Tabelle als unangemeldeter User 6	5
36	RoR-Seite für die Ressourcen - Tabelle als angemeldeter User	5
37	RoR-Seite für die Ressourcen - Bearbeitung 6	7
38	RoR-Seite für die Ressourcen - Übersicht	7

39	RoR-Seite für die Ressourcen - Erstellung	<sup>37</sup>
40	RoR-Seite für die Ressourcen - Anzeige	67
41	RoR-Seite für die Optimierungsseite zur Projektplanung	38
42	RoR-Seite für die Startseite	70
43	Kopfzeile der Web-Apllikation	71
44	RoR-Seite bzgl. der Lösung von Usern über die Übersichtsseite	72
45	RoR-Seite für die User - Bearbeitung	72
46	RoR-Seite für die User - User-/Mitarbeiterübersicht	73
47	RoR-Seite für die User - Erstellung	73
48	RoR-Seite für die User - Anzeige	74
49	RoR-Datenbankschema	75
50	Beispieldaten für die Datenbank	77

## Symbolverzeichnis

 $d_i$  Dauer von Vorgang i

 $FE_i$  frühestes Ende von Vorgang i

i, h = 1, ..., 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, ..., R Ressourcen

 $SE_i$  spätestes Ende von Vorgang i

 $t, \tau = 1, ..., T$  Perioden

 $\mathcal{V}_i$  Menge der direkten Vorgänger von Vorgang i

 $X_{it} \in \{0,1\}$  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.<sup>1</sup> Dementsprechend von großer Bedeutung ist die vorhergehende und genaue Planung von Projekten.<sup>2</sup> 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.<sup>2</sup> 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.<sup>3</sup> Projektziele können dabei unterschiedlich kategorisiert werden, z. B. in Sach-, Termin- oder Kostenziele.<sup>4</sup>

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 Ablaufelement zur Beschreibung eines bestimmten Geschehens.<sup>5</sup> Trotz der Zerlegung besitzen die einzelnen Arbeitsgänge des Projekts eine Beziehung, mit der die Reihenfolge der Ablauffolge bestimmbar ist.<sup>6</sup> Oft wird zur Darstellung der Vorgangsrelationen ein Vorgangsknoten-Netzplan verwendet.<sup>7</sup> 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.<sup>8</sup> 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. 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

<sup>&</sup>lt;sup>1</sup>Vgl. Voigt und Schewe (2014)

<sup>&</sup>lt;sup>2</sup>Vgl. Zimmermann et al. (2006), S. VI

<sup>&</sup>lt;sup>3</sup>Vgl. Zimmermann et al. (2006), S. 4

<sup>&</sup>lt;sup>4</sup>Vgl. Felkai und Beiderwieden (2011), S. 52

<sup>&</sup>lt;sup>5</sup>Vgl. DIN 69900 (2009), S. 15

<sup>&</sup>lt;sup>6</sup>Vgl. Kellenbrink (2014), S. 6-7

<sup>7?????</sup> 

<sup>&</sup>lt;sup>8</sup>Vgl. Bartels (2009), S. 11-12

<sup>&</sup>lt;sup>9</sup>????

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. <sup>10</sup> Es baut auf der Programmiersprache Ruby auf und ist ursprünglich von David Heinemeier Hansson entwickelt. <sup>11</sup> Die Implementierung bedarf einer Verknüpfung von RoR und GAMS <sup>12</sup>. Unter GAMS wird eine algebraische Modellierungssprache für mathematische Optimierungsprobleme verstanden, mit der das RCPSP gelöst wird. <sup>13</sup> 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 6.

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

## 2.1 Kapazitätsplanung

Ein Großteill an Projekten besitzt die Eigenschaft eines beschränkten Ressourcenkontingents. Abeitselber Schaft der Verfüglichen Arbeitsgänge sichergestellt werden. Mit der Einhaltung des 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. 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. Die im folgenden aufgestellte Zielfunktion des RCPSP zur Minimierung der Projektdauer ist die gängige Version

<sup>10???</sup> 

 $<sup>^{11}???</sup>$ 

<sup>&</sup>lt;sup>12</sup>General Algebraic Modeling System

<sup>13777</sup> 

<sup>&</sup>lt;sup>14</sup>Vgl. Kellenbrink (2014), S. 11

 $<sup>^{15}\</sup>mathrm{Vgl.}$  Neumann-Braun et al. (2003), S. 21-22

<sup>&</sup>lt;sup>16</sup>Vgl. Demeulemeester und Herroelen (2011), S. 23

der Kapazitätsplanung, <sup>17</sup> andere Variationen sind aber ebenfalls möglich. <sup>18</sup>

Nachfolgend wird das deterministische RCPSP in diskreter Zeit formuliert. <sup>19</sup> Charakteristisch für eine mathematische Modellformulierung in diskreter Zeit sind die Zeiteinheiten, die den Perioden  $t, \tau$  entsprechen.

#### Modell RCPSP

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

unter Beachtung der Restriktionen

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

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

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

$$X_{it} \in \{0, 1\}$$
  $i \in \mathcal{I}; t \in \{FE_i, ..., SE_i\}$  (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-

<sup>&</sup>lt;sup>17</sup>Vgl. Drexl et al. (1997), S. 98

<sup>&</sup>lt;sup>18</sup>Vgl. Talbot (1982), S. 1200

<sup>19????</sup> 

zugefügt, welche mit einer Dauer von 0 Zeiteinheiten bewertet werden. <sup>20</sup> 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. <sup>21</sup> Um den Fertigstellungszeitpunkt der einzelnen Arbeitsgänge i festlegen zu können, wird der Modellformulierung in diskreter Zeit die binäre Entscheidungsvariable  $X_{it}$  hinzugefügt. <sup>22</sup> 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 Reihenfolgerestriktion 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 Kapaiztätsbegrenzung 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  $t = \{FE_i, ..., SE_i\}$  formal definiert. Aufgrund der Reihenfolgebeziehung (3) darf der jeweils betrachtete Arbeitsgang nur in diesem Zeitraum fertiggestellt werden. Die gemischtganzzahlige Modellformulierung lässt sich durch Standard-Lösungsverfahren exakt lösen.<sup>23</sup>

## 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,

 $<sup>^{20}</sup>$ Vgl. Zimmermann et al. (2006), S. 66

<sup>&</sup>lt;sup>21</sup>Vgl. Kellenbrink (2014), S. 12

<sup>&</sup>lt;sup>22</sup>Vgl. Pritsker et al. (1969), S. 94

<sup>&</sup>lt;sup>23</sup>z. 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+

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

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

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

$$O_{rt} \ge 0$$
  $r = 1, ..., R; t = 1, ..., T$  (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 Nebenbedienung (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.<sup>24</sup> Mit dem Name von RoR wird klar, das das Framework die Programmiersprache Ruby nutzt. Ruby wird von den den meisten gängigen Betriebssystem 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.<sup>25</sup> 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 Sprachsyntax, keiner typisierten Variablen und einer reinen Objektorientierung.<sup>26</sup> 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.<sup>27</sup>

```
● ● ■ sample_app — ruby — 80×24

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-

<sup>&</sup>lt;sup>24</sup>Vgl. Grimmer (2006)

<sup>&</sup>lt;sup>25</sup>Vgl. Wintermeyer (o. J.)

<sup>&</sup>lt;sup>26</sup>Vgl. ?, S. 297-298

<sup>&</sup>lt;sup>27</sup>Vgl. Hartl (2012), S. 10

tenbankbezug entwicklen, wobei der wesentlichen Vorteile in der Softwarearchitektur des Model-View-Controller-Paradigmas liegt.<sup>28</sup> 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 HTLM-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 sind umfangreiche und an spezifische Anfrage angepasste Web-Applikationen entwickeln. Ein weiterer Vorteil von RoR ist die einfache Implementierung von Unterprogrammen. Ein Unterprogramme ist in Ruby/RoR ein Gemfile, das durch den Bundler zur bestehenden Web-Applikation hinzugefügt wird.<sup>29</sup> 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 Projektplanungsprobem 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 Kapizitäts- und Kostenplanung des RCPSP lässt sich am anschaulichsten mit Hilfe eines Userguides darstellen. Die Applikation kann über nachfolgenden Terminalbefehl auf ein lokales Computerverzeichnis geklont werden.

### \$ git clone https://github.com/rb4k/as-rcpsp.git

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

<sup>&</sup>lt;sup>28</sup>Vgl. Walter (2008), S. 463

<sup>&</sup>lt;sup>29</sup>Vgl. Hartl (2012), S. 9-17

(User) anzumelden. Im Testbetrieb wird der Ruby-Servers gestartet und durch Eingabe der URL "http://localhost:3000/" in die Adresszeile eines beliebigen modernen Browers wird die Startseite der Projektplanung angezeigt (siehe Abbildung 2). Alternativ ist der Betrieb 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 der 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 42 im Anhang A.3.

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.<sup>30</sup> Generiert werden 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 12 im Anhang A.3 zeigt.

Für die static\_pages bedarf es einen speziellen *Match*, der in config/routes.rb Datei hinterlegt wird (Vgl. Quellcode 5). Die config/routes.rb ordnet den Scaffolds und HMTL-Dokumente spezifische Verzeichnisse in der Applikation zu. RoR erkennt die Unterseiten der

<sup>&</sup>lt;sup>30</sup>Vgl. Walter (2008), S. 464

angelegten Scaffolds und ermöglicht die Verlinkung der Seiten auch ohne spezifische Angabe (Vgl. Quellcode 5 im Anhang A.3).

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

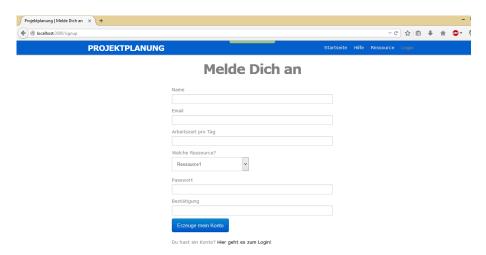


Abbildung 3 Anmeldebildschirm

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 ??). Ausgelöst wird dieser Fehler durch einen Vermerk im zugehörigen RoR models, dass es sich um eine ganzahlilge Zahl handelt (Integer). Der Quellcode 19 im Anhang ?? zeigt dies anhand des hier betrachteten Beispiels.

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 43 im Anhang 43).

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/ressources angezeigt (Siehe Abbildung ???). Der Quellcode 38 im Anhang A.3 zeigt die notwendige Programmierung für die Seite.

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 Browers 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 4).

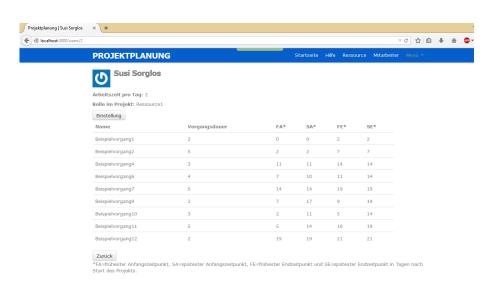


Abbildung 4 Profilseite eines Users

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 48 im Anhang A.3 dargestellt.

Zu jedem Vorgang wird die Dauer und gegebenenfalls die Zeitspanne angegeben, wann er jeweils 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 5), da hier eine andere Quellcode integriert wird (Vgl. Quellcode 38). Der Quellcode 36 im Anhang A.3 zeigt den integrierten Inhalt.

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			63	504	566	Anzeigen				
Ressource2	10	3	42	5	0	0	42	Anzeigen				

Abbildung 5 Übersicht der Ressourcen für User

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-Framework<sup>31</sup>. Alternativ bietet die App die Anzeige der Tabellen anhand einer JavaScript-Tabelle.<sup>32</sup> 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 dieser Anwendung als durchführende Gewalt. In dieser Testsituation ist *Example User*" mit den dafür notwendigen Berechtigungen ausgestattet. Alternativ lässt sich durch Änderung der booleschen Variable admin = true der Datenbank zum RoR models/users die Eigenschaft auch auf andere Datensätze (User) übertragen. Nach der erfolgreichen Anmeldung als User mit administrativen Rechten erscheint zunächst erneut die Profilseite, sofern die Anmeldung über die Startseite erfolgt. Im Gegensatz zu normalen Usern bietet die Seite eines Admins jedoch zusätzliche Handlungsspielräume neben der einfachen Auflistung der Vorgänge (siehe Abbildung 6). Er hat die Möglichkeit, die Dauer der Vorgänge abzuändern oder Vorgänge aus dem Projekt zu löschen. Dies wird wieder über einen *If*-Befehl gesteuert, wie der Quellcode 48 aus dem Anhang A.3 zeigt.

In gleicher Weise stellt sich die Ausweitung der Kompetenzen bei den Ressourcen dar. Uber die Auswahl des Links Ressourcen über den Header wird zur Ressourcenübersicht verbunden und dort können nun die Ressourcen ebenfalls gelöscht oder die Eigenschaften (Kosten und Zusatzkosten je ME) verändert werden. Des Weiteren kann über den Button Neue Ressource anlegen eben dies vollzogen werden (Vgl. Quellcode 38 im Anhang A.3).

Um nun zur Kernaufgabe der Applikation, der Projektplanung, zu gelangen, kann entweder der Button Zurück zur Projektplanung auf der Seite zur Ressourcenübersicht getätigt werden, oder ausgehend von jeder beliebigen Seite der Web-Applikation in der Kopfzeile (Header) wird unter Menü der Unterpunkt Projektplanung ausgewählt (siehe Abbildung 8).

<sup>&</sup>lt;sup>31</sup>http://getbootstrap.com

<sup>&</sup>lt;sup>32</sup>Auf Implementierung wurde jedoch aufgrund der möglichen Inkompatibilität zu bestimmten Browser und aufgrund der Laufzeitverbesserung der Web-Applikation verzichtet.

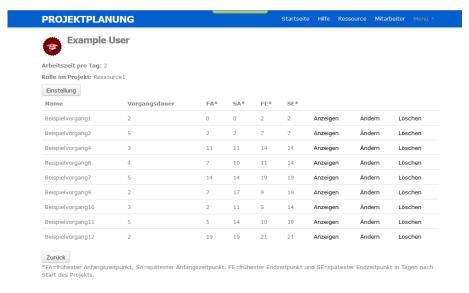


Abbildung 6 Profilseite des Administrators



Abbildung 7 Übersicht der Ressourcen aus Sicht des Administrators

Bei dieser statischen Seite fließen Daten aus dem RoR models/project ein. Bei diesem Modell handelt es sich um eine Hilfsdatenbank ohne weiterer Beziehung zu anderen Modellen (Vgl. Abbildung 13 im Anhang A.1). Sie fungiert als Datenbank für unabhängige Parameter und hat damit nur einen Datensatz. Der Controller der statischen Seiten ruft über die Aktion rcpsp diesen Datensatz auf (Vgl. Quellcode 12 aus Anhang A.3). Dadurch kann das HTML.RB-Dokument views/static\_pages/ den Datensatz aufgreifen und Formularfelder zur Eingabe der unabhängigen Parameter bereitstelle. Zu den unabhängigen Parametern dieses Formulars zählen die Datenfelder path, startdate und deadline, auf die im Verlauf der weiteren Beschreibung der Web-Applikation näher eingegangen wird.

Im oberen Bereich der Seite sind die Verlinkungen zur Verwaltung der nötigen Inputs zur Lösung beider Planungsproblematiken angesiedelt. Neben den bereits behandelten Links zu den Vorgängen und Ressourcen finden sich Verlinkungen zu den Vorgangsrelationen und Vorgang-Ressourcen-Kombinationen.

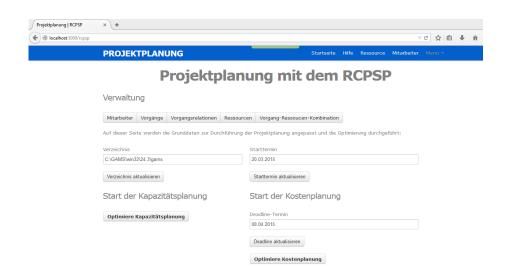


Abbildung 8 Projektplanung mit dem RCPSP - Übersicht

Die Übersicht der Relationen zwischen den Vorgängen stellt eine Auflistung eines jeden Vorgänger und Nachfolger dar. Ein Admin kann diese Relationen löschen oder neue anlegen. Wenn er sich dazu entschließt, eine neue anzulegen, ist zu beachten, dass ein Strukturplan eines Projektes keine Zyklen beinhalten darf. Damit Zyklen verhindert werden, findet beim Prozess des Anlegens einer neuen Vorgangsrelation eine Prüfung statt. Beinhaltet die neu angelegte Relation einen Zyklus, tritt ein Fehler auf und die Relationen muss überarbeitet werden (siehe Abbildung 9). So wird verhindert, dass der Strukturplan Zyklen enthält. Dieser Vorgang wird gesteuert durch die dafür zuständige Aktion create aus dem RoR controllers/procedure-procedures-controller.rb (Vgl. Quellcode 6 im Anhang A.3). Inbegriffen in die Aktion ist ein frei-verfügbares Unterprogramm (Gem)<sup>33</sup>. In Kapitel ?? wird die Integration und Funktionsweise dieses Unterprogramms beschrieben.

Der Button Vorgang-Ressourcen-Kombination führt zu einer Übersicht der verschiedenen Ressourcen zu den Vorgängen. Neben der Auflistung können die Kombinationen verändert, gelöscht oder neu erstellt werden. Bei der Veränderung oder Erstellung ist zu beachten, dass die Angabe des Kapazitätsbedarfs nur mit Hilfe einer ganzen Zahl erfolgen darf. Entsprechend der Kapazitätsangabe bei der Bearbeitung eines Profils erscheint bei jeder anderen Art von Eingabe ein Fehler, der die Datenspeicherung verhindert (Vgl. Quellcode 15 im Anhang A.3).

Nachdem all diese Daten geprüft und gegebenenfalls verändert wurden, steht die Basis sowohl für die Optimierung der Kapazitäts- als auch der Kostenplanung. Bevor der Optimierungsprozess stattfinden kann, müssen noch einige Rahmenbedingungen geprüft werden. Da die

<sup>&</sup>lt;sup>33</sup>https://github.com/monora/rgl



Abbildung 9 Fehler aufgrund eines Zyklus in der topologischen Reihenfolge

Optimierung mit dem Programm *GAMS* stattfindet, muss die Applikation auf dieses Programm zurückgreifen können. Dafür muss *GAMS* auf dem hiesigen Computer installiert sein. Nach der Recherche des Installationsortes muss der korrekte Pfad in das dafür vorgesehene Feld der Übersichtsseite zur Projektplanung eingetragen werden, in dem der Beispielpfad zu sehen ist. Nach der Eingabe wird der Pfadzugriff durch *Verzeichnis aktualisieren* in der Datenbank des RoR models/project gesichert (siehe Abbildung 8). Neben den Programmpfad muss ein Termin ausgewählt werden, zu dem das Projekt startet (siehe Abbildung 10). Anhand dieses Startdatums werden alle Daten bezüglich der Vorgänge berechnet, zudem stellt der Starttermin bei der Kostenplanung einen wichtigen Faktor dar. Durch die Betätigung des Feldes, das ein Muster anzeigt, öffnet sich ein Kalendermenü, in dem ein beliebiges Datum ausgewählt werden kann. Bei dem Datumsfeld handelt es sich ebenfalls um eine Applikationserweiterung (Gem) namens "Bootstrap-Datepicker-Rails"<sup>34</sup> (Vgl. Quellcode ?? im Anhang A.3). Es handelt sich hier um eine Unterprogramm inkl. dazugehöriger JavaScrip-Datei.

Nachdem das GAMS-Verzeichnis und der Starttermin eingestellt wurden, kann die Kapazitätsplanung durch die Betätigung des Buttons Optimiere Kapazitätsplanung durchgeführt werden. Es handelt sich hier um die Aktion optimize des RoR rcpsps\_controller (Vgl. Quellcode 10 im Anhang A.3). Die Aktion dient dazu die Include-Dateien für die GAMS-Optimierung zu schreiben und eben diese durch Aufrufen der GAMS-Software zu starten. Bei der GAMS-Optimierung handelt es sich um die Datei mit dem Quellcode 2 aus Anhang A.2. Nach einer Rechenzeit, währenddessen der Button, mit dem die Optimierung gestartet wurde, auf den Rechenprozess hinweist, leitet die Applikation den Admin direkt zu der Übersicht der Vorgänge. Dieser Schritt wird durch die Hilfsaktion solution des RoR rcpsps\_controller unterstütz, die parallel aufgerufen wird. Nachdem die GAMS-

<sup>&</sup>lt;sup>34</sup>https://github.com/Nerian/bootstrap-datepicker-rails

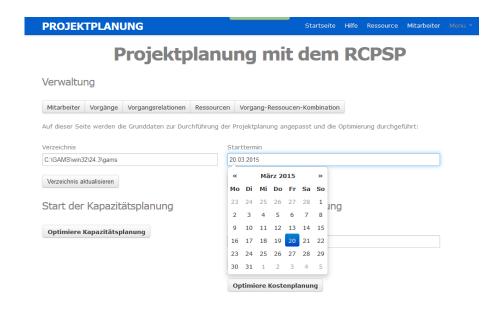


Abbildung 10 Einstellung des Starttermins anhand eines Kalendermenüs

Optimierung vollzogen ist, liest der RoR rcpsps\_controller die von der GAMS-Optimierung erstellten Text-Dateien ein und schreibt diese direkt in die dafür vorgesehen Datenbank. Bei der Kapazitätsplanung wird hauptsächlich die Datenbank des RoR models/procedure angesprochen. Wie bereits in Abbildung 6 dargestellt, sind in der Übersicht der Vorgänge alle möglichen Zeitpunkte dargestellt, an denen die einzelnen Vorgänge stattfinden können. Zusätzlich wird die Projektdauer über der Tabelle und der kritische Pfad, durch den die Projektdauer erzielt wird, in die Tabelle geschrieben (siehe Abbildung 11).



Abbildung 11 Ergebnis der Kapazitätsplanung

Die Besonderheit der Aktion optimize ist, dass RoR die durch die GAMS-Optimierung (Quellcode 3 aus Anhang A.2) erstellte Textdatei zum Parameter  $X_{it}$  trotz der zwei Indizes auslesen kann. Dies erfolgt durch einen If-Befehl, wie der Ausschnitt (Quellcode 1) des Quellcodes 10 aus dem Anhang A.3 zeigt. Die Kapazitätsplanung ist mit dem Einlesen der Ergebnisse abgeschlossen.

Quellcode 1 Ausschnitt aus dem RoR-Controller für das RCPSP

```
if (File.exist?("RCPSP1_solution_zeit.txt") and File.exists?("
RCPSP1_solution_zw.txt"))

fi=File.open("RCPSP1_solution_x.txt", "r")
fi.each { |line|
    sa=line.split(";")
    if sa[0].to_i == 1
    sa0=sa[0]
    sa1=sa[1]
    sa2=sa[2].delete "t" + " \n"
    procedure=Procedure.find_by_name(sa1)
    procedure.optp = sa2
```

Über den Button Zurück zur Projektplanung gelangt ein Admin zurück zur Verwaltungsseite. Sofern die Kostenplanung gewünscht ist, kann diese übe die Verwaltungsseite gestartet werden. Bevor der optimale Kostenplan für das vorhandene Projekt berechnet werden kann, muss zunächst äquivalent zur Einstellung des Starttermins eine Deadline eingerichtet werden. Dies funktioniert erneut über ein Kalendermenü des "Bootstrap-Datepicker-Rails". Es sollte bei der Bestimmung der Deadline darauf geachtet werden, dass die Deadline in einem sinnvollen Verhältnis zum Starttermin steht. Eine zu kurze oder lange Zeitspanne zwischen den beiden Terminen kann zu unbrauchbaren Ergebnissen führen. Wurde eine geeignete Deadline ausgewählt und die Optimierung des Kostenplans gestartet, öffnet sich nach einer kurzen Rechenzeit die Übersicht der Ressourcen. Dieses erfolgt mit der Aktion optimize2 und solution2 des rcpsps\_controller.

Auf der Seite mit der Ressourcenübersicht sind die Projektkosten und die Zusatzkosten, die jede Ressource durch Einhaltung der Deadline verursachen, ausgelesen (siehe Abbildung 7). Bei der Kostenplanung ist jedoch nicht nur relevant, wie hoch die Kosten zur Durchführung des Projektes sind, sondern auch die Zeitpunkte, zu denen die Vorgänge stattfinden. Um dies zu untersuchen, bietet sich dem Admin die Möglichkeit, ein weiteres Mal die Seite mit der Übersicht der Vorgänge aufzurufen. Auf dieser Seite ist angestoßen durch die Berechnung des optimalen Kostenplans der entsprechende kritische Pfad mit allen frühesten und spätesten Zeitpunkten in die Tabelle eingelesen. Die alten Ergebnisse der Kapazitätsplanung

sind gelöscht und dadurch wird kein veralteter Wert angezeigt (siehe Abbildung 12). Somit sind alle Informationen des Kostenplans einsehbar, die Kostenplanung ist ebenfalls abgeschlossen.

PROJEKTPL	ANUNG					Sta	rtseite	Hilfe R	essource	Mitarbeiter	Menü 🔻
		Übers	sicht d	er	V	or	gä	nge			
Name	Vorgangs- dauer	Anzahl Vorgänger	Anzahl Ressourcen	FA*	SA*	FE*	SE*	Kritische Pfad	r		
Beispielvorgang1	2	0	1	0	0	2	2	2	Anzeig	en Ändern	Löschen
Beispielvorgang2	5	1	1	2	2	7	7	7	Anzeig	en Ändern	Löschen
Beispielvorgang3	4	1	1	7	7	11	11	11	Anzeig	en Ändern	Löschen
Beispielvorgang4	3	1	1	11	11	14	14	14	Anzeig	en Ändern	Löschen
Beispielvorgang5	5	1	1	2	5	7	10	7	Anzeig	en Ändern	Löschen
Beispielvorgang6	4	2	1	7	10	11	14	12	Anzeig	en Ändern	Löschen
Beispielvorgang7	5	2	1	14	14	19	19	19	Anzeig	en Ändern	Löschen
Beispielvorgang8	5	1	1	2	12	7	17	16	Anzeig	en Ändern	Löschen
Beispielvorgang9	2	1	1	7	17	9	19	19	Anzeig	en Ändern	Löschen
Beispielvorgang10	3	1	1	2	11	5	14	8	Anzeig	en Ändern	Löschen
Beispielvorgang11	5	1	1	5	14	10	19	17	Anzeig	en Ändern	Löschen
Beispielvorgang12	2	4	1	19	19	21	21	21	Anzeig	en Ändern	Löschen

Abbildung 12 Ergebnis der Kapazitätsplanung

## 4 Kritische Würdigung des Anwendungssystems

In dieser Ausarbeitung werden zunächst die Problematiken der Projektplanung dargelegt und das RCPSP für die Kapazitäts- und Kostenplanung formuliert. Dabei steht neben der Lösung des Modells mit einem Programm die Integration und Verknüpfung mit einer Applikation im Vordergrund. Ziel ist es, eine Applikation zu erstellen, die im Hintergrund der Anwendung auf das Lösungstool, dass die Rechnungen durchführt, zurückgreift, die Ergebnisse abruft und darstellt.

Das RCPSP wird in dem Programm GAMS programmiert und mit dessen Hilfe gelöst. Da der Schwerpunkt dieser Arbeit auf RoR liegt, sind in GAMS nur die essentiellen Bestandteile zur Lösung des RCPSP implementiert. Dank einer Schnittstelle zwischen RoR und GAMS ist es möglich, einen Großteil der Daten in RoR zu programmieren, um sie dann in GAMS einzulesen. Der Schwerpunkt liegt dementsprechend auf der Implementierung in RoR und der Bildung der Schnittstelle.

Bei der Erstellung der Applikation ist die herausragende Rolle der Controller zu erwähnen. Ein Controller ist für jede Art von Programmierungselement erforderlich, unabhängig von der Tatsache, ob es sich zum Beispiel um die Entwicklung eines Datenmodells oder die Strukturierung der Homepage handelt. In den jeweiligen Controllern werden alle Aktionen definiert, auf die in anderen Verzeichnissen zurückgegriffen werden. Nur Aktionen, die im

Controller hinterlegt sind, können an anderer Stelle aufgerufen bzw. genutzt werden. Falls der Controller fehlerhaft programmiert ist, führt dies zwangsläufig an einem gewissen Punkt der Applikation zu Komplikationen. Am Beispiel des RCPSP Controller lässt sich die Signifikanz des Controllers in RoR demonstrieren.

- Quellcode RCPSP Controller(Teil: "def optimize") -

Im RCPSP Controller sind alle Aktionen bezüglich der Optimierung der Kapazitäts- und Kostenplannung hinterlegt. "def optimize" beinhaltet alle Aktionen, die RoR durchführen muss, damit die GAMS-Datei zur Lösung der Kapazitätsplanung den vollständigen Input erhält und der Output korrekt ausgelesen wird. Die Aktion Öptimiere Kapazitätsplanung" (siehe Abbildung 8), die die Betätigung des Buttons auslöst, funktioniert ausschließlich durch die korrekte Programmierung unter "def optimize" im Controller.

Die kritische Betrachtung impliziert neben den Besonderheiten des Programms auch die zukünftigen Verbesserungsmöglichkeiten dieses Anwendungssystems. Die hier implementierte Applikation behandelt ausschließlich das RCPSP bei exakt einem vorhandenen Projekt. Die Multiprojektplanung wurde außer Acht gelassen. Mit der Einführung der Multiprojektplanung eröffnen sich zwar neue Problemstellungen bei der Implementierung, es bietet jedoch auch neue Anwendungsmöglichkeiten der Applikation. Anstatt einzelne Aspekte eines Projektes zu optimieren, kann nun ein Portfolio an mehreren Projekten erstellt werden. Ein Mitarbeiter hat somit die Wahl, an welchen Projekten er partizipieren möchte und die Durchführung des Projektportfolios ist Ziel der Optimierung. Es findet dementsprechend eine Projektion der Kapazitäts- und Kostenplanung auf die Multiprojektplanung statt.

Die tatsächlich durchgeführte Programmierung bietet ebenfalls Verbesserungspotential. Auf der Übersichtsseite der Projektplanung, auf der die Optimierungsvorgänge gestartet werden können, finden sich auch die Felder zur Aktualisierung des Starttermins und der Deadline. Die Felder und Funktionen, die in diesem Zusammenhang ausgelöst werden, funktionieren zwar fehlerfrei, die zeitliche Abhängigkeit ist aber nicht in der Form gegeben, wie es die Bezeichnungen voraussetzen. Wird die zeitliche Reihenfolge vertauscht, führt das Modell die Optimierung trotzdem korrekt durch. Der Termin der Deadline muss laut der implementierten Regularien nicht nach dem Starttermin stattfinden. Die Bezeichnung der Begriffe lässt diese Vertauschung der Reihenfolge unlogisch erscheinen, da es sich für RoR aber nur um reine Strings handelt, existiert keine logische Verknüpfung der Begrifflichkeiten.

Ein weiteres Problem tritt bei der Aktualisierung der Pfade für die Programme *GAMS* und *Graphviz* (siehe Abbildung 8). Die Optimierung aller Planungsprobleme erfolgt erst nach der Angabe des korrekten Pfads von *GAMS*. Nach diesem Schema funktioniert die Erstellung des Graphen der Vorgangsrelationen nur, wenn erfolgreich auf *Graphviz* zurückgegriffen werden kann. Diese Kausalitäten führen zu der Frage, was bei der Angabe von falschen Pfaden für das jeweilige Programm passiert. Faktisch funktionieren sie nicht. Beim Start der Optimierung mit einem flaschen *GAMS*-Pfad sucht der Controller vergeblich nach textitGAMS. Statt den Vorgang abzubrechen, bleibt der Server an dieser Stelle hängen und es der Nutzer

kann keine weiteren Aktionen ausführen. Um dieses Äbstürzen" des Servers zu verhindern, falls der Programmpfad inkorrekt ist, könnte ein *Timeout*-Befehl die Aktion nach einer vorgegebenen Zeitspanne abbrechen. So wäre es möglich, den Pfad zu ändern und den Vorgang erneut zu starten. Eine andere Möglichkeit als *Timeout* ist "terminator". "terminator" ist keine interne RoR-Methode, daher muss der passende Gemfile installiert werden, damit RoR diese Methode verwenden kann.

Auch bei der Verwaltung der Vorgänge und Vorgangsrelationen offenbaren sich Lücken, die in späteren Ausarbeitungen überarbeitet werden sollten. Das Anlegen, Ändern und Löschen von Vorgängen funktioniert reibungslos. Zur Darstellung der Vorgangsrelationen wird das Modell Graphviz verwendet. Dieses Modell kommt aus einer Internetquelle. Der Großteil der Programmcodes dieses Modells ist übernommen, nur eine Anpassung des Problems an die Gegebenheiten findet statt. Diese Datenübernahme hat zur Folge, dass der Name jedes Vorgangs aus mindestens zwölf Zeichen bestehen muss. Bei einem kürzeren Vorgangsnamen zeichnet das Programm keinen Graphen. Die Gründe dafür sind unklar. In dieser Arbeit ist eine Mindestlänge für den Namen eines Vorgangs vorgegeben. Das ursprüngliche Problem von Graphviz besteht dadurch weiterhin, ist im Rahmen dieser Arbeit aber nicht lösbar. Auf andere Funtkionen der Vorgänge oder Vorgangsrelationen hat dieser Umstand keinen Einfluss. Die Vorgangsrelationen stellen einen wichtigen Part der Projektplanung dar. Schließlich ist die Bestimmung der Reihenfolge der abzuarbeitenden Vorgänge die Basis für viele andere Operationen auf dem Weg zur Projektopimierung. Beim Anlegen einer neuen Vorgangsrelation dürfen, wie bereits beschrieben, keine Zyklen entstehen. Es findet aber keine Prüfung statt, ob die neu angelegte Vorgangsrelation bereits existiert. Folglich wird nicht verhindert, dass z.B. die Relation von Vorgang 1 zu Vorgang 2 zwei Mal angelegt wird. Für alle weiteren Prozesse resultieren keine Schäden, da doppelt vorhandene Relationen nicht beachtet werden, trotzdem handelt es sich hier um eine Prüfungslücke im Modell der Vorgangsrelationen.

### 5 Fazit

Ziel dieser Ausarbeitung war es, mit der Programmierumgebung RoR eine Homepage zu erstellen, mit dessen Hilfe eine Fragestellung aus dem Bereich des Operations Management gelöst werden kann. Konkret wurde das RCPSP zur Optimierung der Projektplanung mit einer Schnittstelle zu GAMS implementiert. Es lässt sich festhalten, dass RoR

## 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 Datenbankschema

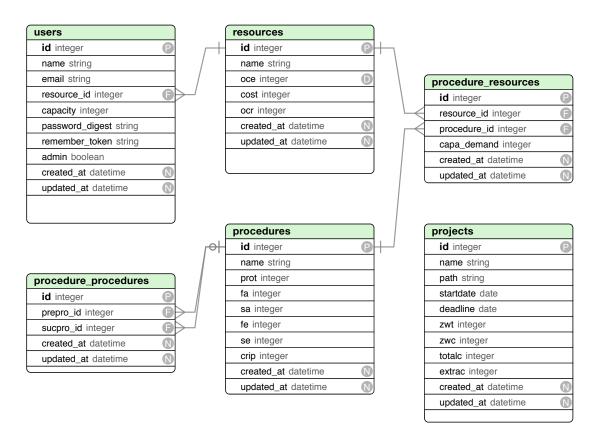


Abbildung 13 Datenbankschema der Web-Applikation Projektplanung

## A.2 GAMS-Implementierung des Beispiels

### Quellcode 2 GAMS-Code zur Kapazitätsplanung

```
* Ressourcenbeschraenkte Projektplanung in diskreter Zeit

* Zwei Modellvarianten:

* Variante 1: Minimierung der Projektdauer bei gegebenen Kapazitaeten

set
    i Vorgang
    t Periode
    r Ressource;

alias(t,tau);
alias(h,i);
```

```
VN(h,i) Vorgaenger-Nachfolger-Relation zwischen h und i;
parameter
    d(i)
            Dauer
    FE(i) Fruehester Endzeitpunkt
    SE(i) Spaetester Endzeitpunkt
    FA(i) Fruehester Anfangszeitpunkt
    SA(i) Spaetester Anfangszeitpunkt
    k(i,r) Kapazitaetsbedarf von Vorgang i auf Ressource r
    \mathtt{KP}(\mathtt{r}) Kapazitaet je Periode von Ressource r
    ihilf
    Deadline
    MinimaleDauer ;
binary variables
    x(i,t) gleich 1 wenn Vorgang i in Periode t beendet wird;
free variables
            Zielfunktionswert;
$include "RCPSP1_Input.inc";
* Zeitrechnung
* Achtung: Topologische Sortierung wird unterstellt
MinimaleDauer = 0;
FA(i)=0;
FE(i)=d(i);
loop(i,
     loop(h$VN(h,i),
         if(FE(h)>FA(i),
             FA(i)=FE(h);
             FE(i)=FA(i)+d(i);
             if ( FE(i)>MinimaleDauer,
                 MinimaleDauer = FE(i)
             );
         );
     );
);
SE(i)=max(MinimaleDauer, Deadline);
SA(i)=SE(i)-d(i);
```

```
for(ihilf=card(i) downto 1,
     loop(i$(ord(i)=round(ihilf)),
          loop(h$VN(i,h),
               if(SA(h) < SE(i),
                   SE(i)=SA(h);
                   SA(i)=SE(i)-d(i);
              );
          );
     );
);
display d, FA, FE, SA, SE, MinimaleDauer;
Equations
    ZielfunktionZeit,
    JederVorgangEinmal(i)
    Projektstruktur(h,i)
    Kapazitaetsrestriktion(r,t);
ZielfunktionZeit..
    z=e=sum(i\$(ord(i)=card(I)),
         sum(t\$(FE(i) \le ord(t) - 1 \text{ and } ord(t) - 1 \le SE(i)),
               (ord(t)-1)*x(i,t));
JederVorgangEinmal(i)..
    sum(t\$(FE(i) \le ord(t) - 1 \text{ and } ord(t) - 1 \le SE(i)), x(i,t)) = e = 1;
Projektstruktur(h,i)$VN(h,i)..
    sum(t\$(FE(h)\leq ord(t)-1 \text{ and } ord(t)-1 \leq SE(h)),
           (ord(t)-1)*x(h,t)) = 1=
    sum(t\$(FE(i) \le ord(t)-1 \text{ and } ord(t)-1 \le SE(i)),
           (ord(t)-1-d(i))*x(i,t));
{\tt Kapazitaetsrestriktion}\,({\tt r}\,,{\tt t}\,)\ldots
    sum(tau\$((ord(tau)-1) = max(ord(t)-1, FE(i))) and
                (ord(tau)-1 \le min(ord(t)-1+d(i)-1, SE(i))))
           k(i,r)*x(i,tau)))=1=KP(r);
model RCPSP1 /
    ZielfunktionZeit
    JederVorgangEinmal
```

```
Projektstruktur
    Kapazitaetsrestriktion /;
RCPSP1.optcr = 0.0;
RCPSP1.limrow=500;
solve RCPSP1 minimizing z using mip;
display z.1, x.1;
file outputfile1 / 'RCPSP1_solution_zeit.txt'/;
put outputfile1;
loop(i,
     put i.tl:0, '; 'FA(i), '; 'SA(i), '; 'FE(i), '; 'SE(i) /
);
putclose outputfile1;
file outputfile2 / 'RCPSP1_solution_x.txt'/;
put outputfile2;
loop(t,
loop(i,
     put x.l(i,t), ';' i.tl:0, ';' t.tl:0 /
);
);
putclose outputfile2;
file outputfile3 /'RCPSP1_solution_zw.txt'/;
put outputfile3;
put 'Zielfunktionswert: ',z.1 /
put '****************
putclose outputfile3;
```

### Quellcode 3 GAMS-Code zur Kostenplanung

```
* Ressourcenbeschraenkte Projektplanung in diskreter Zeit

* Zwei Modellvarianten:

* Variante 2: Minimierung der Kosten fuer Zusatzkapazitaet bei

* gegebener Deadline
```

```
set
    i Vorgang
    t Periode
    r Ressource;
alias(t,tau);
alias(h,i);
set
    {\tt VN}\,({\tt h}\,,{\tt i}) \ {\tt Vorgaenger-Nachfolger-Relation} \ {\tt zwischen} \ {\tt h} \ {\tt und} \ {\tt i}\,;
parameter
    d(i)
             Dauer
    FE(i) Fruehester Endzeitpunkt
    SE(i) Spaetester Endzeitpunkt
    FA(i) Fruehester Anfangszeitpunkt
    SA(i)
            Spaetester Anfangszeitpunkt
    k(i,r) Kapazitaetsbedarf von Vorgang i auf Ressource r
    KP(r) Kapazitaet je Periode von Ressource r
    oc(r)
            Kosten einer Einheit Zusatzkapazitaet
    ihilf
    Deadline
    MinimaleDauer
binary variables
    x(i,t) gleich 1 wenn Vorgang i in Periode t beendet wird;
free variables
             Zielfunktionswert;
positive variables
    O(r,t) Zusatzkapazitaet von Ressource r in Periode t;
$include "RCPSP2_Input.inc";
* Zeitrechnung
* Achtung: Topologische Sortierung wird unterstellt
MinimaleDauer = 0;
FA(i)=0;
FE(i)=d(i);
loop(i,
     loop(h$VN(h,i),
         if(FE(h)>FA(i),
             FA(i)=FE(h);
```

```
FE(i)=FA(i)+d(i);
              if ( FE(i)>MinimaleDauer,
                  MinimaleDauer = FE(i)
              );
         );
     );
);
SE(i)=max(MinimaleDauer, Deadline);
SA(i)=SE(i)-d(i);
for(ihilf=card(i) downto 1,
     loop(i$(ord(i)=round(ihilf)),
         loop(h$VN(i,h),
              if(SA(h) < SE(i),
                  SE(i)=SA(h);
                  SA(i)=SE(i)-d(i);
              );
         );
     );
);
display d, FA, FE, SA, SE, Deadline, MinimaleDauer;
Equations
    {\tt ZielfunktionKosten}\;,
    JederVorgangEinmal(i)
    Projektstruktur(h,i)
    Kapazitaetsrestriktion(r,t)
    KapazitaetsrestriktionFlex(r,t);
ZielfunktionKosten..
    z=e=sum((r,t),oc(r)*O(r,t));
JederVorgangEinmal(i)..
    sum(t\$(FE(i) \le ord(t)-1 \text{ and } ord(t)-1 \le SE(i)), x(i,t)) = = 1;
Projektstruktur(h,i)$VN(h,i)..
    sum(t\$(FE(h) \le ord(t)-1 \text{ and } ord(t)-1 \le SE(h)),
           (ord(t)-1)*x(h,t)) = 1=
```

```
sum(t\$(FE(i) \le ord(t)-1 \text{ and } ord(t)-1 \le SE(i)),
              (ord(t)-1-d(i))*x(i,t));
KapazitaetsrestriktionFlex(r,t)...
     sum(i,
     \label{eq:condition} \begin{split} & \sup \big( \, \mathsf{tau} \, \big( \, \big( \, \mathsf{ord} \, \big( \, \mathsf{tau} \, \big) - 1 \, > = \, \max \big( \, \mathsf{ord} \, \big( \, \mathsf{t} \, \big) \, - 1 \, , \, \, \, \mathsf{FE} \, ( \, \mathsf{i} \, \big) \, \big) \big) \quad \text{and} \end{split}
                    (ord(tau)-1 \le min(ord(t)-1+d(i)-1, SE(i))),
              k(i,r)*x(i,tau))=1=KP(r)+O(r,t);
model RCPSP2 /
     ZielfunktionKosten
      JederVorgangEinmal
     Projektstruktur
     KapazitaetsrestriktionFlex/;
RCPSP2.optcr = 0.0;
RCPSP2.limrow=500;
solve RCPSP2 minimizing z using mip;
parameter
     zkr(r) Berechnung Zusatzkosten;
zkr(r) = sum(t,0.1(r,t));
display x.1, 0.1, zkr;
file outputfile1 / 'RCPSP2_solution_kosten.txt'/;
put outputfile1;
loop(r,
     put r.tl:0, '; 'zkr(r) /
);
putclose outputfile1;
file outputfile2 / 'RCPSP2_solution_x.txt'/;
```

```
put outputfile2;
loop(t,
loop(i,
    put x.l(i,t), ';' i.tl:0, ';' t.tl:0 /
);
);
putclose outputfile2;
file outputfile3 / 'RCPSP2_solution_zeit.txt'/;
put outputfile3;
loop(i,
    put i.tl:0, '; 'FA(i), '; 'SA(i), '; 'FE(i), '; 'SE(i) /
putclose outputfile3;
file outputfile4 / 'RCPSP2_solution_zw.txt'/;
put outputfile4;
put 'Zielfunktionswert: ',z.1 /
put '****************
putclose outputfile4;
```

## A.3 Ruby on Rails Programmcodes

Quellcode 4 Gemfile der Web-Applikation Projektplanung

```
source 'http://rubygems.org'
gem 'rails', '3.2.8'
{\tt gem} \quad {\tt 'bootstrap-sass', \quad {\tt '2.0.4'}}
gem 'bcrypt-ruby', '3.0.1'
gem 'faker', '1.0.1'
gem 'will_paginate', '3.0.3'
gem 'bootstrap-will_paginate', '0.0.6'
gem 'jquery-rails', '2.0.2'
gem 'best_in_place'
gem "chartkick"
gem 'bootstrap-datepicker-rails', '~> 1.3.1.1'
gem 'ruby-graphviz', '~> 1.2.1'
gem 'graphviz', '~> 0.1.0'
gem 'rgl'
group :development, :test do
  gem 'sqlite3', '1.3.5'
```

```
gem 'rspec-rails', '2.11.0'
  gem 'guard-rspec', '1.2.1'
  gem 'annotate', '2.5.0'
  gem 'wdm', '~> 0.0.3'
  gem 'guard-spork', '1.2.0'
  gem 'spork', '0.9.2'
end
group :development do
  gem 'better_errors'
  gem 'binding_of_caller'
end
# Gems used only for assets and not required
# in production environments by default.
group :assets do
  gem 'sass-rails', '3.2.5'
  gem 'coffee-rails', '3.2.2'
  gem 'uglifier', '1.2.3'
  gem 'jquery-datatables-rails'
  gem 'jquery-ui-rails'
end
group :test do
  gem 'capybara', '1.1.2'
  gem 'factory_girl_rails', '4.1.0'
  gem 'cucumber-rails', '1.2.1', :require => false
  gem 'database_cleaner', '0.7.0'
  {\tt gem} \ '{\tt growl'}, \ '1.0.3'
  gem 'rb-fchange', '0.0.5'
  \texttt{gem} \ 'rb-notifu', \ '0.0.4'
  gem 'win32console', '1.3.0'
end
group : production do
  gem 'pg', '0.12.2'
end
if RUBY_VERSION = /1.9/ # assuming you're running Ruby ^{\sim}1.9
  Encoding.default_external = Encoding::UTF_8
  Encoding.default_internal = Encoding::UTF_8
end
```

Quellcode 5 Routes-Datei der Web-Applikation Projektplanung

```
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 'procedure_procedure/graph', :to => 'procedure_procedures#graph'
 match 'rcpsp/optimize', :to => 'rcpsps#optimize'
 match 'rcpsp/solution', to: 'rcpsps#solution'
 match 'rcpsp/optimize2', :to => 'rcpsps#optimize2'
 match 'rcpsp/solution2', to: 'rcpsps#solution2'
end
```

#### Quellcode 6 RoR-Controller für die Vorgangsrelationen

```
class ProcedureProceduresController < ApplicationController
    respond_to :html, :json
    before_filter :signed_in_user
  before_filter :admin_user
  def graph
    @procedure_procedures = ProcedureProcedure.all
    Qproject = Project.find(1)
    require 'rgl/adjacency'
    require 'rgl/dot'
    result = RGL::DirectedAdjacencyGraph.new
    @procedure_procedures.each { |x|
      result.add_edge x.prepro.name, x.sucpro.name }
    result.write_to_graphic_file('png')
    require 'graphviz'
    \texttt{GraphViz.parse}(\ \ "graph.dot"\ ,\ :\texttt{path} \implies \texttt{@project.gvp.to\_s}\ ) \ . \ \texttt{output}\ (:\texttt{png} \implies \longleftarrow
         "app/assets/images/graph.png", :path => @project.gvp.to_s)
  end
```

```
def index
  @procedure_procedures = ProcedureProcedure.all
def show
end
def new
  @procedure_procedure = ProcedureProcedure.new
  respond_to do |format|
    format.html # new.html.erb
    format.json { render json: @procedure_procedure }
  end
end
def edit
end
def create
require 'rgl/adjacency'
require 'rgl/topsort'
{\tt @procedure\_procedure} = {\tt ProcedureProcedure.new(params[:procedure\_procedure} {\hookleftarrow})
   1)
  respond_to do |format|
    if @procedure_procedure.save
      result = RGL::DirectedAdjacencyGraph.new
      ProcedureProcedure.all.each { |x|
         result.add_edge x.prepro_id, x.sucpro_id }
       if result.acyclic? == true
      format.html \{ redirect_to procedure_procedures_path, notice: \vdash \leftarrow
          Relation wurde erfolgreich angelegt! ' }
      format.json { render json: procedure_procedures_path, status: :←
          created, location: @procedure_procedure }
       else
         @procedure_procedure.destroy
         sleep(5)
         format.html { redirect_to :back, notice: 'Zyklen sind in der ←
            Projektplanung nicht erlaubt!' }
         \texttt{format.json} \ \{ \ \texttt{render json: Qprocedure\_procedure.errors} \,, \ \texttt{status: } : \hookleftarrow
            unprocessable_entity }
      end
    end
  end
end
```

```
def update
 end
  def destroy
    @procedure_procedure = ProcedureProcedure.find(params[:id])
    @procedure_procedure.destroy
   respond_to do |format|
      format.html { redirect_to procedure_procedures_url }
      format.json { head :no_content }
   end
 end
 private
  def signed_in_user
    unless signed_in?
      store_location
     redirect_to signin_path, notice: "Bitte anmelden."
    end
 end
  def admin_user
   redirect_to(root_path) unless current_user.admin?
 end
end
```

### Quellcode 7 RoR-Controller für die Vorgangs-Ressourcen-Kombinationen

```
{
m class} ProcedureResourcesController < ApplicationController
 respond_to :html, :json
 before_filter :signed_in_user
 before_filter :admin_user
 # GET /procedure_resources
 # GET /procedure_resources.json
  def index
    @procedure_resources = ProcedureResource.all
   respond_to do |format|
      format.html # index.html.erb
      format.json { render json: @procedure_resources }
   end
 end
 # GET /procedure_resources/1
 # GET /procedure_resources/1.json
  def show
```

```
@procedure_resource = ProcedureResource.find(params[:id])
  respond_to do |format|
    format.html # show.html.erb
    format.json { render json: @procedure_resource }
  end
end
# GET /procedure_resources/new
# GET /procedure_resources/new.json
  def new
    @procedure_resource = ProcedureResource.new
    respond_to do |format|
      format.html # new.html.erb
      format.json { render json: @procedure_resource }
    end
  end
# GET /procedure_resources/1/edit
def edit
  @procedure_resource = ProcedureResource.find(params[:id])
end
# POST /procedure_resources
# POST /procedure_resources.json
  def create
    @procedure_resource = ProcedureResource.new(params[:procedure_resource←
        1)
    respond_to do |format|
      if @procedure_resource.save
        format.html { redirect_to @procedure_resource, notice: 'Vorgangs-←
            Ressourcen-Relation wurde erfolgreich angelegt!' }
        \texttt{format.json} \ \{ \ \texttt{render json: @procedure\_resource} \ , \ \texttt{status: :created} \ , \ \hookleftarrow
            location: @procedure_resource }
      else
        format.html { render action: "new" }
        format.json { render json: @procedure_resource.errors, status: :←
            unprocessable_entity }
      end
    end
  end
# PUT /procedure_resources/1
# PUT /procedure_resources/1.json
def update
```

```
@procedure_resource = ProcedureResource.find(params[:id])
    respond_to do |format|
      if @procedure_resource.update_attributes(params[:procedure_resource])
        format.html { redirect_to @procedure_resource, notice: 'Vorgang-←
            Ressourcen-Relation wurde erfolgreich aktualisiert.' }
        format.json { head :no_content }
      else
        format.html { render action: "edit" }
        \texttt{format.json} \ \{ \ \texttt{render json: @procedure\_resource.errors} \ , \ \texttt{status: } : \hookleftarrow
            unprocessable_entity }
      end
    end
  end
 # DELETE / procedure_resources / 1
 # DELETE /procedure_resources/1.json
    def destroy
      @procedure_resource = ProcedureResource.find(params[:id])
      @procedure_resource.destroy
     respond_to do |format|
       format.html { redirect_to procedure_resources_url }
       format.json { head :no_content }
     end
   end
  private
  def signed_in_user
    unless signed_in?
      store_location
      redirect_to signin_path, notice: "Bitte anmelden."
    end
 end
  def admin_user
    redirect_to(root_path) unless current_user.admin?
 end
end
```

#### Quellcode 8 RoR-Controller für die Vorgänge

```
class ProceduresController < ApplicationController
  respond_to :html, :json
  before_filter :signed_in_user</pre>
```

```
before_filter :admin_user
# GET /procedures
# GET /procedures.json
  def index
     @procedures = Procedure.all
     Qproject = Project.find(1)
    respond_to do |format|
       format.html # index.html.erb
       format.json { render json: @procedures }
       format.json {render json: timeline }
    end
  end
# GET /procedures/1
# GET /procedures /1. json
  def show
     @procedure = Procedure.find(params[:id])
    respond_to do |format|
       format.html # show.html.erb
       format.json { render json: @procedure }
     end
  end
# GET /procedures/new
# GET /procedures/new.json
  def new
     @procedure = Procedure.new
    respond_to do |format|
       format.html # new.html.erb
       format.json { render json: @procedure }
     end
  end
# GET /procedures/1/edit
  def edit
     @procedure = Procedure.find(params[:id])
# POST / procedures
# POST /procedures.json
  def create
     @procedure = Procedure.new(params[:procedure])
     respond_to do |format|
       if Oprocedure.save
         \texttt{format.html} \ \{ \ \texttt{redirect\_to} \ \texttt{Oprocedure} \,, \ \texttt{notice:} \ \ \texttt{'Vorgang} \ \ \texttt{wurde} \, \leftarrow \,
              angelegt' }
         \texttt{format.json} \ \{ \ \texttt{render} \ \texttt{json:} \ \texttt{Oprocedure} \,, \ \texttt{status:} \ \texttt{:created} \,, \ \texttt{location:} \ \hookleftarrow
              @procedure }
```

```
else
        format.html { render action: "new" }
        format.json { render json: @procedure.errors, status: :←
            unprocessable_entity }
      end
    end
  end
# PUT /procedures/1
# PUT /procedures/1.json
  def update
    @procedure = Procedure.find(params[:id])
    respond_to do |format|
      if @procedure.update_attributes(params[:procedure])
        format.html { redirect_to @procedure, notice: 'Vorgang wurde ←
            aktualisiert ' }
        format.json { head :no_content }
      else
        format.html { render action: "edit" }
        format.json { render json: Qprocedure.errors, status: :\leftarrow
            unprocessable_entity }
      end
    end
  end
# DELETE /procedures/1
# DELETE /procedures/1.json
  def destroy
    @procedure = Procedure.find(params[:id])
    Oprocedure.destroy
    respond_to do |format|
      format.html { redirect_to procedures_url }
      format.json { head :no_content }
    end
  end
  private
  def signed_in_user
    unless signed_in?
      store_location
      redirect_to signin_path, notice: "Bitte anmelden."
    end
  end
  def admin_user
    redirect_to(root_path) unless current_user.admin?
  end
```

### Quellcode 9 RoR-Controller für das Projekt

```
{\tt class} \ {\tt ProjectsController} \ < \ {\tt ApplicationController}
  respond_to :html, :json
  #before_filter :admin_user
# GET / projects
# GET /projects.json
  def index
    @projects = Project.all
    respond_to do |format|
       format.html # index.html.erb
       format.json { render json: @projects }
    end
  end
# GET /projects/1
# GET /projects/1.json
  def show
    @project = Project.find(params[:id])
    respond_to do |format|
       {\tt format.html} \ \# \ {\rm show.html.erb}
       format.json { render json: @project }
    end
  end
# GET /projects/new
# GET /projects/new.json
  def new
    @project = Project.new
    respond_to do |format|
       format.html # new.html.erb
      format.json { render json: @project }
    end
  end
# GET /projects/1/edit
  def edit
    @project = Project.find(params[:id])
  end
# POST / projects
# POST /projects.json
  def create
    @project = Project.new(params[:project])
    respond_to do |format|
       if Oproject.save
         format.html { redirect_to @project, notice: 'Projekt wurde angelegt'←
         \texttt{format.json} \ \{ \ \texttt{render} \ \texttt{json: @project}, \ \texttt{status: :created}, \ \texttt{location:} \leftarrow \\
             @project }
```

```
else
        format.html { render action: "new" }
        format.json { render json: @project.errors, status: :←
            unprocessable_entity }
      end
    end
  end
# PUT /projects/1
# PUT /projects/1.json
  def update
    @project = Project.find(params[:id])
    respond_to do |format|
      if @project.update_attributes(params[:project])
        format.html { redirect_to rcpsp_path, notice: 'Daten wurden ←
            aktualisiert ' }
        format.json { head :no_content }
      else
        format.html { render action: "edit" }
        format.json { render json: Qproject.errors, status: :\leftarrow
            unprocessable_entity }
      end
    end
  end
# DELETE / projects/1
# DELETE /projects/1.json
  def destroy
    @project = Project.find(params[:id])
    @project.destroy
    respond_to do |format|
      format.html { redirect_to projects_url }
      format.json { head :no_content }
    end
  end
  private
  def signed_in_user
    unless signed_in?
      store_location
      redirect_to signin_path, notice: "Bitte anmelden."
    end
  end
  def admin_user
    redirect_to(root_path) unless current_user.admin?
  end
```

#### Quellcode 10 RoR-Controller für das RCPSP

```
#encoding: UTF-8
{\tt class} \ {\tt RcpspsController} \ < \ {\tt ApplicationController}
  respond_to :html, :json
  before_filter :signed_in_user
  before_filter :admin_user
   def optimize
         if File.exist?("RCPSP1_solution_x.txt")
      File.delete("RCPSP1_solution_x.txt")
    if File.exist?("RCPSP1_solution_zeit.txt")
      File.delete("RCPSP1_solution_zeit.txt")
    if File.exist?("RCPSP1_solution_zw.txt")
      File.delete("RCPSP1_solution_zw.txt")
    @resources = Resource.all
    @procedures = Procedure.all
    @procedure_procedures = ProcedureProcedure.all
    @procedure_resources = ProcedureResource.all
    @projects = Project.all
    Qproject = Project.find(1)
    @procedures.each { | proc |
      proc.fa=nil
      proc.sa=nil
      proc.fe=nil
      proc.se=nil
      proc.save
    @projects.each { | projekt |
      projekt.zwc=nil
      projekt.save
    @resources.each { | resources |
      resources.oce=0
      resources.save
    }
```

```
if File.exist?("RCSPSP1_Input.inc")
  File.delete("RCPSP1_Input.inc")
f=File.new("RCPSP1_Input.inc", "w")
printf(f, "set r / \n")
Oresources.each { |res| printf(f, res.name + "\n") }
printf(f, "/;" + " \setminus n \setminus n")
printf(f, "set i / \n")
@procedures.each { | proc| printf(f, proc.name + "\n") }
printf(f, "/;" + "\n\n")
printf(f, "set t / t0*t")
{\tt printf} \, (\, {\tt f} \, , \, \, \, {\tt Procedure.sum} \, (\, : {\tt prot} \, ) \, . \, {\tt to\_s} \, ) \,
printf(f, "/;"+"\n\n")
printf(f, "VN(h,i)=no; \n\n")
@procedure_procedures.each { | proc_proc |
  printf(f, "VN('" + proc_proc.prepro.name+"', '" + proc_proc.sucpro.name←
     +"')=yes;\n")
}
printf(f, "\n")
@procedures.each { | time |
  printf(f, "d('" + time.name + "') = "+ time.prot.to_s + "; \n")
printf(f, "\n")
printf(f, "k(i,r)=0; \n\n")
@procedure_resources.each { |k|
  printf(f, "k(!" + k.procedure.name + "', !" + k.resource.name + "') = "+ \leftarrow
       k.capa_demand.to_s + "; \n"
}
printf(f, "\n")
@resources.each { | grenze |
  \texttt{conditions} \implies \{\texttt{:resource\_id} \implies \texttt{grenze}\}).\texttt{to\_s} \, + \, "; \setminus n")
printf(f, "\n")
```

```
printf(f, "Deadline= "+ Procedure.sum(:prot).to_s + ";\n")
f.close
if File.exist?("RCPSP1_solution.txt")
 File.delete("RCPSP1_solution.txt")
end
system @project.path.to_s + " RCPSP1"
redirect_to url_for(:controller => :rcpsps, :action => :solution)
if (File.exist?("RCPSP1_solution_zeit.txt") and File.exists?("\leftarrow
   RCPSP1_solution_zw.txt"))
  fi=File.open("RCPSP1_solution_x.txt", "r")
  fi.each { |line|
  sa=line.split(";")
  if sa[0].to_i == 1
  sa0=sa[0]
  sa1=sa[1]
  sa2=sa[2].delete "t" + " \n"
  procedure=Procedure.find_by_name(sa1)
  procedure.optp = sa2
  procedure.save
  end
  fi.close
  fi=File.open("RCPSP1_solution_zeit.txt", "r")
  fi.each { |line|
    sa=line.split(";")
    sa0=sa[0]
    sa1=sa[1]
    sa2=sa[2]
    sa3=sa[3]
    sa4=sa[4].delete "\n"
    procedure=Procedure.find_by_name(sa0)
    procedure.fa = sa1
    procedure.sa = sa2
    procedure.fe = sa3
    procedure.se = sa4
    procedure.save
```

```
fi.close
    fi=File.open("RCPSP1_solution_zw.txt", "r")
    line=fi.readline
    fi.close
    sa=line.split(" ")
    sa0=sa[0]
    sa1=sa[1].delete " \n"
    project=Project.find_by_id(1)
    project.zwt = sa1
    project.save
  #else
  # flash.now[:not_available] = "Die Lösung wurde noch nicht berechnet!"
  end
end
def solution
  @procedures = Procedure.all
  Qproject = Project.find(1)
  until File.exist?("RCPSP1_solution_zeit.txt") and File.exists?("
     RCPSP1_solution_zw.txt")
    sleep(5)
  end
  render 'procedures/index'
end
def optimize2
  if File.exist?("RCPSP2_solution_x.txt")
    File.delete("RCPSP2_solution_x.txt")
  end
  if File.exist?("RCPSP2_solution_kosten.txt")
    File.delete("RCPSP2_solution_kosten.txt")
  end
  if File.exist?("RCPSP2_solution_zeit.txt")
    File.delete("RCPSP2_solution_zeit.txt")
  end
  if File.exist?("RCPSP2_solution_zw.txt")
    File.delete("RCPSP2_solution_zw.txt")
  end
  @resources = Resource.all
  @procedures = Procedure.all
  @procedure_procedures = ProcedureProcedure.all
  @procedure_resources = ProcedureResource.all
  @projects = Project.all
```

```
Qproject = Project.find(1)
@procedures.each { | proc |
 proc.fa=nil
  proc.sa=nil
  proc.fe=nil
  proc.se=nil
  proc.save
@projects.each { | projekt |
  projekt.zwt=nil
  projekt.save
@resources.each { |res|
  res.oce=nil
  res.save
}
if File.exist?("RCSPSP2_Input.inc")
  File.delete("RCPSP2_Input.inc")
end
f=File.new("RCPSP2_Input.inc", "w")
printf(f, "set r / \n")
Oresources.each { |res| printf(f, res.name + "\n") }
printf(f, "/;" + " \setminus n \setminus n")
printf(f, "set i / \n")
{\tt @procedures.each \{ |proc| printf(f, proc.name + "\n") \}}
printf(f, "/;" + " \setminus n \setminus n")
printf(f, "set t / t0*t")
printf(f, Procedure.sum(:prot).to_s)
printf(f, "/;"+" \n\n")
printf(f, "VN(h, i)=no; \n\n")
@procedure_procedures.each { | proc_proc |
  printf(f, "VN('" + proc_proc.prepro.name+"', '" + proc_proc.sucpro.name←
     +"')=yes;\n")
}
printf(f, "\n")
```

```
@procedures.each { | time |
  printf(f, "d('" + time.name + "') = "+ time.prot.to_s + "; \n")
printf(f, "\n")
\texttt{printf(f, "k(i,r)=0;} \backslash n \backslash n")
@procedure_resources.each { |k|
  printf(f, "k(!" + k.procedure.name + "', !" + k.resource.name + "') = "+\leftarrow"
      k.capa_demand.to_s + ";\n")
}
printf(f, "\n")
@resources.each { | grenze |
  printf(f, "KP(!" + grenze.name + "!) = " + User.sum(:capacity, : \leftarrow
     conditions => {:resource_id => grenze}).to_s + ";\n")
}
printf(f, "\n")
@resources.each { | zusatz |
  printf(f, "oc('" + zusatz.name + "') = " + zusatz.ocr.to_s + "; \n")
printf(f, "\n")
deadline = (@project.deadline - @project.startdate).to_i
printf(f, "Deadline=" + deadline.to_s + ";\n")
f.close
if File.exist?("RCPSP2_solution.txt")
  File.delete("RCPSP2_solution.txt")
end
system @project.path.to_s + " RCPSP2"
redirect_to url_for(:controller => :rcpsps, :action => :solution2)
if (File.exist?("RCPSP2_solution_kosten.txt") and File.exists?("←
   RCPSP2_solution_zw.txt"))
```

```
fi=File.open("RCPSP2_solution_x.txt", "r")
 fi.each { |line|
    sa=line.split(";")
    if sa[0].to_i == 1
      sa0=sa[0]
      sa1=sa[1]
      sa2=sa[2].delete "t" + " \n"
      procedure=Procedure.find_by_name(sa1)
      procedure.optp = sa2
      procedure.save
    end
  }
  fi.close
 fi=File.open("RCPSP2_solution_kosten.txt", "r")
 fi.each { |line|
    sa=line.split(";")
    sa0=sa[0]
    sa1=sa[1].delete " n"
    resource=Resource.find_by_name(sa0)
    resource.oce = sa1
    resource.save
  }
 fi.close
end
if (File.exist?("RCPSP2_solution_zeit.txt"))
 fi=File.open("RCPSP2_solution_zeit.txt", "r")
  fi.each { |line|
    sa=line.split(";")
    sa0=sa[0]
    sa1=sa[1]
    sa2=sa[2]
    sa3=sa[3]
    sa4=sa[4].delete " \n"
    procedure=Procedure.find_by_name(sa0)
    procedure.fa = sa1
    procedure.sa = sa2
    procedure.fe = sa3
    procedure.se = sa4
    procedure.save
  }
  fi.close
end
```

```
if File.exist?("RCPSP2_solution_zw.txt")
      fi=File.open("RCPSP2_solution_zw.txt", "r")
      line=fi.readline
      fi.close
      sa=line.split(" ")
      sa0=sa[0]
      sa1=sa[1].delete " \n"
     project=Project.find_by_id(1)
     project.zwc = sa1
     project.save
   end
 end
  def solution2
    @resources = Resource.all
    Qproject = Project.find(1)
    until File.exist?("RCPSP2_solution_zeit.txt") and File.exists?("
       RCPSP2_solution_zw.txt") and File.exist?("RCPSP2_solution_kosten.txt"←
      sleep(5)
   end
   render 'resources/index'
 end
 private
  def signed_in_user
    unless signed_in?
      store_location
      redirect_to signin_path, notice: "Bitte anmelden."
 end
  def admin_user
   redirect_to(root_path) unless current_user.admin?
 end
end
```

#### Quellcode 11 RoR-Controller für die Ressourcen

```
# GET /resources
# GET /resources.json
  def index
   @resources = Resource.all
    @procedures = Procedure.all
    Qproject = Project.find(1)
   respond_to do |format|
     format.html # index.html.erb
     format.json { render json: @resources }
    end
 end
# GET /resources/1
# GET /resources/1.json
  def show
    @resource = Resource.find(params[:id])
   respond_to do |format|
     format.html # show.html.erb
     format.json { render json: @resource }
    end
 end
# GET /resources/new
# GET /resources/new.json
  def new
    @resource = Resource.new
   respond_to do |format|
     format.html # new.html.erb
     format.json { render json: @resource }
    end
 end
# GET /resources/1/edit
  def edit
    @resource = Resource.find(params[:id])
 end
# POST /resources
# POST /resources.json
  def create
    @resource = Resource.new(params[:resource])
    respond_to do |format|
      if @resource.save
       angelegt' }
       format.json { render json: @resource, status: :created, location: \leftarrow
           @resource }
      else
       format.html { render action: "new" }
```

```
format.json { render json: @resource.errors, status: :\leftarrow
             unprocessable_entity }
       end
    end
  end
# PUT /resources/1
# PUT /resources/1.json
  def update
     @resource = Resource.find(params[:id])
    respond_to do |format|
       if @resource.update_attributes(params[:resource])
         \texttt{format.html} \ \{ \ \texttt{redirect\_to} \ \texttt{@resource} \ , \ \texttt{notice:} \ \ {}^{}_{} \\ \texttt{Ressource} \ \ \texttt{wurde} \ \hookleftarrow \\
             aktualisiert' }
         format.json { head :no_content }
       else
         format.html { render action: "edit" }
         format.json { render json: @resource.errors, status: :\leftarrow
             unprocessable_entity }
       end
     end
  end
# DELETE /resources/1
# DELETE /resources/1.json
  def destroy
    @resource = Resource.find(params[:id])
    @resource.destroy
    respond_to do |format|
       format.html { redirect_to resources_url }
       format.json { head :no_content }
  end
  private
  def signed_in_user
    unless signed_in?
       store_location
      {\tt redirect\_to}({\tt root\_path})
    end
  end
  def correct_user
     @user = User.find(params[:id])
    redirect_to(root_path) unless current_user?(@user)
  end
  def admin_user
```

```
redirect_to(root_path) unless current_user.admin?
end
end
```

#### Quellcode 12 RoR-Controller für die statischen Seiten

## Quellcode 13 RoR-Controller für die Users

```
{
m class} UsersController < ApplicationController
  \texttt{before\_filter} : \texttt{signed\_in\_user}, \ \texttt{only} \colon \ \big[ : \texttt{index} \,, \ : \texttt{show} \,, \ : \texttt{edit} \,, \ : \texttt{update} \,, \ : \hookleftarrow
      destroy
  before_filter :correct_user, only: [:edit, :update]
  before_filter :admin_user, only: [:destroy]
  def show
     @user = User.find(params[:id])
     @resource = Resource.find(@user.resource_id)
  end
  def new
     @user = User.new
  end
  def create
     @user = User.new(params[:user])
     if @user.save
       sign_in @user
```

```
flash[:success] = "Willkommen zur Projektplanung!"
    redirect_to @user
  else
    render 'new'
  end
end
def edit
end
def update
  if @user.update_attributes(params[:user])
    flash[:success] = "Profile updated"
    sign_in @user
    redirect_to @user
  else
    render 'edit'
  end
end
def index
  Qusers = User.paginate(page: params[:page])
end
def destroy
  User.find(params[:id]).destroy
  flash[:success] = "User destroyed."
  redirect_to users_url
end
private
def signed_in_user
  unless signed_in?
    store_location
    redirect_to signin_path, notice: "Bitte anmelden."
  end
end
def correct_user
  @user = User.find(params[:id])
  \verb|redirect_to(root_path)| unless | current_user?(@user)|
end
def admin_user
  redirect_to(root_path) unless current_user.admin?
```

```
end end
```

### Quellcode 14 RoR-Modell für die Vorgangsrelationen

```
class ProcedureProcedure < ActiveRecord::Base
  attr_accessible :prepro_id, :sucpro_id

belongs_to :prepro, class_name: "Procedure"
  belongs_to :sucpro, class_name: "Procedure"

end</pre>
```

### Quellcode 15 RoR-Modell für die Vorgangs-Ressourcen-Kombinationen

```
class ProcedureResource < ActiveRecord::Base
  attr_accessible :procedure_id, :resource_id, :capa_demand

belongs_to :procedure, class_name: "Procedure"
  belongs_to :resource, class_name: "Resource"

validates :capa_demand, :numericality => {:only_integer => true}

end
```

#### Quellcode 16 RoR-Modell für die Vorgänge

#### Quellcode 17 RoR-Modell für das Projekt

```
class Project < ActiveRecord::Base
  attr_accessible :created_at, :startdate, :deadline, :gvp, :updated_at, :←
    path, :zwt, :zwc, :totalc, :extrac</pre>
```

#### Quellcode 18 RoR-Modell für die Ressourcen

```
class Resource < ActiveRecord::Base
  attr_accessible :created_at, :name, :ocr, :cost, :oce

has_many :procedure_resources, :dependent => :destroy
has_many :procedures, through: :procedure_resources
has_many :users, :dependent => :destroy

validates :ocr, :numericality => {:only_integer => true}
validates :cost, :numericality => {:only_integer => true}
end
```

### Quellcode 19 RoR-Modell für die Users

```
# == Schema Information
# Table name: users
#
  id
                                        not null, primary key
#
                    :integer
#
  name
                    : string (255)
                    : string (255)
#
  email
#
  created_at
                    : datetime
                                        not null
  updated_at
                    : datetime
                                        not null
#
  password_digest : string (255)
  remember_token : string (255)
#
  admin
                    : boolean
                                        default (FALSE)
#
class User < ActiveRecord::Base</pre>
  attr\_accessible : email, :name, :password, :password\_confirmation, : \leftarrow
     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+ -.] + 0[a-z d-.] + ..[a-z] + z/i
  validates : \verb|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
```

### Quellcode 20 RoR-Seite für die Vorgangsrelationen - Formular

```
<\!\!\%\!\!= form_for(@procedure_procedure) do |f| \!\!\%\!\!>
    <% if @procedure_procedure.errors.any? %>
        <div id="error_explanation">
          <h2>
<h2>
pluralize(@procedure_procedure.errors.count, "error") %> ←
              prohibited this procedure_procedure from being
            saved:</h2>
          ul>
            <\!\% <code>Oprocedure_procedure.errors.full_messages.each</code> do |msg| \%\!>
                <% end %>
          </div>
    <% end %>
    <div id= "prepro_id" class="field">
      <%= f.label 'Von Vorgang' %><br />
      <%= f.collection_select :prepro_id, Procedure.all, :id, :name %>
    </div>
    <div class="field">
      <%= f.label 'Zu Vorgang' %><br />
      <\!\!\%\!\!= f.collection_select :sucpro_id, Procedure.all, :id, :name \%\!\!>
    </div>
    <div class="actions">
      <%= f.submit 'Relation zwischen den Vorgängen anlegen', :class ⇒ "btn←
         " %>
    </div>
<% end %>
```

# Quellcode 21 RoR-Seite für die Vorgangsrelationen - Übersicht

```
<% provide(:title, 'Übersicht der Vorgangsrelationen') %>
<h1>Übersicht der Vorgangsrelationen</h1>
<div class="center">
```

```
btn btn-info btn-large" %>
</div>
<br/>
<thead>
 >
   Von Vorgang
  <th>Zu Vorgang</th>
   </thead>
 <\!\% @procedure_procedures.each 
m do |procedure_procedure| \%\!>
    <= procedure_procedure.prepro.name %>
      % procedure_procedure.sucpro.name % /td>
      <td>%= link_to 'Löschen', procedure_procedure, method: :delete, \leftarrow
        data: {confirm: 'Sind Sie sicher?'} %>
    <\% end \%
 <br/>
<div class="btn-group btn-lg">
< = link_to  'Neue Relation anlegen', new_procedure_procedure_path, : class \implies \leftarrow
<%= link_to 'Zurück zur Projektplanung', rcpsp_path, :class => "btn"%>
</div>
```

#### Quellcode 22 RoR-Seite für die Vorgangsrelationen - Erstellung

#### Quellcode 23 RoR-Seite für die Vorgangsrelationen - Grafische Darstellung

```
<% provide(:title, 'Graph zur aktuellen Projektplanung') %>
<h1>Graph zur aktuellen Projektplanung</h1>
```

# Quellcode 24 RoR-Seite für die Vorgangs-Ressourcen-Kombinationen - Formular

```
<\% form_for(@procedure_resource) do |f| \%
   <% if @procedure_resource.errors.any? %>
       <div id="error_explanation">
         prohibited this procedure_resource from being
           saved:</h2>
        ul>
          <% @procedure_resource.errors.full_messages.each do |msg| %>
              <% end %>
         </div>
   <% end %>
   <div class="field">
     <%= f.label 'Vorgang' %><br />
     < \% = f.collection\_select: procedure\_id, Procedure.all,: id,: name \%
   </div>
   <div class="field">
     <%= f.label 'Ressource' %><br />
     <%= f.collection_select :resource_id, Resource.all, :id, :name %>
   </div>
   <div class="field">
     <%= f.label 'Kapazitätsbedarf' %><br />
     <\!\!\%\!\!= f.text_field :capa_demand \%\!\!>
   </div>
```

```
<div class="actions">
    <%= f.submit 'Vorgang zur Ressource zuordnen oder aktualisieren', :↔
        class ⇒ "btn" %>
        </div>
</ end %>
```

#### Quellcode 25 RoR-Seite für die Vorgangs-Ressourcen-Kombinationen - Bearbeitung

```
<% provide(:title, 'Zuordnung des Vorgangs zur Ressource ändern') %>
<h1>Zuordnung des Vorgangs zur Ressource ändern</h1>

<pr
```

# Quellcode 26 RoR-Seite für die Vorgangs-Ressourcen-Kombinationen - Übersicht

```
<% provide(:title, 'Übersicht der Zuordnung der Vorgänge zu den Ressourcen')↔
<h1>Übersicht der Zuordnung der Vorgänge zu den Ressourcen</h1>
<thead>
 >
   Vorgang
   Ressource
   <th>Kapazitätsbedarf</th>

 </thead>
 <\% <code>Oprocedure_resources.each</code> do | procedure_resource| \%

procedure = Procedure.find(procedure_resource.procedure_id)
             procedure.name %>
      %= resource = Resource.find(procedure_resource.resource_id)
             resource.name %>
      %= procedure_resource.capa_demand %>
      %= link_to 'Anzeigen', procedure_resource %>
      <td>%= link_to 'Ändern', edit_procedure_resource_path(\leftarrow
         procedure_resource) %>
      . ink_to 'Löschen', procedure_resource, method: :delete, data←
         : {confirm: 'Sind Sie sicher?'} %>
```

### Quellcode 27 RoR-Seite für die Vorgangs-Ressourcen-Kombinationen - Erstellung

```
<% provide(:title, 'Neue Vorgangs-Ressourcen-Kombination') %>
<h1>Neue Vorgangs-Ressourcen-Kombination</h1>
```

#### Quellcode 28 RoR-Seite für die Vorgangs-Ressourcen-Kombinationen - Anzeige

```
<% provide(:title, 'Vorgangs-Ressourcen-Kombination') %>
<h1>Vorgangs-Ressourcen-Kombination</h1>
>
 <b>Vorgang:</b>

procedure = Procedure.find(@procedure_resource.procedure_id)
       procedure.name %>
>
 <b>Ressource:</b>

%= resource = Resource.find(@procedure_resource.resource_id)
       resource.name %>
>
 <b>Kapazitätsbedarf:</b>
<= @procedure_resource.capa_demand %>
<div class="btn-group btn-lg">
```

```
<%= link_to 'Ändern', edit_procedure_resource_path(@procedure_resource), :\(\circ\)
    class => "btn" %>
<%= link_to 'Zurück', procedure_resources_path, :class => "btn" %>
</div>
```

# Quellcode 29 RoR-Seite für die Vorgänge - Formular

```
<\% form_for(@procedure) do |f| \%
 <% if @procedure.errors.any? %>
    <div id="error_explanation">
     <h2>
<h2>
%= pluralize(@procedure.errors.count, "error") %> prohibited this←
          procedure from being saved:</h2>
     ul>
     <% Oprocedure.errors.full_messages.each do |msg| %>
        \% = msg \% > 
     <% end %>
     </div>
 <% end %>
  <div class="field">
   <%= f.label :name %><br />
   <\% f.text_field :name \%
  </div>
  <div class="field">

f.label 'Vorgangsdauer' %><br />
   <\% f.text_field :prot \%
  </div>
  <div class="actions">
   <%= f.submit 'Vorgang anlegen oder aktualisieren', :class => "btn" %>
  </div>
<% end %>
```

### Quellcode 30 RoR-Seite für die Vorgänge - Bearbeitung

Quellcode 31 RoR-Seite für die Vorgänge - Übersicht

```
<% provide(:title, 'Übersicht der Vorgänge') %>
<h1>Übersicht der Vorgänge</h1>
<div class="center">

fink_to "Graph anzeigen", procedure_procedure_graph_path, class: "←
                 btn btn-info btn-large" %>
</div>
<br/>
<\% if Oproject.zwt!=nil %>
            Die minimale Projektdauer beträgt <strong ➤ Procedure.maximum("optp") ←
                         \%></	ext{strong}> Zeiteinheiten. Die Optimierung erfolgte mit einem \hookleftarrow
                        \texttt{Datenstand vom} < \texttt{strong} \\ \not\sim \texttt{@project.updated\_at.strftime}("\%d.\%m.\%Y") \leftarrow \texttt{``datenstand vom} < \texttt{``
                       \%>. </strong><br> <br>
<% end %>
<thead>
      <th>Name</th>
            Vorgangs- dauer
            Anzahl Vorgänger
            <th>Anzahl Ressourcen</th>
             FA * 
            SA*
             FE * 
            SE*
            <th><b><Vorgangs- abschluss</b></th>

       </thead>
      <\!\% @procedures.each 
m do |procedure| \%\!>
       \% = procedure.name \% > 
             \% = procedure.prot \% > 
            <td>%= procedure.reverse_procedure_procedures.count %>
            %= procedure.resources.count %>
             \% = procedure.fa \% > 
             \% = procedure.sa \% > 
             \% = procedure.fe \% > 
             \% = procedure.se \% > 
             b > \% = procedure.optp \% > </b >
```

```
%= link_to 'Anzeigen', procedure %>
                 %= link_to 'Ändern', edit_procedure_path(procedure) %>
                 <td<%= link_to 'Löschen', procedure, method: :delete, data: { confirm: \leftarrow
                                 'Sind sie sicher?' } %>
        <% end %>
 <br />
<div class="btn-group btn-lg">
<%= link_to 'Neuer Vorgang anlegen', new_procedure_path, :class => "btn" %>
<%= link_to 'Zurück zur Projektplanung', rcpsp_path, :class => "btn" %>
</div>
<br />
<thead>
*FA=frühester Anfangszeitpunkt, SA=spätester Anfangszeitpunkt, FE=frühester \hookleftarrow
              Endzeitpunkt und SE=spätester Endzeitpunkt in Tagen nach Start des \hookleftarrow
              Projekts.
</thead>
<br/>
<!--<= javascript_include_tag "//www.google.com/jsapi", "chartkick" %>->
<!--<= \  \, timeline \, \left[ \  \, ["Washington" \, , \, "1789-04-29" \, , \, "1797-03-03" \, ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, \right] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ] \, , \, \, ["Adams" \, , \, \, " \hookleftarrow ]
             1797-03-03", "1801-03-03"], ["Jefferson", "1801-03-03", "1809-03-03"]] \leftrightarrow
             %>--->
<!--<= @procedures.map{|p| "#{p.name} #{p.fa.day.from_now} #{p.fe.day.} \leftarrow
              from_now \}" \} \%>>>
 <!--"<= Time.now.strftime("%Y-%d-%m").to_s %>\"-->
```

### Quellcode 32 RoR-Seite für die Vorgänge - Erstellung

```
<% provide(:title, 'Neuer Vorgang') %>
<h1>Neuer Vorgang</h1>
<%= render 'form' %>
<%= link_to 'Zurück', procedures_path, :class => "btn" %>
```

#### Quellcode 33 RoR-Seite für die Vorgänge - Anzeige

```
<% provide(:title, 'Vorgang') %>
<h1>Vorgang</h1>
```

```
>
 <b>Name:</b>
 < \% @procedure.name \%
>
 <b>Vorgangsdauer:</b>
 < \% @procedure.prot \%
<div class="btn-group btn-lg">
<%= link_to 'Zurück', procedures_path, :class => "btn" %>
< = link_to 'Andern', edit_procedure_path(@procedure), : class => "btn" %>
:class => "btn" %>
< \% = link_to 'Neue Vorgangsrelation anlegen', new_procedure_procedure_path, :<math>\hookleftarrow
   class => "btn"%>
</div>
<br/><br/>
<div class="row">
 <div class="span6">
<thead>
 >
   <\!\!\texttt{th}\!\!>\!\!\texttt{Ressource}\!<\!\!/\texttt{th}\!\!>
   <th>Kapazitätsbedarf</th>

 </thead>
 <% <code>Qprocedure.procedure_resources.each do | procedure_resource| %> <%</code>
       % Resource.find(procedure_resource.resource_id).name % / td>
       <= procedure_resource.capa_demand %>
       <td>%= link_to 'Ändern', edit_procedure_resource_path(\leftarrow
          procedure_resource) %>
       <td>%= link_to 'Löschen', procedure_resource, method: :delete, data\leftarrow
          : {confirm: 'Sind Sie sicher?'} %>
     <% end %>
 </div>
```

```
<div class="span6">
<thead>
 >
  <th>Vorgänger</th>
  </thead>
 <% @procedure.reverse_procedure_procedures.each 
m do |procedure_procedure| \leftarrow
    >
      %= procedure_procedure.prepro.name %>
      <td>%= link_to 'Löschen', procedure_procedure, method: :delete, \leftarrow
        data: {confirm: 'Sind Sie sicher?'} %>
    <% end %>
 </div>
</div>
```

# Quellcode 34 RoR-Seite für die Ressourcen - Formular

```
<\!\!\%\!\!= form_for(@resource) do |f| \!\%\!\!>
 <% if @resource.errors.any? %>
    <div id="error_explanation">
      <h2>%= pluralize(@resource.errors.count, "error") %> prohibited this \leftarrow
          resource from being saved:</h2>
      <\!\% Oresource.errors.full_messages.each do |msg| \%\!>
         \% = msg \% > 
      <% end %>
      </div>
 <% end %>
  <div class="field">
   <%= f.label :name %><br />
   <%= f.text_field :name %>
  </div>
    <div class="field">
     <%= f.label 'Kosten je Kapazität' %><br />
     <%= f.text_field :cost %>
    </div>
    <div class="field">
```

#### Quellcode 35 RoR-Seite für die Ressourcen - Tabelle als unangemeldeter User

```
<thead>
 >
  <th>Name</th>
  </thead>
 <\% elements = [] \%
 <% @resources.each do |resource| %>
    \% = resource.name \% > 
        = link_to 'Bewerben', signup_path %>
    <% end %>
 <%= link_to 'Hier geht es zur Anmeldung', signup_path, :class ⇒ "btn" ←
    %>
```

#### Quellcode 36 RoR-Seite für die Ressourcen - Tabelle als angemeldeter User

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

 <thead>     \text{Ath} Name    \text{Amme} \text{Amme}  
    \text{Cesamt} - kapazität
```

```
<th>Kosten/ ME</th>
   <th>Grund- kosten</th>
   <th>Zusatz- kosten/ ME</th>
   <th>Zusatz- einheiten</th>
   Zusatzkosten Gesamt
   Gesamtkosten
   </thead>
 <% @resources.each do |resource| %>
       \% = resource.name \% > /td >
      <%= User.sum(:capacity, :conditions ⇒> {:resource_id ⇒> resource←
         }) %>
      %= resource.cost * resource.procedures.sum(&:prot) %>
      * resource.oce %>
      <th>%= resource.ocr * resource.oce + (resource.cost * resource.\leftarrow
         procedures.sum(&:prot) ) %>
      %= link_to 'Anzeigen', resource %>
      <% if signed_in? && current_user.admin? %>
         %= link_to 'Ändern', edit_resource_path(resource) %>
          <td>%= link_to 'Löschen', resource, method: :delete, data: { \leftarrow
            confirm: 'Sind sie sicher?' } %>
      <% end %>
     <% end %>
 <br/>
<br/>
<% begin %>
<\% if <code>@project.zwc!=nil</code> \%

javascript_include_tag "//www.google.com/jsapi" %>
   <%= javascript_include_tag "chartkick" %>
   <% require 'chartkick' %>
   <br/>b>Benötigte Zusatzeinheiten:</b>
   <%= bar_chart Resource.group(:name).sum(:oce) %>
<% end %>
```

```
<
```

## Quellcode 37 RoR-Seite für die Ressourcen - Bearbeitung

```
<% provide(:title, 'Bearbeitung der Ressource') %>
<h1>Bearbeitung der Ressource</h1>
<%= render 'form' %>
<div class="btn-group btn-lg">
<%= link_to 'Anzeigen', @resource, :class => "btn" %>
<%= link_to 'Zurück', resources_path, :class => "btn" %>
</div>
```

# Quellcode 38 RoR-Seite für die Ressourcen - Übersicht

```
<% provide(:title, 'Übersicht der Ressourcen') %>
<h1>Übersicht der Ressourcen</h1>
<% if signed_in? %>
<%= render 'resources/signed_in' %>
<% else %>
<%= render 'resources/free' %>
<% end %>
```

## Quellcode 39 RoR-Seite für die Ressourcen - Erstellung

```
<% provide(:title, 'Neue Ressource') %>
<h1>Neue Ressource</h1>
<%= render 'form' %>
<%= link_to 'Zurück', resources_path, :class => "btn" %>
```

#### Quellcode 40 RoR-Seite für die Ressourcen - Anzeige

```
<% provide(:title, 'Ressource') %>
```

```
<h1>Ressource</h1>
>
 <b>Name:</b>
 <%= @resource.name %>
>
 <b>Kosten je Kapazität:</b>
 <%= @resource.cost %>
<b>Zusatzkosten je Kapazität:</b>
 <%= @resource.ocr %>
>
 <br/>b>Aktuelle Gesamkapazität:</b>
 <%= User.sum(:capacity, :conditions => {:resource_id => @resource}) %>
<% if current_user.admin? %>
   <div class="btn-group btn-lg">
<%= link_to 'Ändern', edit_resource_path(@resource), :class => "btn" %>
   <% end %>
< = link_to 'Zurück', resources_path, : class => "btn" %>
</div>
```

#### Quellcode 41 RoR-Seite für die Optimierungsseite zur Projektplanung

```
>
  Auf dieser Seite werden die Grunddaten zur Durchführung der Projektplanung←
      angepasst und die Optimierung durchgeführt: <br > <br >
<% if signed_in? %>
<div class="row">
<div class="span5">
    <h2>Programme</h2>
   <\% form_for(Oproject) do |f| \%
        <div class="field">
          <%= f.label :path, "GAMS-Verzeichnis" %>
          < \% f.text_field :path \%
          <a href="http://www.gams.com/download/">(GAMS-Download)</a>
        </div>
        <br/>
        <div class="field">
          <%= f.label :gvp, "GraphViz-Verzeichnis" %>
          <%= f.text_field :gvp %>
          <a href="http://www.graphviz.org/Download..php">(GraphViz-Download↔
              )</a>
        </div>
        <div class="actions">
          <%= f.submit 'Verzeichnisse aktualisieren', :class => "btn" %>
        </div>
     </div>
     <div class="span6">
      <h2>Start der Kapazitätsplanung</h2>
      <div class="field">
           <%= f.label :startdate, "Starttermin" %>
           <\% f.text_field :startdate, :class \Rightarrow "datepicker", :value \Rightarrow \leftarrow
               Oproject.startdate.strftime("%d.%m.%Y")
         </div>
         <div class="actions">
           <%= f.submit 'Starttermin aktualisieren', :class => "btn" %>
         </div>
        <= link_to "Optimiere Kapazitätsplanung", ←
               rcpsp\_optimize\_path, :class \Rightarrow "btn", data: {disable_with: "<i \leftarrow
                class='fa fa-spinner fa-spin'></i> Berechnung gestartet..."} \leftrightarrow
```

```
\%>
         <% end %>
        <br/>
    <h2>Start der Kostenplanung</h2>
   <\% form_for(@project) do |f| \%
       <div class="field">
         <%= f.label :deadline, "Deadline-Termin" %>
          <%= f.text_field :deadline, :class \Rightarrow "datepicker", :value \Rightarrow \leftarrow
             @project.deadline.strftime("%d.%m.%Y")
       </div>
       <div class="actions">
           <%= f.submit 'Deadline aktualisieren', :class => "btn" %>
       </div>
        >
            %= link_to "Optimiere Kostenplanung", rcpsp_optimize2_path,
                : class \Rightarrow "btn", data: {disable_with: "<i class = 'fa fa - \leftarrow
               spinner fa-spin'></i> Berechnung gestartet..."} %>
          <% end %>
<% end %>
</div>
</div>
<script>
    $('.datepicker').datepicker({
        src: 'js/bootstrap-datepicker.de.js',
        language: 'de',
        format: 'dd.mm.yyyy'
    });
</script>
```

### Quellcode 42 RoR-Seite für die Startseite

```
<div class="center hero-unit">
  <h1>Projektplanung</h1>
  <br/>
  <br/>
  <br/>
  <% if signed_in? %>
```

### Quellcode 43 Kopfzeile der Web-Apllikation

```
<header class="navbar navbar-fixed-top">
 <div class="navbar-inner">
   <div class="container">
    <%= link_to "Projektplanung", root_path, id: "logo" %>
    <nav>
      <= link_to "Startseite", root_path, id: "main_links" %>
        %= link_to "Hilfe", help_path, id: "main_links" %>
        <= link_to "Ressource", resources_path, id: "main_links" %></</pre>
           li>
        <% if signed_in? %>
           %= link_to "Mitarbeiter", users_path, id: "main_links" \Leftarrow
           id="fat-menu" class="dropdown">
             <a href="#" class="dropdown-toggle" data-toggle="dropdown"> ←
                Menü <b class=" caret"></b>
             </a>
             class="dropdown-menu">
               <= link_to "Einstellungen", edit_user_path(←)</li>
                  current_user) %>
               <% if current_user.admin? %>
                  <= link_to "Projektplanung", rcpsp_path %>
               <% end %>
               class="divider">
               <1i>>
                <%= link_to "Logout", signout_path, method: "delete" %>
               <% else %>
           Login", signin_path %>
        <% end %>
```

```
</nav>
</div>
</div>
</header>
```

# Quellcode 44 RoR-Seite bzgl. der Lösung von Usern über die Übersichtsseite

```
<
```

### Quellcode 45 RoR-Seite für die User - Bearbeitung

```
<\% provide(:title, 'Profil') %>
<h1>Aktualisiere Dein Profil</h1>
<div class="row">
  <div class="span6 offset3">
   < \% = form_for(@user) do |f| \%
        <%= render 'shared/error_messages' %>
        <% f.label :name %
        <%= f.text_field :name %>
        <%= f.label :email %>
        <%= f.text_field :email %>
        <%= f.label :capacity, "Arbeitszeit pro Tag" %>
        <%= f.text_field :capacity %>
        <%= f.label :resource_id, "Welche Ressource?" %>
        <%= f.collection_select :resource_id, Resource.all, :id, :name %>
        <%= f.label :password, "Passwort" %>
        <%= f.password_field :password %>
        <%= f.label :password_confirmation, "Bestätigung" %>
        <\!\%\!\!= f.password_field :password_confirmation \%\!\!>
```

### Quellcode 46 RoR-Seite für die User - User-/Mitarbeiterübersicht

### Quellcode 47 RoR-Seite für die User - Erstellung

```
<% provide(:title, 'Melde Dich an') %>
<h1>Melde Dich an</h1>
<div class="row">
  <div class="span6 offset3">
   <\% form_for(@user) do |f| %>

render 'shared/error_messages' %>
     <\% f.label :name \%
     <%= f.text_field :name %>
     < \% = f.label : email %
     <%= f.text_field :email %>
     <%= f.label :capacity, "Arbeitszeit pro Tag" %>
     <%= f.text_field :capacity %>
     <%= f.label :resource_id, "Welche Ressource?" %>
     <%= f.collection_select :resource_id, Resource.all, :id, :name %>
     <%= f.label :password, "Passwort" %>
     <%= f.password_field :password %>
     <%= f.label :password_confirmation, "Bestätigung" %>
     <%= f.password_field :password_confirmation %>
```

### Quellcode 48 RoR-Seite für die User - Anzeige

```
<\% provide(:title, @user.name) \%
<div class="row">
 <aside class="span4">
   <section>
    <h1>
      <%= gravatar_for @user %>
      <%= @user.name %>
    </h1>
   </section>
 </aside>
</div>
>
 <br/>b>Arbeitszeit pro Tag:</b>
 <%= @user.capacity %>
<br/>b>Rolle im Projekt:</b>
 <%= @resource.name%>
<% if current_user?(@user) %>
   <%= link_to 'Einstellung', edit_user_path(current_user), :class ⇒ "btn"←</pre>
<% end %>
<thead>
 >
   <th>Name</th>
   <th>Vorgangsdauer</th>
    FA * 
   SA*
    FE * 
   SE*
   <th><b><fth>
```

```
 </thead>
 <\!\% @resource.procedures.each 
m do |procedure| \%\!>
       \% = procedure.name \% > 
       \% = procedure.fa \% > 
       \% = procedure.sa \% > 
       \% = procedure.fe \% > 
       \% = procedure.se \% > 
       <td><b></b>
      <% if current_user.admin? %>
          %= link_to 'Anzeigen', procedure %>
          %= link_to 'Ändern', edit_procedure_path(procedure) %>
          <td>%= link_to 'Löschen', procedure, method: :delete, data: { \leftarrow
             confirm: 'Sind sie sicher?' } %>
      <% end %>
     <% end %>
 <%= link_to 'Zurück', users_path, :class => "btn" %>
<br>>
<thead>
*FA=frühester Anfangszeitpunkt, SA=spätester Anfangszeitpunkt, FE=frühester \hookleftarrow
   Endzeitpunkt und SE=spätester Endzeitpunkt in Tagen nach Start des \hookleftarrow
   Projekts.
</thead>
```

### Quellcode 49 RoR-Datenbankschema

```
\# from scratch. The latter is a flawed and unsustainable approach (the more \hookleftarrow
    migrations
# you'll amass, the slower it'll run and the greater likelihood for issues).
\# It's strongly recommended to check this file into your version control \leftrightarrow
    system.
ActiveRecord::Schema.define(:version \Rightarrow 20150310092848) do
  create_table "procedure_procedures", :force => true do |t|
    t.integer "prepro_id"
    \verb|t.integer| "sucpro_id"
    t.datetime "created_at", :null => false
    t.datetime "updated_at", :null => false
  end
  create_table "procedure_resources", :force => true do |t|
    t.integer "resource_id"
    t.integer "procedure_id"
    t.integer "capa_demand"
    t.datetime "created_at",
                                 : null \Rightarrow false
    t.datetime "updated_at", :null \Rightarrow false
  end
  create_table "procedures", :force => true do |t|
                "name"
    t.string
                                                 :null => false
    t.datetime "created_at",
    t.datetime "updated_at",
                                                 : null \Longrightarrow false
    t.integer "prot"
    t.integer "fa",
                               : default \implies 0
                               : default \Rightarrow 0
    t.integer "sa",
    t.integer "fe",
                               : default \Rightarrow 0
                                : default \Rightarrow 0
    t.integer "se",
    {\tt t.integer} \quad "optp" \, ,
                                : \mathtt{default} \implies 0
  end
  \texttt{create\_table} \ "projects" \, , \ : \texttt{force} \implies \texttt{true} \ do \ | \texttt{t} \, |
    t.string "gvp"
    t.string "path"
               "startdate"
    t.date
    t.date
                "deadline"
    t.datetime "created_at", :null => false
    t.datetime "updated_at", :null => false
    t.integer "zwt"
    t.integer "zwc"
    t.integer "totalc"
     t.integer "extrac"
```

```
end
  create_table "resources", :force => true do |t|
    t.string "name"
    t.datetime "created_at",
                                                 : null \Rightarrow false
    t.integer "oce",
                         : default \Rightarrow 0
    t.integer "cost"
    t.integer "ocr"
    t.datetime "updated_at",
                                           : null \Longrightarrow false
  end
  create_table "users", :force => true do |t|
    t.string "name"
    t.string "email"
    t.integer "resource_id"
    t.integer "capacity"
    t.datetime "created_at",
                                                            : null \Rightarrow false
    t.datetime "updated_at",
                                                            : null \Rightarrow false
    t.string "password_digest"
    t.string "remember_token"
                                      : default \implies false
    t.boolean "admin",
  end
  \verb"add_index" users", ["email"], : \verb"name" \Rightarrow "index_users_on_email", : \verb"unique" \Rightarrow \leftarrow
      true
  add_index "users", ["remember_token"], :name => "
      index_users_on_remember_token"
end
```

#### Quellcode 50 Beispieldaten für die Datenbank

```
password: "foobar",
                  password_confirmation: "foobar",
                  capacity: 2,
                  resource_id: 1)
    (3..5) .each do |n|
       name = Faker::Name.name
      name = "Nutzer-\#\{n\}"
      email = "example-#{n}@railstutorial.org"
      password = "password"
      capacity = 2
      {\tt resource\_id} = 1
      User.create!(name: name,
                    email: email,
                    password: password,
                    password_confirmation: password,
                    capacity: capacity,
                    resource_id: resource_id)
    end
    (6..10) .each do |\mathbf{n}|
#
       name = Faker::Name.name
      name = "Nutzer-\#\{n\}"
      email = "example-#{n}@railstutorial.org"
      password = "password"
      capacity = 2
      resource_id = 2
      User.create!(name: name,
                    email: email,
                    password: password,
                    password_confirmation: password,
                    capacity: capacity,
                    resource_id: resource_id)
    end
    (1..12) .each do |\mathbf{n}|
      name = "Beispielvorgang#{n}"
      prot = rand(2..5)
      Procedure.create!(name: name,
                    prot: prot)
    end
    (1...2) .each do |n|
      name = "Ressource#{n}"
      ocr = rand(5..10)
```

```
cost = rand(1..3)
  Resource.create!(name: name,
                ocr: ocr,
                cost: cost)
end
ProPro1 = ProcedureProcedure.create!(prepro_id: 1, sucpro_id: 2)
ProPro2 = ProcedureProcedure.create!(prepro_id: 1, sucpro_id: 5)
ProPro3 = ProcedureProcedure.create!(prepro_id: 1, sucpro_id: 8)
ProPro4 = ProcedureProcedure.create!(prepro_id: 1, sucpro_id: 10)
ProPro5 = ProcedureProcedure.create!(prepro_id: 2, sucpro_id: 3)
ProPro6 = ProcedureProcedure.create!(prepro_id: 2, sucpro_id: 6)
ProPro7 = ProcedureProcedure.create!(prepro_id: 3, sucpro_id: 4)
ProPro8 = ProcedureProcedure.create!(prepro_id: 4, sucpro_id: 7)
ProPro9 = ProcedureProcedure.create!(prepro_id: 4, sucpro_id: 12)
ProPro10 = ProcedureProcedure.create!(prepro_id: 5, sucpro_id: 6)
ProPro11 = ProcedureProcedure.create!(prepro_id: 6, sucpro_id: 7)
ProPro12 = ProcedureProcedure.create!(prepro_id: 7, sucpro_id: 12)
ProPro13 = ProcedureProcedure.create!(prepro_id: 8, sucpro_id: 9)
ProPro14 = ProcedureProcedure.create!(prepro_id: 9, sucpro_id: 12)
ProPro15 = ProcedureProcedure.create!(prepro_id: 10, sucpro_id: 11)
ProPro16 = ProcedureProcedure.create!(prepro_id: 11, sucpro_id: 12)
ProRes1 = ProcedureResource.create!(procedure_id: 1, resource_id: 1, ←
   capa_demand: 8)
ProRes2 = ProcedureResource.create!(procedure_id: 2, resource_id: 1, \leftarrow
   capa_demand: 8)
ProRes3 = ProcedureResource.create!(procedure_id: 3, resource_id: 2, ←
   capa_demand: 10)
ProRes4 = ProcedureResource.create!(procedure_id: 4, resource_id: 1, \leftarrow
   capa_demand: 8)
ProRes5 = ProcedureResource.create!(procedure_id: 5, resource_id: 2, \leftarrow
   capa_demand: 8)
ProRes6 = ProcedureResource.create!(procedure_id: 6, resource_id: 1, \leftarrow
   capa_demand: 9)
ProRes7 = ProcedureResource.create!(procedure_id: 7, resource_id: 1, ←
   capa_demand: 8)
{\tt ProRes8} = {\tt ProcedureResource.create!} (\, {\tt procedure\_id} \colon \ 8 \,, \ {\tt resource\_id} \colon \ 2 \,, \ \hookleftarrow
   capa_demand: 10)
{\tt ProRes9} = {\tt ProcedureResource.create!} (\, {\tt procedure\_id} \colon \ 9 \,, \ {\tt resource\_id} \colon \ 1 \,, \ \hookleftarrow
   capa_demand: 8)
ProRes10 = ProcedureResource.create!(procedure_id: 10, resource_id: 1, \leftarrow
   capa_demand: 9)
ProRes11 = ProcedureResource.create!(procedure_id: 11, resource_id: 1, ←
   capa_demand: 8)
```

```
ProRes12 = ProcedureResource.create!(procedure_id: 12, resource_id: 1, ← capa_demand: 8)

Proj1 = Project.create!(gvp: "C:\\Program Files (x86)\\Graphviz2.38\\bin← ", path: "C:\\GAMS\\win64\\24.3\\gams", startdate: Date.today, ← deadline: Date.today + 19.days)

#(1..10).each do |n|
#procedure_id = n
#resource_id = rand(1..5)
#ProcedureResource.create!(resource_id: resource_id, procedure_id: ← procedure_id)
#end

end
end
```