ulm university universität
uulm

Universität Ulm | 89069 Ulm | Germany

Fakultät für Ingenieurwissenschaften und Informatik
Institut für Datenbanken und Informationssysteme

Entwicklung eines Visualisierungskonzeptes für Time Patterns in Prozessen

Bachelorarbeit an der Universität Ulm

Vorgelegt von:
Zenib Awan
zenib.awan@uni-ulm.de

Gutachter:
Prof. Dr. Manfred Reichert

Betreuer:
Andreas Lanz

2012

Fassung 28. November 2012

ABSTRACT Aufgrund stetigen Konkurrenzdrucks, sind Unternehmen heutzutage dazu gezwungen, ihre internen Abläufe zu optimieren. In diesem Kontext kommt die Business Process Model and Notation ins Spiel: Diese Modellierungssprache ermöglicht es, Prozesse abstrakt zu erfassen. Daneben spielen zeitliche Aspekte, wie Termine oder Deadlines in Prozessen eine große Rolle, sind jedoch in BPMN 2.0 nicht sinnvoll modellierbar. Um diese Lücke zu schließen, wurden die sogenannten Time Patterns vorgeschlagen. Diese Arbeit beschäftigt sich mit der Visualisierung der ersten acht dieser Time Patterns. Dabei wird zunächst die Funktionsweise und eine Idee zur Visualisierung der Time Patterns und danach die tatsächliche Umsetzung vorgestellt. Die vorgestellte Visualisierungen sollen dabei vor allem eine Hilfestellung zum Verständnis der Time Patterns bieten.

Inhaltsverzeichnis

1 Einleitung **1**

2 Grundlagen **3**

2.1 Business Process Model and Notation 4

2.1.1 Prozess 4

2.1.2 Aktivität 5

2.1.3 Sequenzfluss 5

2.1.4 Token 5

2.1.5 Gateways 6

2.1.6 Ereignisse 7

2.1.7 Ausnahmen 8

2.2 Time Patterns 9

3 Visualisierung der Time Patterns **10**

3.1 Anforderungen 11

3.2 Styleguide 11

3.2.1 Bedienkonzept 12

3.2.2 Visuelles Design 14

3.3 Grundlagen der Visualisierung 18

3.3.1 Kanten 18

3.3.2 Aktivitäten 18

3.3.3 Fortschrittsbalken 19

3.3.4 Token 19

3.3.5 Uhr 20

3.3.6 Kalender . . . 21
3.3.7 Steuerung der Aufmerksamkeit des Betrachters . . . 22
3.4 Time Patterns . . . 22
3.4.1 TP1: Time Lags between two Activities . . . 22
3.4.2 TP2: Durations . . . 26
3.4.3 TP3: Time Lags between Arbitrary Events . . . 29
3.4.4 TP4: Fixed Date Elements . . . 30
3.4.5 TP5: Schedule Restricted Elements . . . 32
3.4.6 TP6: Time-based Restrictions . . . 34
3.4.7 TP7: Validity Period . . . 35
3.4.8 TP8: Time-dependent Variability . . . 37

4 Software **39**
4.1 Flash und HTML5 im Vergleich . . . 40
4.2 Anforderungen an die Software . . . 41
4.3 Adobe Edge . . . 42
4.4 Sencha Animator . . . 43
4.5 Tumult Hype . . . 44
4.6 Softwarevergleich . . . 45

5 Realisierung **47**
5.1 Fortschrittsbalken . . . 48
5.2 Uhr . . . 48
5.3 Interaktive Animationen . . . 50

6 Zusammenfassung **53**

Abbildungsverzeichnis **55**

Listings **57**

Tabellenverzeichnis **58**

Literaturverzeichnis **59**

1

Einleitung

Unternehmen sind heutzutage großem Konkurrenzdruck ausgesetzt. Um die internen Abläufe zu optimieren, wird daher häufig versucht, diese zu erfassen und präzise als Prozesse zu modellieren. Prozessmodellierung dient der Abstraktion komplizierter Sachverhalte und deren Optimierung. Zur Prozessmodellierung stehen unterschiedliche Notationen zur Verfügung. Dabei hat sich die Business Process Model and Notation als Standard der graphischen Notationen etabliert [3]. Häufig spielen in Prozessen auch zeitliche Aspekte eine wichtige Rolle: Es sind Fristen oder Deadlines vorgegeben, die eingehalten werden müssen. Die sogenannten Time Patterns beschreiben, welche Elemente notwendig sind, um solche zeitlichen Aspekte zu realisieren. Die Time Patterns sind jedoch nicht immer einfach zu verstehen. Diese Arbeit präsentiert daher eine Visualisierung der Time Patterns in Form von Animationen. Diese soll dem Nutzer eine Hilfestellung bieten, um die teils komplexen Time Patterns besser zu verstehen. Um dieses Ziel zu erreichen werden die Time Patterns dabei abstrakt dargestellt und auf das Wesentliche reduziert.

Aufgabenstellung und Zielsetzung

Die Aufgabenstellung besteht im Wesentlichen aus der Visualisierung einer Auswahl der Time Patterns. Die Visualisierung soll in Form von Animationen realisiert werden und sowohl für Anfänger als auch für Experten geeignet sein. Durch diese (interaktiven) Animationen soll ein besseres Verständnis für die Time Patterns vermittelt werden. Außerdem sollten die Animationen zu den gängigen Browsern kompatibel sein.

Aufbau der Arbeit

In Kapitel 2 wird zunächst die Business Process Model and Notation eingeführt und anschließend die Time Patterns vorgestellt. Der Hauptteil dieser Arbeit bildet Kapitel 3: In diesem Kapitel werden die Grundsätze, wie die Anforderung und der Styleguide der Visualisierung, festgelegt. Darüber hinaus wird im Abschnitt „Time Patterns“ die Funktionsweise und die Realisierungsidee zur Visualisierung der Time Patterns erklärt. Kapitel 4 dieser Arbeit beschäftigt sich mit der Software, die zur Visualisierung der Time Patterns in Frage kommt. Dabei werden die Vor- und Nachteile der verschiedenen Software erörtert. Außerdem wird die Fragestellung „HTML5 vs. Flash“ geklärt. Abschließend beschäftigt sich Kapitel 5 mit der Realisierung der zentralen Elemente der Animationen. Außerdem werden die interaktive Animationen anhand von Snippets näher erklärt.

2

Grundlagen

Das nachfolgende Kapitel befasst sich mit Business Process Model and Notation und soll dem besseren Verständnis der grundlegenden Thematik dienen. Begriffe wie Prozess, Aktivität, Token, etc. werden erklärt und in den Kontext eingeordnet. Anschließend wird ein kurzer Überblick über die Time Patterns vermittelt.

2.1 Business Process Model and Notation

Mithilfe von **B**usiness **P**rocess **M**odel and **N**otation (BPMN) werden Prozesse beschrieben, dokumentiert und analysiert. Außerdem wird mit dieser Notation eine einheitliche Sprache festgelegt: BPMN legt fest, „mit welchen Symbolen die verschiedenen Elemente von Prozessen dargestellt werden, was sie genau bedeuten und wie sie miteinander kombiniert werden können“ [3, S. 8f]. Dadurch wird ein Standard für die Prozessmodellierung festgelegt. Auf dieser Grundlage gibt es unterschiedliche Muster zur Prozessmodellierung, zum Beispiel die Workflow Patterns [12], die zur Realisierung von Kontrollflussabhängigkeiten eingesetzt werden. Abbildung 2.1 zeigt einen Prozess, der mithilfe von BPMN modelliert wurde.
BPMN wurde ursprünglich im Jahr 2004 von der Business Process Management Initiative (BPMI) entwickelt. Nachdem die Organisation „Object Management Group“ (OMG) BPMI übernommen hatte, wurde im Jahr 2006 die erste offizielle Version von BPMN veröffentlicht. Nach mehreren Zwischenversionen mit einigen Neuerungen erschien im Januar 2011 die Version 2.0. Diese Version wurde um verschiedene Konstrukte und neue Modelltypen erweitert (vgl. [3, S. 8ff]).

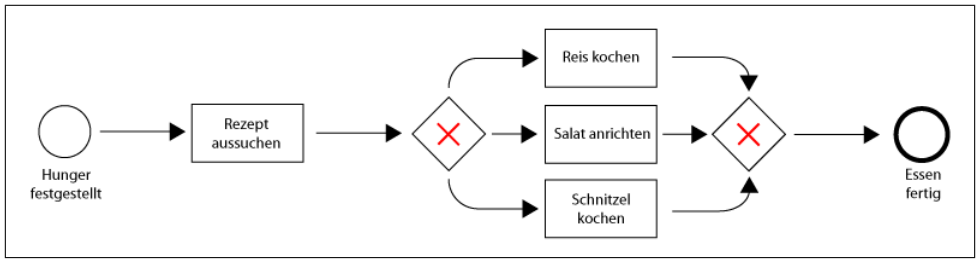

Abbildung 2.1: Prozess mit wichtigen Elementen

2.1.1 Prozess

Ein Prozess (siehe Abbildung 2.1) stellt die Gesamtheit aller Aktivitäten, Kanten und Gateways dar. Ein Prozess beginnt mit einem Startereignis. Danach durchläuft es eine Folge von Aktivitäten und Gateways und endet schließlich in einem Endereignis. Ein Prozess kann aus mehreren Teilprozessen bestehen.

2.1.2 Aktivität

Die grundlegenden Elemente von Prozessen sind die Aktivitäten. Dabei unterscheidet man zwei Arten: Zum einen gibt es einfache Arbeitseinheiten („Task“) und zum anderen Teilprozesse („Sub-Process“). Teilprozesse werden durch ein „+“-Symbol innerhalb der Aktivität (siehe Abbildung 2.2) gekennzeichnet. Sie verweisen auf einen anderen Prozess, der innerhalb der Aktivität des Prozesses ausgeführt wird. Dieser Teilprozess wird wie jeder andere Prozess durchlaufen. „Für den Sequenzfluss ist nur wichtig, dass der Teilprozess komplett abgeschlossen ist, bevor eine Marke ausgegeben wird“ [3, S.86]. Nachdem ein Teilprozess beendet ist, werden die von ihm erzeugten Daten und die Kontrolle an den übergeordneten Prozess weitergegeben. Arbeitseinheiten können in verschiedene Typen (zum Beispiel Service-, Sende-, Empfangs-Task) unterteilt werden, welche durch ein entsprechendes Symbol in der Aktivität gekennzeichnet werden (vgl. [3, S. 86ff]).

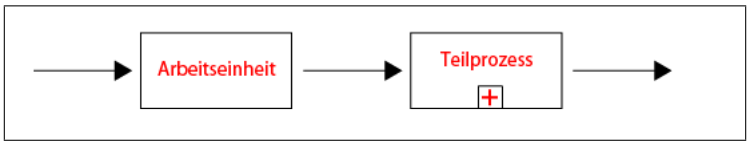

Abbildung 2.2: Arbeitseinheit und Teilprozess

2.1.3 Sequenzfluss

Der Sequenzfluss bezeichnet die Abarbeitungsreihenfolge eines Prozessabschnittes nach den Regeln der BPMN. Während der Abarbeitung werden Aktivitäten entlang des Sequenzflusses ausgeführt. Der Sequenzfluss kann durch Gateways beeinflusst werden [3, S. 19].

2.1.4 Token

Token (oder auch Marken) sind ein Konzept zur besseren Übersicht über den Prozessablauf. Jedes Mal wenn ein Prozess gestartet wird, wird durch ein Startereignis ein Token erzeugt. Das Token wandert dann entlang des Sequenzflusses durch den

Prozess. Sobald das Token auf eine Aktivität stößt, wird es von der Aktivität über deren eingehenden Pfeil aufgenommen und die Aktivität ausgeführt. Nachdem die Aufgabe erledigt ist, erscheint das Token auf dem ausgehenden Pfeil. So wandert das Token von Aktivität zu Aktivität, bis es das Endereignis erreicht hat und der Prozess somit beendet ist. Es können auch mehrere Token, zum Beispiel durch eine parallele Verzweigung im Prozess, erzeugt werden. Diese Token sind völlig unabhängig und beeinflussen sich in keiner Weise. Eine Vereinigung wartet dann auf die Token aller eingehenden Pfade und führt diese zu einem einzelnen Token zusammen. Token werden im Prozess durch schwarze Kreise dargestellt (vgl. [3, S. 19]).

2.1.5 Gateways

Gateways stellen in den Prozessen Verzweigungen und Zusammenführungen im Ablauf dar. Dargestellt werden Gateways als Raute mit einem Symbol, welches die jeweilige Art des Gateways repräsentiert. Die wichtigsten Arten von Gateways sind [3, S. 25ff]:

- Exklusives Gateway
  Mit einem exklusiven Gateway können alternative Pfade realisiert werden. Wie Abbildung 2.3 zeigt, wird dieser Gateway mit einem „X“ in der Raute dargestellt. Dieser Gateway wird auch als „XOR“-Gateway bezeichnet. Mithilfe dieses Gateway wird genau ein Sequenzfluss aus mehreren ausgewählt. Welcher Sequenzfluss ausgewählt wird, hängt von den jeweiligen Bedingungen ab, die auf den Kanten notiert sind.

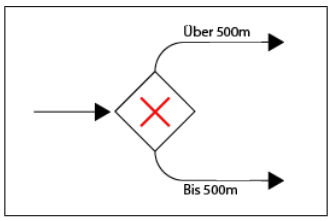

Abbildung 2.3: XOR-Gateway

- Paralleles Gateway
  Bei einem parallelen Gateway, welcher durch ein „+“-Symbol in der Raute

gekennzeichnet wird (Abbildung 2.4), werden alle ausgehenden Sequenzflüsse parallel ausgeführt. „Dies entspricht einem logischen UND“ [3, S. 28]. Dauert einer dieser Sequenzflüsse länger, so müssen bei der Zusammenführung die anderen auf ihn warten.

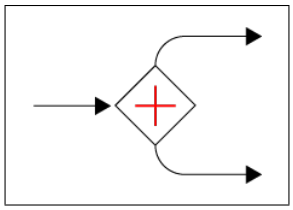

Abbildung 2.4: UND-Gateway

- Inklusives Gateway
  Bei diesem Gateway geht es darum, dass mindestens ein Sequenzfluss ausgeführt werden muss. Das inklusive Gateway (durch ein „O“ gekennzeichnet) stellt das logische ODER dar (siehe Abbildung 2.5). Wie beim parallelen Gateway wird auch hier bei der Zusammenführung auf alle ausgeführten Sequenzflüsse gewartet.

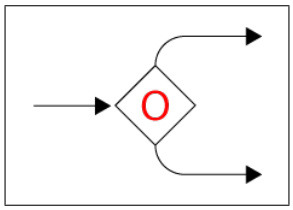

Abbildung 2.5: ODER-Gateway

2.1.6 Ereignisse

Ein Ereignis beschreibt zu einem Zeitpunkt, „dass etwas passiert ist“ [3, S. 63f], und hat selbst keine Dauer. Ereignisse können verschiedene Auslöser haben, zum Beispiel:

- Eintreffende Nachricht
- Bestimmter Zeitpunkt

- Bedingung, die wahr wird
- Eintretende Ausnahme

Eine spezielle Form der Ereignisse bilden die Start- bzw. Endereignisse. Ein Startereignis, welches den Anfang eines Prozess kennzeichnet, kann etwas empfangen, wohingegen ein Endereignis etwas versenden kann. Oft ist der Auslöser eines Startereignisses nicht bekannt. Neben diesen Ereignissen gibt es noch Zwischenereignisse: Diese können sowohl etwas empfangen als auch etwas versenden (vgl. [13, S. 63ff]).

2.1.7 Ausnahmen

Bei der Durchführung einer Aktivität „kann ein Ereignis auftreten, das zum vorzeitigen Abbruch der Aktivität führen kann“ [3, S.107f]. Dies kann auch zum vorzeitigen Abbruch des gesamten Prozesses führen. Bei diesen Ausnahmen kann es sich beispielsweise um den Ablauf einer vorgegebenen maximalen Zeitspanne oder um einen Fehler innerhalb der Aktivität handeln. Ausnahmen sind eine spezielle Form von Ereignissen (vgl. [3, S. 107ff]). Für die Behandlung von Ausnahmen kann ein Ausnahmefluss modelliert werden (siehe „Auftraggeber informieren“ in Abbildung 2.6): Dauert der Aufbau des Gerüsts zu lange, so wird der Auftraggeber informiert. Ansonsten kann mit der Arbeit begonnen werden. Die Ausnahme wird durch den Timer an der Aktivität „Gerüst aufbauen“ dargestellt.

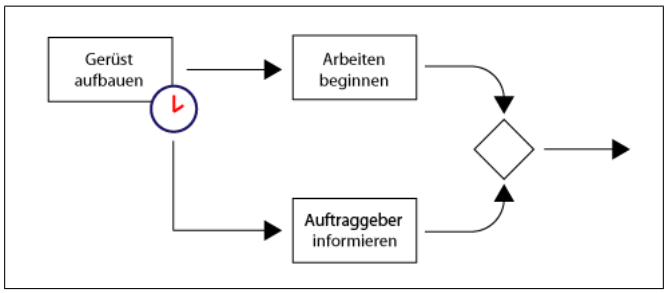

Abbildung 2.6: Spezielle Form der Ereignisse: Ausnahme

2.2 Time Patterns

Die Time Patterns [7] stellen eine Vielzahl von Prozessmustern zur Verfügung, um zeitliche Aspekte in Prozessen abzubilden. Tabelle 2.1 zeigt eine Übersicht der Time Patterns: Kategorie 1 beschäftigt sich mit Zeitspannen zwischen zwei Aktivitäten und den Dauern von Aktivität. In Kategorie 2 geht es um die Einschränkung der Zeitpunkte, zu denen eine Aktivität ausgeführt werden darf. Die dritte Kategorie stellt die Möglichkeit dar, dass die Gültigkeit von Aktivitäten zeitlich beschränkt ist. Kategorie 4 bietet sowohl Muster zur Beschreibung von Einschränkungen bezüglich zyklischer Aktivitäten als auch Periodizität. Im Rahmen dieser Bachelorarbeit wird eine Visualisierung der ersten acht der zehn Time Patterns vorgestellt.

Category 1: Durations and Time Lags
TP1: Time Lags between two Activities
TP2: Durations
TP3: Time Lags between Arbitrary Events
Category 2: Restricting Executions Times
TP4: Fixed Date Elements
TP5: Schedule Restricted Elements
TP6: Time-based Restrictions
TP7: Validity Period
Category 3: Variability
TP8: Time-dependent Variability
Category 4: Recurrent Process Elements
TP9: Cyclic Elements
TP10: Periodicity

Tabelle 2.1: Übersicht der Time Patterns

3

Visualisierung der Time Patterns

In diesem Kapitel werden die Time Patterns in ihrer Funktionsweise und die grundlegenden Regeln der Visualisierung dargestellt. Zuerst werden die Anforderungen an die Visualisierung beschrieben, danach der Styleguide, um ein einheitliches Design zu gewährleisten. Zuletzt wird die Visualisierung der einzelnen Time Patterns vorgestellt.

3.1 Anforderungen

Durch die Visualisierungen sollen dem Nutzer der Ablauf und die Funktionsweise der Time Patterns bildlich und leicht verständlich vermittelt werden. Hierbei werden die Aktivitäten, das Token und der Sequenzfluss abstrakt dargestellt (siehe Abbildung 3.2). So kann der Nutzer sich auf die Funktionsweise der Time Patterns konzentrieren. Die Visualisierung soll in Form von Animationen umgesetzt und teilweise auch interaktiv gestaltet werden. Sind die Animationen interaktiv, dann hat der Nutzer selbst die Möglichkeit, den Ablauf zu steuern. Der Lerneffekt wird dadurch gesteigert, da die interaktive Realisierung der Time Patterns auf die Eingaben des Nutzers reagiert.
Um den Nutzer visuell nicht zu überfordern, sollten möglichst wenige Sequenzen parallel zu einander ablaufen. Die Aufmerksamkeit wird somit auf ein Objekt gelegt. Dadurch verliert der Nutzer den Überblick nicht. Um Klarheit zu gewährleisten, sollte ein einheitliches Design, welches sich in Farbe, Form, Größe und Schrift widerspiegelt, gewählt werden. Die Time Patterns stehen im Mittelpunkt und die Animationen sollten sich daher stets auf die notwendigen Bestandteile konzentrieren. Die Animationen sollten in einer angemessenen Geschwindigkeit ablaufen. Dabei muss darauf geachtet werden, dass die Animationen nicht zu schnell und nicht zu langsam ablaufen. Neben Form und Größe sollten wichtige Ereignisse oder Objekte farblich gekennzeichnet sein. Das Ziel ist dabei, die Aufmerksamkeit des Nutzers auf das Wesentliche zu lenken.

3.2 Styleguide

Im Nachfolgenden werden Regeln festgelegt, um ein einheitliches Design der Visualisierung zu gewährleisten. So werden Verwechslungen ausgeschlossen und es wird ein roter Faden innerhalb der Time Patterns erzeugt. Alle Time Patterns sind nach den Richtlinien, die im Styleguide festgelegt worden sind, zu visualisieren.

3.2.1 Bedienkonzept

Dieser Abschnitt befasst sich mit der Interaktion, den Kontrollelementen und dem Beschreibungs-Button, die bei der Bedienung eine Rolle spielen. Durch diese Schnittstellen kann der Nutzer das Timing und das Abspielen der Animationen selbst kontrollieren.

Kontrollelemente

Die Kontrollelemente, die im Rahmen dieser Arbeit zum Einsatz kommen, sind in der Technik bereits weitläufig vertreten. Zur Steuerung der Animation werden dem Nutzer folgende Kontrollelemente zur Verfügung gestellt:

- Play-Button Startet die Animation
- Pause-Button Hält die Animation am aktuellen Zeitpunkt an
- Stop-Button Stoppt die Animation und setzt sie auf den Anfang zurück

Um die Animation zu starten, muss der Nutzer den in der Abbildung 3.1 dargestellten Play-Button drücken.

Möchte der Nutzer während der Animation eine Pause machen, dann drückt er auf den Pause-Button (Abbildung 3.1). Um sich die Animation anschließend weiter anzusehen, drückt er erneut auf den Play-Button.

Klickt der Nutzer während der Animation auf den Stop-Button, welcher in Abbildung 3.1 zu sehen ist, so wird die Animation angehalten und auf den Anfang der Animation zurück gesetzt. Mit dem Play-Button kann er die Animation erneut starten.

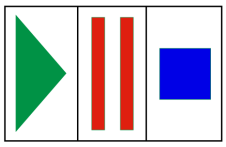

Abbildung 3.1: Kontrollelemente

Interaktion

Durch die Möglichkeit, den Ablauf selbst zu steuern, soll der Nutzer die Funktionsweise des Time Patterns besser nachvollziehen können. Hierfür bietet ein Interaktionselement (Play-Button) die Möglichkeit, die Ausführung der aktuell anstehenden Aktivität zu starten. Als Hilfestellung für den Nutzer dienen dabei kurze Anweisungen, die ihm seine aktuellen Interaktionmöglichkeiten beschreiben. Der Play-Button befindet sich stets in den Aktivitäten, die gerade ausgeführt werden können, wie Abbildung 3.2 zeigt. Um

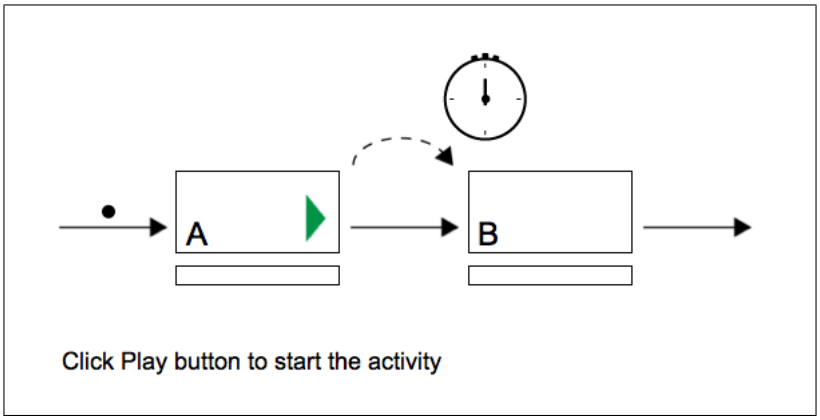

Abbildung 3.2: Interaktionsmöglichkeit durch Play-Button

die Aufmerksamkeit des Nutzers auf den Button und dessen Interaktionsmöglichkeit zu lenken und den Zusammenhang zu verdeutlichen, erscheinen Play-Button und Hilfestellung gleichzeitig.

Wenn der Nutzer auf den Play-Button klickt, dann verschwindet die Hilfestellung und der Play-Button, um ihm zu zeigen, dass die Animation reagiert hat. Ist die Aktivität abgeschlossen, so erscheint in Form einer neuen Hilfestellung und Play-Button die nächste Interaktionsmöglichkeit, wie in Abbildung 3.3 illustriert.

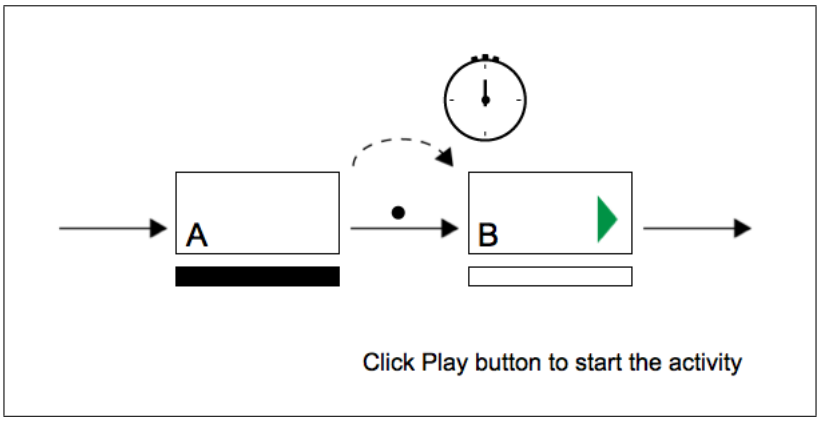

Abbildung 3.3: Neue Interaktionsmöglichkeit

3.2.2 Visuelles Design

In diesem Abschnitt werden die visuellen Aspekte, wie Form und Größe, Farbe, Geschwindigkeit der Animation und Schrift diskutiert. Die hier festgelegten Richtlinien sollen den Nutzer durch ein einheitliches Design bei der visuellen Wahrnehmung der verschiedenen Time Patterns unterstützen.

Form und Größe

Die Objekte (Token, Aktivität und Kante) sollen bei allen Time Patterns eine einheitliche Form und Größe haben. Dies führt zu einem einheitlichen Design bei den Patterns. Dadurch wird gewährleistet, dass alle vorkommenden Objekte einen hohen Wiedererkennungswert haben. Für Aktivitäten werden Rechtecke mit einer Breite von 120 Pixel und einer Höhe von 60 Pixel verwendet. Kanten werden als Pfeile mit einer Länge von 80 Pixel dargestellt. Dies entspricht zwei Dritteln der Breite einer Aktivität. Liegen zwei durch eine Kante verbundene Aktivitäten nicht auf einer horizontalen Linie, so wird der zugehörige Pfeil um eine Ecke geführt. Das Token wird durch einen schwarz gefüllten Kreis mit einem Durchmesser von zehn Pixel umgesetzt.

Farbe

Es gibt kaum ein anderes Gestaltungselement, das so ausdrucksstark und emotional wie Farbe ist. So hat jede Farbe ihre eigene Bedeutung und ruft bei Menschen bestimmte Emotionen hervor. Die Bedeutungen der Farben kann von Kultur zu Kultur variieren [13, S. 196f].
Im Rahmen dieser Bachelorarbeit werden die Farben nach ihren Assoziationen eingesetzt. Sie sollen dem Nutzer bei den Animationen helfen, die Aufmerksamkeit auf das Wesentliche zu fokussieren. Der Nutzer braucht keine Vorkenntnisse über die Assoziationen der Farben, da er sie aus dem Alltag bereits kennt (beispielsweise die Farben einer Ampel: Rot = Stehen, Grün = Gehen).

Rot ist eine besonders ausdrucksstarke und auffällige Farbe. Keine andere Farbe bleibt so stark in der Erinnerung haften wie Rot. Die Farbe Rot hat viele (wenn auch widersprüchliche) Assoziationen: Zum einen steht Rot für Liebe und Leidenschaft, zum anderen für Hass, Gefahr und Warnung [13, S. 196ff]. Das in Abbildung 3.4 dargestellte Rot (rgb(240,15,15)) wird in den Animationen vielfältig genutzt: Um dem Nutzer bildlich zu zeigen, dass die Zeit für eine Aktivität knapp wird, ist das letzte Viertel der Uhr rot gefärbt. So wird dem Nutzer vermittelt, dass es Konsequenzen haben könnte, wenn die Aktivität nicht zeitnah abgeschlossen wird. Des Weiteren sind Deadlines oder andere wichtige Kommentare (wie beispielsweise bei Knappheit einer Ressource), ebenfalls in Rot dargestellt. Hier ist es wichtig, dass sie vom Nutzer schnell wahrgenommen werden, da zum Beispiel eine Deadline ein zeitliches Hindernis darstellt.

Blau hat eine sehr harmonische und weiche Wirkung. Mit dieser Farbe wird Kälte, Ruhe, Unendlichkeit, Himmel, Wasser und Harmonie verbunden [13, S. 199ff]. Bei den Animationen wird Blau für den Stop-Button verwendet. Speziell bei dem Pattern „Durations“ spielt dieser Button eine große Rolle: Der Nutzer darf selbst entscheiden, wann die Aktivität beendet wird. Die Aktivität wird förmlich eingefroren. Deshalb wurde dafür die Farbe Blau (rgb(0,0,230)) gewählt, wie in Abbildung 3.4 zu erkennen ist.

Grün „löst viele positive Assoziationen aus“ und steht für Wachstum, Natur, Gesundheit, Hoffnung und Leben [13, S. 202ff]. Die Farbe Grün (rgb(0,146,69)), wie sie in Abbildung 3.4 dargestellt wird, signalisiert bei der Ampel, dass die Straße überquert werden kann. Ebenso soll die Farbe Grün in den Animationen dafür stehen, dass die jeweilige Aktivität gestartet werden kann.

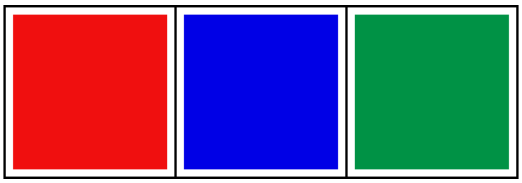

Abbildung 3.4: Verwendete Farben

Geschwindigkeit der Animation

Der Nutzer kann bei der interaktiven Visualisierung die Geschwindigkeit selbst beeinflussen. So können Anfänger und Fortgeschrittene die Time Patterns, an ihre Vorkenntnisse angepasst, steuern: Anfänger werden mehr Zeit brauchen, die Time Patterns zu verstehen, wohingegen Fortgeschrittene sich schneller damit zurechtfinden. Bei den Time Patterns ohne Interaktionsmöglichkeit wurde die Geschwindigkeit so gewählt, dass sie sowohl für Anfänger als auch für Fortgeschrittene angemessen ist. Die Geschwindigkeit darf nicht zu schnell sein, da die Anfänger nicht mehr mitkommen würden. Ist sie jedoch zu langsam, dann wird der Fortgeschrittene schnell gelangweilt sein. Wenn Elemente ein- bzw. ausgeblasst werden, so ist für diesen Effekt eine halbe Sekunde vorgesehen. Zwischen dem Ein- bzw. Ausblassen liegen mindestens eine Sekunde. Sobald das Token in einer Aktivität erscheint, fängt der Fortschrittsbalken an zu laufen. Das bietet dem Anfänger genug Zeit, dem Token zu folgen, und ist dennoch schnell genug, um dem Fortgeschrittenen eine angenehme Ablaufgeschwindigkeit zu bieten. Die Dauer des Fortschrittsbalken liegt bei mindestens 2,5 Sekunden. Eine Ausnahme bildet Time Pattern 8: Die Dauer von der Aktivität „Check Best-Before Date“ liegt bei fünf Sekunden. Hier ist es wichtig, dass der Nutzer realisiert, dass das Mindesthaltbarkeitsdatum abgelaufen ist.

Schrift

Für die Schrift wurde einheitlich die serifenlose Schriftart „Arial, Helvetica, Sans-Serif“ gewählt. Serifenlose Schriftarten sind auf dem Bildschirm besser zu lesen, wohingegen Schriftarten mit Serifen im Druckbereich besser geeignet sind: Serifenlose Schriftarten wirken harmonischer, weicher, freundlicher und lassen sich besser auf Bildschirmen darstellen, wohingegen Schriftarten mit Serifen „produktionstechnisch gutmütig“ sind und „sich durch die damit verbundene ausgeprägte Zeilenbildung gut lesen“ lassen [13, S.289ff]. Es wurden folgende Schriftgrößen verwendet:

- Titel 19 Point
- Interaktionsanweisung 18 Point
- Aktivitäten mindestens 15 Point
- Kommentare 13 Point, 16 Point

Die jeweiligen Schriftgrößen sollen die Prioritäten der Labels widerspiegeln: Um eine gute Übersicht zu erzielen, wurde für den Titel die Schriftgröße 19 Point gewählt. So weiß der Nutzer sofort, bei welchem Pattern er sich befindet. Neben dem Titel sind auch die Interaktionsanweisungen bei den interaktiven Patterns wichtig und haben deshalb die Schriftgröße 18 Point. Wichtig ist hier, dass die Anweisungen beim Auftauchen gleich wahrgenommen werden, damit der Nutzer schnell darauf reagieren kann. Die Aktivitäten sind meist nur mit einem Buchstaben oder wenigen Wörtern gekennzeichnet. Hier wurde je nach Textlänge die Schriftgröße an das Kästchen der Aktivität angepasst. Jedoch sollte die Schriftgröße der Labels 15 Point nicht unterschreiten, damit sie gut lesbar sind. Die Kommentare beschreiben zusätzlich Aspekte, die bei den Time Patterns eine Rolle spielen: Sie verwenden andere Effekte (aufblinken, rot einfärben oder ein- bzw. ausblassen), um auf sich aufmerksam zu machen. So werden sie auch bei einer kleinen Größe gut wahrgenommen.

3.3 Grundlagen der Visualisierung

Im Folgenden werden die für die Animationen wichtigsten Elemente dargestellt und ihre Funktionen erläutert. Außerdem werden Richtlinien diskutiert, mit deren Hilfe die Aufmerksamkeit des Benutzers stets auf das Wesentliche gelenkt werden soll

3.3.1 Kanten

Kanten visualisieren den Übergang von einer Aktivität zu einer anderen. Sie werden mithilfe eines einfachen Pfeils dargestellt. Da die Übergänge gerichtet sind, bietet sich der Pfeil (siehe Abbildung 3.5) besonders gut dafür an. Die geraden Kanten sind nicht direkt an Aktivitäten angedockt, sondern haben einen Abstand von mindestens zehn Pixel zu den Aktivitäten. Das rote Rechteck steht stellvertretend für das Token: Es taucht immer mittig oberhalb auf der Kante auf.

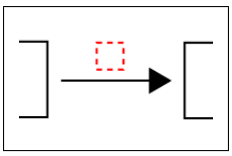

Abbildung 3.5: Kante

3.3.2 Aktivitäten

Aktivitäten werden durch ein Kästchen dargestellt. Jede Aktivität hat ein Textfeld, welches den Namen der Aktivität beinhaltet. Die Position des Labels ist so gewählt, dass es in der linken unteren Ecke platziert wird. Unter jeder Aktivität befindet sich ein Fortschrittsbalken, der den Fortschritt der Ausführung der Aktivität darstellt. Das rote Rechteck in Abbildung 3.6 zeigt die Position des Tokens an, welches in der Aktivität erscheint, wenn diese zur Ausführung bereit ist. Das grüne Rechteck in der rechten unteren Ecke wiederum steht für die Position der Steuerelemente bei interaktiven Animationen.

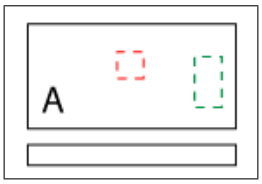

Abbildung 3.6: Aktivität mit Beschriftung und Fortschrittsbalken

3.3.3 Fortschrittsbalken

Der Fortschrittsbalken soll den Fortschritt einer Aktivität darstellen. Der Balken (Abbildung 3.7) wird, wie die Aktivität auch, durch ein Kästchen symbolisiert. Dieses ist genau so breit wie die Aktivität und hat eine Höhe von 15 Pixel. Zunächst ist der Fortschrittsbalken leer.

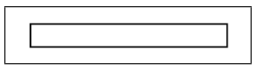

Abbildung 3.7: Leerer Fortschrittsbalken

Wird die Aktivität gestartet, so fängt der Fortschrittsbalken an, sich zu füllen (vgl. Abbildung 3.8). Ist die Aktivität abgeschlossen, so erkennt man dies an ihrem vollen Fortschrittsbalken.

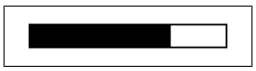

Abbildung 3.8: Sich füllender Fortschrittsbalken

3.3.4 Token

Das Token, welches durch einen gefüllten Kreis dargestellt wird (Abbildung 3.9), zeigt die aktuelle Position der Ausführung des Prozesses an. Erscheint das Token in einer Aktivität, so startet es die Aktivität. Nachdem eine Aktivität beendet wurde und das Token auf der ausgehenden Kante erschienen ist, kann die nachfolgende Aktivität gestartet werden.

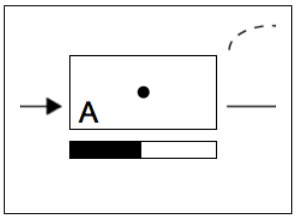

Abbildung 3.9: Token in einer Aktivität

3.3.5 Uhr

Die Uhr spielt in den Animationen eine große Rolle: Sie dient zur Visualisierung der Zeit. Eine Uhr kann beispielsweise einen Timeout oder eine Deadline darstellen. Bei einem Timeout wird eine Aktion (zum Beispiel Fehlermeldung) nach Zeitüberschreitung ausgelöst. Beim Ablauf eines Timeouts ist der Prozess anschließend nicht mehr ausführbar. Die Uhr besteht aus einem Kreis, vier Strichen, für drei, sechs, neun und zwölf Uhr, und drei Knöpfe, die an eine Stoppuhr erinnern sollen (Abbildung 3.10). Der Zeitablauf wird durch das Füllen der Uhr symbolisiert.

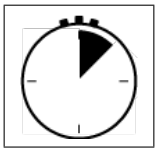

Abbildung 3.10: Sich füllende Uhr

Wird die Zeit knapp, das heißt sind drei Viertel der Zeit abgelaufen, wechselt die Farbe der Uhr von Schwarz auf Rot und die Uhr beginnt zu blinken. Die Aufmerksamkeit des Nutzers wird sofort auf die Uhr gelenkt und ihm wird verdeutlicht, dass die Zeit kritisch ist.

Im Falle eines Timeouts wird der Nutzer durch eine entsprechende Meldung darüber informiert, wie in Abbildung 3.11 zu sehen ist. Auch an der roten vollen Uhr, die das Timeout symbolisiert, kann man sich orientieren.

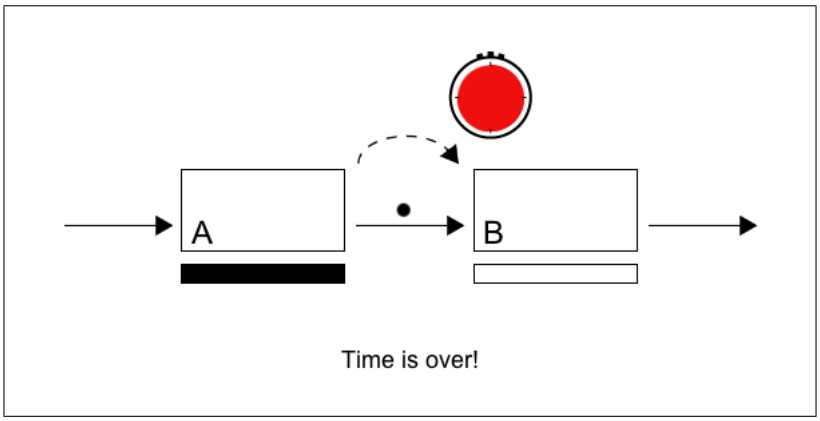

Abbildung 3.11: Meldung bei Timeout

3.3.6 Kalender

Der Kalender wird dann eingesetzt, wenn es in der Animation einen Termin gibt. Die Termine stehen als Kommentar oberhalb des Kalender, wie in der Abbildung 3.12 zu sehen ist.

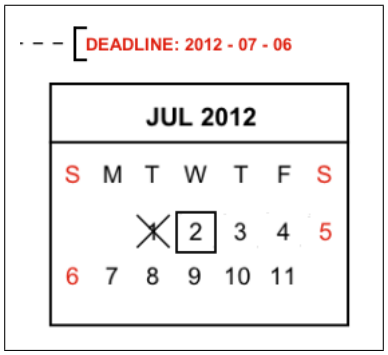

Abbildung 3.12: Deadline und Kalender

Während die Animation abläuft, werden im Hintergrund die Tage im Kalender durchgestrichen. Ist die Deadline erreicht und die Animation noch nicht fertig, dann wird die Animation beendet und ein Fehler als Kommentar angezeigt.

3.3.7 Steuerung der Aufmerksamkeit des Betrachters

Um die Aufmerksamkeit des Nutzers stets auf das Wesentliche zu lenken, werden folgende Regeln für die Umsetzung der Animationen festgelegt:

1. Die Objekte dürfen nicht zu schnell auftauchen bzw. verschwinden. Dem Nutzer könnte dies sonst entgehen wodurch er die Animation nur schlecht weiterverfolgen könnte.
2. Um den Nutzer nicht zu überfordern, soll vermieden werden, dass viele Elemente der Animation parallel ablaufen. Elemente dürfen nur dann parallel ablaufen, wenn
   a) sie räumlich nah sind und
   b) ihre Geschwindigkeit zueinander passt, sodass sie zusammen harmonieren.
3. Soll die Aufmerksamkeit des Nutzers auf ein Objekt gelenkt werden, so blinkt das Objekt zunächst auf und wird eventuell farblich gekennzeichnet.
4. Ist ein Timer kurz davor abzulaufen, so muss dieser farblich hervorgehoben werden: Der Timer soll bei dem letzten Viertel der Uhr zunächst blinken und dann rot werden. Damit wird dem Nutzer der Zeitmangel klar verdeutlicht.

3.4 Time Patterns

Im Folgenden werden die Visualisierungen der einzelnen Time Patterns im Detail vorgestellt. Time Patterns eins bis vier sind interaktiv gestaltet, wohingegen Time Patterns fünf bis acht als reine Animationen umgesetzt sind.

3.4.1 TP1: Time Lags between two Activities

Das Pattern „Time Lags between two Activities“ erlaubt es, zwischen zwei Aktivitäten eine Zeitspanne zu definieren, die eingehalten werden muss. Bei dieser Zeitspanne handelt es sich entweder um einen Minimal- oder Maximalabstand. Bei einem Minimalabstand muss die Zeit abgelaufen sein, bevor das nachfolgende Ereignis eintreten kann, wohingegen bei

einem Maximalabstand das Ereignis eingetreten sein muss bevor die Zeit abgelaufen ist. Man gibt somit eine Zeit vor, in der die Aktivität gestartet bzw. beendet sein muss. Dabei kann es folgende Beziehungen zwischen zwei Aktivitäten geben (vgl. [7, S. 7ff]):

a) „Start-Start“
   Diese Beziehung gibt an, dass, sobald die eine Aktivität gestartet ist, die andere in der angegebenen Zeit auch starten muss bzw. erst danach gestartet werden darf.

b) „Start-End“
   Hier muss, nach Start der ersten Aktivität, die zweite innerhalb einer vorgegebenen Zeitspanne abgeschlossen werden, bzw. darf erst nach Ablauf der Zeitspanne beendet werden.

c) „End-Start“
   Bei dieser Variation, fängt die Zeit an, abzulaufen, wenn die erste Aktivität beendet ist. Bevor bzw. nachdem die Zeit abgelaufen ist, muss die zweite starten.

d) „End-End“
   Nach Beendigung der ersten Aktivität muss innerhalb bzw. nach Ablauf der vorgegebenen Zeit auch die andere abgeschlossen werden.

Im Folgenden wird als Beispiel die Visualisierung der „End-Start“-Beziehung mit Maximalabstand zwischen zwei Aktivitäten vorgestellt. Die vollständige Animation ist in Abbildung 3.13 zu sehen.

Realisierungsidee

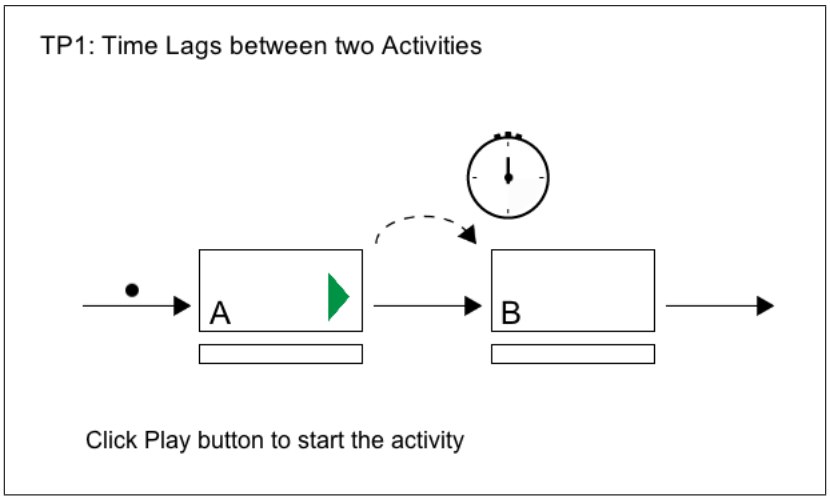

Abbildung 3.13: TP1 - Time Lags between two Activities

Der Nutzer wird anhand von Anweisungen durch die Animation geführt. Die entsprechende Anweisung für den aktuellen Schritt wird jeweils unterhalb der auszuführenden Aktivität eingeblendet. Der Hauptteil dieser Animation besteht aus den Aktivitäten, A und B, und der Uhr, die sich oberhalb der Aktivität B befindet. Der Prozess ist in Abbildung 3.13 dargestellt. Der Benutzer muss zunächst auf den Play-Button in Aktivität A klicken, um deren Ausführung zu starten. Daraufhin verschwindet das Token von dem vorangehenden Pfeil und erscheint innerhalb von Aktivität A wieder. Die Aktivität wird nun ausgeführt, was durch den Fortschritt des darunter eingeblendeten Balkens dargestellt wird (siehe Abbildung 3.14).

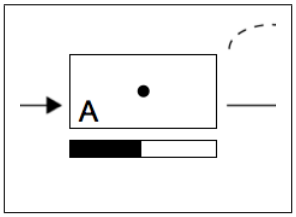

Abbildung 3.14: TP1 - Aktivität A wird ausgeführt

Sobald Aktivität A fertig ist, verschwindet das Token aus der Aktivität und erscheint auf dem nachfolgenden Pfeil. Wegen der „End-Start“-Beziehung beginnt jetzt, die Uhr abzulaufen, wie Abbildung 3.15 zeigt.

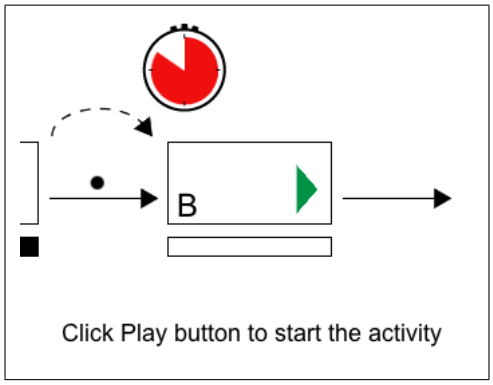

Abbildung 3.15: TP1 - Zeit läuft ab

Drückt der Nutzer innerhalb der vorgesehenen Zeit auf den in Aktivität B erschienenen Play-Button, und erscheint das Token somit rechtzeitig in Aktivität B, so wird der Timer angehalten und die Animation wird fortgesetzt: Der Fortschrittsbalken von B füllt sich. Die Animation ist zu Ende, sobald der Fortschrittsbalken von B voll ist und das Token auf dem nachfolgenden Pfeil erschienen ist. Abbildung 3.16 zeigt dies.

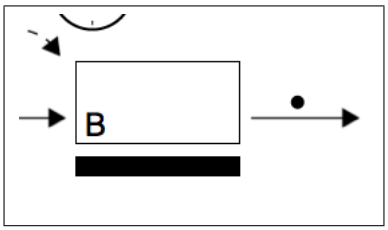

Abbildung 3.16: TP1 - Prozess abgeschlossen

Wartet der Nutzer aber zu lange und drückt nicht in der vorgesehenen Zeit auf den Play-Button von Aktivität B, bricht die Animation ab. Der Nutzer wird durch die Meldung „Time is over“ auf den Fehler bzw. das Ablaufen der Zeit hingewiesen (vgl. Abbildung 3.17).

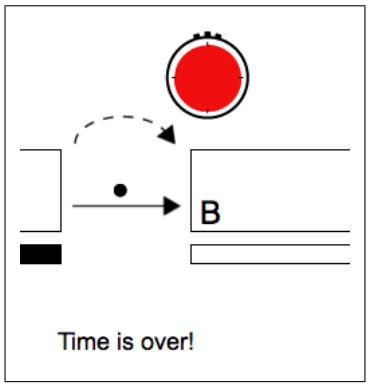

Abbildung 3.17: TP1 - Zeit ist abgelaufen

3.4.2 TP2: Durations

Dieses Pattern beschreibt, wie lange eine Aktivität höchstens andauern darf oder nach welcher Zeit sie frühestens beendet werden darf. Man unterscheidet 3 Arten von Dauern:

a) Minimal
   Die Aktivität darf nicht vor Ablauf der Zeit beendet werden.

b) Maximal
   Die Aktivität muss vor Ablauf der Zeit beendet werden.

c) Zeitintervall
   Innerhalb eines Intervalls muss eine Aktivität beendet werden. Hier ist ein frühester und spätester Zeitpunkt gegeben, zu dem die Aktivität beendet werden darf bzw. muss.

Einzelne Aktivitäten oder der gesamte Prozess können von solchen Arten der Dauer betroffen sein (vgl. [7, S. 8ff]).

Realisierungsidee

Bei diesem Time Pattern geht es darum, dass die Aktivität innerhalb der vorgesehenen Zeit abgeschlossen werden muss. Die Animation besteht aus der Aktivität A. Über dieser

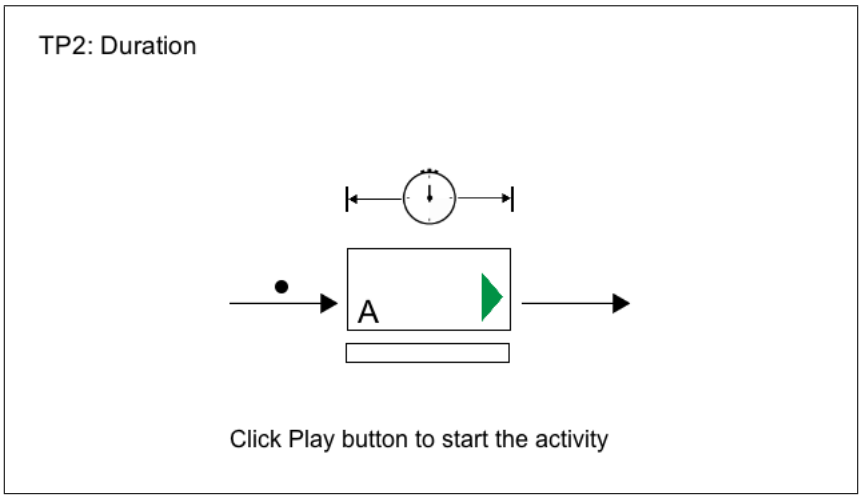

Abbildung 3.18: TP2 - Durations

Aktivität befindet sich eine Uhr. Die Uhr und die zwei Pfeile sind genauso lang wie der Fortschrittsbalken (siehe Abbildung 3.18). Das soll dem Nutzer bildlich zeigen, dass die Aktivität innerhalb der Zeit fertig sein muss, um die Animation positiv zu beenden. Dauert die Aktivität länger als die Zeitspanne, so wird die Animation abgebrochen. In diesem Fall wird der Nutzer durch eine Fehlermeldung darauf aufmerksam gemacht.
Die Animation wird gestartet, indem der Nutzer auf den Play-Button, der in der Aktivität A angezeigt wird, klickt. Daraufhin verschwindet das Token und erscheint in der Aktivität A wieder. Dies zeigt die Abbildung 3.19.

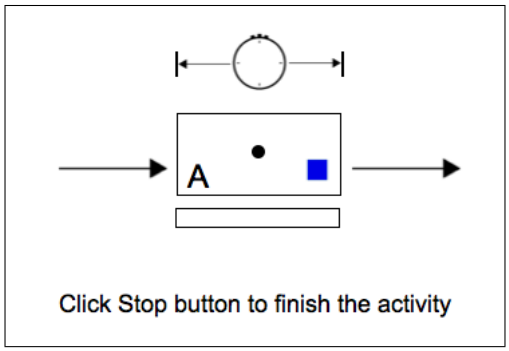

Abbildung 3.19: TP2 - Aktivität A wird gestartet

Die Uhr fängt an, abzulaufen. Das Besondere bei dieser Animation ist, dass der Benutzer selbst entscheiden darf, wann die Aktivität beendet ist. Dafür klickt der Benutzer auf den Stop-Button, der sich in der Aktivität A befindet. Deswegen gibt es bei dieser Animation keinen sich füllenden Fortschrittsbalken wie bei den anderen, sondern einen Rahmen, durch den ein schwarzer Balken durchläuft. Dieser Balken zeigt an, dass die Aktivität gerade bearbeitet wird, aber nicht, wann sie zu Ende ist (siehe Abbildung 3.20).

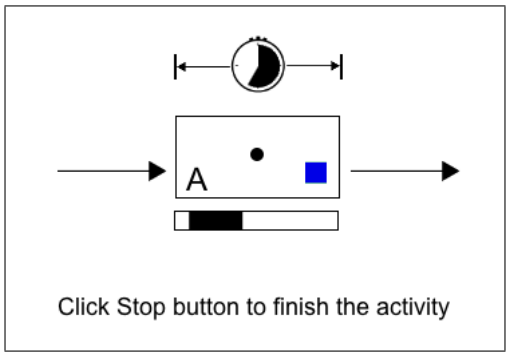

Abbildung 3.20: TP2 - Aktivität A in Bearbeitung

Klickt der Benutzer innerhalb der Zeitspanne auf den Stop-Button, so wird die Animation positiv weiter ausgeführt: Es erscheint ein schwarzer voller Balken für den Fortschrittsbalken und das Token verschwindet aus der Aktivität A und erscheint auf dem Pfeil wieder. Die Animation ist beendet (vgl. Abbildung 3.21).

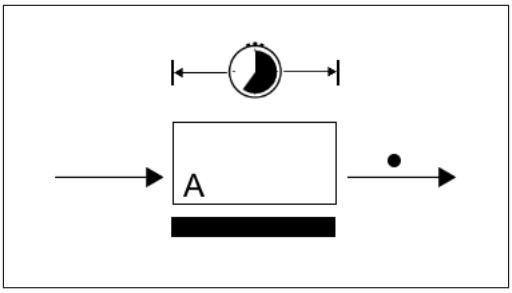

Abbildung 3.21: TP2 - Prozess abgeschlossen

Andernfalls kommt sobald die Zeit abgelaufen ist die Meldung „Time is over!“.

3.4.3 TP3: Time Lags between Arbitrary Events

Das Besondere bei TP3 ist, dass ein externes Event einen Timer starten kann. Dieses Event ist dabei völlig unabhängig und an keine Aktivität gebunden. Sobald dieses Event erscheint, wird der Timer für den Zeitabstand ausgelöst. Wie bei Abschnitt 3.4.2 kann die Aktivität in der vorgesehenen minimalen oder maximalen Zeit oder dem Intervall (minimal oder maximal) starten oder enden. Das auslösende Event kann, zum Beispiel, eine eintreffende Nachricht oder ein eintretendes Ereignis sein (vgl. [7, S. 9f]).

Realisierungsidee

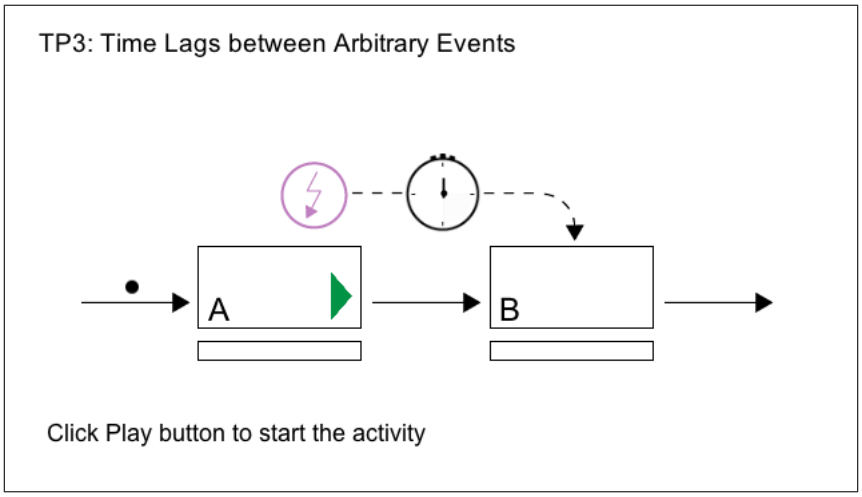

Abbildung 3.22: TP3 - Time Lags between Arbitrary Events

Die Animation beginnt, indem der Nutzer auf den Play-Button in Aktivität A klickt. Er wird durch die Interaktionsanweisung „Click Play button to start the activity“ hierzu aufgefordert (vgl. Abbildung 3.22). Daraufhin fängt Aktivität A an abzulaufen. Gleichzeitig erscheint über dem Blitzsymbol eine Anweisung, wie Abbildung 3.23 zeigt. Der Nutzer hat nun zwei Möglichkeiten:

1. Er kann das Blitzsymbol ignorieren und, nachdem Aktivität A beendet ist und unter Aktivität B die neue Interaktionsanweisung erschienen ist, Aktivität B starten. Aktivität B wird dann ausgeführt und die Animation ist danach beendet.

2. Der Nutzer kann aber auch das Blitzsymbol anklicken. Um die Animation erfolgreich zu beenden, muss Aktivität B nun in der vorgesehenen Zeit starten. Dass heißt, der Nutzer muss innerhalb dieser Zeit auf den Play-Button in der Aktivität B klicken. Die Zeit wird in diesem Fall angehalten. Folgt der Nutzer nicht der Anweisung, dann erscheint irgendwann die Timeoutmeldung und die Animation ist beendet.

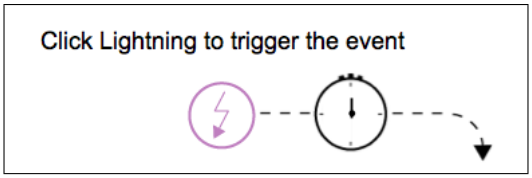

Abbildung 3.23: TP3 - Blitzsymbol löst Timer aus

3.4.4 TP4: Fixed Date Elements

Oftmals spielen Deadlines, wie der Abgabetermin eines Projektes oder das pünktliche Erledigen einer Aufgabe, eine große Rolle bei Prozessen. Das Pattern „Fixed date element“ gibt genau solche Deadlines vor. Auch bei diesem Pattern kann sich der zeitliche Aspekt auf eine Aktivität oder auf den gesamten Prozess beziehen. Die Aktivität kann einen

- frühesten Startzeitpunkt,
- spätesten Startzeitpunkt,
- frühesten Endzeitpunkt oder
- spätesten Endzeitpunkt

haben [7, S. 9ff].

Realisierungsidee

Die Abbildung 3.24 zeigt den gesamten Prozess. Der Nutzer klickt, durch die Anweisung „Click Play button to start the activity“ geleitet, zu Beginn auf den Play-Button, um die Aktivität „Make Appointment“ zu starten. Während diese Aktivität läuft, wird die Deadline festgesetzt: Diese erscheint als Kommentar oberhalb des Kalenders und blinkt dabei

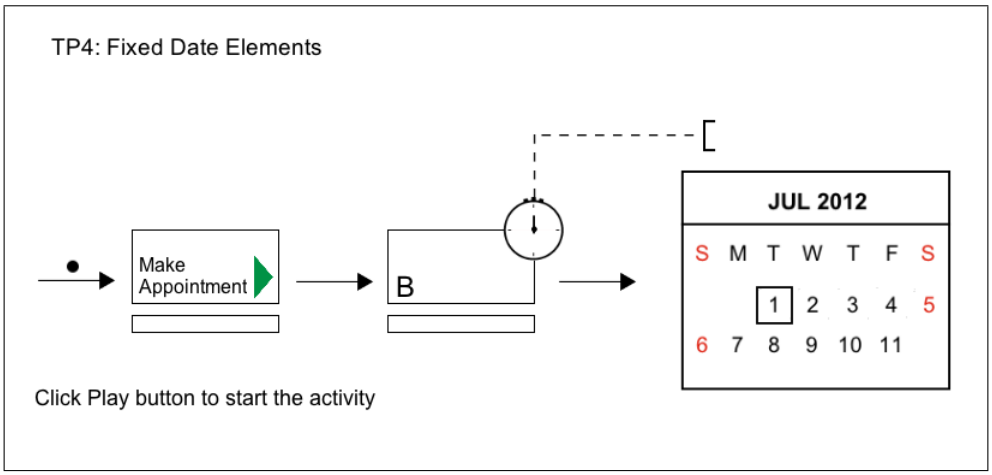

Abbildung 3.24: TP4 - Fixed Date Elements

zweimal auf (vgl. Abbildung 3.25). Danach wird die Uhr vollständig gefüllt und blinkt ebenfalls zweimal auf. Dadurch wird der Timer ausgelöst. Sobald die Aktivität „Make Appointment“ beendet ist und das Token auf dem Pfeil erscheint, fängt die Uhr an, abzulaufen. Gleichzeitig werden im Hintergrund in dem Kalender die Tage durchgestrichen, wie Abbildung 3.26 illustriert. Startet der Nutzer die Aktivität nicht innerhalb in der vorgesehenen Zeit, erscheint eine Timeout-Meldung.

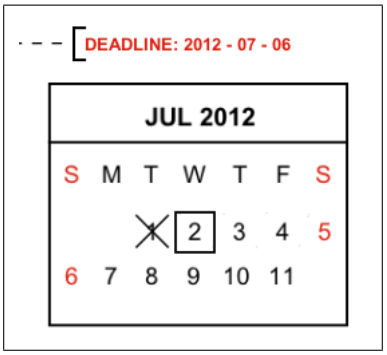

Abbildung 3.25: TP4 - Deadline wird festgesetzt

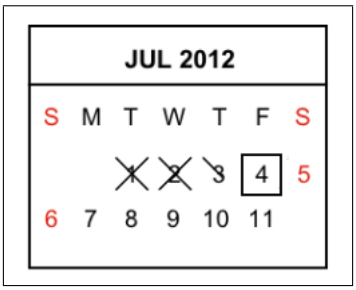

Abbildung 3.26: TP4 - Tag 3 wird durchgestrichen

3.4.5 TP5: Schedule Restricted Elements

Prozesse müssen sich an reale Gegebenheiten, wie Öffnungszeiten oder Stundenpläne, halten. Diese geben Zeitslots vor, in denen Aktivitäten ausgeführt werden dürfen. Wie bei TP4 gibt es auch bei diesem Pattern vier Zeitpunkte, auf die sich das Pattern beziehen kann (frühester/spätester Startzeitpunkt, frühester/spätester Endzeitpunkt). Eine Exception tritt auf, wenn eine Aktivität (ein Prozess) nicht in einem solchen Zeitslot ausgeführt wird. (vgl. [7, S. 10ff]).

Realisierungsidee

Dieses Time Pattern ist das Erste, welches als reine Animation umgesetzt wird. Ein Grund hierfür ist, die Komplexität der vielen parallelen Ereignisse, weswegen eine interaktive Umsetzung nicht geeignet ist. Die Abbildung 3.27 stellt einen Prozess dar, der einen Einkaufsablauf modellieren soll. Bei „Schedule Restricted Elements“ geht es darum, dass eine Aktivität nur zu bestimmten Zeiten ausgeführt werden darf: Der Supermarkt „GOOD FOOD“ hat von 10 - 18 Uhr geöffnet. Rechts neben den Öffnungszeiten befindet sich eine Uhr, deren Uhrzeiger von 10 - 18 Uhr grün ist und andernfalls rot. Auch die Farbe der Öffnungszeiten auf dem Schild ändert sich entsprechend zum Zeiger dieser Uhr: Wenn der Laden offen ist, dann sind die Öffnungszeiten grün, andernfalls rot. Die Aktivität kann nur gestartet werden, wenn der Laden offen ist.
Ein Kunde wartet schon, bevor der Laden geöffnet ist. Sobald es 10 Uhr ist, öffnet sich die

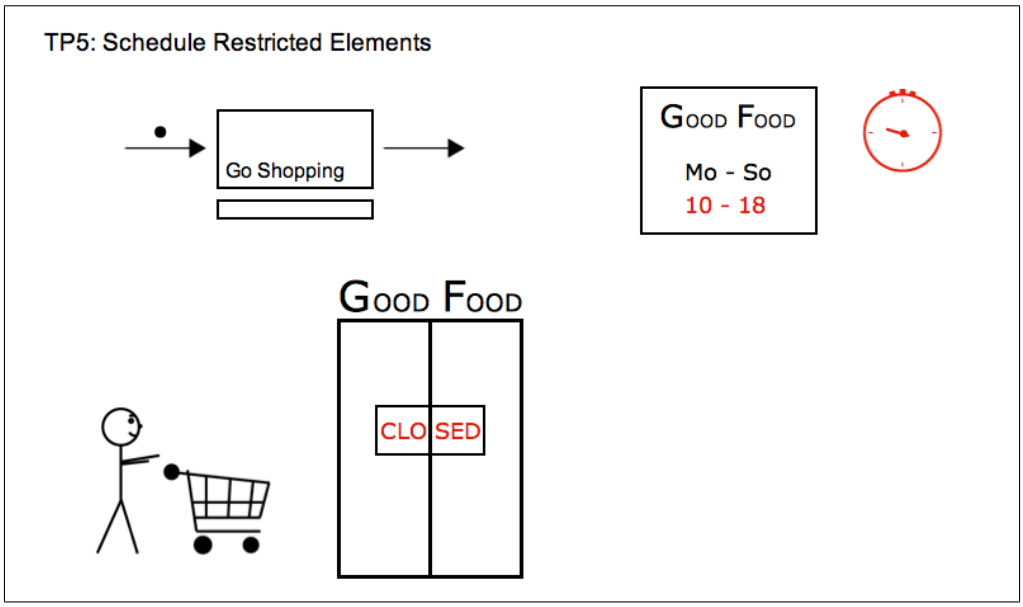

Abbildung 3.27: TP5 - Duration

Tür und der Kunde kann hineingehen. Zeitgleich wird die Aktivität „Go Shopping“ gestartet (siehe Abbildung 3.29). Die drei zentralen Symbole der Animation werden in Abbildung 3.28 illustriert.

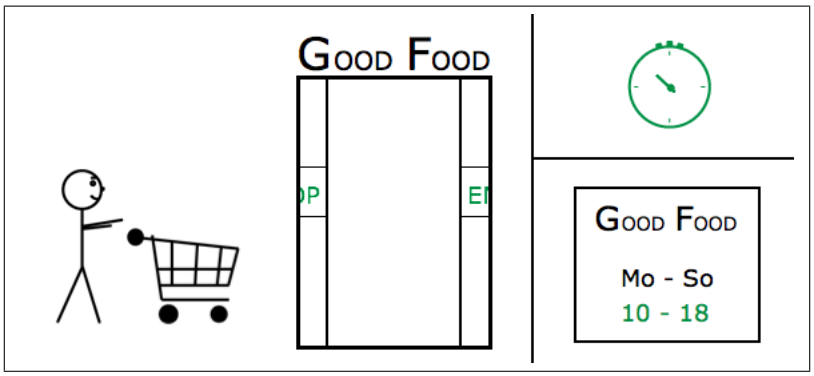

Abbildung 3.28: TP5 - Kunde kann Laden betreten

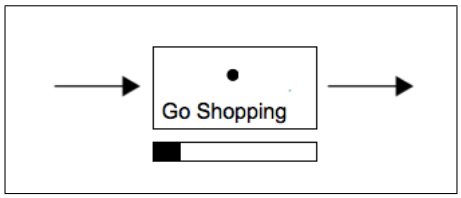

Abbildung 3.29: TP5 - Aktivität „Go Shopping“ nach Betreten des Ladens

Kurz vor Ladenschluss, kommt der Kunde mit vollem Einkaufswagen wieder heraus. Ein anderer Kunde kommt zu spät, es ist bereits nach 18 Uhr. Er muss nun die ganze Nacht warten, bis der Laden wieder geöffnet wird. Erst dann kann er einkaufen gehen.

3.4.6 TP6: Time-based Restrictions

Manchmal ist es wichtig, die Anzahl, wie oft sich ein Prozess (oder eine Aktivität) innerhalb eines vordefinierten Zeitfensters wiederholen darf, zu beschränken. Das kann zum Beispiel an den verfügbaren Ressourcen liegen. Die Anzahl der Ausführungen kann dabei durch ein Minimum oder Maximum beschränkt werden (vgl. [7, S. 12f]).

Realisierungsidee

Die Einschränkung bei „Time-based Restrictions“ ist, dass dem „Strichmännchen“ nur zwei Kuchenstücke pro Tag zur Verfügung stehen. Wenn er mehr haben möchte, so muss er bis zum nächsten Tag abwarten. Wenn das Strichmännchen ein Kuchenstück (Aktivität „Take Cake“ in Abbildung 3.30) nimmt, wird der Kommentar über der Raute mit der Uhr geändert: Jedes Mal wenn ein Kuchenstück gegessen wird, wird die Anzahl der Kuchenstücke um eins reduziert und ein Stück verschwindet. Wenn die Anzahl der Kuchenstücke kritisch wird (eins oder null), wird die Zahl in rot dargestellt (siehe Abbildung 3.31). Bei null Kuchenstücken muss abgewartet werden, bis neue geliefert werden.

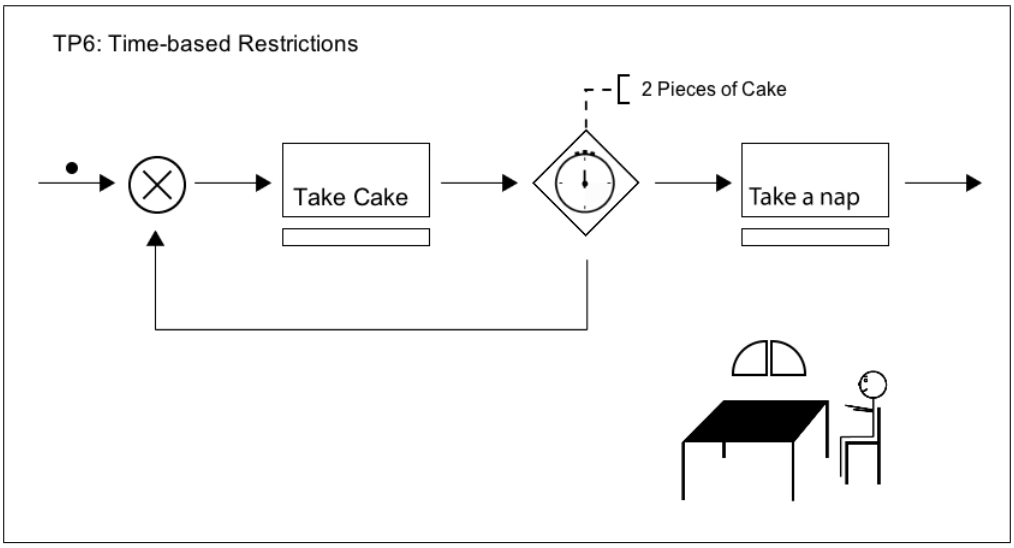

Abbildung 3.30: TP6 - Time-based Restrictions

Abbildung 3.31: TP6 - Anzahl der Küchenstücke

Das Strichmännchen nimmt genau zwei Stücke. Nachdem neue Kuchenstücke am nächsten Tag gebracht werden und das Strichmännchen noch ein Kuchenstück isst, entscheidet es sich, keine weiteren Stücke mehr zu nehmen. Die Aktivität „Take a nap“ wird gestartet. Danach ist die Animation beendet.

3.4.7 TP7: Validity Period

Prozesse können sich über die Zeit verändern: Eine Aktivität kann hinzukommen, wegfallen oder der ganze Prozess wird geändert. Durch solche Änderungen entstehen unterschiedliche Versionen eines Prozesses. Zu einem bestimmten Zeitpunkt soll jedoch nur eine der Versionen gültig sein. Hierfür bietet TP7 die Möglichkeit Gültigkeitsperioden („Validity Periods“) für Prozesse zu definieren [7, S. 12ff].

Realisierungsidee

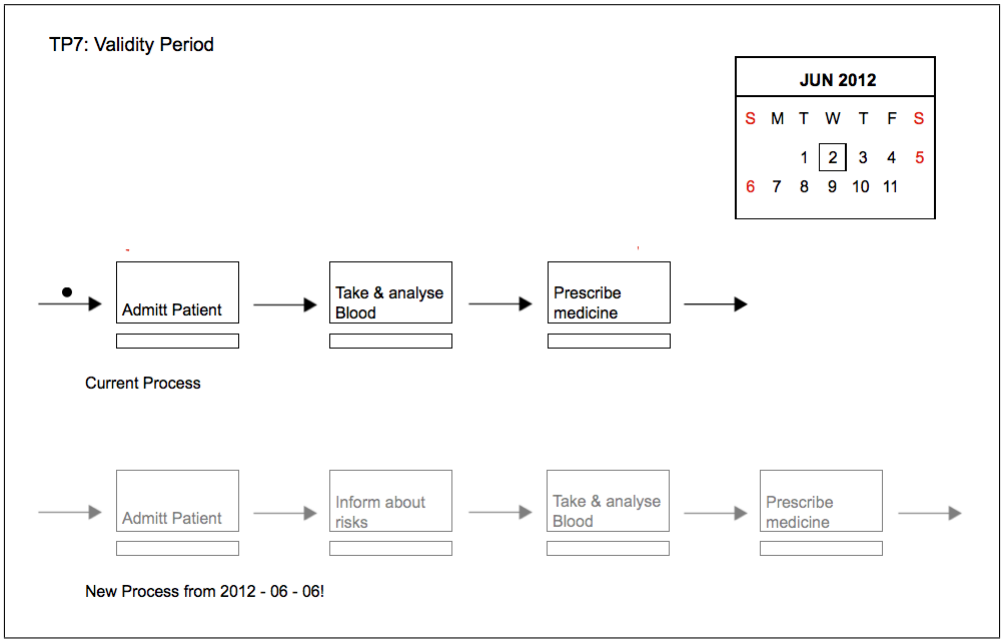

Abbildung 3.32: TP7 - Validity Period

Bei dem Time Pattern „Validity Period“ läuft parallel neben dem Prozess ein Kalender, wie in Abbildung 3.32 zu sehen ist. In dieser Animation sind zwei Prozesse dargestellt, die sich nur durch die Aktivität „Inform about risks“ unterscheiden. Der obere Prozess ist der aktuell gültige.

Ab dem 2012 - 06 - 06 soll sich dieser Prozess ändern: Die Aktivität „Inform about risks“ soll zwischen den Aktivitäten „Admitt Patient“ und „Prescribe medicine“ eingefügt werden. Der entsprechende Prozess ist bereits unterhalb des aktuellen Prozesses ausgegraut zu erkennen. Zunächst sieht man den aktuellen Prozess ablaufen, während die Tage im Kalender durchgestrichen werden. Ist der sechste Juni erreicht, so wird der aktuelle Prozess ausgegraut und durchgestrichen und durch den neuen ersetzt. Außerdem wird der Text „Current Process“ unter dem oberen Prozess durch „Old Process“ ersetzt. Beim unteren Prozess ändert sich der Text von „New Process from 2012 - 06 - 06!“ zu „Current

Process“ (siehe Abbildung 3.33). Der neue Prozess läuft ab, während die Tage weiter durchgestrichen werden.

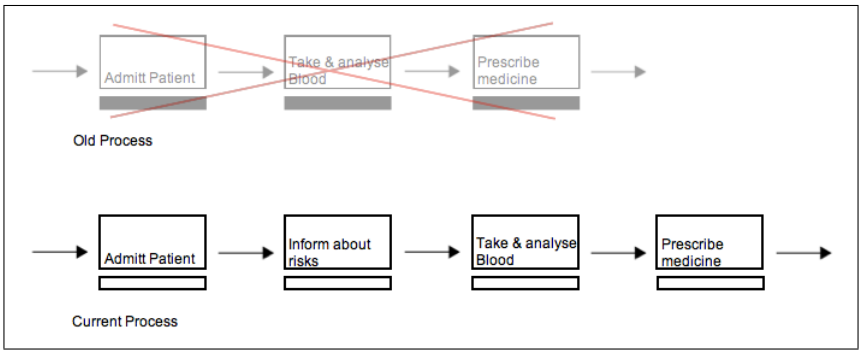

Abbildung 3.33: TP7 - Neuer Prozess

3.4.8 TP8: Time-dependent Variability

Das Pattern „Time-dependent Variability“ bietet die Möglichkeit, in Prozessen Entscheidungen zu modellieren, die von Zeitaspekten abhängig sind. Erst zur Laufzeit eines Prozesses wird basierend auf der aktuellen Zeit entschieden, welcher Pfad zur Ausführung kommt [7, S. 13f].

Realisierungsidee

Die Abbildung 3.34 zeigt einen Prozess, bei dem es um die Überprüfung des Mindesthaltbarkeitsdatums geht. Die Animation beginnt, indem das Token in der Aktivität „Check Best-Before Date“ erscheint. Daraufhin erscheint oberhalb der Aktivität der Kommentar „Best-Before Date: 2012 - 06 - 03“, welcher das entsprechende Mindesthaltbarkeitsdatum darstellt. (vgl. Abbildung 3.35).

Abbildung 3.35: TP8 - Best-Before Date erscheint

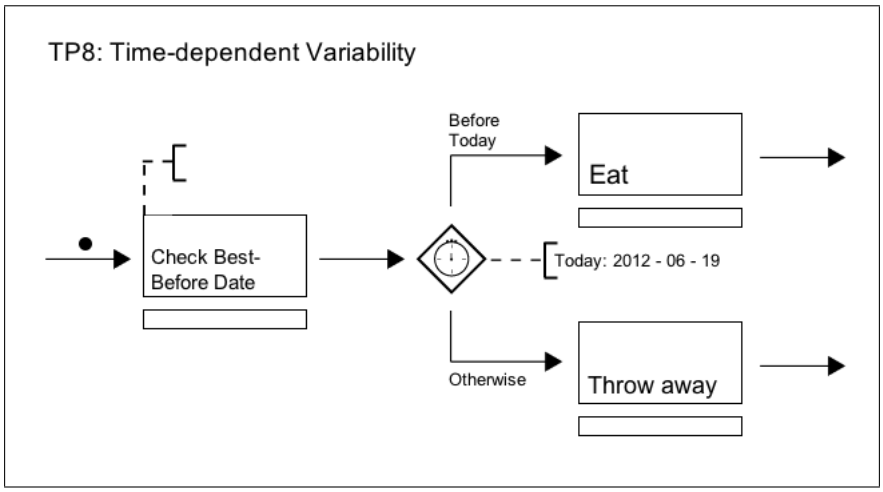

Abbildung 3.34: TP8 - Time-dependent Variability

Nachdem die Aktivität abgeschlossen ist, und das Token auf dem Pfeil auftaucht wird das Entscheidungs-Gateway ausgewertet. Die beiden Kommentare „Best-Before Date: 2012 - 06 - 03“ und „Today: 2012 - 06 - 19“ blinken mehrmals um die Aufmerksamkeit des Nutzers auf die Kommentare zu lenken. Dadurch sieht er, dass das Lebensmittel abgelaufen ist. Daraufhin wird der obere XOR-Pfad ausgeblendet, weil dieser nicht mehr ausführbar ist, was in Abbildung 3.36 dargestellt ist. Das Token erscheint auf dem „Otherwise“-Pfeil und die Aktivität „Throw away“ wird ausgeführt. Damit ist die Animation zu Ende.

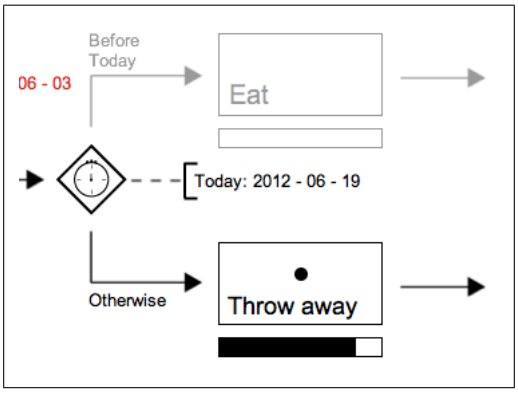

Abbildung 3.36: TP8 - Alternativpfad wird deaktiviert

4

Software

Dieses Kapitel beschäftigt sich mit der Software, die im Rahmen dieser Bachelorarbeit auf ihre Eignung zur Gestaltung der Animationen geprüft wird. Dabei werden Adobe Edge, Sencha Animator und Tumult Hype, die alle HTML5 unterstützen, zunächst vorgestellt und dann miteinander verglichen. Für die Umsetzung der Animationen wird eine Software benötigt, die den heutigen technischen Anforderungen gerecht wird. Was die technischen Anforderungen sind, wird in den Abschnitten „Flash und HTML5 im Vergleich“ und „Anforderungen an die Software“ diskutiert. Im Folgenden wird jedoch „Edge Animate 1.0“ vorgestellt, weil die Vorgängerversion nicht mehr verwendbar ist.

4.1 Flash und HTML5 im Vergleich

Seit seiner Veröffentlichung durch die Firma Macromedia im Jahre 1997 genießt Flash große Beliebtheit bei Webentwicklern für die Animation von Webinhalten. Heute wird Flash von Adobe vertrieben und steht für die gängigen Browser als Plug-In zur Verfügung. Es kommt heute vor allem auf Video-Streaming Plattformen und bei Browser-Spielen zum Einsatz. Zwar konnte sich Flash in der Vergangenheit auf Desktop-PCs durchsetzen, doch der Sprung auf die Smartphones und Tablets gelang ihm nicht. Das liegt unter anderem daran, dass, einer der Vorreiter in diesem Bereich, die Firma Apple, die Installation des Flash-Players auf iOS (Apples Betriebssystem für mobile Geräte, wie iPhone, iPod und iPad) verhindert. Steve Jobs, Mitgründer und langjähriger CEO von Apple, begründete diese Entscheidung damit, dass Flash den heutigen Anforderungen an Applikationen für mobile Geräte (Zuverlässigkeit, Sicherheit, Leistung und Akkulaufzeit) nicht gerecht wird. So ist - nach Steve Jobs - Flash „the number one reason Macs crash“ [5]. Außerdem verweist er auf einen Bericht der Firma Symantec (eine der führenden Firmen im Bereich der PC Sicherheit), welcher aufzeigt, dass Flash in Fragen der Sicherheit zu den schlechtesten Anwendungen gehört. Außerdem geht nach Ansicht von Steve Jobs bei Flash zu viel Leistung und Akkulaufzeit für die Dekodierung von Videos in Software verloren, da Flash viele verschiedene Codecs erlaubt, die Hardware mobiler Geräte jedoch nur den Codec H.264 dekodieren kann. Hinzu kommt, dass Flash für die Eingabe durch Maus und Tastatur ausgelegt ist, welche in einem mobilen Gerät in der Regel nicht zum Einsatz kommen. Als wichtigsten Punkt nennt Steve Jobs jedoch, die Abhängigkeit der Plattform- und Anwendungsentwickler von einer Drittsoftware. So könne der Anwendungsentwickler von den Neuerungen und Verbesserungen einer Plattform erst Gebrauch machen, nachdem Adobe diese in den Flash-Player integriert hat [5].

Hinzu kommt, dass Adobe inzwischen seine Unterstützung für den Flash Player auf mobilen Geräten weitgehend eingestellt und ihn aus Googles PlayStore entfernt hat. Auch die Unterstützung des Players für Linux Betriebssysteme wurde bereits eingestellt [9].

Im Gegensatz dazu ist es durch die Einführung des „Canvas“-Elements in die Spezifikation von HTML5 nun möglich mittels HTML5 in Kombination mit JavaScript und CSS3 gleichwertige Ergebnisse zu erzielen. Dadurch stellt HTML5 einen ernstzunehmenden

Konkurrenten für Flash dar. Obwohl HTML5 noch nicht offiziell veröffentlicht wurde, ist es in den gängigen Browsern bereits (wenn auch unvollständig) implementiert und genießt eine immer größere Beliebtheit bei Webentwicklern. Für HTML wurden darüber hinaus klare Gestaltungsprinzipien festgelegt. Diese umfassen die Bereiche Kompatibilität, Verwendbarkeit, Sicherheit, Konsistenz, Vereinfachung, Universalität und Barrierefreiheit [6].

Insgesamt ist festzustellen, dass Flash immer mehr an Bedeutung verliert und inzwischen selbst von Adobe immer weniger unterstützt wird. HTML5 dagegen setzt sich mehr und mehr durch. Selbst Adobe stellt inzwischen Werkzeuge zur Erstellung von Webinhalten in HTML5 (Adobe Edge) zur Verfügung. Aus diesen Gründen fiel die Wahl zur Realisierung der Animationen schließlich auf HTML5.

4.2 Anforderungen an die Software

Wie zuvor diskutiert ist Flash seit einiger Zeit nicht mehr die erste Wahl, wenn es um Animationen und Videos im Internet geht. Stattdessen wird HTML5 den neuen Anforderungen besser gerecht. Die Software für die Umsetzung sollte deshalb diese Neuerungen der Technologie (z.B. smartphones und Tablets) unterstützen. Außerdem muss sie die Möglichkeit bieten, die Animation interaktiv zu gestalten: Dem Nutzer soll bei den Time Patterns 1 - 4 die Möglichkeit geboten werden, diese interaktiv zu steuern. Dazu sollte eine Möglichkeit zur Erstellung und Bearbeitung von JavaScript vorhanden sein. Die Software sollte ebenso eine Einstiegshilfe (in Form von einem Tutorial oder Video) bieten, um einen schnellen Einstieg zu ermöglichen. Darüber hinaus sollte das Programm für gängige Betriebssysteme, wie Windows oder Mac OS, erhältlich sein, damit die Animationen plattformübergreifend gestaltet werden können. Um eine breite Nutzerschicht erreichen zu können ist es zudem notwendig, dass weit verbreitete Browser (oder auch Smartphones), wie Firefox, Safari, Chrome, etc. unterstützt werden. Damit wird dem Nutzer die Möglichkeit geboten, sich die Animationen, unabhängig von seinem Browser, anzusehen.

4.3 Adobe Edge

„Adobe Edge“, welches durch Adobe Systems veröffentlicht wurde, ist derzeit in der Version „Edge Animate 1.0“ erhältlich. Die grafische Benutzeroberfläche gliedert sich in verschiedene Bereiche (siehe Abbildung 4.1): In der Mitte befindet sich die Arbeitsfläche, in der die Bühne zu sehen ist. Die Bühne stellt den Bereich dar, der später im Browser angezeigt wird. Zur besseren Darstellung, wird die Arbeitsfläche in Abbildung 4.1 mit einem grünen Rahmen hervorgehoben. Unterhalb der Arbeitsfläche befindet sich die Zeitleiste (violetter Rahmen). Die Zeitleiste dient zur Steuerung des zeitlichen Ablaufs einer Animation. Links daneben lassen sich die Eigenschaften der Animation ändern. Hier werden die Eigenschaften des aktuell ausgewählten Elements angezeigt (Position und Größe, Transformieren, etc.). Dieser Bereich hat einen türkisfarbenen Rahmen. Die Fenster Elemente (rot) und Bibliothek (gelb) befinden sich jeweils zur rechten Seite der Arbeitsfläche. Das Fenster Elemente listet alle Objekte auf, die in der Bühne vorkommen. Mit der Bibliothek können Bilder oder Symbole geladen und wiederverwendet werden. Die graphische Benutzeroberfläche wird durch Einzelheiten wie

- Lineal am Arbeitsbereich für eine bessere Anordnung der Elemente
- Neue Icons für Elemente (Symbole, Bilder, Text)

ergänzt. Die Grundlage von Edge bildet ein Animationsframework, das auf dem jQuery Core Framework basiert [2]. Als unterstützte Browser zählt Adobe Firefox, Chrome, Safari und Internet Explorer 9 auf. Zu den unterstützten Smartphone-Betriebssystemen zählen Apple iOS und Android.

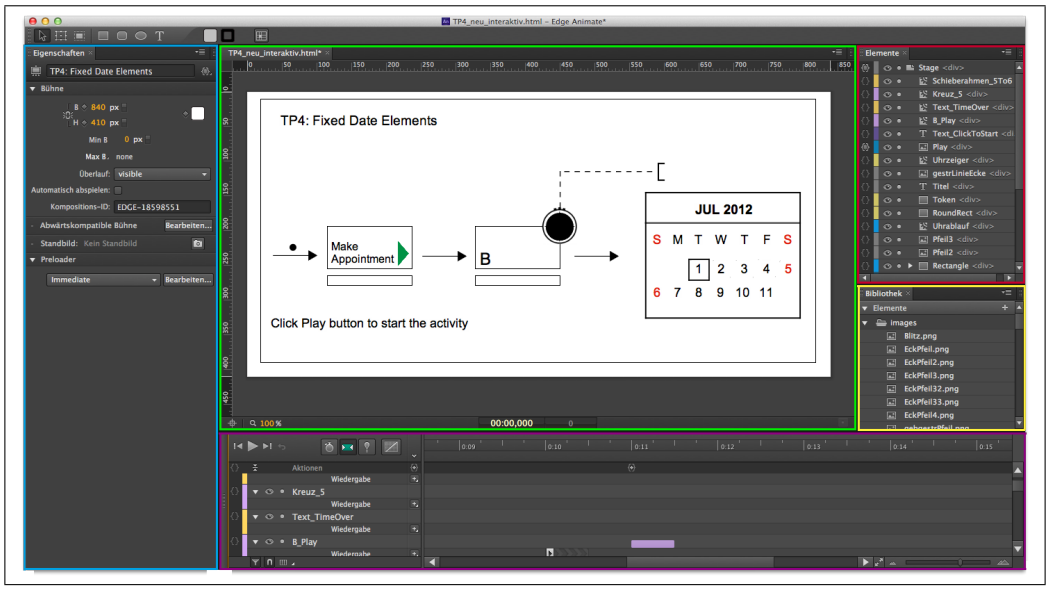

Abbildung 4.1: Adobe Edge Animate 1.0

4.4 Sencha Animator

„Sencha Animator“ wird von Sencha Inc. entwickelt. Wie bei Adobe Edge, wird bei Sencha mit einer Timeline gearbeitet (siehe Abbildung 4.2). Sencha arbeitet mit einem eigenen JavaScript-Framework, dem „Sencha Touch“ [8]. Animator ist ähnlich wie Edge Animate aufgebaut: Neben den Fenstern wie Arbeitsfläche, Zeitleiste, Objekte (unterteilt in Eigenschaften, Bibliothek und Projekte) gibt es noch „Scenes“: Sie erlauben die Unterteilung der Animation in Szenen, die unabhängig von einander abgespielt werden können. Ähnlich wie Symbole in Adobe Edge haben diese Szenen ihre eigene Timeline. Die einzelnen Szenen findet man auch in einem zusätzlichen Fenster, in dem sie durch ein Thumbnail und ihren Titel dargestellt werden. Darüber hinaus können die Szenen auch exportiert und in anderen Animationen wiederverwendet werden. Mit Sencha Animator erstellte Animationen erfordern einen CSS3-fähigen Browser (**C**ascading **S**tyle **S**heets). Im Gegensatz zu Adobe Edge unterstützt Sencha Animator zusätzlich noch das BlackBerry-Betriebssystem [8].

Abbildung 4.2: Sencha Animator 1.3

4.5 Tumult Hype

Die von Tumult Inc. veröffentlichte Software „Tumult Hype“ stellt ein drittes Tool zur Erstellung von Animationen auf HTML5 Basis dar. Auch hier liegt ein Animationsframework basierend auf CSS3 zugrunde. Wie die zwei anderen HTML5-Tools, unterstützt auch Tumult Hype JavaScript. Man kann in diesem Tool Unterseiten, die als „Scenes“ bezeichnet werden, erstellen und mit anderen Seiten verlinken. Zusätzlich zu den von Adobe Edge unterstützten Browsern wird noch Opera aufgezählt. Zu den unterstützten Smartphone-Betriebssystemen gehören Apple iOS und Android Phones [10]. Tumult Hype bietet die Möglichkeit Änderungen an den Objekten aufzuzeichnen und dementsprechend Key-Frames zu setzen, um hieraus eine Animation zu erzeugen. Die Benutzeroberfläche bei Tumult Hype besteht aus dem Szeneeditor, der Szeneliste, Kontrollelementen zur

Animationssteuerung und der Timeline. Die Timeline ist wiederum unterteilt in Objekte und Eigenschaften. Abbildung 4.3 zeigt die Oberfläche von Tumult Hype [11].

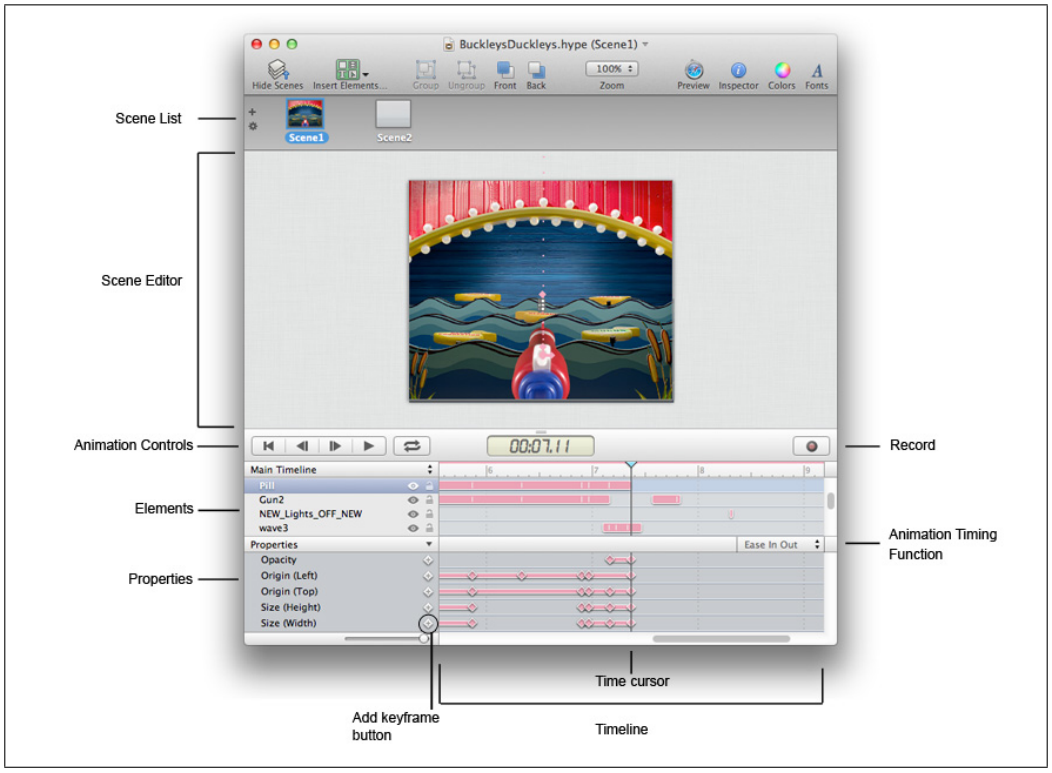

Abbildung 4.3: Tumult Hype 1.5

4.6 Softwarevergleich

In Tabelle 4.1 werden die Eigenschaften von Adobe Edge, Tumult Hype und Sencha Animator mit einander verglichen. Alle drei Tools werden den vorher definierten Anforderungen gerecht, mit Ausnahme von Tumult Hype, welches ausschließlich für Mac OS X erhältlich ist. Sowohl Adobe Edge als auch Sencha Animator bieten eine gute Einstiegshilfe und unterstützen die gängigsten Browser. Die Möglichkeit zur Unterteilung der Animationen in Szenen gehört nicht zu den Anforderungen, da die Animationen der Time Patterns in der Regel aus einer einzigen Szene bestehen. Die Unterstützung

animierter Symbole in Adobe Edge hat sich jedoch als nützlich erwiesen, weil es mehrere gemeinsame Bestandteile gibt, die einzeln animiert werden müssen. Außerdem verwendet Edge jQuery, eine weit verbreitete JavaScript Bibliothek, anstelle einer Eigenentwicklung. Damit ist eine breite Unterstützung sichergestellt. Aufgrund dieser positiven Aspekte, fiel die Wahl letztendlich auf Adobe Edge.

	Adobe Edge	Sencha Animator	Tumult Hype
Aktuelle Version	Animate 1.0	1.3	1.5
Betriebssystem	Mac OS X, Windows	Linux, Mac OS X, Windows	Mac OS X
Technische Grundlage	Animationsframework basierend auf jQuery	JavaScript-Framework Sencha Touch	CSS3
Einstiegshilfe	Lessons	Guided Tour online	
Skriptunterstützung	JavaScript	JavaScript	JavaScript
Unterstützung für Wieder-verwendbarkeit	Symbole	Scenes	Scenes
Umgang mit existierendem HTML Code	Problemlos, da Animation von HTML Code getrennt		Änderungen in eigenem HTML Editor
Unterstützte Browser	Firefox, Chrome, Safari, IE 9+	CSS3-fähige Browser	Safari 5+, Chrome 9+, Firefox 3.5+, IE 6+, Opera 11.10+
Unterstützte Smartphone-Betriebssysteme	Apple iOS, Android	Verbesserte Versionen für Android 2.3+, Apple iOS 4+, BlackBerry OS6+	Apple iOS 4.3.1+, Android 2.3+,

Tabelle 4.1: Vergleich der Software

5

Realisierung

Für die Animationen spielt die Uhr eine besondere Rolle: Sie ist das Kernstück, da sie den zeitlichen Aspekt darstellt. Dynamische Objekte, wie der Fortschrittsbalken und die Uhr, werden mithilfe von mehreren Bausteinen realisiert. Dieses Kapitel zeigt auf, wie diese Objekte zusammengesetzt sind und wie die interaktive Animationen realisiert wird. Die in den Animationen dargestellten Bilder sind mithilfe von Adobe Illustrator entworfen und die Visualisierungen der Time Patterns wurden mit der zuvor veröffentlichte Version „Preview 6“ animiert. Zur Erstellung der Skripte für die interaktiven Animationen wurde die API (Programmierschnittstelle) von Adobe Edge herangezogen [1].

5.1 Fortschrittsbalken

Der Fortschrittsbalken besteht aus einem schwarzen Rahmen, der als ein Rechteck mit Rahmenstärke ein Pixel und durchsichtiger Füllung umgesetzt ist (siehe Abbildung 5.1). In diesen Rahmen ist ein Rechteck gleicher Größe gesetzt, das selbst keinen Rahmen hat, aber komplett schwarz gefüllt ist. Im Gegensatz zu dem schwarzen Rahmen ist das schwarz gefüllte Rechteck dynamisch. Zu Beginn hat dieses Rechteck eine Breite von null. Mit dem Fortschritt der Zeit vergrößert sich dessen Breite linear bis der Rahmen vollständig gefüllt ist. Die Füllung spiegelt den Fortschritt der darüber liegenden Aktivität wieder.

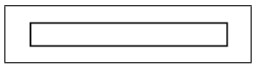

Abbildung 5.1: Rahmen mit durchsichtiger Füllung

5.2 Uhr

Die Uhr spielt eine wichtige Rolle in den Animationen: Sie stellt den Timer dar und regelt die zeitlichen Aspekte. Der in Abbildung 5.2 dargestellte Uhrrahmen dient als Basis. Alle nachfolgenden Elemente liegen über diesem Rahmen.

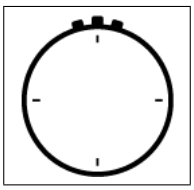

Abbildung 5.2: Rahmen der Uhr

Den Hauptteil der Uhr stellt der Dreiviertelkreis in Schwarz dar. Dieser Kreis besteht aus zwei schwarz gefüllten Halbkreisen, die sich senkrecht zueinander überlappen (vgl. Abbildung 5.3).

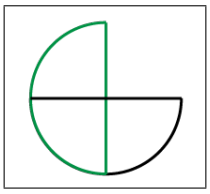

Abbildung 5.3: Dreiviertelkreis aus zwei Halbkreisen

Über diesem Dreiviertelkreis liegen passend drei Quadrate, die jeweils ein Viertel verdecken, so, dass er anfangs vollständig verdeckt ist (vgl. Abbildung 5.4). Mit Start des Timers, beginnt sich der Dreiviertelkreis im Uhrzeigersinn zu drehen.

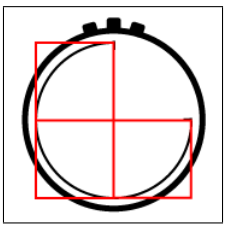

Abbildung 5.4: Quadrate, die den Dreiviertelkreis verdecken

Die Deckflächen werden jeweils nach einer viertel Umdrehung des Dreiviertelkreises nacheinander ausgeblendet. Abbildung 5.5 zeigt die Uhr, nachdem über ein Viertel der Zeit abgelaufen ist. Das Quadrat, das rechts unten auf dem Dreiviertelkreis liegt, wird ausgeblendet, um den darunter liegenden Bereich sichtbar zu machen. So werden nach und nach im Uhrzeigersinn alle drei Quadrate ausgeblendet.

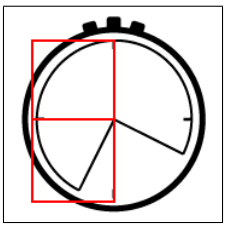

Abbildung 5.5: Über ein Viertel der Zeit schon abgelaufen

Sobald das letzte Viertel der Zeit erreicht ist, wird der schwarze Dreiviertelkreis durch einen Roten ersetzt. Der Nutzer wird somit darauf aufmerksam gemacht, dass für die Aktivität nicht mehr viel Zeit übrig ist. Das letzte Viertel wird durch den roten Dreiviertelkreis und einen roten Halbkreis dargestellt: Der Dreiviertelkreis ist statisch und der Halbkreis dreht sich, um das letzte Viertel (Abbildung 5.6) darzustellen. Der Halbkreis ist vom Dreiviertelkreis anfangs komplett verdeckt und dreht sich dann im Uhrzeigersinn, bis der Kreis vollständig gefüllt ist.

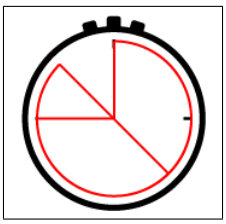

Abbildung 5.6: Letztes Viertel der Uhr

5.3 Interaktive Animationen

Durch die Interaktionen soll der Nutzer die Time Patterns besser verstehen. Die interaktive Animation geht dabei auf Nutzereingaben ein: Der Nutzer kann entscheiden, wann eine Aktivität gestartet wird. Dadurch hat er die Gelegenheit, die unterschiedlichen Fälle auszuprobieren (zum Beispiel bei einem Timeout oder das erfolgreiche Beenden der Animation). Zuerst wird die Animation so erstellt, wie bei den nicht interaktiven. Die Interaktionsmöglichkeiten werden durch Skripte (Javascript), die den Ablauf der Animation steuern, umgesetzt. Anschließend wird das „Automatische Abspielen“ der Animation deaktiviert, um dem Nutzer die vollständige Kontrolle über die Animation zu gewährleisten.

Im Folgenden werden die einzelnen Möglichkeiten der Interaktion, wie das Abspielen oder das Anhalten eines Symbols, anhand von Quellcode (in JavaScript realisiert) erklärt. Die Variable „sym“ referenziert das aktuelle Symbol. Wenn ein Skript auf der Bühne angelegt wird, referenziert „sym“ die Bühne. Ein wichtiges Steuerelement in den Interaktionen ist der Play-Button: Die Aktivität wird per Klick auf diesen Button gestartet. Das Listing 5.1 zeigt das Skript, das ausgeführt wird, wenn auf den Play-Button geklickt wird. Das Skript

wird auf der Bühne dem Klickereignis des jeweiligen Play-Buttons zugewiesen. Das Skript startet die Animation, indem „play()“ an der Variablen „sym“ (referenziert hier die Bühne) aufgerufen wird.

```
1 sym.play();
```

Listing 5.1: Aktuelles Symbol wird ausgeführt

Analog zu „sym.play();“ bietet die Programmierschnittstelle von Adobe Edge eine Funktion zum Anhalten der Symbole (vgl. Listing 5.2). Dieses Skript wird dazu verwendet, um die Animation an den Stellen, die eine Interaktion erfordern, anzuhalten (wenn die nächste Aktivität, durch einen Klick auf den Play-Button, gestartet werden kann): Beispielsweise wird die Bühne angehalten, nachdem die vorangehende Aktivität abgeschlossen wurde und das Token auf dem folgenden Pfeil erschienen ist.

```
1 sym.stop();
```

Listing 5.2: Skript hält aktuelles Symbol an

Im Falle eines Timeouts darf es keine weiteren Interaktionsmöglichkeiten mehr geben. Dafür wird der Play-Button mit der dazugehörigen Interaktionsanweisung ausgeblendet (vgl. Zeile 1 und 2 des Listings 5.3) und es erscheint die Meldung, dass die Zeit abgelaufen ist (siehe Zeile 3 in Listing 5.3).

```
1 sym.getComposition().getStage().$("Play").hide();
2 sym.getComposition().getStage().$("Text_ClickToStart").hide();
3 sym.getComposition().getStage().getSymbol("Text_TimeOver").play();
```

Listing 5.3: Timeout

Time Pattern 3 enthält, im Gegensatz zu den anderen Time Patterns, ein Ereignis, das den Timer auslöst. Dieses Ereignis ist an keine Aktivität gekoppelt. Das Ereignis tritt ein, sobald der Nutzer auf das Blitz-Symbol klickt. Dabei wird das in Listing 5.4 dargestellte Skript ausgeführt: Zunächst wird geprüft, ob der Timer bereits gestartet wurde und ob sich das Token noch vor der zweiten Aktivität befindet. Dies gilt nach 10250 Millisekunden. Wurde ein Symbol noch nicht abgespielt, so gibt „getPosition()“ den Wert -1 zurück (vgl. Zeile 1f). Gilt diese Bedingung, so reagiert der Blitz, der

Timer wird gestartet und die Interaktionsanweisung wird komplett ausgeblasst (siehe Zeile 3ff). Andernfalls geschieht nichts. „getCompostion()“ „gibt die Komposition zurück, die Eigentümer dieser Symbolinstanz ist“ [1]. „getStage()“ liefert eine Referenz auf die Bühne und mit „getSymbol()“ erhält man eine Referenz auf das jeweilige Symbol. Durch „sym.getComposition().getStage().getPosition()“ lässt sich die aktuelle Position in Millisekunden abfragen.

```
1 if (sym.getComposition().getStage().getSymbol("Uhrablauf").getPosition() == -1 &&
      sym.getComposition().getStage().getPosition() <= 10250) {
2        sym.getComposition().getStage().getSymbol("Blitz").play();
3        sym.getComposition().getStage().getSymbol("Uhrablauf").play();
4        sym.getComposition().getStage().getSymbol("Text_ClickToTrigger_Ausblassen")
             .play();
5 }
```

Listing 5.4: Skript, welches durch einen Klick auf das Blitz-Symbol ausgelöst wird

6

Zusammenfassung

Im Rahmen dieser Arbeit wurden Möglichkeiten zur Visualisierung der Time Patterns erarbeitet. Dabei wurden für die Erstellung der Animationen zunächst Skizzen auf Papier angefertigt, die nach und nach optimiert wurden. Hier wurde darauf geachtet, wie die Aufmerksamkeit des Nutzers auf das Wesentliche gelenkt werden kann und wie man sicherstellt, dass der Nutzer nicht überfordert wird. Anschließend wurden verschiedene Tools für das Erstellen von Animationen evaluiert und die Animationen schließlich mit Adobe Edge umgesetzt. Die Time Patterns 1 - 4 wurden dabei mithilfe von JavaScript interaktiv gestaltet, um einen besseren Lerneffekt zu erzielen. Sowohl für Anfänger als auch für Fortgeschrittene bieten diese Visualisierungen eine gute Hilfestellung zur Aneignung der Time Patterns. Die interaktiven Visualisierungen bieten darüber hinaus die Möglichkeit, die Time Patterns „spielerisch“ zu erlernen: Sie reagieren auf die Eingaben des Nutzers und geben ihm Feedback.

Für die Animationen wurde HTML5 verwendet, da Flash keine Zukunft mehr hat. Es ist veraltet und kommt den heutigen Anforderung an Animationen und Videos im Internet nicht mehr nach. Zwar ist Flash in der Internetwelt noch sehr weit verbreitet, wird aber in ein paar Jahren komplett durch HTML5 ersetzt werden. Auch in Hinsicht auf die Skriptsprache zur Darstellung (CSS3) hat sich einiges getan: Änderungen an der Darstellung können

einfach übernommen und wiederverwendet werden, ohne, dass der Designer diese über einen umständlichen Prozess importieren muss [4].

Aus Zeitgründen konnte im Rahmen dieser Arbeit leider nur für einen Teil der Time Patterns eine Visualisierung erarbeitet werden. Für die Zukunft wäre es daher wünschenswert auch für die anderen Time Patterns (*Time Pattern 9: Cyclic Elements* und *Time Pattern 10: Periodicity*) eine passende Visualisierung zu finden. Um möglichst viele Time Patterns zu visualisieren, wurde die Arbeit jeweils auf eine Variante beschränkt. Es wäre daher für die Zukunft wünschenswert auch die anderen Varianten, wie zum Beispiel „End-Start“ oder „End-End“ bei Time Pattern 1, zu realisieren. Für die Zukunft wären auch interaktive Versionen der bisher nur statisch animierten (nicht interaktiven) Time Patterns denkbar. Der in dieser Arbeit definierte Styleguide und die vorgestellten Grundlagen zur Visualisierung stellen dabei für die Erweiterung eine gute Grundlage dar und sollten bei möglichen Erweiterungen eingehalten werden, um eine einheitliches Aussehen der Visualisierungen sicherzustellen.

Abbildungsverzeichnis

2.1 Prozess mit wichtigen Elementen 4
2.2 Arbeitseinheit und Teilprozess 5
2.3 XOR-Gateway 6
2.4 UND-Gateway 7
2.5 ODER-Gateway 7
2.6 Spezielle Form der Ereignisse: Ausnahme 8

3.1 Kontrollelemente 12
3.2 Interaktionsmöglichkeit durch Play-Button 13
3.3 Neue Interaktionsmöglichkeit 14
3.4 Verwendete Farben 16
3.5 Kante 18
3.6 Aktivität mit Beschriftung und Fortschrittsbalken 19
3.7 Leerer Fortschrittsbalken 19
3.8 Sich füllender Fortschrittsbalken 19
3.9 Token in einer Aktivität 20
3.10 Sich füllende Uhr 20
3.11 Meldung bei Timeout 21
3.12 Deadline und Kalender 21
3.13 TP1 - Time Lags between two Activities 24
3.14 TP1 - Aktivität A wird ausgeführt 24
3.15 TP1 - Zeit läuft ab 25
3.16 TP1 - Prozess abgeschlossen 25
3.17 TP1 - Zeit ist abgelaufen 26
3.18 TP2 - Durations 27
3.19 TP2 - Aktivität A wird gestartet 27
3.20 TP2 - Aktivität A in Bearbeitung 28
3.21 TP2 - Prozess abgeschlossen 28
3.22 TP3 - Time Lags between Arbitrary Events 29

3.23 TP3 - Blitzsymbol löst Timer aus 30
3.24 TP4 - Fixed Date Elements 31
3.25 TP4 - Deadline wird festgesetzt 31
3.26 TP4 - Tag 3 wird durchgestrichen 32
3.27 TP5 - Duration 33
3.28 TP5 - Kunde kann Laden betreten 33
3.29 TP5 - Aktivität „Go Shopping“ nach Betreten des Ladens 34
3.30 TP6 - Time-based Restrictions 35
3.31 TP6 - Anzahl der Küchenstücke 35
3.32 TP7 - Validity Period 36
3.33 TP7 - Neuer Prozess 37
3.35 TP8 - Best-Before Date erscheint 37
3.34 TP8 - Time-dependent Variability 38
3.36 TP8 - Alternativpfad wird deaktiviert 38

4.1 Adobe Edge Animate 1.0 43
4.2 Sencha Animator 1.3 44
4.3 Tumult Hype 1.5 45

5.1 Rahmen mit durchsichtiger Füllung 48
5.2 Rahmen der Uhr 48
5.3 Dreiviertelkreis aus zwei Halbkreisen 49
5.4 Quadrate, die den Dreiviertelkreis verdecken 49
5.5 Über ein Viertel der Zeit schon abgelaufen 49
5.6 Letztes Viertel der Uhr 50

Listings

5.1 Aktuelles Symbol wird ausgeführt 51
5.2 Skript hält aktuelles Symbol an 51
5.3 Timeout 51
5.4 Skript, welches durch einen Klick auf das Blitz-Symbol ausgelöst wird 52

Tabellenverzeichnis

2.1 Übersicht der Time Patterns 9

4.1 Vergleich der Software 46

Literaturverzeichnis

[1] ADOBE SYSTEMS: *Adobe Edge Animate JavaScript API - Version 1.0.0*. http://www.adobe.com/devnet-docs/edgeanimate/api/current/index.html. – Zuletzt abgerufen am 11.10.2012

[2] ADOBE SYSTEMS INCORPORATED: *Adobe Edge Animate*. http://html.adobe.com/edge/animate/faq.html. – Zuletzt abgerufen am 05.11.2012

[3] ALLWEYER, Thomas: *BPMN 2.0 - Business Process Model and Notation: Einführung in den Standard für die Geschäftsprozessmodellierung*. 2., aktual. u. erw. Aufl. Norderstedt : Books on Demand, 2009. – ISBN 978–3–8391–2134–4

[4] ANTHES, Gary: HTML5 Leads a Web Revolution. In: *Commun. ACM* 55 (2012), Juli, Nr. 7, 16–17. http://doi.acm.org/10.1145/2209249.2209256. – ISSN 0001–0782

[5] JOBS, Steve: *Thoughts on Flash*. http://www.apple.com/hotnews/thoughts-on-flash/. – Zuletzt abgerufen am 09.11.2012

[6] KESTEREN, Anne van ; STACHOWIAK, Maciej: *HTML Design Principles*. http://www.w3.org/TR/html-design-principles/. – Zuletzt abgerufen am 30.10.2012

[7] LANZ, Andreas ; WEBER, Barbara ; REICHERT, Manfred: Workflow Time Patterns for Process-Aware Information Systems. In: *BMMDS/EMMSAD*, 2010

[8] SENCHA INC.: *Sencha Touch*. http://www.sencha.com/products/. – Zuletzt abgerufen am 09.11.2012

[9] THOMA, Jörg: *Adobe entfernt Flash Player aus dem Play Store*. http://www.golem.de/news/android-adobe-entfernt-flash-player-aus-dem-play-store-1208-93872.html. – Zuletzt abgerufen am 02.11.2012

[10] TUMULT INC.: *Tumult Hype*. http://tumult.com/hype. – Zuletzt abgerufen am 07.11.2012

[11] TUMULT INC.: *Tumult Hype. Tumult Hype Documentation — Overview*. `http://tumult.com/hype/documentation/overview/#userinterface`. – Zuletzt abgerufen am 07.11.2012

[12] WORKFLOW PATTERNS INITIATIVE: *Workflow Patterns*. `http://www.workflowpatterns.com`. – Zuletzt abgerufen am 23.09.2012

[13] WÄGER, Markus: *Grafik und Gestaltung. Das umfassende Handbuch*. Bonn : Galileo Press, 2010. – ISBN 978–3–8362–1206–9

Name: Zenib Awan Matrikelnummer: 692283

Erklärung

Ich erkläre, dass ich die Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Ulm, den .....

Zenib Awan