

TRAVEL.CH BPMN

Simon Lang

25. Juni 2016

Version 1.0.0

STUDIENGANG	Informatik 5 Ba 2012
SEMESTERARBEIT	Travel.ch BPMN
DOZENT	Daniel Liebhart
SCHULE	ZHAW - School of Engineering

Kurzfassung

In der Softwareentwicklung wird traditionell nach dem Wasserfall-Prinzip¹ vorgegangen. Viele Teams wird dieses alte Vorgehensmodel durch neue, agile Methoden abgelöst. In kurzen abständen (drei bis vier Wochen) wird dem Klient eine lauffähige Version vorgestellt. Kunden werden dabei stärker in den Entwicklungsprozess eingebunden, wodurch neue Probleme entstehen. In jedem Zyklus ist eine Spezifikations-, eine Umsetzungs- und eine Test-Phase enthalten. Bei umfangreicher Software kann der letzte Abschnitt sehr umfangreich werden und wird deshalb gerne vernachlässigt.

Um diesem Problem entgegenzutreten, kann das Testen der Software Automatisiert werden. Diesem Thema nimmt sich diese Arbeit an. Sie versucht am Beispiel der travel.ch aufzuzeigen, wie ein Automated Testing geplant und umgesetzt werden kann, und wann sich dieses lohnt und wann nicht.

Schlagwörter: Integration Tests, Selenium, Testing

¹ Wasserfallmodell.

Inhaltsverzeichnis

1	Einleitung	1
1.1	Aufgabenstellung	1
1.1.1	Ausgangslage	1
1.1.2	Ziele der Arbeit	1
1.1.3	Aufgabenstellung	1
1.1.4	Erwartete Resultate	1
1.2	Dieses Dokument	1
1.2.1	Recherche	1
1.2.2	Anforderungen und Analyse	1
1.2.3	Konzept	1
1.2.4	Umsetzung	1
1.2.5	Summary	1
2	Recherche	2
2.1	BPMN	2
2.1.1	Notation	2
2.2	Rahmenbedingungen	3
2.2.1	Travelwindow AG Prozesse	3
2.2.2	Travelwindow AG API	4
2.2.3	Betriebssystem	4
2.3	travel.ch Webseite	4
2.4	Programme	7
2.4.1	Anforderungen	7
2.4.2	Programmanalyse	7
3	Anforderungen und Analyse	9
3.1	Anaylse der travelwindow AG API	9
3.1.1	Testumgebung	19
3.1.2	Checkout	19
3.2	Bonita	19
3.2.1	Interface	19
3.2.2	Business Data Model	20
3.2.3	Connectors	22

3.2.4 Forms, Contracts und Operations	25
4 Konzept	28
4.1 Mockups	28
4.2 Prozess	30
5 Umsetzung	32
5.1 Prozess	32
5.2 Business Data Models und Pool Variables	32
5.3 Forms	34
5.4 Connectors	34
5.4.1 Abhängigkeiten	34
5.4.2 Definitions	34
5.4.3 Implementations	34
5.5 Tasks	34
5.5.1 Search	34
5.5.2 Select Hotel	35
5.5.3 Select Room Config	35
5.5.4 Select Flight	36
5.6 Probleme	36
5.6.1 Rückgabewert der Connectors	37
5.6.2 Business Data Models (BDMs) befüllen	38
6 Zusammenfassung & Fazit	41
A Testübersicht	42
Quellenverzeichnis	43

Abbildungsverzeichnis

2.1 Flow Objects	2
2.2 Events	2
2.3 Pool and Swimlanes	3
2.4 Example	3
2.5 Hotel plus Flight BPMN Model auf travel.ch	4
2.6 Hotel und Flug suche auf travel.ch	4
2.7 Hotel und Flug Hotelresultate auf travel.ch	5
2.8 Hotelconfiguration von Hotel und Flug auf travel.ch	6
2.9 Hotel und Flug Flugresultate auf travel.ch	6
3.1 Bonita Interface	20
3.2 Bonita BPM	21
3.3 Bonita Pool Variables	22
3.4 Bonita Connectors	23
3.5 Bonita Connector Implementation	24
3.6 Bonita Form Contract	25
3.7 Bonita Form Operations	26
3.8 Bonita Form erstellung aus Contract und Operations	27
4.1 Suchformular Mockup	28
4.2 Mockup für die Hotel Suchresultate	29
4.3 Mockup für die Hotelzimmer Konfiguration	30
4.4 Mockup für die Flugresultate	30
4.5 Hotel plus Flight BPMN Model für dieses Projekt	31
5.1 Prozess im Bonita	32

Tabellenverzeichnis

2.1	Programme - Feature Gegenüberstellung	8
5.1	Business Data Models	33
5.2	Pool Variables	33
5.3	Connector Definitions	34
5.4	Search Task Contract	35
5.5	Search Task Operations	35
5.6	Select Hotel Task Contract	35
5.7	Select Hotel Task Operations	35
5.8	Select Room Config Task Contract	36
5.9	Select Room Config Task Operations	36
5.10	Select Flight Task Contract	36
5.11	Select Flight Task Operations	36

Akronyme

Bezeichnung	Beschreibung
BDM	Business Data Model
URL	Uniform Resource Locator

Glossar

API

API steht für Application Programming Interface. Dabei handelt es sich um eine Programmierschnittstelle, die den Austausch von Daten oder das starten von Prozessen erlaubt.

HAL

HAL steht für Hypertext Application Language. HAL ist eine Art wie APIs aufgebaut werden können. Das Ziel ist eine vereinfachung der API sowie diese frei erkundbar zu machen. Dies ermöglicht HAL über Links zwischen den einzelnen Ressourcen. Für weitere Informationen: http://stateless.co/hal_specification.html

JSON

JSON steht für JavaScript Object Notation. Dies ist eine definierte Schreibweise um Daten auszutauschen.

KAPITEL 1

Einleitung

1.1 Aufgabenstellung

In diesem Abschnitt wird die Aufgabenstellung gemäss Eingabe im Einschreibe und Bewertungssystem (EBS) der ZHAW aufgeführt.

1.1.1 Ausgangslage

Bislang besteht keine Abbildung der Travel.ch Prozesse.

1.1.2 Ziele der Arbeit

Der Prozess soll abgebildet und ausgeführt werden können.

1.1.3 Aufgabenstellung

Eine Abbildung des Prozesses soll erstellt werden. Initiale Eingaben müssen definiert werden und Antworten der Services müssen mit XSLT transformiert werden.

1.1.4 Erwartete Resultate

Ein Visualisierter Prozess welcher mit einer "Prozess Engine" ausgeführt werden kann.

1.2 Dieses Dokument

Dieses Dokument dokumentiert die Vorbereitung sowie die Umsetzung. Dieser Abschnitt soll eine Übersicht bieten über die Struktur der Arbeit.

1.2.1 Recherche

1.2.2 Anforderungen und Analyse

1.2.3 Konzept

1.2.4 Umsetzung

1.2.5 Summary

KAPITEL 2

Recherche

In diesem Kapitel wird erklärt was BPMN ist, es werden die Rahmenbedingungen umrissen und die verschiedenen Programme analysiert welche für die Umsetzung in Frage kommen.

2.1 BPMN

BPMN steht für Business Process Model and Notation. Dabei handelt es sich um eine genormte Visualisierung für Business Processes. Entwickelt wurde der Standart im Jahre 2001 durch die Firma IBM.

2.1.1 Notation

In diesem Abschnitt soll die grundlegenden Objekte von BPMN vorgestellt werden.

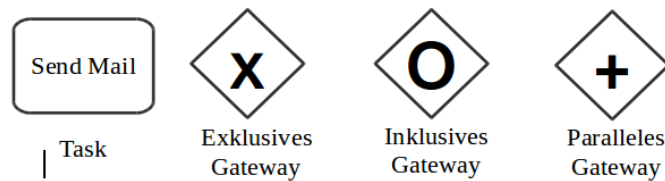


Abbildung 2.1: Flow Objects

Der Task ist ein Schritt eines Prozesses, welcher ausgeführt wird. Dieser kann entweder durch einen Menschen oder automatisiert durchgeführt werden. Die Rauten sind Gateways. Das Exklusive (XOR) Gateway, steht für einen Entscheidung. Es kann nur ein Pfad weitergeführt werden. Deshalb heisst es exklusiv. Das Inklusive (OR) Gateway ist wie das Exklusive Gateway eine Entscheidung, es können jedoch mehrere Wege parallel ausgeführt werden. Das Parallele (AND) Gateway führt alle Pfade parallel aus. Es kann auch dazu verwendet werden mehrere Pfade zu synchronisieren und als einzelner Weg weiterzuführen.



Abbildung 2.2: Events

Ein Event beschreibt, wenn etwas während eines Prozesses passiert. Oben aufgeführt sind die drei Grundevents. Der Start Event kennzeichnet der Beginn und der End Event das Ende eines Prozesses. Ein Intermediate Event kann irgendwo zwischen dem Start und dem Ende eines Prozesses stehen. Es gibt diverse Ausprägungen von Events, welche mit Icons in den Kreisen gekennzeichnet werden. Zum Beispiel ein Mail Event, Timer Event, Error Event, Cancel Event, Link Event, Signal Event und Terminate Event, um einige zu nennen.

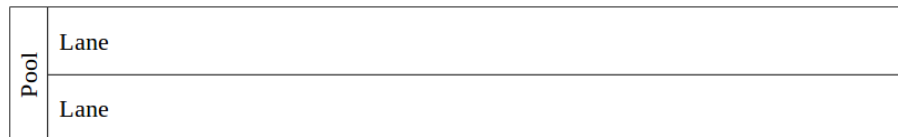


Abbildung 2.3: Pool and Swimlanes

Ein Pool kennzeichnet einen Participant, einen Benutzer oder Benutzerrolle in einem Prozess. Swimlanes ziehen sich über den gesamten Pool und werden dazu benutzt diesen weiter zu unterteilen.

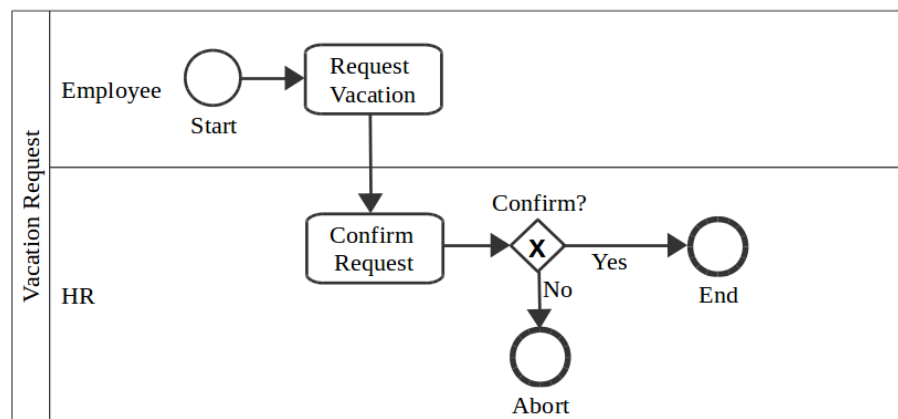


Abbildung 2.4: Example

Dies ist ein einfaches Beispiel eines Vacation Request Prozesses. Der Arbeitnehmer startet den Prozess und stellt seine Ferienanfrage. Die Anfrage wird von der Personalabteilung weiterverwendet und entscheidet, ob die Anfrage genehmigt wird oder nicht.

2.2 Rahmenbedingungen

Zu den Rahmenbedingungen gehört die Infrastruktur der travel.ch Webseite, das Entwicklungssystem des Programmierers sowie die etablierten Prozesse der travelwindow AG, welche die Webseite travel.ch betreibt.

2.2.1 Travelwindow AG Prozesse

Die Travelwindow AG ist der Betreiber der Seite travel.ch, auf welcher verschiedene Produkte gekauft werden können. Flüge, Hotel, Badeferien, Städtereisen sowie Hotel und Flug Kombinationen. Da der Buchungsprozess für Hotel und Flug Kombinationen der längste ist, wurde entschieden dass dieser in dieser Arbeit modelliert werden soll.

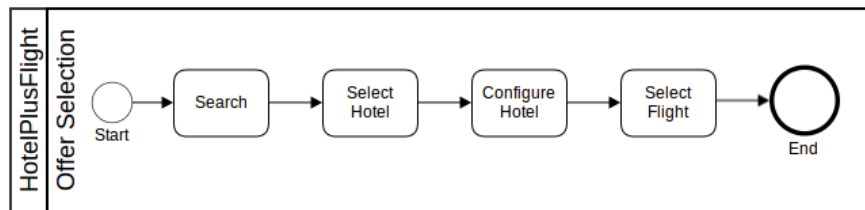


Abbildung 2.5: Hotel plus Flight BPMN Model auf travel.ch

Der Prozess der Hotel und Flug Suche auf der travel.ch Webseite ist linear. Nach einer Produktsuche muss der Kunde ein Hotel wählen, welches er danach noch weiter Konfigurieren kann. Dabei kann er andere Zimmer- und Verpflegungstypen wählen. Zum Schluss erhält er eine Auswahl von Flügen bevor der Prozess mit einem End Event terminiert.

2.2.2 Travelwindow AG API

Travel.ch besteht aus einer Webseite und einer [API](#). Für die Ausführung der Prozesse via BPMN muss die API angesprochen werden. Diese liefert daten in [JSON](#) aus. Programme für das ausführen des travel.ch BPMN Prozesses muss demnach den Austausch von Daten über eine [API](#) mittels [JSON](#) ermöglichen.

2.2.3 Betriebssystem

Die travel.ch Webseite wird über eine [API](#) betrieben, welche mittels eines BPMN Programmes abgefragt werden soll. Diese [API](#) ist nur vom Firmennetzwerk aus erreichbar. In der Firma sind nur Rechner mit dem Windows Betriebssystem im Einsatz.

Das Entwicklungssystem des Programmierers ist eine Linux Mint Computer. Dabei handelt es sich um ein Unix basiertes Betriebssystem.

Es ist demnach zwingend Notwendig, dass das BPMN Programm auf Windows, sowie auf Unix basierten Betriebssystemen läuft.

2.3 travel.ch Webseite

Da die Webseite der Travelwindow AG in BPMN dargestellt werden soll, wird nachfolgend das Interface der Seite aufgezeigt.

The screenshot shows the search interface of the travel.ch website. It features a red header with the text "[Not set: Checkout/MandantSpecific/HotelPlusFlight/EngineTitle] suchen und buchen". Below the header, there are four input fields: "Reiseziel" (with a placeholder "z.B. Stadt, Hotelname"), "Reisedaten" (with a placeholder "16.06.16 - 16.03.17, [Not set: DurationOfS...]", "Personen/Zimmer" (with a placeholder "2 Pers. in 1 Zim." and a dropdown arrow), and "Abflug ab" (with a placeholder "Alle Abflughäfen" and a dropdown arrow). A yellow "Suche" button is located to the right of the input fields.

Abbildung 2.6: Hotel und Flug suche auf travel.ch

Das Reiseziel definiert das Ziel des Fluges und wo das Hotel gesucht wird. Als Reisedaten gibt es eine exakte und eine flexible Suche. Bei einer flexiblen Suche gibt man einen Reisezeitraum (z.B. 6 Monate) an und definiert, wie lange die Reise sein soll (z.B. 7 Tage). Dann kann man das günstigste Objekt in diesem Zeitraum aussuchen. Bei der exakten Suche gibt man das Hinreise- und Rückreisedatum genau an. Im Feld Personen/Zimmer kann

man Räume definieren und wie viele Passagiere in diesen übernachten. Es wird zwischen Erwachsenen und Kinder unterschieden, da bei den Kindern zusätzlich das Geburtsdatum angegeben werden muss. Der Abflug ab gibt an, ab wo man fliegen möchte. Dies ist eine fixe Liste mit folgenden Flughäfen:

- Basel-Mülhausen
- Bern-Belp
- Genève-Cointrin
- Zürich
- Friedrichshafen
- Mailand-Malpensa

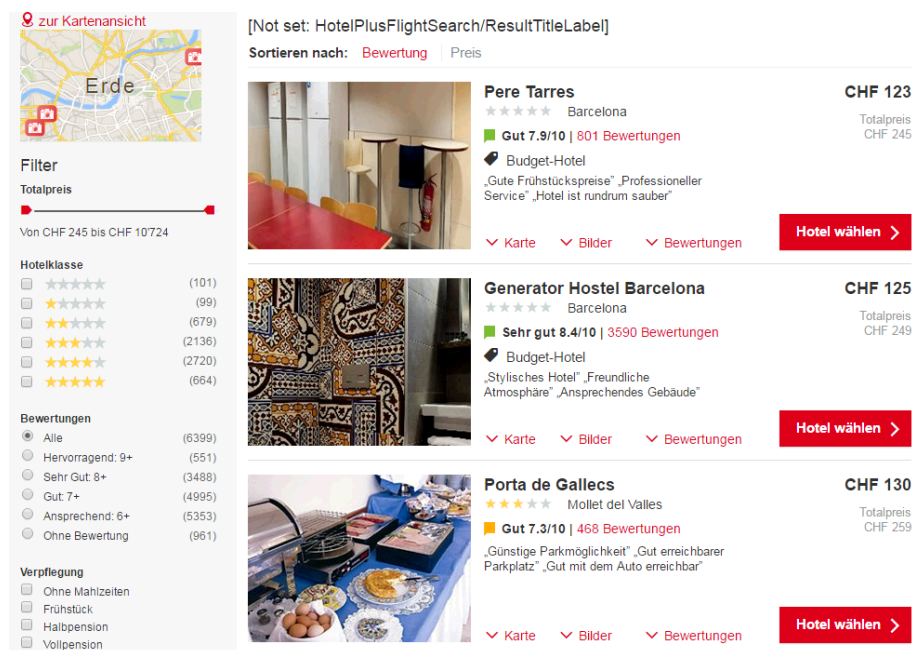


Abbildung 2.7: Hotel und Flug Hotelresultate auf travel.ch

Bei den Hotelresultaten gibt es links eine Filtrierung der Ergebnisse, welche Rechts dargestellt werden. Diese bestehen aus:

- Hauptbild
- Namen des Hotels
- Günstigster Preis
- Hotelklasse (Sterne)
- Bewertungen
- Karte, auf welcher die Geolocation des Hotels angezeigt wird

- Weiteren Bildern

Wählen Sie Zimmer und Verpflegung

Sie können hier den Zimmertyp sowie die Art der Verpflegung wählen. Der angepasste Gesamtpreis Ihrer Reise wird oben rechts angezeigt.

	Ohne Verpflegung	Frühstück
Doppelzimmer (1 oder 2 Betten), Standardzimmer	<input checked="" type="radio"/> CHF 384	<input type="radio"/> CHF 414
Familienzimmer, Standardzimmer	<input type="radio"/> CHF 412	<input type="radio"/> CHF 446
Doppelzimmer (1 oder 2 Betten), 3-Mann-Zimmer	<input type="radio"/> CHF 440	<input type="radio"/> CHF 479
Doppelzimmer, Superior-Zimmer	—	<input type="radio"/> CHF 451
Totalpreis (Hotel)	CHF 384	

Abbildung 2.8: Hotelconfiguration von Hotel und Flug auf travel.ch

Auf der Konfigurationsseite des Hotels werden weitere Informationen wie Checkin-Daten, Hoteldetails, Detaillierten Bewertungen, Hotelfacts, etc. angezeigt. Das wichtigste ist jedoch die oben dargestellte Matrix. Sie erlaubt dem Kunden das Hotelzimmer zu konfigurieren. Die Spalten der Matrix sind die Zimmer-, und die Kolonnen die Verpflegungstypen.

Wählen Sie Ihre Flugverbindung

Wir haben 16 Flüge von Zürich nach Lissabon und zurück gefunden.
Die angezeigten Preise beinhalten Hotel und Flug (inkl. Steuern und Gebühren).

vueling.com Vueling Mo. 27.02.2017 ZRH > LIS <input checked="" type="radio"/> 11:00 – 13:00 Direktflug	vueling.com Vueling Mi. 01.03.2017 LIS > ZRH <input checked="" type="radio"/> 13:40 – 17:20 Direktflug	pro Person Economy CHF 192 Totalpreis CHF 384	Flug wählen >
SWISS Swiss Mo. 27.02.2017 GVA > LIS <input type="radio"/> 07:00 – 08:30 Direktflug	SWISS Swiss Mi. 01.03.2017 LIS > GVA <input type="radio"/> 09:10 – 12:35 Direktflug	pro Person Economy CHF 239 Totalpreis CHF 478	
TAP AIR PORTUGAL TAP Air Portugal Mo. 27.02.2017 GVA > LIS <input type="radio"/> 06:35 – 08:10 Direktflug	TAP AIR PORTUGAL TAP Air Portugal Mi. 01.03.2017 LIS > GVA <input type="radio"/> 14:05 – 17:35 Direktflug <input type="radio"/> 19:25 – 22:55 Direktflug	pro Person Economy CHF 253 Totalpreis CHF 505	

Abbildung 2.9: Hotel und Flug Flugresultate auf travel.ch

Zum Schluss wird dem Kunden noch eine Auswahl von Flügen angeboten.

Danach gelangt der User in den Checkout Prozess, wo er seine persönlichen Daten eingeben muss und die Zahlung tätigen kann. Dies wurde jedoch an ein externes System ausgelagert und kann deshalb nicht mehr im BPMN dargestellt werden, da es keine zugängliche API dafür gibt.

2.4 Programme

In diesem Abschnitt werden verschiedene Programme vorgestellt und analysiert, ob sie für die Umsetzung dieses Projekts in Frage kommen. Dazu wird zuvor definiert, was die Anforderungen an das Programm sind.

2.4.1 Anforderungen

Eine zentrale Eigenschaft des Programmes ist, dass es eine Process Engine enthält. Dieses ermöglicht es, ein Prozess nicht nur in dem Tool zu zeichnen, sondern diesen auch auszuführen. Gemäss Aufgabenstellung muss das Tool auch die Datentransformation mittels XSLT unterstützen.

In den Abschnitten Abschnitt [2.2.1 Travelwindow AG Prozesse](#) und Abschnitt [2.2.2 Travelwindow AG API](#) wurde zudem beschrieben, dass das Programm auf den Betriebssystemen Windows und Unix lauffähig sein muss, sowie die Unterstützung von [JSON](#) basierten [APIs](#) beinhalten muss.

Zusätzlich sollte die Software gratis sein.

2.4.2 Programmanalyse

Die folgende Tabelle vergleicht verschiedene BPMN Tools auf die zuvor definierten Anforderungen (siehe Abschnitt [2.4.1 Anforderungen](#)).

Tabelle 2.1: Programme - Feature Gegenüberstellung

Programm	Process Engine	Windows & Linux	API mittels JSON	XSLT	Gratis
Activiti ^a	x	x			x
Yaoqiang ^b		x			
jBPM ^c	x	x			x
Imixs ^d	x	x			x
IBM Business Process Manager ^e	x		x		
Camunda ^f	x	x			x
Drools ^g	x	x			x
Bonita ^h	x	x		x	x
BizTalk ⁱ	x		x	x	

^a *Activiti*. URL: <http://activiti.org/> (besucht am 12.06.2016).

^b *Yaoqiang BPMN Editor - an Open Source BPMN 2.0 Modeler*. URL: <http://bpmn.sourceforge.net/> (besucht am 12.06.2016).

^c *jBPM - Open Source Business Process Management - Process engine*. URL: <http://www.jbpm.org/> (besucht am 12.06.2016).

^d *Imixs Workflow Project - Open Source Java Workflow Engine for BPMN 2.0*. URL: <http://www.imixs.org/> (besucht am 12.06.2016).

^e *IBM - Software - IBM Business Process Manager*. URL: <http://www-03.ibm.com/software/products/en/business-process-manager-family> (besucht am 12.06.2016).

^f *BPMN Workflow Engine*. URL: <https://camunda.org/> (besucht am 12.06.2016).

^g *Drools - Drools - Business Rules Management System (Java™, Open Source)*. URL: <http://www.drools.org/> (besucht am 12.06.2016).

^h *Bonitasoft*. URL: <http://www.bonitasoft.com/> (besucht am 12.06.2016).

ⁱ *Microsoft BizTalk Integration | Microsoft*. URL: <https://www.microsoft.com/en-us/server-cloud/products/biztalk/> (besucht am 12.06.2016).

Von den Funktionen ist Microsoft's BizTalk die geeignetste Variante. Jedoch läuft dieses Programm nicht auf Linux und es ist nicht gratis. Die zweitbeste Wahl ist Bonita von Bonitasoft, welches keine API Anfragen mittels JSON unterstützt.

KAPITEL 3

Anforderungen und Analyse

In diesem Kapitel werden die technischen Aspekte der API der travelwindow AG beschrieben, sowie die eingesetzte BPMN Software beschrieben.

3.1 Analyse der travelwindow AG API

Die [API](#) der travelwindow AG ist mit [HAL](#) aufgebaut. Sprich es gibt einen Einstiegspunkt, welcher unter „/“ erreichbar ist. Danach navigiert man sich über Links durch die Schnittstelle durch. Dies macht die API frei erkundbar, jedoch die Umsetzung mittels BPMN komplexer.

Nachfolgend werden die benötigten Anfragen an die API aufgeführt, welche für die Suche eines Hotel und Flug Angebotes benötigt werden (siehe Abschnitt [2.2.1 Travelwindow AG Prozesse](#)). Die aufgeführten Antworten sind für eine bessere Übersichtlichkeit verkürzt.

```
1 {
2   "href": "https://dev-api.travel.ch/",
3   [...]
4   "hotelPlusFlightSearch": { "href": "https://dev-api.travel.ch/
  HotelPlusFlight/Search" },
5   [...]
6 }
```

Dies ist die Root Ressource. Das Feld hotelPlusFlightSearch ist der Einstiegspunkt für die Hotel und Flug Suche.

```
1 {
2   "href": "https://dev-api.travel.ch/HotelPlusFlight/Search",
3   "form": {
4     "description": { "href": "https://dev-api.travel.ch/HotelPlusFlight/
  SearchFormDescription/HotelPlusFlight" },
5     "submit": {
6       "href": "https://dev-api.travel.ch/HotelPlusFlight/Search{?
  destination,periodOfStay,roomOccupancies*,departureAirport,
  departureAirports*,targetPeriodOfStay,hotelCategories*,ratings,
  mealTypeCategories*,directFlight,flightClasses*,matchHotels,productType}"
```

```
7     },
8     "data":{ [...] },
9     "alternatives":null
10  },
11     "results":null
12 }
```

/HotelPlusFlight/Search führt eine Suche aus. Wird die Ressource jedoch wie hier ohne Parameter aufgerufen, so sind keine Suchresultate definiert. Hier wird auch beschrieben, wie eine Suche ausgeführt werden kann. Im Feld form->submit ist die [Uniform Resource Locator \(URL\)](#) definiert, welche für eine Suche aufgerufen werden muss. Bei dieser sind jedoch Parameter zwingend, welche in der SearchFormDescription unter form->description weiter beschrieben werden.

```
1 {
2   "href":"https://dev-api.travel.ch/HotelPlusFlight/SearchFormDescription/
HotelPlusFligh",
3   "destination":{
4     "type":{ "href":"https://dev-api.travel.ch/FieldType/Destination" },
5     "config":{
6       "suggestUri":{ "href":"https://dev-api.travel.ch/
DestinationSuggestions/HotelPlusFligh?q=%7B_%7D" }
7     }
8   },
9   "periodOfStay":{
10     "type":{ "href":"https://dev-api.travel.ch/FieldType/PeriodOfStay" },
11     "config":{
12       "minCheckInDate":"2016-06-16",
13       "maxCheckInDate":"2017-06-16",
14       "minNumberOfNights":1,
15       "maxNumberOfNights":90
16     }
17   },
18   "roomOccupancies":{
19     "type":{ "href":"https://dev-api.travel.ch/FieldType/RoomOccupancies"
20   },
21     "config":{
22       "minRooms":1,
23       "maxRooms":2,
24       "minAdultsPerRoom":1,
25       "maxAdultsPerRoom":6,
26       "minChildrenPerRoom":0,
27       "maxChildrenPerRoom":4,
28       "minChildAge":0,
```

```
28     "maxChildAge":18
29   },
30 },
31 "departureAirport":{
32   "type":{"href":"https://dev-api.travel.ch/FieldType/Airport" },
33   "config":[
34     { "href":"https://dev-api.travel.ch/Destination/A6299466" },
35     { "href":"https://dev-api.travel.ch/Destination/A6299720" },
36     { "href":"https://dev-api.travel.ch/Destination/A2660644" },
37     { "href":"https://dev-api.travel.ch/Destination/A6299722" },
38     { "href":"https://dev-api.travel.ch/Destination/A3208823" },
39     { "href":"https://dev-api.travel.ch/Destination/A3174133" }
40   ]
41 },
42 [...]
43 }
```

Dies ist die SearchFormDescription für die Hotel und Flug Suche. Sie beschreibt die Parameter, welche für eine Suche benutzt werden können.

```
1 {
2   "href":"https://dev-api.travel.ch/HotelPlusFlight/Search?destination=
https%3A%2F%2Fdev-api.travel.ch%2FDestination%2F6547539&periodOfStay.
checkInDate=2016-10-05&periodOfStay.checkOutDate=2016-10-08&[...]",
3   "form":{
4     [...]
5     "data":{
6       "destination":{"href":"https://dev-api.travel.ch/Destination/654753
9" },
7       "periodOfStay":{
8         "checkInDate":"2016-10-05",
9         "checkOutDate":"2016-10-08"
10      },
11      [...]
12      "roomOccupancies":[
13        { "dateOfBirths":[ null, null ] }
14      ],
15      "departureAirport":{"href":"https://dev-api.travel.ch/Destination/A
6299722" },
16      [...]
17    },
18    "alternatives":null
19  },
20  "results":{
```

```

21     "href": "https://dev-api.travel.ch/HotelPlusFlight/SearchResultPage/19E
22     2B6BB-1F06-4396-B86E-169EDB368B0E?hotelCategories%5B0%5D=https%3A%2F%2Fdev
23     -api.travel.ch%2FHotelCategory%2FNone[...]"
    }
  }

```

Bei dieser Anfrage handelt es sich um eine Suche, dieses mal mit Suchparametern. Das Feld results ist nun mit einer SearchResultPage befüllt.

```

1  {
2    "href": "https://dev-api.travel.ch/HotelPlusFlight/SearchResultPage/19E2B
3    6BB-1F06-4396-B86E-169EDB368B0E?sortMethod.type=https%3A%2F%2Fdev-api.
4    travel.ch%2FSortType%2FPrice&ratings.minimal=0[...]",
5    "citytripSearch": {
6      "href": "https://dev-api.travel.ch/HotelPlusFlight/Search?destination=
7      https%3A%2F%2Fdev-api.travel.ch%2FDestination%2F6547539&periodOfStay.
8      checkInDate=2016-10-05&periodOfStay.checkOutDate=2016-10-08[...]"
9    },
10   "form": {
11     "description": {
12       "href": "https://dev-api.travel.ch/HotelPlusFlight/
13       SearchResultPageFormDescription/19E2B6BB-1F06-4396-B86E-169EDB368B0E?
14       searchResultPageId=E5241069-1272-4DD0-946E-46DD2E9D55E6"
15     },
16     "data": { [...] },
17     "facets": { [...] },
18     "submit": {
19       "href": "https://dev-api.travel.ch/HotelPlusFlight/SearchResultPage/1
20       9E2B6BB-1F06-4396-B86E-169EDB368B0E{?paging.offset,paging.size,[...],
21       luggageOptions*}"
22     },
23     "alternatives": null
24   },
25   "teaserForm": { [...] },
26   "totalCount": 263,
27   "count": 263,
28   "results": {
29     "href": "https://dev-api.travel.ch/HotelPlusFlight/SearchResults/E52410
30     69-1272-4DD0-946E-46DD2E9D55E6"
31   },
32   "mapFeatures": {
33     "href": "https://dev-api.travel.ch/HotelPlusFlight/SearchResultPage/E52
34     41069-1272-4DD0-946E-46DD2E9D55E6/Map"
35   }
36 }

```

```
26 }
```

Dies ist die SearchResultPage. Auf travel.ch wird diese verwendet, um die Resultate weiter zu filtern. Dies ist jedoch bei diesem Projekt nicht vorgesehen. Von dieser Anfrage wird nur das Feld results benötigt, welches die Suchresultate beinhaltet.

```
1 {
2   "href":"https://dev-api.travel.ch/HotelPlusFlight/SearchResults/E5241069
  -1272-4DD0-946E-46DD2E9D55E6",
3   "items":[
4     {
5       "href":"https://dev-api.travel.ch/HotelPlusFlight/SearchResult/21ca6
  5ff-976a-4a3e-bc92-b17608db3b25?pageId=58B8D81D-12B5-4EEE-9CE9-902CCDF0179
  1",
6       "citytripSearchResultPage":{
7         "href":"https://dev-api.travel.ch/HotelPlusFlight/SearchResultPage
  /19E2B6BB-1F06-4396-B86E-169EDB368B0E?paging.offset=0&paging.size=12&
  sortMethod.type=https%3A%2F%2Fdev-api.travel.ch%2FSortType%2FPrice[...]"
8       },
9       "hotelInformation":{
10        "href":"https://dev-api.travel.ch/HotelInformation/DDA48396-8105-4
  33D-B75B-668F96EC89D8"
11      },
12      "offer":{
13        "href":"https://dev-api.travel.ch/HotelPlusFlight/Offer?
  destination=https%3A%2F%2Fdev-api.travel.ch%2FDestination%2FH_63CA0931EAD8
  A33EF988144360A8056C&offerHotelInformation=https%3A%2F%2Fdev-api.travel.ch
  %2FHotelInformation%2FDDA48396-8105-433D-B75B-668F96EC89D8[...]",
14        "price":{
15          "actualTotal":{
16            "amount":45512,
17            "currency":{ "href":"https://dev-api.travel.ch/Currency/CHF" }
18          },
19          "previousTotal":null,
20          "averagePrice":{
21            "type":{ "href":"https://dev-api.travel.ch/AveragePriceType/
  PerPerson" },
22            "value":{
23              "amount":22800,
24              "currency":{ "href":"https://dev-api.travel.ch/Currency/CHF"
25            }
26          }
27        }
28      }
29    }
30  ]
31 }
```

```
28     },
29     "availability":null,
30     "labels":null,
31     "distanceInMeters":null
32   },
33   [...]
34 ]
35 }
```

Das sind die Hotelresultate der Suche. Diese Antwort ist sehr verkürzt. Es wird nur ein Resultat hier angezeigt um die Übersicht zu behalten.

```
1 {
2   "href":"https://dev-api.travel.ch/HotelInformation/dda48396-8105-433d-b7
3   5b-668f96ec89d8",
4   "name":"Generator Berlin Prenzlauer Berg",
5   "descriptions":{
6     "href":"https://dev-api.travel.ch/HotelInformation/dda48396-8105-433d-
7     b75b-668f96ec89d8/Descriptions",
8     "hotel":{ "href":"https://dev-api.travel.ch/HotelInformation/dda48396-
9     8105-433d-b75b-668f96ec89d8" },
10    "items":[
11      {
12        "type":{ "href":"https://dev-api.travel.ch/HotelDescriptionType/
13        Hotel" },
14        "text":"Das Stadthotel ist in einem futuristischen Design erbaut
15        und befindet sich in der Nahe der Berlin Arena und der S-Bahnstation
16        Landsberger Allee, nur sechs Strassenbahnhaltestellen vom Alexanderplatz
17        entfernt. Das stilvolle Hostel verfuegt ueber farbenfrohe Zimmer mit
18        Schliessfach und eigenem Bad bzw. Gemeinschaftsbad und die Bar organisiert
19        taegliche Veranstaltungen und woechentliche Partys. Darueber hinaus
20        werden Karaoke-Abende, Filmvorfuehrungen und kostenlose Wanderungen
21        angeboten. Dieses Hostel verspricht einen einzigartigen und
22        unvergesslichen Aufenthalt."
23      }
24    ]
25  },
26  "images":{
27    "href":"https://dev-api.travel.ch/HotelInformation/dda48396-8105-433d-
28    b75b-668f96ec89d8/ImageGallery",
29    "hotel":{ "href":"https://dev-api.travel.ch/HotelInformation/dda48396-
30    8105-433d-b75b-668f96ec89d8" },
31    "main":{
32      "href":"https://dev-api.travel.ch/HotelInformation/dda48396-8105-433
```

```
d-b75b-668f96ec89d8/Image/SG90ZWxcRERBNFw4Mzk2LTgxMDUtNDMzRC1CNzVCLTY2OEY5
NkVDOD1EOFwwMTU5NzVhX2hiX2JhXzAxMi5qcGc=",
19     "url": "//dev-images.travel.ch/pictures/Hotel/DDA4/8396-8105-433D-B75
B-668F96EC89D8/015975a_hb_ba_012.jpg",
20     "alt": null,
21     "description": null
22 },
23     "items": [
24     {
25         "href": "https://dev-api.travel.ch/HotelInformation/dda48396-8105-4
33d-b75b-668f96ec89d8/Image/SG90ZWxcRERBNFw4Mzk2LTgxMDUtNDMzRC1CNzVCLTY2
OEY5NkVDOD1EOFwwMTU5NzVhX2hiX2JhXzAxMi5qcGc=",
26         "url": "//dev-images.travel.ch/pictures/Hotel/DDA4/8396-8105-433D-B
75B-668F96EC89D8/015975a_hb_ba_012.jpg",
27         "alt": null,
28         "description": null
29     },
30     {
31         "href": "https://dev-api.travel.ch/HotelInformation/dda48396-8105-4
33d-b75b-668f96ec89d8/Image/SG90ZWxcRERBNFw4Mzk2LTgxMDUtNDMzRC1CNzVCLTY2
OEY5NkVDOD1EOFwwMTU5NzVhX2hiX2xfMDA5LmpwZw==",
32         "url": "//dev-images.travel.ch/pictures/Hotel/DDA4/8396-8105-433D-B
75B-668F96EC89D8/015975a_hb_l_009.jpg",
33         "alt": null,
34         "description": null
35     },
36     [...]
37 ]
38 },
39     "facts": {
40         "href": "https://dev-api.travel.ch/HotelInformation/dda48396-8105-433d-
b75b-668f96ec89d8/Facts",
41         "hotel": {
42             "href": "https://dev-api.travel.ch/HotelInformation/dda48396-8105-433
d-b75b-668f96ec89d8"
43         },
44         "items": [
45         {
46             "type": { "href": "https://dev-api.travel.ch/FactGroupType/
Environment" },
47             "facts": [
48             {
49                 "type": { "href": "https://dev-api.travel.ch/FactType/Simple" },
50                 "data": "Busbahnhof/Bahnhof 3500 m"
```

```
51     },
52     {
53       "type":{ "href":"https://dev-api.travel.ch/FactType/Simple" },
54       "data":"Vergnuegungsviertel 3500 m"
55     },
56   ]
57 },
58 {
59   "type":{ "href":"https://dev-api.travel.ch/FactGroupType/Hotel" },
60   "facts":[
61     {
62       "type":{ "href":"https://dev-api.travel.ch/FactType/Simple" },
63       "data":"NO Haustiere auch ueber 5 kg erlaubt"
64     },
65     {
66       "type":{ "href":"https://dev-api.travel.ch/FactType/Simple" },
67       "data":"Waescherel"
68     },
69     [...]
70   ]
71 },
72 {
73   "type":{ "href":"https://dev-api.travel.ch/FactGroupType/Room" },
74   "facts":[
75     {
76       "type":{ "href":"https://dev-api.travel.ch/FactType/Simple" },
77       "data":"NO Raucherzimmer"
78     },
79     {
80       "type":{ "href":"https://dev-api.travel.ch/FactType/Simple" },
81       "data":"Dusche"
82     },
83     [...]
84   ]
85 }
86 ]
87 },
88 "ratings":null,
89 "geoLocation":{
90   "type":"Point",
91   "coordinates":[ 13.455673, 52.529681 ]
92 },
93 "destination":{ "href":"https://dev-api.travel.ch/Destination/H_63CA0931
EAD8A33EF988144360A8056C" },
```



```

94   "keyFacts":[
95     { "href":"https://dev-api.travel.ch/KeyFact/Bar" },
96     { "href":"https://dev-api.travel.ch/KeyFact/WLAN" }
97   ],
98   "labels":[
99     { "href":"https://dev-api.travel.ch/Label/CustomerRecommendation" }
100  ],
101   "hotelCategory":{"href":"https://dev-api.travel.ch/HotelCategory/One" }
102 ,
103   "addressLines":[ "10407 Prenzlauer Berg", "Deutschland" ],
104   "tourOperator":null,
105   "tourOperatorGtcUrl":null
106 }

```

Jedes Suchresultat der vorherigen Anfrage beinhaltet eine solche (oben) HotelInformation. Diese beherbergt alle Informationen eines Hotels, die nicht an ein Angebot gebunden ist. Zum Beispiel Bilder, Beschreibungen, etc. Was nicht enthalten ist sind Preise.

```

1  {
2    "href":"https://dev-api.travel.ch/HotelPlusFlight/Offer?destination=
https%3A%2F%2Fdev-api.travel.ch%2FDestination%2FH_9675C54E32056B7DC2393EE1
D758FEB9&offerHotelInformation=https%3A%2F%2Fdev-api.travel.ch%2
FHHotelInformation%2F46FFEA61-6A24-4FCB-AE6A-524CE6E752DF[...]",
3    "tourOperator":{"href":"https://dev-api.travel.ch/TourOperator/TWCH" },
4    "form":{"
5      "data":{"
6        "destination":{"href":"https://dev-api.travel.ch/Destination/H_9675
C54E32056B7DC2393EE1D758FEB9" }
7      }
8    },
9    "selectForm":{" [...] },
10   "hotelForm":{"
11     "description":{"href":"https://dev-api.travel.ch/HotelPlusFlight/
HotelConfigurationFormDescription?offerHotelInformation=https%3A%2F%2Fdev-
api.travel.ch%2FHHotelInformation%2F46FFEA61-6A24-4FCB-AE6A-524CE6E752DF[
...]" },
12     [...]
13   },
14   "flightForm":{"
15     "description":{"href":"https://dev-api.travel.ch/HotelPlusFlight/
FlightConfigurationFormDescription?offerHotelInformation=https%3A%2F%2Fdev-
api.travel.ch%2FHHotelInformation%2F46FFEA61-6A24-4FCB-AE6A-524CE6E752DF&
roomOccupancies%5B0%5D.dateOfBirths%5B0%5D=null[...]" },
16     [...]

```

```

17   },
18   "orderForm":{
19     "description":{ "href":"https://dev-api.travel.ch/HotelPlusFlight/
OrderFormDescription" },
20     "submit":{ "href":"https://dev-api.travel.ch/HotelPlusFlight/
SearchResult/Offer/Order?destination=https%3A%2F%2Fdev-api.travel.ch%2
F46FFEA61-6A24-4FCB-AE6A-524CE6E752DF[...] ",
21     "action":"POST"
22   }
23 },
24 "price":{
25   "actualTotal":{
26     "amount":174370,
27     "currency":{ "href":"https://dev-api.travel.ch/Currency/CHF" }
28   },
29   "previousTotal":null,
30   "averagePrice":{
31     "type":{ "href":"https://dev-api.travel.ch/AveragePriceType/
PerPerson" },
32     "value":{
33       "amount":87200,
34       "currency":{ "href":"https://dev-api.travel.ch/Currency/CHF" }
35     }
36   }
37 },
38 "travelPeriod":{
39   "startDate":"2016-10-05",
40   "endDate":"2016-10-12"
41 },
42 "hotelInformation":{ "href":"https://dev-api.travel.ch/HotelInformation/
46FFEA61-6A24-4FCB-AE6A-524CE6E752DF" },
43 "transfer":{ "href":"https://dev-api.travel.ch/HotelPlusFlight/Transfer/
NotIncluded" }
44 }

```

Dies ist ein Offer und Zeigt die Hotelzimmer Konfigurationen (RoomTypes, MealTypes) und die Flüge, welche zum Hotel gebucht werden können.

Aus dieser Auflistung wird ersichtlich, dass es sehr viele API Anfragen benötigt, um ein Produkt auszuwählen. Hier wurden noch diverse andere Requests weggelassen. Zum Beispiel müssen noch Destinationen, Preisbeschreibungen, etc. abgeholt werden. Diese sind für das Verständnis jedoch nicht nötig und wurden deshalb weggelassen.

3.1.1 Testumgebung

Für dieses Projekt muss die Testumgebung verwendet werden. Dies stellt sicher, dass keine Testbuchungen ins produktive System gelangen und die Authorisierungsdaten der Live-API nicht veröffentlicht werden. Die [URL](https://dev-api.travel.ch) der Umgebung lautet: <https://dev-api.travel.ch>

3.1.2 Checkout

Der Checkout gibt es in der Testumgebung seit dem letzten Projekt nicht mehr. Dieser wurde ausgelagert in eine Umgebung die nicht mehr unserem Team unterstellt ist und kann somit nicht nur BPMN modelliert werden.

3.2 Bonita

Also BPMN Programm wurde Bonita festgelegt (siehe Abschnitt [2.4.2 Programmanalyse](#)). Es bietet zwar nicht die gesamte benötigte Features, ist jedoch auf Unix und auf Windows lauffähig und gratis. In diesem Abschnitt wird eine Einführung in das Tool gegeben.

Bonita wird von der Firma Bonitasoft entwickelt und besteht aus einem Programm, welches in Java geschrieben ist, sowie aus einer Webseite, welche standardmässig auf einem mitgelieferten Tomcat Webserver¹ läuft. Nachfolgend wird das Interface des Programmes und die verwendeten Funktionen der Software beschrieben.

3.2.1 Interface

Wenn man ein neues Projekt erstellt, sieht das Programm folgendermassen aus:

¹ *Apache Tomcat® - Welcome!* URL: <https://tomcat.apache.org/> (besucht am 12.06.2016).

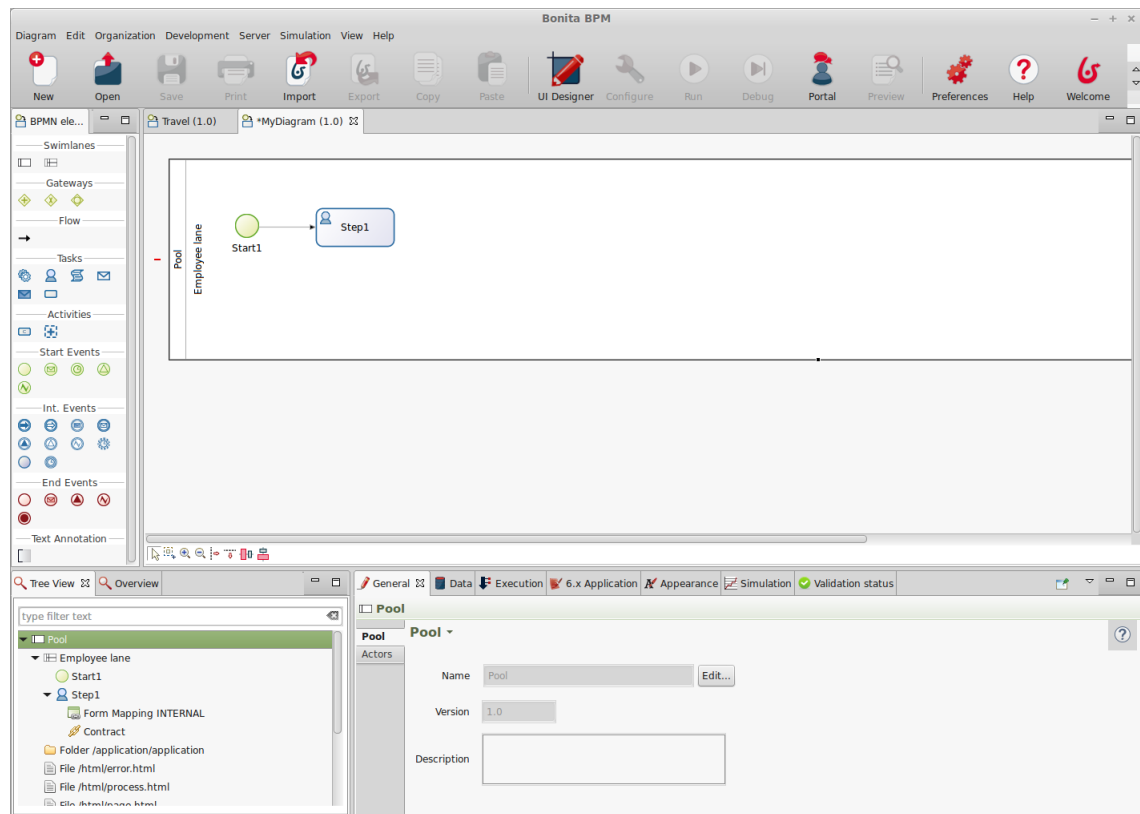


Abbildung 3.1: Bonita Interface

Zuerst ist die Toolbar. Darunter auf der linken Seite sind die BPMN Elemente (siehe Abschnitt 2.1.1 Notation) wie z.B. die Swimlanes, Gateways und Events. Daneben kann der Prozess definiert werden in der Process View. Bei einem neuen Projekt hat es einen Start Event (mit dem Namen „Start1“) und ein Task „Step1“. Bei dem Task hat es links oben ein Benutzer Icon. Dieses sagt aus, dass es sich um einen Human Task handelt, bei welchem eine Benutzerinteraktion benötigt wird.

Zuunterst links ist eine Übersicht über alle Elemente im Projekt (Tree View) und daneben können die BPMN Elemente im Projekt konfiguriert werden (Configuration View).

Bonita ist mit Java implementiert. Auch die Erweiterungen (siehe Abschnitt 3.2.3 Connectors) werden in dieser Sprache entwickelt. Für das Verständnis wird vorausgesetzt, dass man grundlegendes Wissen für Java Begriffe besitzt (z.B. Namespace, Class, Implementation, Interface, etc.).

3.2.2 Business Data Model

BDM werden für die Modellierung der Daten benötigt. Connectoren können später diese Daten entgegen nehmen oder zurückgeben. Intern handelt es sich dabei um Java Classes.

Ein **BDM** hat einen Namen und Felder. Felder haben wiederum einen Namen und einen Java Datentypen (string, int oder ein weiteres **BDM**). Ein Feld kann zusätzlich noch die zwei Attribute Multiple oder Mandatory haben. Ist Multiple gesetzt, so handelt es sich bei dem Feld um eine Liste, und Mandatory macht es zwingend, dass bei der Erstellung ein

Wert zugewiesen wird.

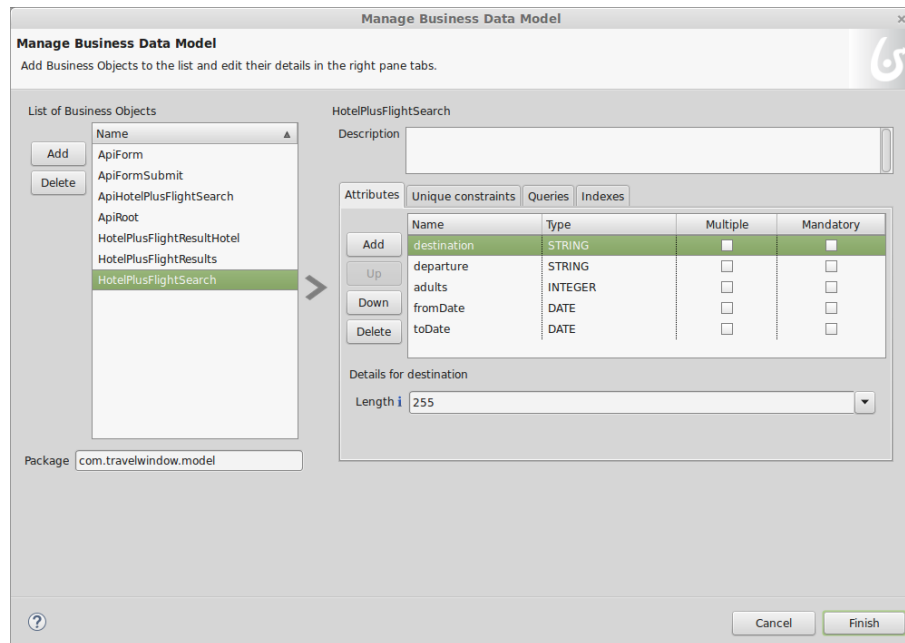
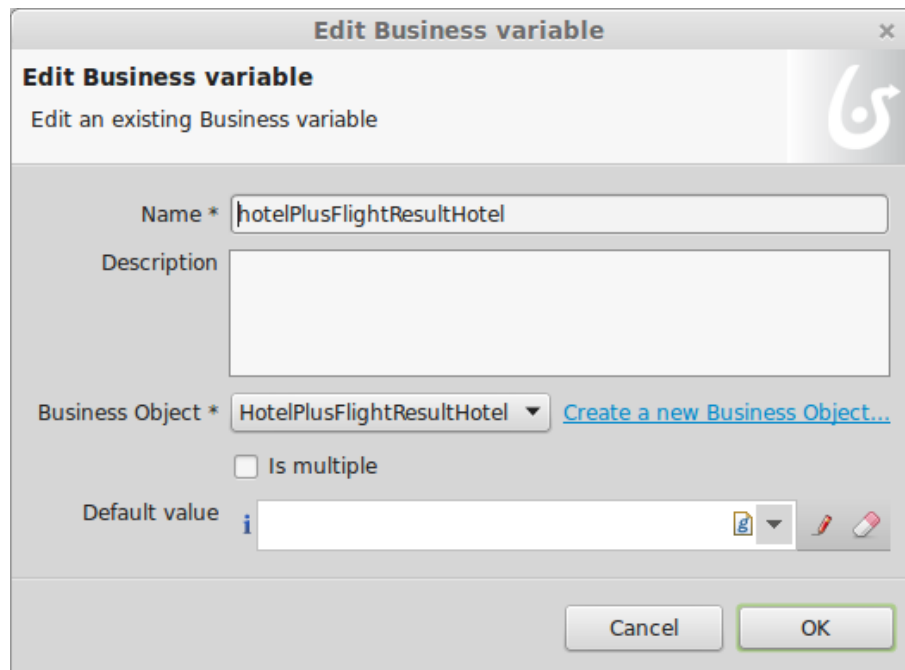


Abbildung 3.2: Bonita BPM

Pool Variables

Ist in der Process View ein Pool angewählt, so kann in der Configuration View unter „Data -> Pool Variables“ neue Pool Variablen definiert werden. Diese sind für alle Tasks in diesem Pool und Swimlanes sichtbar und können von den Connectors verwendet werden. Dabei handelt es sich um Instanzen von **BDMs**. Eine Pool Variable besteht aus einem Namen, einem **BDM** und einem Initialisierungs-Skript, um die Mandatory Fields abzufüllen.



Edit Business variable

Edit an existing Business variable

Name * hotelPlusFlightResultHotel

Description

Business Object * HotelPlusFlightResultHotel [Create a new Business Object...](#)

☐ Is multiple

Default value

Cancel OK

Abbildung 3.3: Bonita Pool Variables

3.2.3 Connectors

Connectors werden in Bonita dazu verwendet, um eine Aktion durchzuführen.

Bonita wird mit folgenden Connectors ausgeliefert:

- CMS
- CRM
- Calendar
- Database
- ERP
- LDAP
- Messaging
- Reporting
- SOAP Web Services
- Script
- Social
- Talend

Connectors können einem Task angehängt werden. Klickt man einen Task in der Process View an, kann man in der Configuration View unter Execution Connectors hinzufügen.

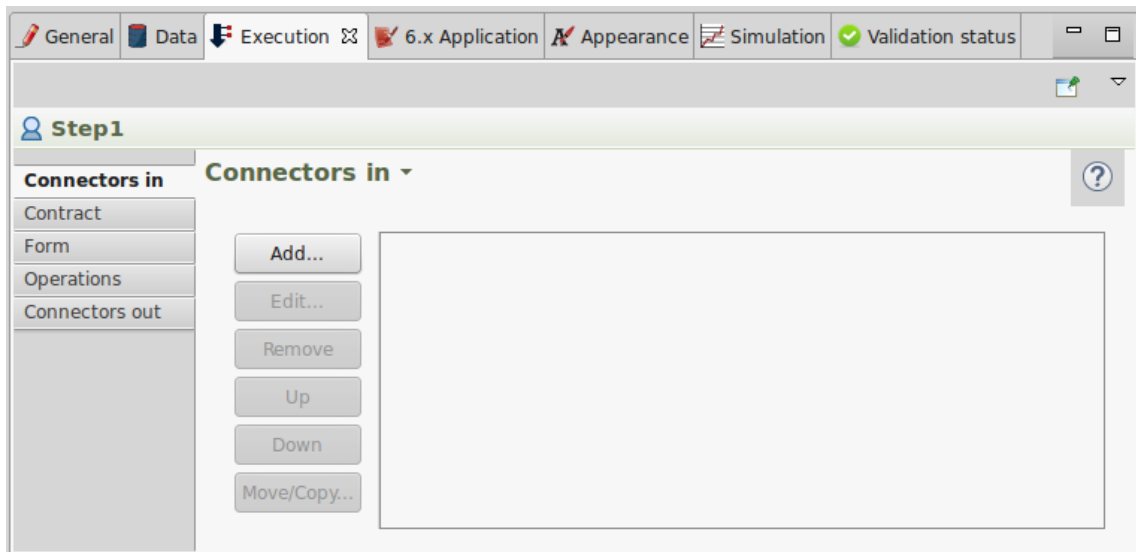


Abbildung 3.4: Bonita Connectors

Es gibt in und out Connectors. Diese werden entweder vor dem Task, oder nach dem Task ausgeführt. Bei einem Human Task kann so definiert werden, ob der Connector vor der Benutzerinteraktion ausgeführt wird oder danach.

Ein Connector besteht aus einer Definition und einer Implementation, welche nachfolgend beschrieben sind. In der Definition ist spezifiziert, was die Eingaben und Ausgaben sind. Wird ein Connector einem Task angehängt, so müssen die Eingaben definiert sowie ein Mapping der Ausgabeparameter gemacht werden.

Definition

Die Definition eines Connectors besteht aus Beschreibungsdaten (Name, Namespace, Kategorie, etc.), Input-Parameter, Wizard Pages und Output-Parameter. Als Input- und Output-Parameter können Java Classes oder [BDMs](#) definiert werden. Wizard Pages werden dazu benötigt, dass wenn ein Connector einem Task hinzugefügt wird die entsprechenden Input-Parameter definiert werden können.

Implementation

Um eine Implementation eines Connectors anzulegen, muss zuerst eine Definition gewählt und Beschreibungsdaten angegeben werden.

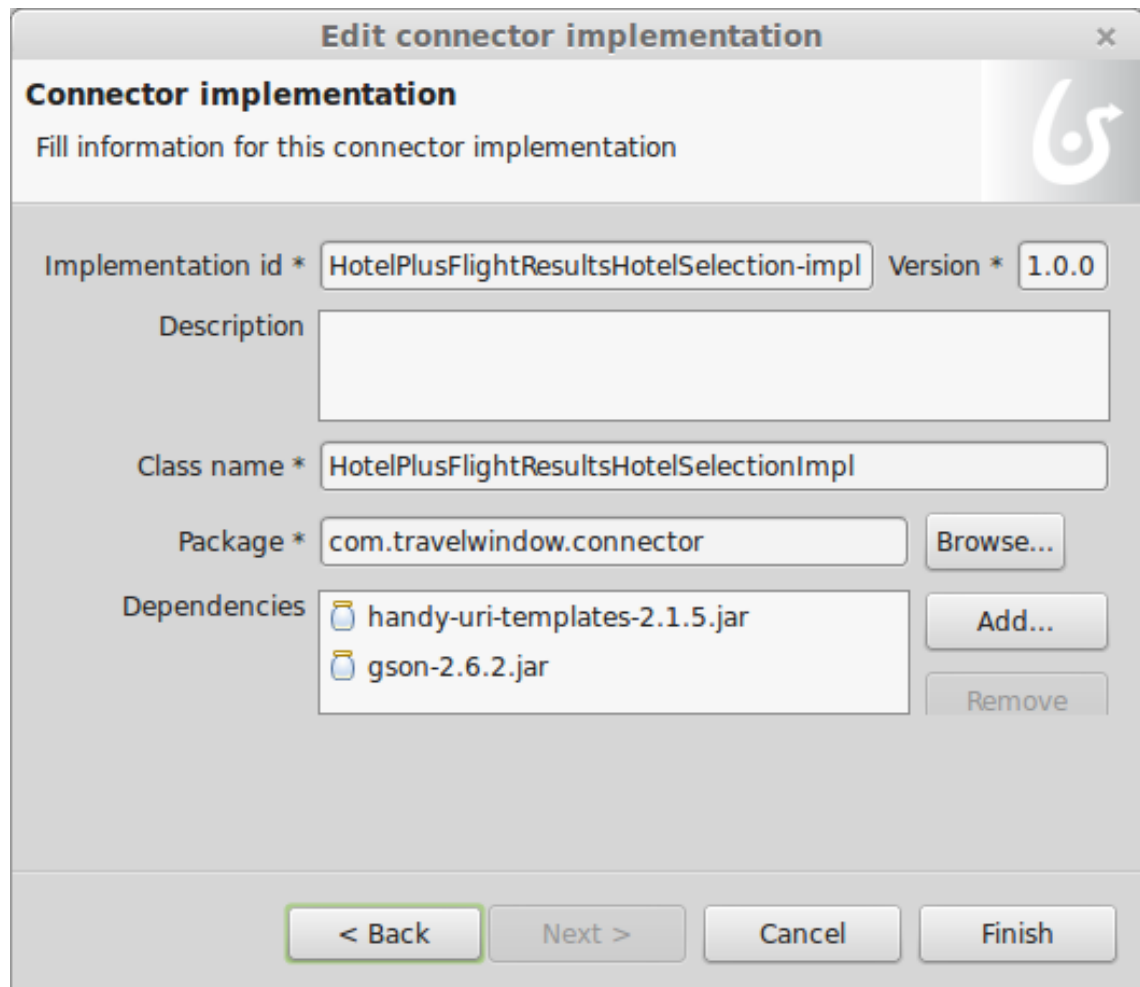


Abbildung 3.5: Bonita Connector Implementation

Die Implementation id kann frei gewählt werden, muss im Projekt jedoch eindeutig sein. Der Class name ist der Name der Java Klasse die erstellt wird. Dependencies sind JAR Files welche von innerhalb der Java Klasse benötigt werden.

Bestätigt man den obigen Dialog, so wird eine Java Klasse und abstrakte Klasse erzeugt. Nach dem obigen Beispiel werden die beiden Classes `HotelPlusFlightResultsHotelSelectionImpl` und `AbstractHotelPlusFlightResultsHotelSelectionImpl` erstellt. Die Hauptmethode der ersten ist die Methode `executeBusinessLogic`. In der Definition dieses Connectors wurde definiert, dass ein [BDM](#) `HotelPlusFlightResultHotel` zurückgegeben wird, welches den namen „selection“ hat. Deshalb muss die Methode `executeBusinessLogic` die Methode „setSelection“ aufrufen und dieser eine Klasse vom Typen `HotelPlusFlightResultHotel` übergeben, um der Definition zu genügen. Die Methode `setSelection` ist dabei in der `AbstractHotelPlusFlightResultsHotelSelectionImpl` definiert.

Die Funktionalität eines Connectors kann somit frei implementiert werden. Zwingend ist nur das die Methode `setSelection` aufgerufen wird, sonst die die Ausführung des Connectors

fehlerhaft.

3.2.4 Forms, Contracts und Operations

Bei einem Human Task muss ein Form angehängt werden, um dem Benutzer eine Interaktion in dem Prozess zu ermöglichen. In einem Task kann direkt ein neues Formular erstellt werden. Es empfiehlt sich jedoch, zuerst den Contract zu definieren, da die dort angegebenen Informationen automatisch bei der Erstellung des Forms und den Operations mitberücksichtigt werden. Weitere Informationen dazu werden den folgenden beiden Abschnitten gegeben.

Contract

Ein Contract definiert die Daten, die vom Formular erwartet werden. Sie sind dafür verantwortlich, welche Informationen aus dem Process an das Form übergeben werden. Die dort definierten Werte können Java Klassen oder BDMs sein. Beim Task für die Auswahl eines Hotels in der Hotel und Flug suche, könnte dies die Liste der Hotelresultate sein. Dazu wird im Contract eine Liste von Hotels angegeben.

HotelPlusFlight Search

Task Inputs

A contract defines the information that a task requires to execute. If inputs are missing or constraints are not fulfilled on form submission, the task is not executed and stays ready. After you have defined the contract, you can generate the form by clicking the UI Designer icon on the right of this panel. This generates a form with the relevant widgets and data bindings for the task inputs. You must handle constraints with validation in the form. You can use a contract input value in operations and connectors out (on finish).

Name *	Type	Multiple	Description
hotelPlusFlightSearchInput	COMPLEX	<input type="checkbox"/>	
destination	TEXT	<input type="checkbox"/>	
departure	TEXT	<input type="checkbox"/>	
adults	INTEGER	<input type="checkbox"/>	
fromDate	DATE	<input type="checkbox"/>	
toDate	DATE	<input type="checkbox"/>	

Abbildung 3.6: Bonita Form Contract

Operations

Operations werden nach der Ausführung des Formulars ausgeführt. Sie definieren, wie die Informationen aus dem Formular zurück in den Prozess fließen.

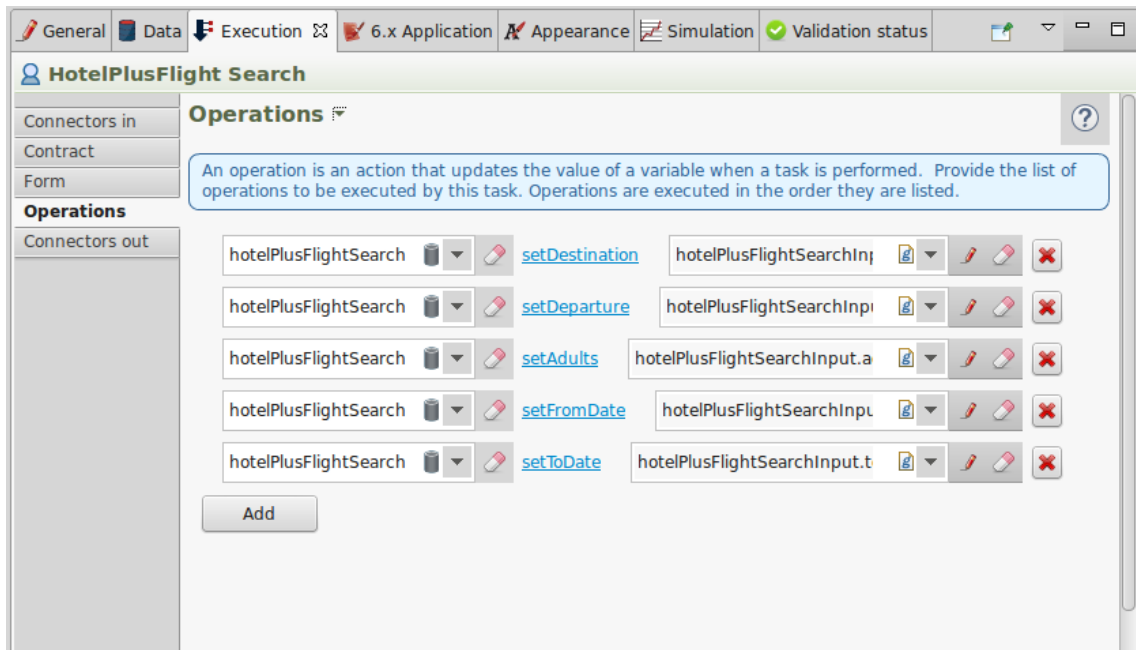


Abbildung 3.7: Bonita Form Operations

Im obigen Beispiel wird definiert, dass auf der Pool Variable `hotelPlusFlightSearch` die Java Methode `setDestination` aufgerufen wird. Der Methode wird der Wert `hotelPlusFlightSearchInput.destination` übergeben. Bei `hotelPlusFlightSearchInput` handelt es sich um eine Variable, welche auf dem Formular definiert wurde (siehe [Forms](#)).

Forms

Ein Form ermöglicht die Interaktion eines Benutzers. Die Daten die an das Form übergeben werden und wie die bearbeiteten Daten zurück in den Process fließen, ist in dem Contract und den Operations definiert (siehe [Contract](#) und [Operations](#)). Es ist zu empfehlen, dass zuerst diese beiden Informationen definiert werden, da sie bei der Erstellung des Forms direkt weiterverwendet werden. Definiert man bei dem Contract dass eine Liste von Hotels übergeben wird, so wird bei der Erstellung des Formulars automatisch eine Liste generiert welche diese Daten anzeigt.

Um ein Formular aus den Daten des Contracts und der Operations zu erstellen, muss dieses in der Configuration View erstellt werden.

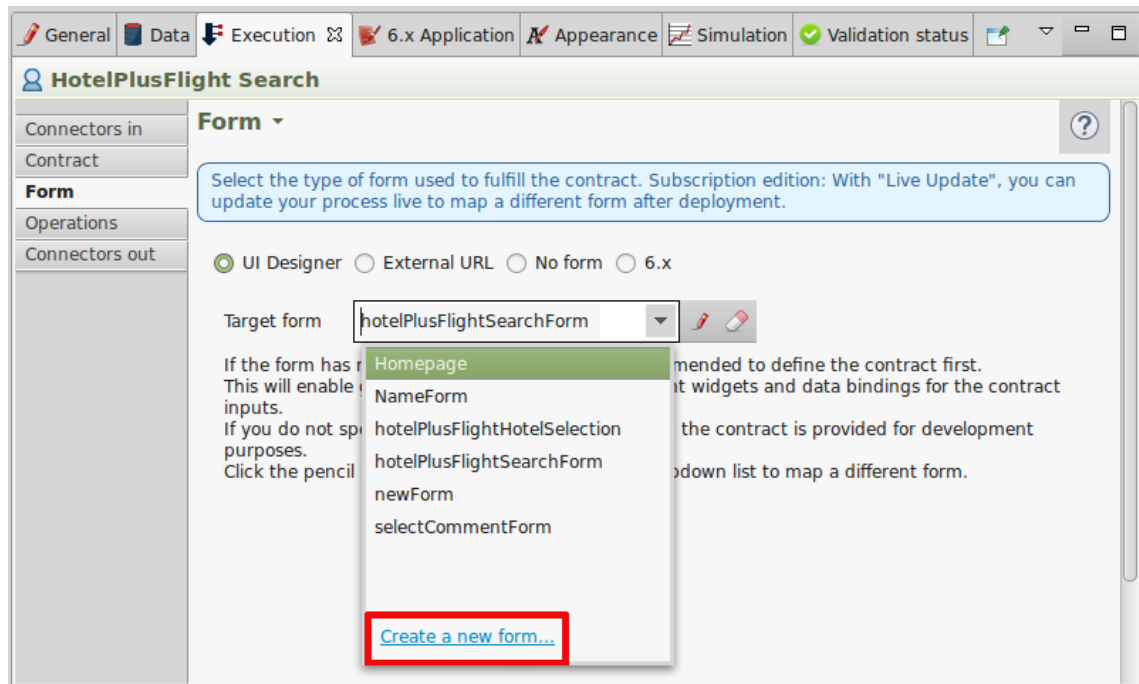


Abbildung 3.8: Bonita Form Erstellung aus Contract und Operations

Danach öffnet sich der UI Designer. Dies ist ein Webinterface, welches auf dem mitgelieferten Tomcat Webserver läuft. Dort kann das Formular angepasst werden. Die Funktionalität des UI Designers ist gross. Einen guten Überblick bietet ein Video Tutorial, welches von Bonitasoft bereit gestellt wird und auf folgender URL eingesehen werden kann <http://www.bonitasoft.com/resources/videos/getting-started-tutorial>. Ab 35:55 wird der UI Designer vorgestellt.

KAPITEL 4

Konzept

Das Konzept umfasst die Mockups der Formulare, welche in Bonita dargestellt werden, sowie der Prozess welcher modelliert werden soll.

4.1 Mockups

Der Buchungsprozess vom Hotel und Flug Produkt von travel.ch benötigt vier Ansichten, für welche hier Mockups aufgezeigt werden. Diese wurden mit dem UI Designer von Bonita erstellt (siehe Abschnitt [3.2.4 Forms](#)).

Preview - hotelPlusFlightSearchForm

Hotel Plus Flight Search Input

Destination

Departure

Adults

From Date

To Date

Submit

Abbildung 4.1: Suchformular Mockup

Das Suchformular ist vereinfacht dargestellt. Für die Destination muss man einen gültigen Namen eingeben. Auf der Webseite travel.ch wird dem User beim Tippen Vorschläge angezeigt. Dies wurde für dieses Projekt nicht umgesetzt.

Für die Passagiere kann im obigen Formular nur eine Anzahl für die Erwachsenen angegeben werden. Kinder können nicht angewählt werden, da für diese zusätzlich noch das Geburtsdatum angegeben werden muss. Auch die Wahl nach mehreren Zimmern ist

nicht möglich. Für die Anzahl der Erwachsenen wird ein Hotelzimmer gesucht.

Preview - hotelPlusFlightHotelSelection

Hotel Plus Flight Results Input

Hotels



Name	Price	
Radisson Blu Hotel Berlin	482	Select



Name	Price	
NH Berlin Mitte	403	Select



Name	Price	
Wyndham Berlin Excelsior	350	Select

Abbildung 4.2: Mockup für die Hotel Suchresultate

Diese Ansicht zeigt ein Foto des Hotels, dessen Namen und der günstigste Preis. Von der [API](#) werden noch mehrere Fotos, eine Geokoordinate für die Anzeige einer Maps, Bewertungsdaten, etc. übermittelt. Diese Informationen werden jedoch nicht angezeigt.

Preview - hotelPlusFlightSelectRoomConfig

Hotel Plus Flight Room Configs Input

Configs

Room Type	Meal Type	Price	
Junior Suite	All-Inclusive	850	Select
Double Room King Size	Halfboard	650	Select
Besenkammer	Brot & Wasser	50	Select

Abbildung 4.3: Mockup für die Hotelzimmer Konfiguration

Auf der travel.ch Seite werden die Zimmer-Konfigurationen als Matrix dargestellt (siehe ?? ??). In Bonita ist dies jedoch nur sehr umständlich abbildbar. Deshalb wurde die Ansicht vereinfacht und als Liste dargestellt.

Preview - hotelPlusFlightSelectFlight

Hotel Plus Flight Flight Results Input

Flights

Airline	From Airport	To Airport	Flight1 from time	Flight1 to time
LH	ZRH	TXL	20.10.2016 20:15	20.10.2016 22:15
	Select		Flight2 from time	Flight2 to time
			22.10.2016 12:25	22.10.2016 14:35
LX	ZRH	TXL	20.10.2016 16:00	20.10.2016 18:05
	Select		Flight2 from time	Flight2 to time
			22.10.2016 20:45	22.10.2016 22:40
LX	ZRH	TXL	20.10.2016 06:30	20.10.2016 08:35
	Select		Flight2 from time	Flight2 to time
			22.10.2016 16:00	22.10.2016 18:00

Abbildung 4.4: Mockup für die Flugresultate

4.2 Prozess

Der Prozess der travel.ch Seite ist linear und wurde im Abschnitt [2.2.1 Travelwindow AG Prozesse](#) abgebildet. Für die Modellierung in Bonita wird eine Veränderung vorgenommen.

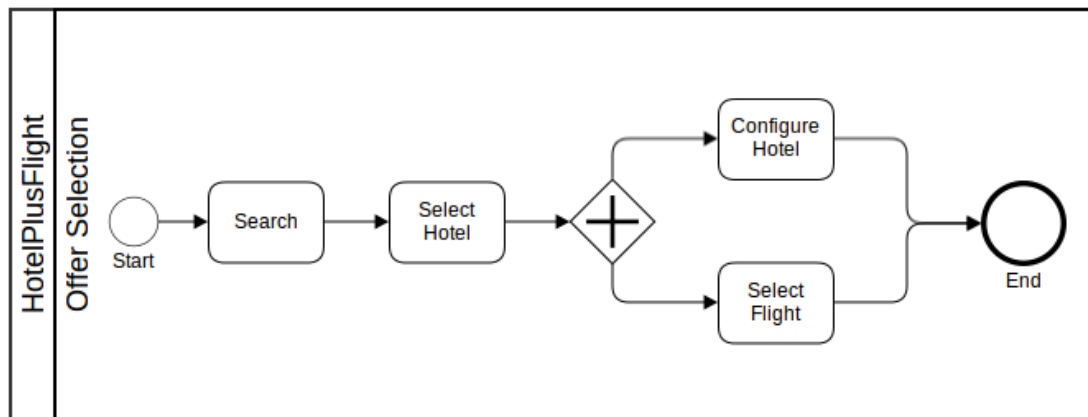


Abbildung 4.5: Hotel plus Flight BPMN Model für dieses Projekt

Die API von travelwindow AG gibt vor, dass zuerst gesucht und ein Hotel ausgewählt wird. Ob danach das Hotel konfiguriert oder der Flug gewählt wird ist hinfällig. Dies ist in der API so modelliert, dass im HotelPlusFlightOffer direkt das Formular für die Hotelkonfiguration und die Flugauswahl vorhanden sind (siehe den letzte Anfrage im Abschnitt [3.1 Analyse der travelwindow AG API](#)). Deshalb wurde ein parallel Gateway eingefügt der diesen Umstand im BPMN abbildet.

KAPITEL 5

Umsetzung

5.1 Prozess

Der Prozess wurde equivalent aufgesetzt, wie es im Abschnitt [4.2 Prozess](#) beschrieben wurde.

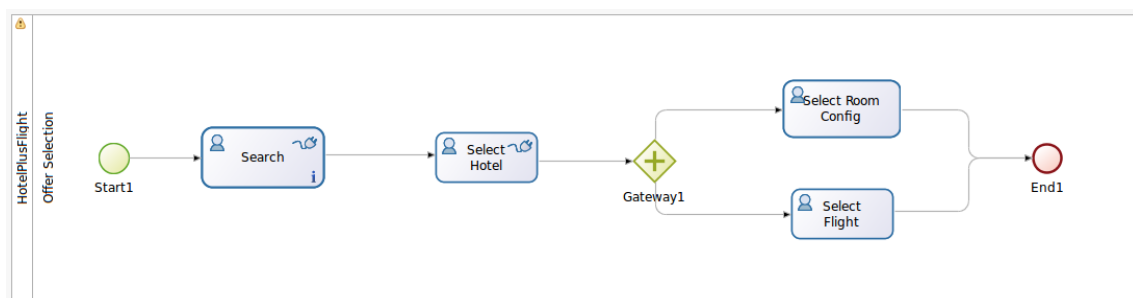


Abbildung 5.1: Prozess im Bonita

Das Icon oben Links in den Task signalisiert, dass es sich dabei um einen Human Task handelt und eine Benutzerinteraktion erforderlich ist. Das Bild oben rechts in den Task zeigt an, dass es einen Connector auf dem Task gibt.

5.2 Business Data Models und Pool Variables

Als [BDMs](#) wurden folgende Modelle definiert:

Tabelle 5.1: Business Data Models

Name	Type	Multiple
HotelPlusFlightSearch		
destination	string	
departure	string	
adults	integer	
fromDate	date	
toDate	date	
id	string	
HotelPlusFlightResults		
hotels	HotelPlusFlightResultHotel	x
HotelPlusFlightResultHotel		
name	string	
price	string	
image	string	
roomConfigs	HotelPlusFlightRoomConfig	x
flights	HotelPlusFlightFlightResult	x
id	string	
HotelPlusFlightRoomConfig		
roomType	string	
mealType	string	
id	string	
HotelPlusFlightFlightResult		
airline	string	
fromAirport	string	
toAirport	string	
flight1FromTime	string	
flight1ToTime	string	
flight2FromTime	string	
flight2ToTime	string	
id	string	

Auf der Poolebene wurde für jedes BDM eine Pool Variable erstellt. Der Name der Variable entspricht dem BDM, beginnt jedoch mit einem kleinen Buchstaben.

Tabelle 5.2: Pool Variables

Name	BDM
hotelPlusFlightSearch	HotelPlusFlightSearch
hotelPlusFlightResults	HotelPlusFlightResults
hotelPlusFlightResultHotel	HotelPlusFlightResultHotel
hotelPlusFlightRoomConfig	HotelPlusFlightRoomConfig
hotelPlusFlightFlightResult	HotelPlusFlightFlightResult

5.3 Forms

Die Formulare wurden bereits für die Mockups im Abschnitt [4.1 Mockups](#) erstellt und konnten gleich wiederverwendet werden. Deshalb wird hier auf eine erneute Auflistung verzichtet.

5.4 Connectors

Nachfolgend werden die Definitionen und Implementationen der Connectors beschrieben. Zusätzlich werden für deren Ausführung weitere Abhängigkeiten benötigt, welche in Bonita definiert werden müssen.

5.4.1 Abhängigkeiten

Für die Ausführung der Connectoren können in Bonita weitere Libraries hinterlegt werden. Für dieses Projekt wurde zusätzlich zwei Abhängigkeiten definiert. Zum einen Gson¹, für die Bearbeitung von JSON, and Handy URI Templates², für die Verarbeitung von URI Templates gemäss RFC 6570³

5.4.2 Definitions

Es wurden vier Definitionen erstellt, für jeden Task einen.

Tabelle 5.3: Connector Definitions

Name	Input	Wizard pages	Outputs
H+FSearch	H+FSearch		H+FResults
H+FResultsHotelSelection	HotelPlusFlightResults	Results	H+FResultHotel
H+FHHotelConfiguration	H+FResultHotel	Hotel	H+FResults
H+FFlightSelection	H+FResultHotel	Hotel	H+FFlightResult

5.4.3 Implementations

5.5 Tasks

Nachfolgend werden die Tasks im Prozess beschrieben. Diese bestehen aus einem Bezeichner und einem Connector. Zusätzlich müssen noch der Contract, das Formular sowie die Operations (siehe Abschnitt [3.2.4 Forms, Contracts und Operations](#)) angegeben werden.

5.5.1 Search

Der Search Task muss ein Formular darstellen damit der Benutzer seine Suchkriterien eingeben kann. Dazu wird der Connector HotelPlusFlugSearch hinterlegt und das Formular hotelPlusFlightSearchForm unter Connectors out definiert.

Folgender Contract und Operations mussten spezifiziert werden:

-
- ¹ A Java serialization/deserialization library that can convert Java Objects into JSON and back. URL: <https://github.com/google/gson> (besucht am 12.06.2016).
 - ² A Java URI Template processor implementing RFC6570. URL: <https://github.com/damnhandy/Handy-URI-Templates> (besucht am 12.06.2016).
 - ³ RFC 6570 - URI Template. URL: <https://tools.ietf.org/html/rfc6570> (besucht am 21.06.2016).

Tabelle 5.4: Search Task Contract

Name	Type	Multiple
hotelPlusFlightSearchInput	complex	
destination	text	
departure	text	
adults	integer	
fromDate	date	
toDate	date	

Tabelle 5.5: Search Task Operations

Business Data Model	Operation	Form Variable
hotelPlusFlightSearch	setDestination	H+FSearchInput.destination
hotelPlusFlightSearch	setDeparture	H+FSearchInput.departure
hotelPlusFlightSearch	setAdults	H+FSearchInput.adults
hotelPlusFlightSearch	setFromDate	H+FSearchInput.fromDate
hotelPlusFlightSearch	setToDate	H+FSearchInput.toDate

5.5.2 Select Hotel

Der Select Hotel Task nimmt eine Liste von Hotels entgegen und speichert danach das selektierte ab. Dazu wird der Connector HotelPlusFlightResultsHotelSelection unter Connectors out und das Formular hotelPlusFlightHotelSelection hinterlegt.

Folgender Contract und Operations mussten spezifiziert werden:

Tabelle 5.6: Select Hotel Task Contract

Name	Type	Multiple
hotelPlusFlightResultsInput	complex	
hotels	complex	x
name	text	
price	text	
image	text	

Tabelle 5.7: Select Hotel Task Operations

Business Data Model	Operation	Form Variable
hotelPlusFlightSearch	setDestination	H+FSearchInput.destination
hotelPlusFlightSearch	setDeparture	H+FSearchInput.departure
hotelPlusFlightSearch	setAdults	H+FSearchInput.adults
hotelPlusFlightSearch	setFromDate	H+FSearchInput.fromDate
hotelPlusFlightSearch	setToDate	H+FSearchInput.toDate

5.5.3 Select Room Config

Beim Schritt Select Room Config wird das selektierte Hotel übergeben und der Benutzer kann eine Zimmer konfiguration auswählen. Dafür ist das From hotelPlusFlightSelec-

tRoomConfig und der Connector HotelPlusFlightHotelConfiguration (im Connector out) vorgesehen.

Folgender Contract und Operations mussten spezifiziert werden:

Tabelle 5.8: Select Room Config Task Contract

Name	Type	Multiple
hotelPlusFlightRoomConfigsInput	complex	
configs	complex	x
RoomType	text	
MealType	text	
Price	text	

Tabelle 5.9: Select Room Config Task Operations

Business Data Model	Operation	Form Variable
hotelPlusFlightRoomConfig	setConfig	H+FRoomConfigsInput.config

5.5.4 Select Flight

Beim letzten Task muss der User ein Flug auswählen. Dazu wird dem Task das gewählte Hotel übergeben. Definiert wurde das Formular hotelPlusFlightSelectFlight und der Connector out HotelPlusFlightFlightSelection.

Folgender Contract und Operations mussten spezifiziert werden:

Tabelle 5.10: Select Flight Task Contract

Name	Type	Multiple
hotelPlusFlightFlightResultsInput	complex	
flights	complex	x
Airline	text	
FromAirport	text	
ToAirport	text	
Flight1FromTime	text	
Flight1ToTime	text	
Flight2FromTime	text	
Flight2ToTime	text	

Tabelle 5.11: Select Flight Task Operations

Business Data Model	Operation	Form Variable
hotelPlusFlightFlightResult	setFlight	H+FFlightsInput.flight

5.6 Probleme

Es gab zwei Probleme, welche es verunmöglichten einen sauberen Prozess durchzuführen. Diese werden nachfolgend beschrieben.

5.6.1 Rückgabewert der Connectors

Der Connector HotelPlusFlight Search gibt ein HotelPlusFlightResults [BDM](#) zurück. Dieses wird beim Connector Output in eine Pool Variable geschrieben. Dabei tritt folgender Fehler auf:

```

1 java.lang.reflect.InvocationTargetException
2 com.thoughtworks.xstream.converters.ConversionException: com.travelwindow.model.
  HotelPlusFlightResults : com.travelwindow.model.HotelPlusFlightResults
3 ---- Debugging information ----
4 message : com.travelwindow.model.HotelPlusFlightResults
5 cause-exception : com.thoughtworks.xstream.mapper.CannotResolveClassException
6 cause-message : com.travelwindow.model.HotelPlusFlightResults
7 class : java.util.HashMap
8 required-type : java.util.HashMap
9 converter-type : com.thoughtworks.xstream.converters.collections.MapConverter
10 path : /map/entry/com.travelwindow.model.HotelPlusFlightResults
11 line number : 5
12 version : null
13
14 com.thoughtworks.xstream.mapper.CannotResolveClassException: com.travelwindow.
  model.HotelPlusFlightResults

```

Bonita unterstützt Groovy Scripts¹, mit welcher man Daten verarbeiten kann. Es wurde versucht damit das Problem zu beheben.

Folgender Groovy Code wurde verwendet um das Mapping für die HotelPlusFlightResults durchzuführen:

```

1 import groovy.json.JsonBuilder
2 import com.travelwindow.model.HotelPlusFlightResults
3 import com.travelwindow.model.HotelPlusFlightResultHotel
4
5 def mappedData = new HotelPlusFlightResults()
6 for (hotel in hotelPlusFlightResults.getHotels()) {
7     def tempHotel = new HotelPlusFlightResultHotel()
8     tempHotel.name = comment.name
9     tempHotel.image = comment.image
10    tempHotel.price = comment.price
11    mappedData.hotels.add(tempHotel)
12 }
13 return mappedData

```

Leider konnte der Fehler dadurch nicht behoben werden. Es wurden stattdessen Standartwerte auf den Pool Variablen definiert damit der Prozess durchgeführt werden kann.

¹ The Groovy programming language. URL: <http://www.groovy-lang.org/> (besucht am 25.06.2016).

5.6.2 BDMs befüllen

Das Aufsetzen der Formulare und die Konfiguration der Contract und Operations verlief Problemlos. Jedoch gab es Probleme beim Abfüllen der BDMs nach den Formularen. Dadurch gab es im Bonita Log folgende Meldung:

```

1 SEVERE: THREAD_ID=119 | HOSTNAME=Slang-ThinkPad | TENANT_ID=1 | The work [
  ExecuteConnectorOfActivity: flowNodeInstanceId = 240003, connectorDefinitionName
  = HotelPlusFlightResultsHotelSelection] failed. The failure will be handled.
2 2016-06-25 15:07:57.211 +0200 org.bonitasoft.engine.execution.work.
  FailureHandlingBonitaWork org.bonitasoft.engine.log.technical.
  TechnicalLoggerSLF4JImpl log
3 SEVERE: THREAD_ID=119 | HOSTNAME=Slang-ThinkPad | TENANT_ID=1 | org.bonitasoft.
  engine.persistence.SRetryableException : "Groovyx.persistence.
  PersistenceException: org.hibernate.PropertyValueException: not-null property
  references a null or transient value: com.travelwindow.model.
  HotelPlusFlightResultHotel._hotels_HOTELPLUSFLIGHTRESULTS_PIDBackref"
4 org.bonitasoft.engine.persistence.SRetryableException: javax.persistence.
  PersistenceException: org.hibernate.PropertyValueException: not-null property
  references a null or transient value: com.travelwindow.model.
  HotelPlusFlightResultHotel._hotels_HOTELPLUSFLIGHTRESULTS_PIDBackref

```

Der Fehler tritt nach dem Select Hotel Task auf, bevor der Connector HotelPlusFlightResultsHotelSelection aufgerufen wird. Ziel des Formulars ist es, aus mehreren Einträgen einer Auszuwählen. Dieser kann jedoch nicht mehr zurück ins Bonita überführt werden, weshalb es zu obigen Fehler kommt wenn der Connector aufgerufen wird.

Folgendes wurde versucht den Fehler zu beheben

Script zum Abspeichern der Formularwerte

Mit Groovy¹ kann man die Formularwerte bearbeitet, welche von den Formularen zurückgeliefert werden. Es wurde folgendes Script definiert, um den Wert des Formulars in ein BDM zu überführen damit Bonita diesen weiterverwenden kann.

```

1 def hotels = new java.util.ArrayList()
2 hotelPlusFlightResultsInput.hotels.each{
3   hotels.add({ currentHotelInput ->
4     def tempHotel = new com.travelwindow.model.HotelPlusFlightResultHotel()
5     tempHotel.name = currentHotelInput.name
6     tempHotel.price = currentHotelInput.price
7     tempHotel.image = currentHotelInput.image
8     return tempHotel
9   }(it))
10 }
11 def results = new com.travelwindow.model.HotelPlusFlightResults()
12 results.setHotels(hotels)
13 return results

```

¹ *The Groovy programming language.*

Der Fehler blieb jedoch der selbe.

Standartwerte definieren

Es wurde ein neues Pool Variable vom Typ `HotelPlusFlightResults`, welche mit fix vorgegebenen Werten befüllt wurde. Dies sollte testen, ob somit der Fehler nicht mehr angezeigt wird. Auch dies wurde mit einem Groovy Script bewerkstelligt.

```
1 def hotel1 = new com.travelwindow.model.HotelPlusFlightResultHotel()
2 hotel1.name = "Radisson Blu Hotel Berlin"
3 hotel1.image = "https://t-static.net/pictures/Hotel/Hotelbeds/9274/009274
  a_hb_ba_003.jpg?width=240&height=180&scale=both&mode=crop"
4 hotel1.price = "482 CHF"
5
6
7 def hotel2 = new com.travelwindow.model.HotelPlusFlightResultHotel()
8 hotel2.name = "NH Berlin Mitte"
9 hotel2.image = "https://t-static.net/pictures/Hotel/Hotelbeds/8819/008819
  a_hb_ba_008.jpg?width=240&height=180&scale=both&mode=crop"
10 hotel2.price = "403 CHF"
11
12 def hotel3 = new com.travelwindow.model.HotelPlusFlightResultHotel()
13 hotel3.name = "Wyndham Berlin Excelsior"
14 hotel3.image = "https://t-static.net/pictures/Hotel/Hotelbeds/5847/005847
  a_hb_l_010.jpg?width=240&height=180&scale=both&mode=crop"
15 hotel3.price = "350 CHF"
16
17 def hotels = new java.util.ArrayList()
18 hotels.add(hotel1)
19 hotels.add(hotel2)
20 hotels.add(hotel3)
21
22 def results = new com.travelwindow.model.HotelPlusFlightResults()
23 results.setHotels(hotels)
24 return results
```

Der Fehler blieb jedoch der gleiche.

Input Parameter des Connectors entfernen

Nun wurde versucht, der Input Parameter des `HotelPlusFlightResultsHotelSelection` auf der Connector Definition zu entfernen.

Danach musste der Out Connector auf dem Select Hotel Task noch angepasst werden, um der Änderung in der Definition zu entsprechen.

Jedoch blieb der Fehler der Selbe.

Connector entfernen

Als letzte Möglichkeit wurde der Connector aus dem Task Select Hotel entfernt. Damit verschwand auch der Fehler und der Prozess konnte durchgeführt werden.

Konsequenz

Da der Connector aus dem Select Hotel Task entfernt werden musste, wurden keine Daten mehr ins Bonita übernommen (siehe Abschnitt [5.6.2 Connector entfernen](#)). Deshalb wurde für alle Formulare Standartwerte Definiert, so dass der Prozess zumindest durch getestet werden kann.

Ein Beispiel für Standartwerte des Flugselections-Formular:

```
1 {
2   "hotelPlusFlightFlightResultsInput" : {
3     "flights" : [
4       { "Airline": "LH", "FromAirport": "ZRH", "ToAirport": "TXL", "
Flight1FromTime": "20.10.2016 20:15",
5         "Flight1ToTime": "20.10.2016 22:15", "Flight2FromTime": "22.10.201
6 12:25", "Flight2ToTime": "22.10.2016 14:35"},
7       { "Airline": "LX", "FromAirport": "ZRH", "ToAirport": "TXL", "
Flight1FromTime": "20.10.2016 16:00",
8         "Flight1ToTime": "20.10.2016 18:05", "Flight2FromTime": "22.10.201
6 20:45", "Flight2ToTime": "22.10.2016 22:40"},
9       { "Airline": "LX", "FromAirport": "ZRH", "ToAirport": "TXL", "
Flight1FromTime": "20.10.2016 06:30",
10        "Flight1ToTime": "20.10.2016 08:35", "Flight2FromTime": "22.10.201
11 6 16:00", "Flight2ToTime": "22.10.2016 18:00"}
12    ]
13  }
14 }
```


KAPITEL 6

Fazit

ANHANG A

Testübersicht

Quellenverzeichnis

- [1] *A Java serialization/deserialization library that can convert Java Objects into JSON and back.* URL: <https://github.com/google/gson> (besucht am 12.06.2016) (siehe S. 34).
- [2] *A Java URI Template processor implementing RFC6570.* URL: <https://github.com/damnhandy/Handy-URI-Templates> (besucht am 12.06.2016) (siehe S. 34).
- [3] *Activiti.* URL: <http://activiti.org/> (besucht am 12.06.2016) (siehe S. 8).
- [4] *Apache Tomcat® - Welcome!* URL: <https://tomcat.apache.org/> (besucht am 12.06.2016) (siehe S. 19).
- [5] *Bonitasoft.* URL: <http://www.bonitasoft.com/> (besucht am 12.06.2016) (siehe S. 8).
- [6] *BPMN Workflow Engine.* URL: <https://camunda.org/> (besucht am 12.06.2016) (siehe S. 8).
- [7] *Drools - Drools - Business Rules Management System (Java™, Open Source).* URL: <http://www.drools.org/> (besucht am 12.06.2016) (siehe S. 8).
- [8] *IBM - Software - IBM Business Process Manager.* URL: <http://www-03.ibm.com/software/products/en/business-process-manager-family> (besucht am 12.06.2016) (siehe S. 8).
- [9] *Imixs Workflow Project - Open Source Java Workflow Engine for BPMN 2.0.* URL: <http://www.imixs.org/> (besucht am 12.06.2016) (siehe S. 8).
- [10] *jBPM - Open Source Business Process Management - Process engine.* URL: <http://www.jbpm.org/> (besucht am 12.06.2016) (siehe S. 8).
- [11] *Microsoft BizTalk Integration | Microsoft.* URL: <https://www.microsoft.com/en-us/server-cloud/products/biztalk/> (besucht am 12.06.2016) (siehe S. 8).
- [12] *RFC 6570 - URI Template.* URL: <https://tools.ietf.org/html/rfc6570> (besucht am 21.06.2016) (siehe S. 34).
- [13] *The Groovy programming language.* URL: <http://www.groovy-lang.org/> (besucht am 25.06.2016) (siehe S. 37, 38).
- [14] *Yaoqiang BPMN Editor - an Open Source BPMN 2.0 Modeler.* URL: <http://bpmn.sourceforge.net/> (besucht am 12.06.2016) (siehe S. 8).