Universität Potsdam

Institut für Informatik und
Computational Science

Simulation des Verhaltens von kontextsensitiven Anwendungen

Dem Lehrstuhl für
Komplexe Multimediale Anwendungsarchitekturen
am Institut für Informatik und Computational Science
der Universität Potsdam
zur Erlangung des akademischen Grades eines

Master of Science

vorgelegte Masterarbeit von

Frau Helena Jank, B.Sc.

aus Berlin

Universität Potsdam

Institut für Informatik und Computational Science

Masterarbeit am Lehrstuhl für
Komplexe Multimediale Anwendungsarchitekturen

Simulation des Verhaltens von kontextsensitiven Anwendungen

Kandidatin:	Helena Jank, B.Sc.
Gutachterin:	Prof. Dr.-Ing. habil. Ulrike Lucke
Gutachter:	Dr. Christoph Rensing
Abgabedatum:	9. Mai 2016

Ich versichere, dass ich die von mir vorgelegte Arbeit selbstständig verfasst habe, dass ich die verwendeten Quellen, Internet-Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, in jedem Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Berlin, den 9. Mai 2016

.....
(Unterschrift der Kandidatin)

Zusammenfassung

In dieser Arbeit wird die Konzipierung und Umsetzung eines Simulators für kontextsensitive, adaptive Lernanwendungen thematisiert. Solche Anwendungen können über Sensoren und andere Schnittstellen den Kontext ihrer Benutzer, der Lernenden, erfassen. Sie können erkennen, in welcher Situation der Lernende sich befindet, und dazu passende Lerneinheiten auswählen. Dadurch soll der Lernende mehr Unterstützung im individuellen Lernprozess erfahren.

Um auch Lehrkräften ohne Programmierkenntnisse den Zugang zu solchen adaptiven Anwendungen zu eröffnen, wurde ein prototypisches Autorensystem als Teil eines Frameworks entwickelt, das deren Erstellung auf einfache Weise ermöglichen soll. Diese Arbeit widmet sich der Entwicklung einer Simulationskomponente für ebendieses Autorensystem. Damit soll das Verhalten der erstellten Anwendung getestet und überprüft werden können, ob in jeder Situation die richtige Lerneinheit ausgewählt wird. Die Autoren der Lernanwendung sollen dadurch ein besseres Verständnis der Anwendungslogik und mehr Vertrauen in ihre Anwendung erhalten. Der Schwerpunkt dieser Arbeit liegt auf der Konzipierung einer intuitiv bedienbaren Benutzerschnittstelle, die für die künftigen Benutzer schnell und einfach erlernbar sein sollte.

Dazu wurde ein grobes Konzept in Form eines Paper-Prototyps erstellt und in einer ersten Nutzerstudie getestet. Anhand der Ergebnisse wurde das Konzept überarbeitet, verfeinert und als ein digitaler, lauffähiger Prototyp implementiert. Dieser wurde in einer zweiten, umfangreicheren Nutzerstudie erneut getestet und das Konzept evaluiert. Dabei konnten größere Probleme beim Verständnis und bei der Bedienung aufgedeckt und weitere funktionale Anforderungen ermittelt werden.

In dieser Arbeit werden die Vorgehensweise und die Ergebnisse sowohl der Konzipierung als auch der Umsetzung präsentiert.

Inhaltsverzeichnis

1 Motivation **1**
- 1.1 Mobiles adaptives E-Learning . . . 1
- 1.2 Simulation des adaptiven Verhaltens . . . 2
- 1.3 Fragestellung . . . 3
- 1.4 Aufbau dieser Arbeit . . . 4

2 Grundlagen **5**
- 2.1 Autorensystem für adaptive Anwendungen . . . 5
- 2.2 Begrifflichkeiten . . . 6
  - 2.2.1 Lerneinheit . . . 7
  - 2.2.2 Kontextsensitivität und Adaptivität . . . 7
  - 2.2.3 Kontext . . . 8
  - 2.2.4 Situation . . . 9
  - 2.2.5 Szenario . . . 10
- 2.3 Technologische Grundlagen . . . 10
  - 2.3.1 MOTIVATE-Framework . . . 10
  - 2.3.2 Autorensystem und Regelgenerator . . . 11
  - 2.3.3 Adaptation Engine . . . 13
  - 2.3.4 ContactJS . . . 14
  - 2.3.5 Prototypisches User-Interface . . . 15

3 Verwandte Arbeiten **17**
- 3.1 Automatisches Testen . . . 17
- 3.2 Manuelles Testen . . . 18

4 Konzipierung **21**
- 4.1 Anforderungen . . . 21
  - 4.1.1 Funktionale Anforderungen . . . 22
  - 4.1.2 Nicht-funktionale Anforderungen . . . 22
- 4.2 Methodik . . . 23
  - 4.2.1 Machbarkeitsanalyse . . . 23
  - 4.2.2 Wahl der Methode: Prototyping . . . 24
- 4.3 Use-Cases . . . 28

4.4 Designideen 28
4.4.1 Metaphern 28
4.4.2 Bedien- und Visualisierungskonzepte 32
4.4.3 Progressive Disclosure 33
4.5 Erste Nutzerstudie: Paper-Prototyping 33
4.5.1 Vorgehen 34
4.5.2 Ergebnisse 38
4.6 Konzept für das User-Interface 42
4.6.1 Unterstützte Use-Cases 42
4.6.2 UI-Komponenten und Funktionen 44

5 Implementierung **51**
5.1 Architektur 51
5.1.1 Klassen 52
5.1.2 Schnittstellen 54
5.1.3 Verwendete Bibliotheken 56
5.2 Benutzerschnittstelle 56

6 Evaluierung **65**
6.1 Vorgehen 65
6.1.1 Benutzerprofile 66
6.1.2 Ziele der Studie 66
6.1.3 Rekrutierung 68
6.1.4 Szenarien und Aufgaben 68
6.1.5 Ablauf 69
6.2 Ergebnisse 72
6.2.1 Kritische Vorbetrachtungen 72
6.2.2 Diskussion der Ergebnisse 75

7 Fazit **83**
7.1 Zusammenfassung 83
7.2 Ausblick 85

Literaturverzeichnis **XI**

Web-Quellenverzeichnis **XVII**

Appendices **XIX**
1 Anhang: Ergebnisse der 1. Studie XXI
1.1 Paper-Prototyping: Zusammenfassung der Ergebnisse . XXI
1.2 Paper-Prototyping: Vollständige Ergebnisse XXVI

2 Ergebnisse der 2. Studie XXXVI
2.1 Ergebnisse der Evaluation XXXVII
2.2 Ergebnisse der System Usability Scale (SUS) LXXVIII

1 Motivation

Im Jahr 2016 sind leistungsstarke „intelligente“ Mobilgeräte wie Smartphones oder Tablets in den Alltag eingezogen. Nicht nur die Verbreitung dieser Geräte nimmt stetig zu, sondern auch ihr Funktionsumfang. Moderne Smartphones sind in der Regel mit GPS, Kamera, Mikrophon, mobilem Internetzugang, Kontaktverzeichnis, Kalender und vielem anderen mehr ausgestattet. Derartige Hardware- und Software-Komponenten erlauben es den auf den Geräten laufenden Anwendungen (Applications oder Apps), eine Vielfalt an Informationen über ihre Benutzer (User) zu sammeln und zu erkennen, in welchen Situationen diese sich während der Nutzung befinden. Die Gesamtheit all dieser Informationen, die eine Situation, den darin befindlichen Nutzer und die Anwendung selbst beschreiben, wird als Kontext bezeichnet [Dey01].

Zum Beispiel ist es möglich, den Standort des Users und die aktuelle Uhrzeit zu ermitteln. Daraus lässt sich dann schlussfolgern, ob der User einen Termin, der aus dessen Kalender-App abgerufen wird, einhält oder verpasst. So können sich mobile Apps an die jeweilige Situation anpassen und, um beim Beispiel zu bleiben, eine Erinnerung auslösen oder unterdrücken. Weitere mögliche Ausprägungen solcher situations- oder kontextabhängigen Anpassungen sind das Filtern von Inhalten wie Texten oder Videos nach Interessen des Users oder Hinweise auf interessante Objekte in der Umgebung.

Es gibt verschiedene mögliche Einsatzbereiche für solche Anwendungen, die sich an ihren Nutzungskontext anpassen können. Einer davon ist die Unterstützung von mobilem Lernen.

1.1 Mobiles adaptives E-Learning

Mobiles Lernen ist zeitlich und örtlich ungebunden und wird ganz besonders durch den Einsatz persönlicher Endgeräte ermöglicht und vorangetrieben. Lernende können dank ihrer Smartphones und mobilen Internetzugangs jederzeit und überall lernen - ob im Bus zur Arbeit, beim Einkaufen, im Bett vor dem Einschlafen oder während der Wartezeit beim Arztbesuch. Sie können selbst kleinste Zeitlücken nutzen, um Lerninhalte abzurufen und zu wiederholen, sich zu interessanten Themen zu informieren, Fragen zu beantworten, Gedanken festzuhalten und sich mit anderen Lernenden auszutauschen. Mit dieser Vielzahl an Möglichkeiten geht allerdings auch

eine potenzielle Überforderung der Lernenden einher. Denn mobiles Lernen erfordert ein hohes Maß an Selbstlernkompetenz, also der Fähigkeit, den eigenen Lernprozess zu organisieren, zu steuern und darüber zu reflektieren, um die anvisierten Lernziele zu erreichen [MJZL14].

Hier können Anwendungen, die sich an den Kontext des Lernenden anpassen, unterstützend wirken, indem sie den Lernfortschritt und die Lernziele des Nutzers erfassen und den Lernstoff darauf zuschneiden. Auch können Informationen über den Lernstil des Nutzers verwendet werden, um die Auswahl der Lerninhalte an die persönlichen Stärken und Vorlieben anzupassen. So unterscheiden sich Lernende darin, ob sie „global“ oder „sequenziell“ lernen, also in der Breite lernen und im Stoff springen oder einem linear vorgegebenen Lernpfad folgen [FS88]; ob sie besser anhand von Bildern und Videos oder über die auditive Wahrnehmung lernen [FS88]; ob sich ihre Leistungsfähigkeit morgens oder abends entfaltet [RWJM03] - um nur einige Beispiele zu nennen.

Zusätzlich zu diesem persönlichen Kontext, also Informationen über den Lernenden, können auch alle Informationen über dessen Umwelt und die jeweils aktuelle Situation einbezogen werden, um das Lernerlebnis daran anzupassen und den Lernenden weiter zu unterstützen. Wie im eingangs beschriebenen Beispiel können Termine und geplante Aufgaben berücksichtigt werden, um den Lernenden bei der Selbstorganisation zu unterstützen. Außerdem können Standortinformationen genutzt werden, um passende Lerninhalte zu nahen Objekten bereitzustellen oder Lernende zu vernetzen. Umgebungsgeräusche, Helligkeit, Tageszeit, Signalstärke der Internetverbindung, Fortbewegungsgeschwindigkeit sind Beispiele für Informationen, die etwa in die Auswahl passender Formate für die Präsentation der Lerninhalte verwandelt werden können. Die Möglichkeiten sind vielfältig.

1.2 Simulation des adaptiven Verhaltens

Das Testen von Anwendungen spielt in der Software-Entwicklung eine große Rolle und ist auch für alle Nicht-Programmierer essenziell, die ihre Anwendung überprüfen und fehlerhaftes Verhalten frühzeitig erkennen und beheben wollen.

Deshalb sollte der Benutzer eines Autorensystems für adaptive Anwendungen die Möglichkeit haben, deren Adaptierungsverhalten zu testen, ohne sie erst auf ein Mobilgerät spielen und sich in die vorgesehenen Situationen begeben zu müssen. Denn die App in allen möglichen Situationen in der Realität zu testen, wäre zumindest mit einem immensen Aufwand verbunden, wenn nicht sogar unmöglich. Das lässt sich leicht am Beispiel einer Lernein-

heit zeigen, die im Brandfall in einem bestimmten Gebäude einen sicheren Fluchtweg anzeigen soll.

Alternativ kann auf das Mittel der Simulation zurückgegriffen werden: Kontextinformationen müssen nicht real gemessen werden, sondern können in einem Simulator fingiert werden. Neben dem Testen auf fehlerhaftes Verhalten bietet eine Simulation so auch den Vorteil, dass die Logik der Adaptierung visualisiert werden kann, und zwar kontrolliert durch den Anwender. Dieser kann verschiedene Kontextparameter einstellen, verändern und kombinieren und sich anschauen, wie die adaptive Anwendung jeweils reagiert. So können Rückschlüsse auf das zugrundeliegende Modell für die Adaptierung gezogen werden [Rey09]. Besonders unerfahrene Autoren haben dadurch eine Möglichkeit, den Zusammenhang von Lerneinheiten, deren Relationen untereinander und den verschiedenen Kontextbedingungen sukzessive zu begreifen.

Um die Autoren bei diesem Prozess zu unterstützen und sie nicht zusätzlich kognitiv zu belasten, sollte der Simulator einfach und intuitiv, ohne lange Einarbeitung zu bedienen und leicht verständlich sein. Außerdem sollte er durch einen einfachen, spielerischen Umgang die intrinsische Motivation, sich mit dem Autorensystem auseinanderzusetzen, ansprechen, bewahren und bestenfalls fördern.

1.3 Fragestellung

Diese Arbeit hat zum Thema, einen solchen Simulator als Teil des Autorensystem-Prototyps zu konzipieren und prototypisch umzusetzen. Dabei werden die folgenden Forschungsfragen adressiert:

1. Welche funktionalen und nicht-funktionalen Anforderungen stellen sich aus Nutzerperspektive an einen Simulator für kontextsensitive, adaptive Lernanwendungen, der ein Testen und Verstehen der Anwendungslogik ermöglichen soll?

2. Wie muss das User Interface (die Benutzungsschnittstelle) gestaltet sein, um eine hohe *Usability* [1] bei der Arbeit mit dem Simulator zu gewährleisten?

[1] Die DIN-Norm EN ISO 9241, Teil 11 definiert Usability (Gebrauchstauglichkeit) als „[...] das Ausmaß, in dem ein Produkt durch bestimmte Benutzer in einem bestimmten Nutzungskontext genutzt werden kann, um bestimmte Ziele effektiv, effizient und zufriedenstellend zu erreichen.“ [ISO98]. In der Literatur finden sich verschiedene weitere Definitionen von Usability [RF13], die Kriterien wie Nützlichkeit und Erlernbarkeit einschließen [Kru13], [Bar10].

Um Antworten auf diese Fragen zu finden, soll in dieser Arbeit das entworfene Konzept prototypisch implementiert und mithilfe potenzieller Endanwender getestet werden.

1.4 Aufbau dieser Arbeit

Im Folgenden werden zunächst in Kapitel 2 die Grundlagen für diese Arbeit erörtert. Dazu werden die Ziele und Ergebnisse des Projekts vorgestellt, in dem diese Arbeit entstanden ist, und die relevante Terminologie erklärt. Anschließend wird in Kapitel 3 ein Blick auf das Forschungsumfeld geworfen, in dem diese Arbeit verortet und gegen verwandte Arbeiten abgegrenzt wird.

In Kapitel 4 werden Konzipierung und Vorgehensweise dieser Arbeit erläutert. Dazu gehört eine erste Nutzerstudie in Form eines Paper-Prototypings, die vorgestellt und ausgewertet wird. Basierend auf den Ergebnissen dieser Studie wurde ein erstes User-Interface-Konzept erstellt und als digitaler Prototyp implementiert. In den Kapiteln 5 bis 6 folgt die Beschreibung und Diskussion der praktischen Umsetzung: Die Implementierung des User-Interface wird in Kapitel 5 vorgestellt. Kapitel 6 schildert die Vorbereitung, Durchführung und Auswertung einer abschließenden Nutzerstudie, in welcher der digitale Prototyp des Simulators getestet wurde. Die Arbeit schließt mit einem Fazit und Ausblick in Kapitel 7.

2 Grundlagen

In diesem Kapitel werden die Grundlagen für diese Arbeit erläutert. Zunächst wird kurz das Projekt MOTIVATE vorgestellt, in dessen Rahmen der Grundstein für diese Arbeit gelegt wurde. Danach wird auf die verwendeten Begrifflichkeiten eingegangen und deren Bedeutung im Kontext dieser Arbeit definiert. Anschließend werden Funktionsweise und Aufbau des MOTIVATE-Frameworks sowie der Prototyp eines Autorensystems für dieses Framework vorgestellt.

2.1 Autorensystem für adaptive Anwendungen

Um die Entwicklung kontextsensitiver, adaptiver Lernanwendungen voranzutreiben, wurde im Jahr 2013 das Projekt MOTIVATE (Mobile Training Via Adaptive Technologies)[1] ins Leben gerufen. Dessen Ziel war es, ein Framework zu entwickeln, mit dem Lehrkräfte aus den Bereichen Hochschule und Wirtschaft auf einfache Weise adaptive Lernanwendungen erstellen können, ohne dafür über Programmierkenntnisse verfügen zu müssen.[2]

Der Fokus lag dabei auf Mikrolernanwendungen. Das sind Anwendungen, die dadurch gekennzeichnet sind, dass Lerninhalte in Form atomarer Mikrolerneinheiten bereitgestellt werden. Mikrolernen (oder Microlearning) hat insbesondere in der betrieblichen Personalbildung in den letzten Jahren massiv an Bedeutung gewonnen [BMF12]. Sein Vorteil liegt darin, dass Wissen in Form kleinster „Häppchen“ durch eine Kombination von Wiederholung und unmittelbarem Feedback vermittelt und gefestigt wird. Einen großen Mehrwert bietet hier zusätzliche Kontextsensitivität: Lerneinheiten können dann nicht mehr nur zufällig ausgewählt und dem Lernenden zur Wiederholung vorgeschlagen werden, sondern angepasst an den Lernfortschritt des Lernenden und in Situationen, in denen sie besonders nützlich sind.

Damit sich eine Software an den erfassten Kontext anpassen kann, benötigt sie Regeln. Ein einfaches Beispiel wäre:

[1] `http://www.motivate-project.de/` [Abgerufen am 05.04.2016]

[2] An dieser Stelle sei angemerkt, dass das Framework zwar dem Einsatz im Bereich E-Learning dediziert ist, jedoch für die Erstellung jedweder kontextsensitiver Anwendungen geeignet ist. Denn die einzelnen Bestandteile sind an keinen Zweck gebunden.

```
WENN: der Geräuschpegel Zimmerlautstärke übersteigt,
DANN: spiele keine Audiodateien ab.
```

Komplexer wird es, wenn eine zweite Bedingung hinzukommt:

```
WENN: der Geräuschpegel Zimmerlautstärke übersteigt
 ODER keine Kopfhörer eingesteckt sind,
DANN: spiele keine Audiodateien ab.
```

Diese Regel lässt sich so umformen, dass daraus eine Handlung abgeleitet werden kann:

```
WENN NICHT: der Geräuschpegel Zimmerlautstärke übersteigt
 ODER keine Kopfhörer eingesteckt sind,
DANN: spiele Audiodateien ab.
```

Die Logik für die Anpassung der Anwendung wird also umso weiter verfeinert, je mehr Bedingungen die Regeln beinhalten. Letztendlich gibt es nicht nur Regeln für die Auswahl verschiedener Medienformate, wie im Beispiel, sondern auch für jede einzelne Lerneinheit eine eigene Regel, die bestimmt, in welcher Situation, also unter welchen Bedingungen diese Lerneinheit ausgewählt wird. Dieses Regelwerk muss zunächst durch einen „echten Menschen“ erstellt werden, da noch nicht automatisch erkannt werden kann, zu welchen Situationen eine Lerneinheit passt oder nicht. Deshalb wurde der Prototyp eines Autorensystems entwickelt, das diese Arbeit handhabbar machen soll: Der Autor muss sich darin nicht mit der Formulierung von abstrakten Regeln auseinandersetzen, sondern kann nach dem Baukastenprinzip jeder Lerneinheit die Bedingungen für ihre Auswahl in Form von Kontextbausteinen zuordnen. Durch diese Plastizität soll auch Lehrkräften und Autoren von E-Learning-Content, die kein tieferes Verständnis von mathematischer Logik haben, ermöglicht werden, Schritt für Schritt ein komplexes Regelwerk zu erstellen.

2.2 Begrifflichkeiten

Im MOTIVATE-Framework und in dieser Arbeit spielen die Begriffe Lerneinheit, Kontext, Situation und (Lern-)Szenario eine zentrale Rolle. Sie werden daher im Folgenden näher erläutert. Ferner wird auf die Begriffe Kontextsensitivität, Adaptierung und Adaptivität eingegangen.

2.2.1 Lerneinheit

Da sich diese Arbeit im Bereich E-Learning, genauer: Mobiles Lernen bewegt, wird im weiteren Verlauf dieser Arbeit von Lernanwendungen ausgegangen, die auf mobilen Geräten laufen. Lernanwendungen, die mit dem MOTIVATE-Framework erstellt werden, präsentieren ihren Nutzern die zu lernenden Inhalte in Form von Lerneinheiten. Das sind Mikrolerneinheiten wie Bilder, Podcasts, kurze Texte, Videos oder Quizzes, die in der Regel atomar und abgeschlossen sind. Diese Eigenschaften kommen vor allem den vielen Ablenkungen und der tendenziell kurzen Aufmerksamkeitsspanne des Lernenden beim mobilen Lernen entgegen [KH05]. Eine MOTIVATE-Lerneinheit beinhaltet neben dem Lerninhalt und dem Medienformat für die Präsentation auch jene Metadaten, die für die Adaptierung relevant sind. Das sind all die Kontextinformationen, die die Situationen beschreiben, in welcher die Lerneinheit konsumiert werden soll und folglich durch die Anwendung ausgewählt wird.

2.2.2 Kontextsensitivität und Adaptivität

Die Fähigkeit einer Anwendung, ihren Kontext wahrzunehmen, um eine Situation zu erkennen und darauf zu reagieren, wird als Kontextsensitivität bezeichnet [SAW94]. Adaptivität ist die damit einhergehende Fähigkeit der Anwendung, sich automatisch an die wahrgenommene Situation anzupassen. Demgegenüber steht der Begriff der Adaptierbarkeit: Eine Anwendung ist adaptierbar, wenn sie eine Anpassung zulässt, die zumeist manuell durch den Nutzer erfolgt [HM04].

Diese Anpassung (oder Adaptierung) geschieht in Form der Auswahl passender Lerninhalte oder Formate [HM04]. Passend heißt in diesem Zusammenhang, dass etwa Thema, Vertiefungs- und Schwierigkeitsgrad auf die Interessen und Bedürfnisse des Lernenden abgestimmt sind und die Form der Präsentation für die Lernsituation geeignet ist. Beispielsweise wäre ein Text besser als ein Podcast oder Video geeignet, wenn der Lernende sich in der Bibliothek befindet und keine Kopfhörer parat hat.

Im E-Learning wird Adaptivität oft mit Intelligenten Tutoriellen Systemen in Verbindung gebracht, die sich an den persönlichen lernspezifischen Kontext anpassen [Rey09][HM04]. Für die Auswahl geeigneter Medienformate wird zusätzlich der technische Kontext berücksichtigt [HM04]. Innerhalb des MOTIVATE-Frameworks ist der Begriff jedoch weiter gefasst und auf keine Kontextart beschränkt, wie in Kapitel 1 und im Folgenden beschrieben.

2.2.3 Kontext

Im MOTIVATE-Projekt und -Framework findet die Definition von Kontext nach [Dey01] Anwendung:

> "Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves."[Dey01]

Kontextinformationen beschreiben also den Endanwender der mobilen Lernanwendung sowie alles in seiner Umwelt, was relevant ist, um die aktuelle Situation zu erkennen, einschließlich der Anwendung selbst. Die Gesamtheit aller messbaren oder anderweitig erfassbaren Kontextinformationen wird wie folgt klassifiziert [MJZL14]:

- Physikalischer Kontext: umfasst Wetter, Umgebungsgeräusche, Lichtstärke, Zeit, Luftfeuchtigkeit, Temperatur usw.
- Mobil bzw. Ortung: umfasst Adresse, Entfernung zu einem Ort, Transportmittel, Geschwindigkeit, Ziel usw.
- Situativer Kontext: umfasst Termine, Aufgaben, Aktivitäten, aktuelle Blickrichtung usw.
- Lernszenario: umfasst Lernfortschritt, aktuell bearbeitete Aufgaben, benötigte Zeit für die Bearbeitung usw.
- Persönlicher Kontext: umfasst Vorwissen, Lernstil, Motivation, Emotionen, Kontakte usw.
- Technischer Kontext bzw. Infrastruktur: umfasst Ein- und Ausgabegeräte, Internetzugang, Akkustand, uvm.

Die einzelnen aufgelisteten Informationen sind dabei nur als Beispiele zu verstehen. Das Framework ist nicht auf eine bestimmte Auswahl von Kontextinformationen beschränkt, sondern generisch angelegt. Denn zum einen ist abzusehen, dass mit dem technischen Fortschritt in Zukunft noch mehr Kontextquellen erschlossen werden, die sich zur feineren Anpassung der Anwendung an den Lernenden nutzen lassen. Beispiele dafür sind die Erkennung von Emotionen und Intentionen. Zum anderen ist noch nicht geklärt, welche dieser Informationen in welchen Lernszenarien tatsächlich relevant und nützlich sind oder in Zukunft sein werden [MJZL14].

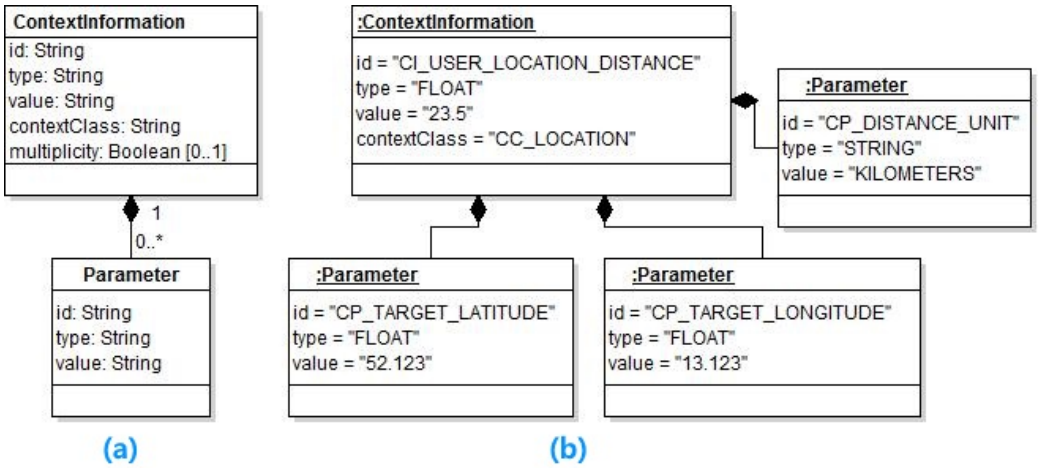


Abbildung 2.1: **(a)** Klassenmodell von Kontextinformation und Parameter. **(b)** Objektmodell einer Beispiel-Kontextinformation.

Für die Modellierung einer Kontextinformation wird im MOTIVATE-Framework ein objektorientierter Ansatz verfolgt. Eine Kontextinformation ist ein Objekt mit den Eigenschaften: eindeutiger Bezeichner, Datentyp, Wert und Kontextklasse. Optional können beliebig viele Parameter hinzukommen und die Information, ob diese Kontextinformation gleichzeitig mehrere Werte annehmen kann. Durch Parameter wird eine komplexere Information beschrieben. Ein Parameter ist selbst wiederum ein Objekt mit den Eigenschaften Bezeichner, Datentyp und Wert.

Abbildung 2.1 zeigt das Modell einer Kontextinformation, illustriert am Beispiel einer komplexen Kontextinformation.

2.2.4 Situation

Wie oben angegeben, wird eine Situation durch eine Menge von Kontextinformationen beschrieben. Somit lässt sich eine Situation wiederum, etwas abstrahiert, definieren als die Konjunktion dieser Kontextinformationen.

Im MOTIVATE-Framework wird diese Definition angewandt. Dadurch wird eine Situation inhärent als eine Art Momentaufnahme verstanden: Sie gilt genau so lange wie ihre Kontextinformationen.[3]

[3]Hier liegt ein feiner semantischer Unterschied zum eher umgangssprachlichen Gebrauch des Begriffs „Situation“, worunter „Verhältnisse, Umstände, die einen allgemeinen Zustand kennzeichnen“ oder eine „allgemeine Lage“ [Red16a] verstanden werden.

2.2.5 Szenario

Ein (Lern-/Lehr-)Szenario ist, ganz allgemein, eine „Beschreibung, [ein] Entwurf, [ein] Modell der Abfolge von möglichen Ereignissen oder der hypothetischen Durchführung einer Sache“ [Red16b] mit Lehr- bzw. Lernbezug. Die „Sache“ ist in diesem Fall das Lehren bzw. Lernen[4] - in bestimmten Situationen, mit bestimmten Inhalten zu bestimmten Themen. Da diese Definition bewusst allgemein gehalten ist, seien einige Beispiele für Lernszenarien genannt: der Besuch einer Vorlesung, das nachmittägliche Lösen von Übungsaufgaben in der Gruppe, das Wiederholen von Vorlesungsinhalten während der Bahnfahrt, ein Museumsbesuch oder eine Exkursion. Im MOTIVATE-Kontext lässt sich ein Szenario aus technischer Sicht betrachten als die Summe einer Menge von Lerneinheiten (für dieses Szenario) und des zu erfassenden Kontexts. Wobei die in einer Situation verfügbaren oder bearbeiteten Lerneinheiten selbst wiederum Teil des Kontexts (der Klasse „Lernszenario“, siehe oben) sind - in der hier angewandten weiten Definition. Am Beispiel des Gruppenlernens würden also die Lerneinheiten den Übungsaufgaben entsprechen. Der (restliche) Kontext würde Informationen wie Uhrzeit, anwesende Kommilitonen, Fortschritt, Thema der Übung beinhalten.

2.3 Technologische Grundlagen

Nachdem die für das weitere Verständnis wichtigsten Begriffe erläutert wurden, soll im Folgenden das MOTIVATE-Framework vorgestellt werden: seine Funktion, seine Komponenten und deren Zusammenspiel.[5] Im Zuge dessen werden zudem die etwas technischeren Konzepte Regelgenerator, Contraints, Adaptation Engine, Rule Engine und Kontexterfassung eingeführt.

2.3.1 MOTIVATE-Framework

Abbildung 2.2 gibt einen Überblick über den Workflow im MOTIVATE-Framework, dessen ausgewählte Aspekte im Folgenden näher beleuchtet werden.

Das Framework lässt sich grob untergliedern in zwei Hauptbestandteile: Den einen bilden auf der einen Seite das bereits erwähnte Autorensystem

[4] Im weiteren Verlauf dieser Arbeit wird sich auf das Morphem „Lern-“ beschränkt werden. Zum einen der Einfachheit halber, zum anderen aber auch, weil im Projekt MOTIVATE der Lernende stets im Mittelpunkt steht, an dessen Bedürfnisse das Lernerlebnis angepasst werden soll.

[5] Dieses Kapitel gibt nur eine kurze Zusammenfassung der Bestandteile des Frameworks, die für diese Arbeit besonders relevant sind (Details zu den einzelnen Komponenten finden sich in: [MJZL14] und [Lem14]).

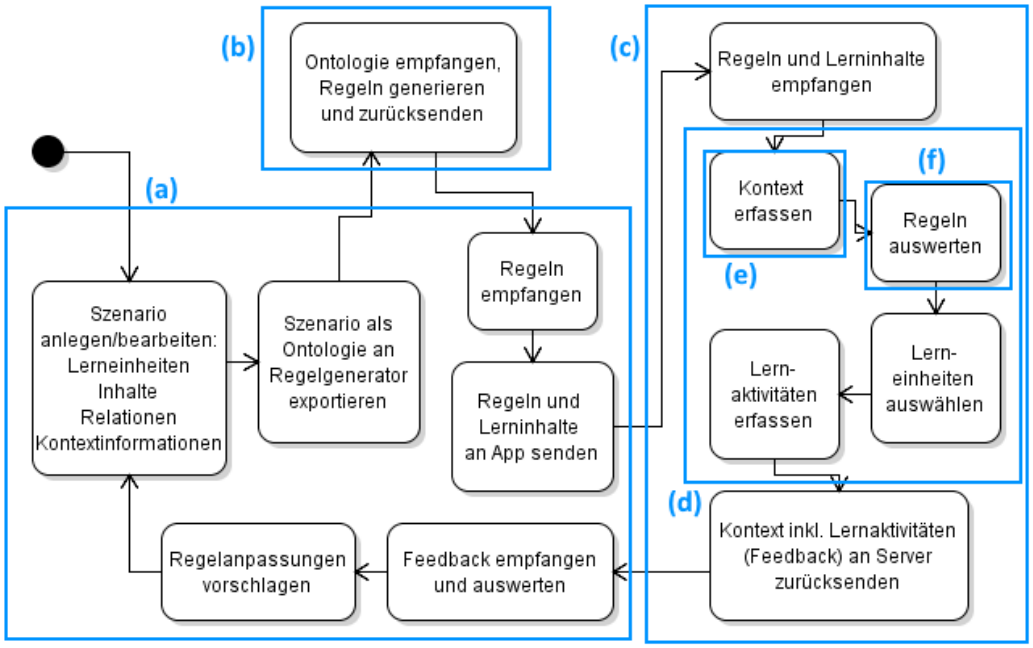


Abbildung 2.2: Workflow im MOTIVATE-Framework: Links die Serverseite mit Autorensystem (a) und Regelgenerator (b). Rechts die mobile App (c) mit Adaptation Engine (d) einschließlich Kontexterfassung (e) und Rule Engine (f).

in Kombination mit einem Regelgenerator. Auf der anderen Seite finden auf dem mobilen Gerät die Auswertung der erzeugten Logik und die Adaptierung der Anwendung statt.

2.3.2 Autorensystem und Regelgenerator

Das Autorensystem ist eine Desktop-Anwendung, die auf Webtechnologien basiert, das heißt in HTML, CSS und JavaScript implementiert ist. Diese Sprachen ermöglichen ein relativ einfaches, schnelles Entwickeln erster User-Interface-Prototypen durch die Entkopplung von Funktion, Daten und Darstellung. Gleichzeitig kann damit der volle Funktionsumfang implementiert und von den meisten Browsern interpretiert werden.

Im Autorensystem legt ein Autor Mikrolerneinheiten für bestimmte Szenarien an und definiert die Adaptierungslogik. Jede Lerneinheit wird mit Inhalt gefüllt und mit Kontextinformationen attribuiert. Diese Kontextinformationen bilden die Situation ab, in der die Anwendung diese Einheit auswählen soll. Zudem werden Lerneinheiten in Relation zueinander gesetzt. Sie können

einem Lernpfad folgend aufeinander aufbauen, andere Lerneinheiten erweitern, näher erläutern oder gleichwertige Alternativen voneinander sein.

Am Beispiel der Übungsgruppe, die sich immer montags um 14 Uhr in Raum III.H1.S16 trifft, könnte so eine Lerneinheit namens „Übung 2“ wie folgt angelegt sein:

- Die Lerneinheit „Übung 1“ muss bereits abgeschlossen sein.
- Der Lernende befindet sich in Raum III.H1.S16.
- Die aktuelle Uhrzeit ist 14 Uhr oder später.

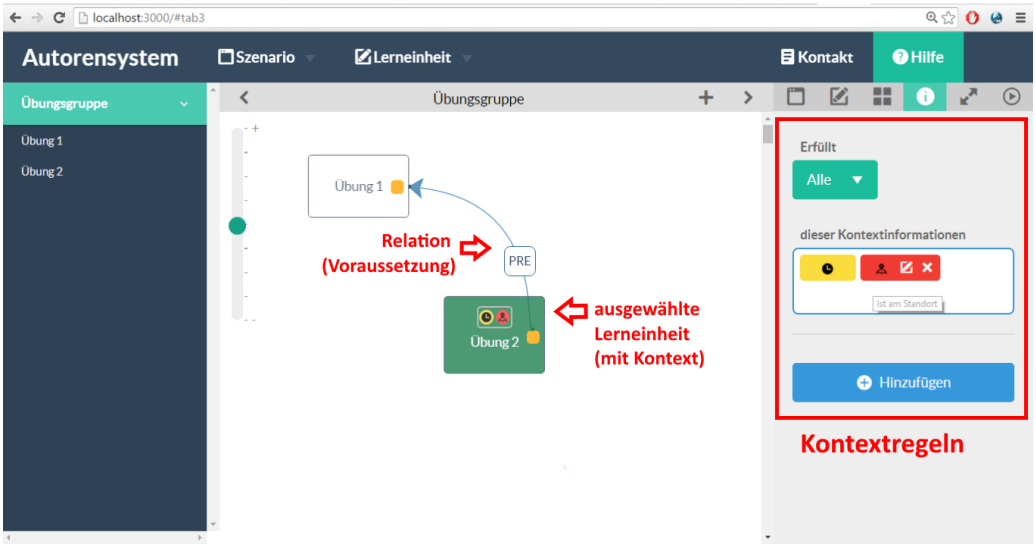


Abbildung 2.3: Screenshot des Autorensystems (Elemente sind rot gekennzeichnet und beschriftet).

Abbildung 2.3 zeigt einen Screenshot vom User-Interface des Autorensystems, in dem das Szenario aus dem Beispiel angelegt wurde. In der Lerneinheit „Übung 2“ und im Panel rechts sind die angelegten Kontext-Regeln für Uhrzeit und Standort zu erkennen. Eine nähere Beschreibung des User-Interface folgt am Ende dieses Kapitels.

Zusätzlich sollen pro Szenario sogenannte *Constraints* festgelegt werden können.[6] Das sind global geltende Bestimmungen, die sich nicht nur auf einzelne, sondern alle Lerneinheiten beziehen. Damit kann beispielsweise festgelegt werden, dass zwischen 23 und 6 Uhr keine Lerneinheiten angezeigt

[6]Diese Funktionalität ist im Prototyp des Autorensystems noch nicht umgesetzt.

werden sollen; oder dass Audio- und Videoeinheiten nur ausgewählt werden sollen, wenn Kopfhörer eingesteckt sind. An dieser Stelle könnten auch die verschiedenen Lernstile berücksichtigt werden, da diese Information über den Lernenden für alle Lerneinheiten in allen Szenarien gleichermaßen gelten.

Die Gesamtheit der Lerneinheiten, Relationen, Kontextinformationen und Constraints werden in Form einer *Ontologie* gespeichert und an einen Regelgenerator übergeben. Eine Ontologie ist ein Modell zur formalen Wissensrepräsentation und besteht aus einer *T-Box* und einer *A-Box*. Die T-Box liegt beim Regelgenerator vor und enthält allgemeine Klassendefinitionen, Relationen und Regeln, die die oben genannten Konzepte repräsentieren. Die A-Box, die im Autorensystem erstellt und an den Regelgenerator gesendet wird, beinhaltet eine Instanziierung dieses Modells mit konkreten Individuen, ihren Eigenschaften und Relationen.

Beide Komponenten (Autorensystem und Regelgenerator) laufen auf demselben Server und tauschen ihre Daten (A-Box und Regeln) über eine REST-Schnittstelle aus. Der Regelgenerator ist als fester Bestandteil des Frameworks mit höheren Performance-Ansprüchen in Java implementiert.

Die Regeln, die dieser aus der empfangenen Ontologie generiert, bilden die Logik der mobilen Anwendung und damit den einen Baustein der Adaptivität. Den anderen Baustein bilden die auf dem mobilen Gerät erfassten Kontextinformationen.

Die Regeln sind einfache „Wenn ..., dann ...“-Konstrukte, ähnlich den Beispielen in Kapitel 1, wobei der „dann“-Teil jeweils die Auswahl einer Lerneinheit beinhaltet. Wiederum am obigen Beispiel orientiert, könnte die Regel für die Auswahl der Lerneinheit „Übung 2“ wie folgt aussehen:

```
WENN: Abgeschlossene Lerneinheit = "Übung 1"
 UND Standort des Benutzers = Raum III.H1.S16
 UND aktuelle Uhrzeit >= 14 Uhr,
DANN: wähle Lerneinheit "Übung 2" aus.
```

Das gesamte Regelwerk wird vom Regelgenerator zurück ans Autorensystem gesendet. Von dort aus werden die Regeln zusammen mit den Inhalten der Lerneinheiten an das Mobilgerät des Endanwenders geschickt, genauer gesagt: an die mobile Lernanwendung, die darauf läuft.

2.3.3 Adaptation Engine

Der mobile Teil des Frameworks basiert ebenfalls auf Webtechnologien. Denn die oben genannten Sprachen können auf jedem Smartphone oder Tablet vom

Browser interpretiert werden, wodurch eine plattformunabhängige Lösung geschaffen wird.

Der „Motor“ der mobilen Anwendung ist die sogenannte *Adaptation Engine*. Diese nimmt die automatische Adaptierung, das heißt die Auswahl der passenden Lerneinheiten vor. Dazu startet und koordiniert sie zwei essenzielle Teilkomponenten: Zum einen die Kontexterfassung, die durch eine Implementierung des Context Toolkit [DAS01] realisiert wird; zum anderen eine *Rule Engine*. Diese verbindet die beiden Bausteine Kontext und Regeln miteinander. Sie vergleicht den erfassten Kontext mit den Bedingungen, dem „Wenn“-Teil der Regeln und informiert die Adaptation Engine über gefundene Übereinstimmungen. Ist die Bedingung einer Regel erfüllt, wird der Adaptation Engine die Lerneinheit aus dem „Dann“-Teil mitgeteilt. Die Adaptation Engine reicht diese Information weiter an die App, die sich schließlich um die Präsentation der Lerneinheit kümmert.

Dieser letzte Teil, die Präsentation der ausgewählten Inhalte, ist Sache der Lernanwendung und fällt nicht mehr ins Ressort des Frameworks. Denn jeder Lehrer oder Anbieter von Lern-Apps soll selbst über die konkrete Ausprägung des Lernerlebnisses entscheiden können. Das MOTIVATE-Framework deckt lediglich die Adaptierung ab.

2.3.4 ContactJS

Zur Erfassung von Kontext wurde im Rahmen einer Masterarbeit [Lem14] das Context Toolkit [DAS01] unter dem Namen ContactJS (Context Acquisition Tooltik) nach JavaScript portiert und anschließend im weiteren Projektverlauf erweitert. Dieses Framework - oder, um Verwechslungen zu vermeiden, Toolkit - folgt einem modularen Aufbau und dem Publish-Subscribe-Muster: einzelne, autonome *Widgets* erfassen rohen Kontext aus jeweils einer Quelle und geben diesen in der Regel normalisiert und ggf. teilweise interpretiert weiter. *Interpreter* sind weitere eigenständige Komponenten, die diese Kontextinformationen als Input „abonnieren“ und zu nochmals höherwertigem Kontext verarbeiten. Eine zentrale Verwaltungseinheit, der *Discoverer*, fungiert als Verzeichnis und Vermittler zwischen den Komponenten. *Aggregatoren* sind darüberhinaus dafür zuständig, Kontext aus verschiedenen Quellen, sowohl Widgets als auch Interpretern, zu aggregieren und an die Anwendung des Toolkits weiterzureichen. In diesem Fall ist die Anwendung die Adaptation Engine, die bei einem oder mehreren Aggregatoren den Kontext anfragt, der für die Auswertung der Adaptierungslogik benötigt wird. Zusätzlich wurde ein spezieller Aggregator implementiert, dem Kontextinformationen manuell (bzw. programmatisch) übergeben werden können. Dieser nimmt in dieser

Arbeit eine wichtige Rolle ein.

2.3.5 Prototypisches User-Interface

Die Abbildung 2.4 zeigt einen Screenshot des Autorensystems, in dem die folgenden Elemente rot gekennzeichnet und beschriftet sind.

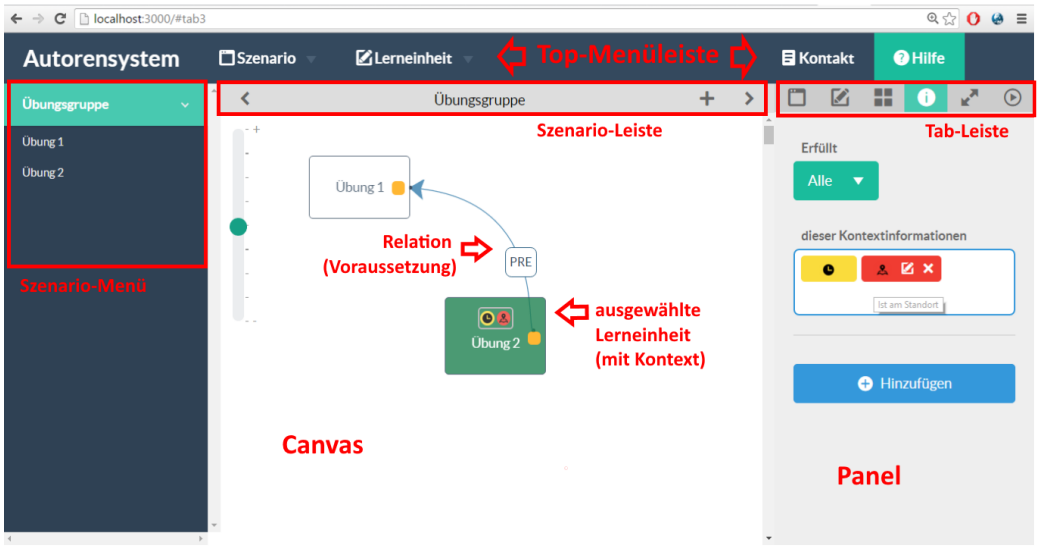


Abbildung 2.4: Screenshot des Autorensystems (Elemente sind rot gekennzeichnet und beschriftet).

Oben befindet sich die Hauptnavigation (Top-Menüleiste) mit Drop-Down-Menüs für Szenarien und Lerneinheiten sowie Kontakt und Hilfe. Szenarien können dort erstellt, gespeichert, gelöscht oder exportiert werden. Lerneinheiten können erstellt oder gelöscht werden.

Links befindet sich ein Drop-Down-Menü aller angelegten Szenarien und Lerneinheiten. Dies gibt zum jetzigen Zeitpunkt lediglich einen Überblick. Auch können bisher nicht mehr als ein Szenario bearbeitet werden.

In der Mitte ist das *Canvas* zu sehen, in dem Lerneinheiten angelegt und durch Drag-and-Drop verschoben und miteinander verbunden werden können. In den Lerneinheiten werden der jeweilige Name angezeigt sowie die Icons der Kontextinformationen, die für die Auswahl der Einheit gelten müssen. Ein Mouseover über diesen Icons zeigt jeweils den Namen der Kontextinformation an.

Lerneinheiten und Relationen werden durch Anklicken ausgewählt. Ihre Eigenschaften werden dann im jeweils zugehörigen *Tab* im *Panel* auf der

rechten Seite angezeigt. Links im Canvas befindet sich noch eine Skala zum Zoomen, die aber derzeit nicht wie gedacht funktioniert.

Im Panel werden alle Einstellungen und Eigenschaften aller Elemente bearbeitet. Navigiert wird durch die Tabs für (von links nach rechts): Szenario (derzeit noch leer), Lerneinheit (Name, Beschreibung und Löschen), Inhalt der Lerneinheit, Kontext der Lerneinheit, Relation und (seit dieser Arbeit) Simulator.

Zu Beginn dieser Arbeit war die Tab-Leiste allerdings noch anders umgesetzt. Die Ansicht änderte sich dynamisch mit der Auswahl eines Elements: Wurde ins Canvas oder Menü geklickt, war der Szenario-Tab zu sehen; wurde eine Lerneinheit angeklickt, wurden stattdessen die drei zugehörigen Tabs angezeigt; beim Klick auf das Label einer Relation erschien in der Tab-Leite „Relation“. Dieser Punkt wird in Kapitel 4.5 noch einmal thematisiert.

Mittig unter der Top-Menüleiste gibt es noch eine Funktionsleiste zum Ändern des Szenario-Namens und schnellen Hinzufügen von Lerneinheiten („+“). Die beiden Pfeile links und rechts davon bewirken bei Klick ein Ein- bzw. Ausblenden der Panels links und rechts, um das Canvas zu vergrößern.

3 Verwandte Arbeiten

Zum Testen von mobiler adaptiver Software durch die Simulation von Kontext gibt es bereits einige Vorarbeiten, obwohl es sich hierbei um ein noch sehr junges Forschungsthema handelt. Die Ansätze finden sich in verschiedenen Bereichen der Informatik wieder und lassen sich grob einteilen in automatisches Testen und manuelles Testen. Im Folgenden sollen beide Forschungsrichtungen kurz vorgestellt werden.

3.1 Automatisches Testen

[CGO15] gibt einen Überblick über Werkzeuge, mit denen System- und UI-Events generiert werden können, um speziell Android-Apps automatisch auf ihre Robustheit, d.h. Performance-Einbrüche, Abstürze und andere Fehler zu testen. Mit System-Events sind Ereignisse gemeint, die das Betriebssystem, in diesem Fall Android, auslöst. Dazu gehören zum Beispiel Benachrichtigungen über eingehende Anrufe oder SMS oder einen niedrigen Akkustand [MTN13]. Unter UI-Events werden hingegen Interaktionen des Nutzers wie Texteingaben und Tipp- oder Wisch-Gesten verstanden [CGO15]. Zu den Werkzeugen, die solche Events generieren, gehören unter anderen Dynodroid [MTN13] und MobiGUITAR [AFT+15]. Letzteres baut auf den Ergebnissen von [AFTA13] auf. Darin wurden neben System- und UI-Events auch Kontext-Events, die durch die im Smartphone eingebaute Sensorik ausgelöst werden, berücksichtigt und typische Event-Muster definiert. Ein Beispiel ist der kurzzeitige Verlust des GPS-Signals [AFTA13]. Auch Dynodroid bezieht Kontext-Events mit ein, sie werden jedoch zu den System-Events gerechnet [MTN13].

Diese Arbeiten haben zum Ziel, konkrete (Android-)Apps auf Robustheit und Fehler zu testen und dazu möglichst viel Programmcode abzudecken [CGO15]. Dieses Testen findet hauptsächlich vollautomatisiert statt und wird durch App-Entwickler initiiert und überwacht.

Andere Arbeiten haben zum Ziel, möglichst umfangreiche Testfallsammlungen zu generieren, die wiederum für automatisierte Tests kontextsensitiver Anwendungen genutzt werden können [PSS12][GG14][LLB+14].

Es geht demnach bei den genannten Ansätzen nicht darum, das adaptive Verhalten einer App nachzuvollziehen und ihre Anpassungslogik zu überprüfen.

Auch können die Ergebnisse nicht verwendet werden, um Ideen für eine Benutzeroberfläche für Nicht-Programmierer abzuleiten.

Es kann für diese Arbeit lediglich die Erkenntnis mitgenommen werden, dass neben der Modellierung und Simulation von Kontextinformationen (Kontext-Events) auch weitere, vom Benutzer oder System ausgelöste Events eine Rolle spielen könnten.

3.2 Manuelles Testen

Hier finden sich vor allem Arbeiten, die im Forschungsbereich Ubiquitous oder Pervasive Computing (kurz: Ubicomp) angesiedelt sind, zum Beispiel [OKL+05][MOG+10][DSSK06][BC13]. Diese Arbeiten sind auf die Entwicklung und Simulation von Anwendungen für „Smart Homes“ oder andere „smarte“ Räume fokussiert. Das sind Räume, die mit einer Vielzahl von Sensoren (Messgliedern) und Aktuatoren (Stellgliedern) ausgestattet sind, um sich den Vorlieben ihrer Bewohner anzupassen. Ein Beispiel ist die Regulation der Beleuchtung in einem Gebäude: Betritt eine Person den Flur oder das Treppenhaus, soll dort das Licht eingeschaltet werden, und nur dann [OCL10].

Einige Beliebtheit hat in diesem Bereich der Einsatz von 3D-Engines aus Ego-Shooter-Spielen erfahren [BV03][OCL10][MBOL11]. Damit lassen sich eine virtuelle Realität gestalten und ein reales Anwendungsszenario nachbilden. So können Räume einschließlich darin befindlicher Personen, Sensoren und Aktuatoren modelliert und deren Interaktion simuliert werden.

Nicht ganz so ausgefallen, aber ähnlich sind Ansätze, in denen 2D-Oberflächen zur Modellierung von Räumen, Personen, Sensoren und Aktuatoren verwendet wurden [CSMH12][BC13].

Für ihren speziellen Ubicomp-Anwendungsbereich sind diese Werkzeuge bereits sehr ausgereift und bieten den Vorteil der spielerischen Bedienbarkeit auch für Laien. Allerdings sind sie auch auf diesen Bereich beschränkt und lassen sich nicht für mobile, ortsungebundene Anwendungen und Kontextinformationen wie Lernstil, Frustlevel, Motivation oder Lernfortschritt generalisieren. Das Modellieren der virtuellen 3D-Welten erfordert außerdem wiederum Experten-Kenntnisse.

Andere Arbeiten befassen sich ebenfalls mit der Simulation von Kontext zum Testen adaptiver Apps, verfolgen dazu jedoch weniger visuelle, interaktive Ansätze. In einem werden Kontext-Events in Script-Form eingegeben [KPPR10]. In einem weiteren wurde ein Framework mit eigenem Kontextmodell und einer Administrator-Oberfläche entwickelt [BVH06]. Diese Oberfläche ist allerdings für Entwickler mit einem tieferen technischen Verständnis des

Kontextmodells entwickelt worden.

Erwähnenswert ist eine Arbeit [DSSK06], die dem MOTIVATE-Framework sehr ähnlich ist, indem sie:

1. sich mit dem einfachen Erstellen adaptiver Anwendungen für Laien beschäftigt,
2. dabei die Endnutzer zentral im Blick hat,
3. ein User-Interface zum einfachen visuellen Definieren von Regeln beinhaltet und
4. ebenfalls auf dem Context Toolkit [DAS01] aufsetzt.

Allerdings kann das Design der Simulationskomponente in diesem Framework nicht als Inspiration dienen. Denn während das User-Interface zum Definieren von Regeln in Nutzerstudien getestet und verbessert wurde [DSSK06], wurde der Simulator hier vernachlässigt.

Zuletzt sei eine zu Beginn dieser Arbeit noch frische Veröffentlichung genannt: In [VHH15] wurde eine Möglichkeit erforscht, „high-level“, also komplexen Kontext zu modellieren und Testfälle abzuleiten. Dazu wurde auch ein User-Interface entwickelt, in dem physikalischer Kontext zu logischem Kontext und dieser zu Situationen zusammengesetzt werden können. Situationen können dann in einer Zeitleiste zu Szenarien angeordnet und simuliert werden. Bei der Simulation werden die komplexen Informationen wieder in ihre Bestandteile zerlegt und in einen Android-Emulator übertragen. In einem zusätzlichen Fenster können Testfälle definiert werden, wofür typische Fehlermuster für jede Kontextquelle ermittelt wurden.

Die Lösung ließ sich im MOTIVATE-Framework nicht einsetzen, da sie an den Android-Emulator gebunden ist und ein anderes Kontextmodell verwendet. Auch das Interface war nicht mit Nutzern getestet worden und ließ einige Schwächen erkennen.

Diese sowie weitere Elemente der Arbeit konnten jedoch als Orientierung und Inspiration genutzt werden. So wirkten etwa die Darstellung einer Zeitleiste und Abspielfunktion, wie sie auch schon in [BC13][CSMH12] zu finden sind, vielversprechend. Auch das Modellieren und Testen von fehlerhaftem Kontext sollte berücksichtigt werden.

4 Konzipierung

Im Folgenden sollen zunächst die initial bestimmten funktionalen und nichtfunktionalen Anforderungen besprochen werden. Anschließend werden die ersten Ideen für ein User-Interface-Konzept vorgestellt. Schließlich wird erläutert und begründet, wie in dieser Arbeit methodisch vorgegangen wurde, um die gestellten Anforderungen zu erfüllen.

4.1 Anforderungen

Gesucht wurde eine *generische* Lösung, die Anwender beim Testen und Begreifen der Arbeitsweise des *Frameworks* (nicht: einer spezifischen App) unterstützt. Diese Lösung besteht in einem Konzept für die Modellierung und Visualisierung der kontextabhängigen Adaptierung. Sie sollte folgende Punkte gewährleisten:

Korrektheit: Das Verhalten der Adaptierung sollte korrekt, das heißt realitätsgetreu simuliert werden. Dazu mussten die Komponenten Regelgenerator und Adaptation Engine an den Simulator angebunden werden. So sollte ausgeschlossen werden, dass eine Abstraktion von der Funktionsweise oder Änderungen am Framework ein fehlerhaftes oder abweichendes Verhalten bewirken.

Generizität: Die Lösung sollte erweiterbar und offen für alle Kontextarten, Relationen, Constraints, Inhalte, Anwendungen und Geräte sein. Das schließt die Simulation einer bestimmten Lernanwendung aus. Simuliert werden konnte und sollte zunächst allein die Funktionsweise der Adaptierung, das heißt der Auswahl der Mikrolerneinheiten. Daher sollte von technischen Details, die nicht Bestandteil des Frameworks sind, abstrahiert werden.

Größtmögliche Nutzerunterstützung: Das Modellieren der Testfälle sollte für die Nutzer so verständlich, das heißt auch unmissverständlich, und intuitiv wie möglich sein. Die Möglichkeit von Fehlern in der Modellierung der Testfälle sollte minimiert werden. Denn der Fokus der Nutzer sollte auf dem Verhalten der Anwendung liegen, nicht auf dem Testen der Testfälle. Zudem gehören

zur Zielgruppe auch und vor allem Personen ohne Programmiererfahrung. Deswegen sollte auch im User-Interface von technischen Details weitestgehend abstrahiert werden.

4.1.1 Funktionale Anforderungen

Diese übergeordneten, groben Ziele ließen sich weiter konkretisieren. Zu den funktionalen Anforderungen gehörte, dass dem Benutzer folgende Möglichkeiten geschaffen werden:

Modellierung und Simulation: Kontext und Benutzer-(inter-)aktionen können (manuell) modelliert werden.[1] Zur Simulation gehörte die Visualisierung der Adaptierung, das heißt, Auswahl der Lerneinheiten.

Dynamische Anpassungen: Der Benutzer kann das Modell des zu simulierenden Kontexts noch ändern, nachdem die Simulation gestartet wurde, ohne diese beenden und neu starten zu müssen.

Modellierung von Fehlerfällen: Fehlerhafte oder fehlende Kontextinformationen können modelliert werden. Dadurch sollte das einfache Testen der Anwendung auf Geräten mit unterschiedlicher Ausstattung ermöglicht werden.

Definition von „Pre-Sets“: Der Benutzer kann Vorlagen für Testfälle definieren und wiederverwenden.

4.1.2 Nicht-funktionale Anforderungen

Die nicht-funktionalen Anforderungen wurden wie folgt spezifiziert:

Integration in den Autorensystem-Prototyp: Der Simulator sollte als neue Komponente in den bestehenden Prototyp des Autorensystems integriert werden. Dazu gehörten sowohl die Verwendung derselben Webtechnologien und Frameworks als auch Konsistenz im Design.

[1] Benutzeraktionen beschränkten sich noch auf die Information, welche Einheiten der Lernende abgeschlossen hat. Diese wird benötigt, um Regeln auszuwerten, die die Relation „Voraussetzung“ beinhalten. Zum Zeitpunkt der Entstehung dieser Arbeit war diese Relation die einzige, die bei der Regelgenerierung berücksichtigt wurde.

Intuitive Interaktionsformen: Die Bedienung des Simulators sollte verständlich und leicht erlernbar sein. Dazu sollten die „Grundsätze der Dialoggestaltung“ nach DIN-Norm EN ISO 9241-110 [ISO06] beachtet werden, insbesondere:[2]

- Selbstbeschreibungsfähigkeit: durch klare Kommunikation der Funktionalität und Rückmeldungen des Systems.
- Erwartungskonformität: durch Konsistenz sowie Berücksichtigung mentaler Modelle und Vorkenntnisse.
- Lernförderlichkeit: durch Anleitungen und geeignete Hilfestellungen.

4.2 Methodik

Im Usability-Engineering gibt es eine große Auswahl an Methoden, um nützliche, gebrauchstaugliche und „benutzerfreundliche“ Software-Produkte zu entwickeln. Als besonders wichtig und wertvoll wird dabei die Nutzerforschung, die sogenannte Contextual Inquiry, betont [RF13][Mos12]. Dabei geht es darum, Informationen über Benutzer in ihrem realen Arbeitsumfeld zu sammeln, um ihre Aufgaben, Arbeitsabläufe, Bedürfnisse, Probleme, Wünsche und Erwartungen zu verstehen und Lösungen zu schaffen, die daran ausgerichtet sind. Die Informationen werden verwendet, um Benutzerprofile und Personas, Use-Cases und Szenarien abzuleiten, die wiederum als Grundlage für die Entwicklung von Prototypen und schließlich eines User-Interface-Konzepts dienen [RF13].

4.2.1 Machbarkeitsanalyse

Als diese Arbeit entstand, existierten weder vergleichbare Kontextsimulatoren für „Technik-Laien“ (siehe Kapitel 3), noch typische Benutzer solcher. Daher konnte auf diese Informationsquellen nicht zurückgegriffen werden. Folglich ließen sich auch typische Anforderungen, Arbeitsabläufe, Szenarien und Personas nicht durch Feldstudien ermitteln. Fokusgruppen [Mos12] konnten ebenfalls schwerlich zusammengestellt werden, da es in diesem Anwendungsbereich bisher keine Experten mit klaren Vorstellungen von der zu entwickelnden Lösung gab.

[2] Des Weiteren sollten bei der Dialoggestaltung nach[ISO06] Aufgabenangemessenheit, Steuerbarkeit, Individualisierbarkeit und Fehlertoleranz grundsätzlich berücksichtigt werden.

Da auch die Zielgruppe dieser Arbeit sehr breit definiert war, ließ sich nicht bestimmen, mit welchen Anwendungen alle potenziellen Endnutzer gleichermaßen vertraut sein würden. Hätte andernfalls davon ausgegangen werden können, dass alle künftigen Nutzer von Autorensystem und Simulator bereits regelmäßig mit derselben App arbeiteten, so hätten ihre Erfahrungen damit und Erwartungen daran in die neue Lösung einfließen können. Doch diese Voraussetzung war hier nicht gegeben.

Um dennoch Informationen über die Benutzer, ihre Erwartungen und Bedürfnisse im Umgang mit dem zu entwickelnden System zu erhalten, konnten sie direkt damit konfrontiert und ihre Reaktionen beobachtet werden [Mos12]. Daraus konnten sich Schlüsse darüber ziehen lassen, welche Unterstützung benötigt würde, um den Umgang schnell zu erlernen, und welcher Funktionalität noch bedurft würde.

4.2.2 Wahl der Methode: Prototyping

User-Interface-Prototyping [RF13] ist dafür das Mittel der Wahl: Dabei wird ein Prototyp mithilfe von Probanden, die potenzielle Endanwender repräsentieren, getestet und weiterentwickelt.

Der Prototyp erfüllt zwei Funktionen: Zum einen kann damit anschaulich vermittelt werden, worum es eigentlich geht, sodass Missverständnisse vermieden werden. Zum anderen dient der Prototyp als Orientierungshilfe, um die Kreativität der Testnutzer anzuregen, ohne sie zu überfordern. Erste Ideen können gemeinsam mit den Testnutzern exploriert und ausgebaut oder auch komplett umgestoßen werden, wenn sich zeigt, dass die Probanden damit zu große Probleme haben.

User-Interface-Prototyping kann auch als *Usability-Testing* verstanden werden: das interaktive Testen eines Prototyps mit repräsentativen Testnutzern, die für sie realistische Aufgaben damit lösen [RC08][Bar10]. Ziel solcher Tests ist es, die Usability des Prototyps einzuschätzen, Probleme und Optimierungen für die weitere Entwicklung zu identifizieren.

Usability-Testing wird als Teil eines iterativen Entwicklungsprozesses gesehen, der durch wiederholte Tests von Beginn an begleitet und maßgeblich vorangetrieben wird [RC08][Mos12][Bar10][EY10]. Dabei kommen Prototypen unterschiedlicher Reifegrade zum Einsatz, beginnend bei Entwürfen auf Papier, bis hin zu voll funktionsfähigen, fertigen Anwendungen [Sny03][Bar10][RC08].

Die Tests werden teilweise als „rein qualitativer Ansatz“ [EY10] betrachtet, teilweise nochmals verschiedentlich klassifiziert, je nach Verortung im Entwicklungsprozess, Zielen und entsprechendem Vorgehen [Bar10][RC08][Hei07]. In [Bar10] wird beispielsweise zwischen formativem und summativem Testen

unterschieden:

- *Formatives Testen* begleitet den Entwicklungsprozess in Form mehrerer kleinerer Studien, deren *qualitative* Ergebnisse in die Verbesserung und Verfeinerung des jeweils getesteten Prototyps einfließen.

- *Summatives Testen* hingegen folgt nach Abschluss der Entwicklung in Form einer umfangreichen Studie, in der *Benchmarks*[3] ermittelt oder verglichen werden.

In dieser Arbeit ist ein qualitativer Ansatz [Bar10][EY10] verfolgt worden, um zu ergründen, *welche* Anforderungen aus Nutzersicht bestehen und *wie* diese erfüllt werden könnten.

Der Vorteil einer qualitativen Studie mit Beobachtung und Interviews der Nutzer - gegenüber quantitativen Tests - liegt im differenzierteren Feedback [Bar10], das nicht nur Probleme offenbart, sondern auch Aufschluss darüber geben kann, woher diese rühren. Gerade früh in der Entwicklung kann so auch herauskristallisiert werden, welche der Probleme nicht nur der „Unreife“ der Anwendung geschuldet sind. Ein weiterer Vorteil ist, dass selbst aus wenigen Tests mit wenigen Nutzern viel Feedback generiert werden kann [Bar10].

Der Nachteil liegt darin, dass die Auswertung der erhobenen Daten deutlich aufwendiger ist als bei einer quantitativen Studie, weil eine gründliche, iterative Analyse erforderlich ist [Bar10].

Der Vollständigkeit halber sei angemerkt, dass Statistiken zu Performanz und Präferenz [Bar10][RC08] zu diesem Zeitpunkt in der Entwicklung, mit einem unausgereiften Prototyp und wenigen Testnutzern, wenig Aussagekraft und Nutzen gehabt hätten [Bar10]. Zumal für diese Art Anwendung noch keinerlei Benchmarks existierten, gegen die getestet werden könnte.

Paper-Prototyping

Beim Paper-Prototyping [Sny03] wird sehr früh im Entwicklungsprozess ein erster Prototyp aus Papier getestet. Bedient wird er vom Testnutzer ähnlich wie ein echtes User Interface, die Maus wird dabei durch einen Stift ersetzt, Testeingaben per Hand geschrieben. Die Interaktion kommt durch einen menschlichen „Computer“ zustande, der das Verhalten der Anwendung imitiert und alle Bestandteile entsprechend hinzufügt, wegnimmt oder umordnet. Ein Paper-Prototype ist bewusst krude gehalten: Alle Beschriftungen

[3] Quantitative Nutzungsdaten wie z.B. Zeit für die Bearbeitung einer Aufgabe, Abbrechen eines Bearbeitungsschritts, Fehlerfälle etc.

sind schnell per Hand getätigt, Linien ohne Lineal gezogen, die einzelnen Bestandteile des User Interface grob aus Papier oder Pappe ausgeschnitten.

Dadurch hat er gleich mehrere zusätzliche Vorteile [Sny03]:

1. Den Testpersonen wird sofort klar, dass es sich lediglich um einen ersten Rohentwurf handelt, was die Scheu, Kritik - auch grundlegende - zu äußern, maßgeblich reduziert. Ein digitaler Prototyp löst hingegen eher Zurückhaltung aus, weil den Testpersonen bewusst ist, dass bereits einige Arbeit in die Entwicklung gesteckt worden ist.

2. Er lässt sich mit geringem Aufwand erstellen und ändern. Änderungen und Ergänzungen können sogar während der Tests schnell und einfach vorgenommen werden, was die Testpersonen dazu einlädt, sich aktiv am Gestaltungsprozess zu beteiligen. Das kann vom Äußern von Kritik, Vorschlägen und Wünschen über Verschieben und Umordnen der Bestandteile bis hin zum Griff zu Papier und Stift reichen.

3. Dadurch dass der Computer durch einen Menschen simuliert wird, wird ein hohes Maß an Interaktivität erreicht. Es gibt im Prinzip keinerlei Beschränkungen der möglichen Anwendungsfälle, Eingaben und Interaktionen. So können auch neuartige Lösungsideen und Anforderungen exploriert werden.

Fragestellung: Durch alle diese Vorteile eignete sich ein Paper-Prototyping als erster Schritt, um das grobe Konzept dieser Arbeit zu testen und Antworten auf die folgenden Fragen zu finden:

- Sind die angenommenen funktionalen Anforderungen valide und vollständig? Welche Use-Cases[4] wurden noch nicht bedacht?

- Wie sollten die geplanten Funktionen des Simulators priorisiert werden?

- Inwieweit fördert das (grobe) Konzept das Verständnis der Anwendung?

Details zu den Zielen und Fragestellungen des Paper-Prototypings sowie zur Durchführung und den Ergebnissen finden sich im Abschnitt 4.5.

[4] Anwendungsfälle, das heißt: Wozu nutzt der Anwender das System?

Schwächen der Methode

Paper-Prototyping kann zwar Antworten auf grundsätzliche Fragen liefern, wie zum Verständnis der Anwendung, ihrer Konzepte und Begrifflichkeiten, zur bereitgestellten Funktionalität, der Navigation darin und zu Arbeitsabläufen [Sny03]. Es kommt allerdings schnell an seine Grenzen, wenn Details der Interaktion mit der Anwendung getestet werden sollen [Sny03]. So können ganze Funktionen nicht getestet werden, wenn sie auf eine Weise implementiert werden sollen, die sich schlecht mit Papier simulieren lässt, wie zum Beispiel:

- Mouseover-Events wie Tooltips oder Änderungen des Cursors.
- Komplexere Visualisierungen wie Verdunkelung des Bildschirms und Aufleuchten von Elementen an verschiedenen Positionen.
- Zoomen und Scrollen.

Hinzu kommt, dass der menschliche „Computer“ deutlich mehr Zeit für seine Reaktionen und die Anpassung der Darstellung benötigt. Dadurch können Aufgaben und Interaktionen als umständlich und ineffizient empfunden werden, die sich in der „echten“ Anwendung schnell, einfach und intuitiv bedienen lassen.

Im zu entwickelnden Simulator trafen die genannten Punkte auf einen großen Teil der geplanten Funktionalität zu. Der gesamte Workflow der Simulation ließ sich nicht abbilden. Deshalb sollten im Paper-Prototyping zunächst lediglich Designidee und Anforderungen validiert und ggf. angepasst werden, wobei der Schwerpunkt auf dem interaktiven Teil, der Modellierung von Kontext, liegen sollte.

Usability-Testing mit digitalem Prototyp

Daran schloss sich die Implementierung einer digitalen Version des Prototyps an, in der die gewonnenen Erkenntnisse und geplanten Funktionen umgesetzt wurden. Der Prototyp wurde, sobald die wichtigsten Funktionen lauffähig waren, erneut mit Nutzern getestet. Dabei sollte ermittelt werden, inwieweit die Bedürfnisse und Anforderungen der Nutzer bereits erfüllt wurden und was noch fehlte oder verbessert werden musste. Die Tests sollten aufzeigen, wie Verbesserungen aussehen könnten, um sie in möglichen Folgearbeiten umzusetzen.

Zeit, Ressourcen und vor allem Probanden waren im Rahmen dieser Arbeit nur beschränkt verfügbar. Daher sollte die Evaluation der umgesetzten Ideen als „kleine Studie“ [Bar10] durchgeführt werden:

In einer solchen werden in kurzen Iterationen Tests durchgeführt, bei denen qualitative Daten zu Usability-Problemen erhoben werden. Die Ergebnisse werden verwendet, um notwendige Anpassungen am Prototyp zu identifizieren. Diese werden, falls machbar, direkt zwischen den Tests vorgenommen, um in den darauffolgenden Tests ihren Effekt zu überprüfen. Eine Quantifizierung der gefundenen Probleme, im Sinne von deren Häufigkeit und Gewichtung, kann dabei als Entscheidungsgrundlage dienen.

4.3 Use-Cases

Um den ersten Prototyp zu entwerfen, mussten die möglichen Use-Cases (Anwendungsfälle) bestimmt werden, das heißt, welche Aufgaben ein Benutzer mit dem Prototyp ausführen und wie er dabei vorgehen würde. In Abbildung 4.1 ist dargestellt, welche grundlegenden Schritte bei der Benutzung des Simulators angenommen werden. Abbildung 4.2 zeigt ein detailliertes Use-Case-Diagramm der zugehörigen Anwendungsfälle (siehe auch Abschnitt 4.1.1).

4.4 Designideen

Neben den Ideen dafür, was der Simulator funktional leisten sollte, wurde ein Konzept für die Gestaltung der Benutzungsoberfläche gesucht. Sie sollte konsistent sein - in sich und mit dem Autorensystem - und einen schnellen Einstieg in die Benutzung ermöglichen. Für die Modellierung und Simulation eines völlig beliebigen Kontextes gibt es noch keine bekannten Anwendungen, deren Interaktionsmuster etabliert sind und sich leicht übertragen lassen. Deswegen wurde zunächst nach einer *Metapher* als Grundlage für das Design gesucht. Auch die im vorigen Kapitel 3 vorgestellten Arbeiten wurden zum Teil als Inspirationsquelle genutzt.

4.4.1 Metaphern

Eine Metapher [Sta10] ist eine Übertragung bekannter Konzepte auf einen neuen, unbekannten Gegenstandsbereich. Dieser soll sich so schneller erschließen lassen, da er nicht in Gänze neu erlernt werden muss, sondern an Bekanntem angeknüpft wird. Dazu muss das mentale Modell [Sta10] des Anwenders berücksichtigt werden: Welche Vorstellungen hat er von der ihm bisher bekannten Welt und an welcher Stelle kann das neue Konzept eingeordnet werden? Beispiele für Metaphern aus dem User-Interface-Bereich

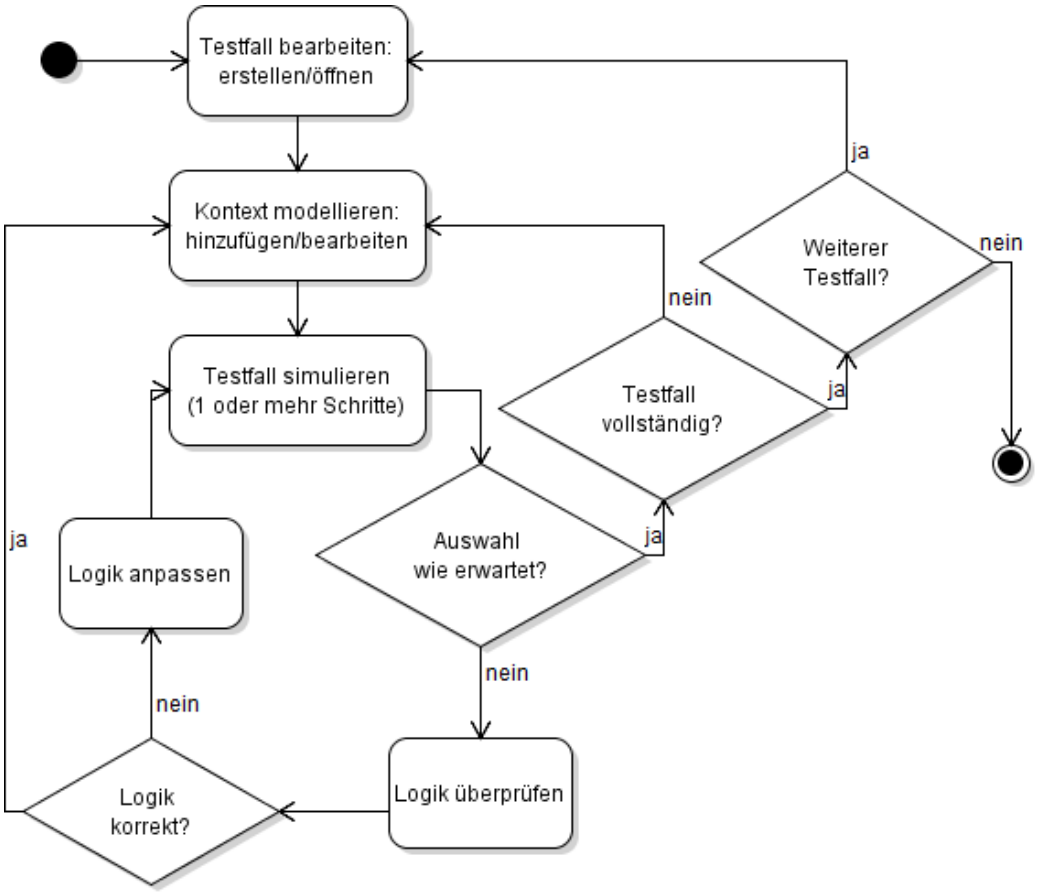


Abbildung 4.1: Grundlegender Workflow der Benutzung des Simulators.

sind das Papierkorb-Icon eines jeden Betriebssystems oder der Warenkorb auf Shopping-Webseiten [Sta10].

Das Modellieren von beliebigem Kontext eines Lernszenarios stellt einen neuen, unbekannten Gegenstandsbereich dar, der erschlossen werden soll. Eine Metapher würde sich hier mitunter als Unterstützung anbieten. Daher wurde nach konzeptionellen Ähnlichkeiten zu anderen Anwendungen, Gegenständen oder Ideen gesucht, die sich als Versinnbildlichung [Sta10] eignen könnten. Zunächst wurde versucht, die Fragen zu beantworten:

Was wird eigentlich bei der Kontextmodellierung abgebildet? Was stellt ein Kontextmodell dar? Auf der unteren, technischen Ebene werden Messwerte angegeben, aber *wofür stehen diese?*

Antwort: *Für einen Ausschnitt der realen Welt und eine Abfolge von Ereignissen.*

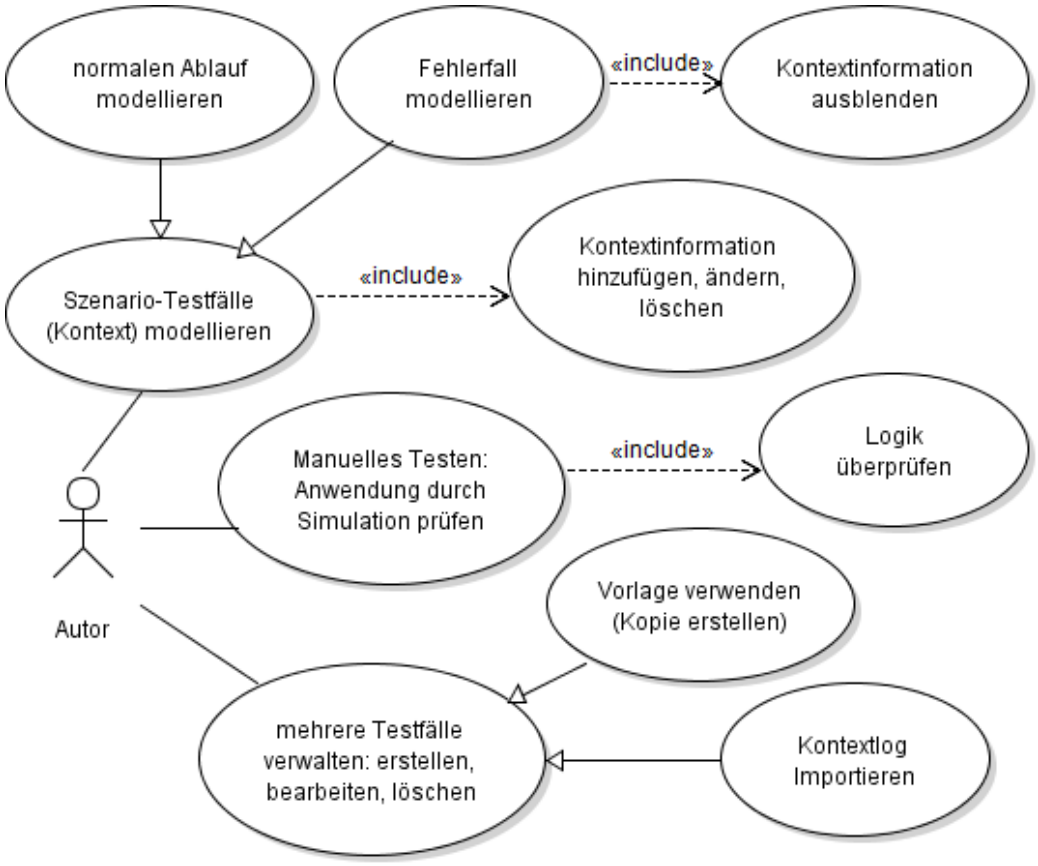


Abbildung 4.2: Use-Case-Diagramm der initial definierten Anwendungsfälle.

Nächste Frage: *Woher ist die Abbildung eines solchen „Stücks Realität“ sonst noch bekannt?*

Storyboard: Die erste damit verbundene Idee war die eines *Storyboards*, mit dem eine kurze Geschichte in einer Abfolge von Bildern erzählt wird. Jedes Bild repräsentiert dabei eine Situation, ähnlich wie in einem Comic.

Diese Metapher bietet zwar theoretisch eine starke Visualisierung, ließ sich jedoch als solche nicht praktisch umsetzen. Denn zum einen war die Übersetzung von einem Bild in konkrete Kontextinformationen zu diesem Zeitpunkt und im Rahmen dieser Arbeit nicht zu bewerkstelligen. Zum anderen hätten die Bilder zunächst einmal erstellt werden müssen, was nicht jedem Autor liegt. Am Ende hätten zu viele Kompromisse und Abstraktionen getroffen werden müssen, sodass die Metapher kaum noch erkennbar gewesen wäre.

Kalender: Daher wurde eine weitere Idee gesucht und in Form eines *Kalenders* gefunden: In einem Kalender werden ebenfalls Ereignisse geplant und visualisiert. Der Kalender als Metapher bietet gegenüber dem Storyboard die Vorteile, dass er für jeden Anwender bekannt und zudem leicht bedienbar ist.

Allerdings hätten auch hier wieder Abstriche bei der Umsetzung gemacht werden müssen: Das wesentliche Merkmal eines Kalenders ist die Orientierung an der realen, absoluten Zeit. Bei der Modellierung von beliebigem Kontext hingegen wäre das eine zu große Einschränkung. Eine bestimmte Uhrzeit ist auch nicht in jedem Lernszenario relevant. Es gibt lediglich eine Abfolge von Situationen, deren Zeitpunkte relativ angeordnet, aber nicht zwangsläufig festen Uhrzeiten zugeordnet werden können. Somit konnte auch der Kalender nicht *per se* als Metapher verwendet werden.

Schnitt-Programm: Eine letzte Möglichkeit der Übertragung von Bekanntem fand sich in Anwendungssoftware für *Audio- bzw. Videobearbeitung*.[5] Auf der konzeptionellen Seite lassen sich Ähnlichkeiten ausmachen, wenn - wie beim Storyboard-Ansatz - das Lernszenario als eine Art Film betrachtet wird, also eine Umsetzung eines fiktiven Stücks Realität.[6] Aber auch Elemente des User-Interface ließen sich bereits in einigen Vorarbeiten (siehe Kapitel 3) wiederfinden: So wurde ein *Play*-Button verwendet, um die Simulation abzuspielen, und auch eine Zeitleiste kam hin und wieder zum Einsatz. Zeitleisten in Programmen für Audio- und Videoschnitt sind zwar in der Regel an reale Zeiteinheiten gebunden, es gibt aber auch die Darstellung von einfachen nummerierten *Frames* in Animationssoftware, wie zu sehen am Beispiel von Adobe Animate[7].

Wahl der Metaphern

Die Wahl fiel schließlich auf eine Kombination der Elemente solcher Schnitt-Programme und einer Kalender-App. Aus Video-Schnitt-Programmen wurde die Idee einer *Zeitleiste* (engl. Timeline) übernommen, in der in jedem Frame die verschiedenen Kontextinformationen wie Spuren nebeneinander liegen. Der Kontext der Simulation wird in dieser Zeitleiste editiert. Gestartet und gesteuert wird die Simulation über eine Kontrollleiste, wie sie von Musik- und

[5] Diese Idee war bei der Vorstellung dieser Arbeit im Forschungsseminar des Lehrstuhls aufgekommen.

[6] Im Forschungsseminar war zudem die *Komposition* von Kontextinformationen zu einem Stück als Assoziation genannt worden.

[7] `https://helpx.adobe.com/animate/using/timeline.html#about_the_timeline` [Abgerufen am 05.04.2016]

Video-Abspielgeräten und -programmen bekannt ist und sich auch in einigen verwandten Arbeiten (siehe Kapitel 3) wiederfindet.

Ähnlichkeiten zum Kalender finden sich im Hochkant-Format wieder, in dem die Zeit von oben nach unten verläuft. Dabei sind die einzelnen „Zeitpunkte“, das heißt *Situationen* bzw. Simulationsschritte, wie die Zeiteinheiten in einem Kalender optisch voneinander abgegrenzt.

4.4.2 Bedien- und Visualisierungskonzepte

Wichtig waren eine einfache, überschaubare Modellierung von Kontext und eine deutliche Visualisierung. Beides sollte durch bekannte Konzepte intuitiv wirken.

Elemente für die Modellierung von Kontext: Die Inhalte der Zeitleiste sollten sich so einfach manipulieren lassen wie die Termine in einem Kalender: Durch einfaches Klicken oder wahlweise Drücken und Ziehen der Maus sollten Kontextinformationen hinzugefügt werden und sich durch erneutes Klicken bearbeiten lassen. An der jeweiligen Stelle sollte dazu ein kleines *Popover*-Fenster erscheinen, in dem der Kontextwert eingetragen bzw. verändert wird. Dessen Gültigkeitsdauer sollte außerdem durch direkte Manipulation, mittels Drag-and-Drop in der Zeitleiste verändert werden können. Die einmal eingetragenen Kontextwerte sollten mithilfe von *Super-Tooltips* bei Mouseover angezeigt werden: Dabei handelt es sich um eine Erweiterung des Standard-Tooltips, in der nicht nur reiner Text, sondern auch Markup angezeigt werden kann.

Elemente für die Visualisierung der Simulation: Sowohl in Audio- und Video-Programmen als auch in einigen Kalender-Apps wird der zeitliche Fortschritt an der Zeitachse durch einen schmalen Querbalken angezeigt. Ein solcher Indikator sollte auch für die Simulation verwendet werden und sich im Play-Modus mit der eingestellten Geschwindigkeit von oben nach unten über die Zeilen bewegen.

Gleichzeitig sollten die zur jeweils aktuellen Zeile passenden Lerneinheiten im Canvas als ausgewählt angezeigt werden.[8] Für das Highlighting der Lerneinheiten sollte eine *Lightbox* verwendet werden: Dabei wird der Hintergrund des Canvas abgedunkelt und nur das bzw. die selektierte(n) Element(e)

[8] Da zum Zeitpunkt der Konzipierung und Umsetzung noch keine Inhalte im Autorensystem hinterlegt werden konnten, sollte ein Highlighting der Lerneinheiten vorerst genügen.

„angeleuchtet“. Durch das Abdunkeln kann zum einen signalisiert werden, dass die Simulation läuft und im Canvas etwas passieren wird. Zum anderen werden die ausgewählten Lerneinheiten so besonders prominent und können schwer übersehen werden.

4.4.3 Progressive Disclosure

Ein Mittel, um Klarheit und Überblick zu schaffen, ist „Progressive Disclosure“ [Nie06]. Dabei werden die Funktionen danach priorisiert, wie oft sie von der Mehrheit der Benutzer gebraucht und genutzt werden. Die wichtigsten Funktionen - und nur diese - sollten in der ersten Ebene jederzeit verfügbar und sichtbar sein. Seltener benötigte, erweiterte Funktionalität wird für den Anwender in einer zweiten Ebene verborgen und nur bei Bedarf angezeigt. Ein typisches Beispiel dafür ist die Schaltfläche „Eigenschaften“ im „Drucken“-Dialog [Nie06].

Dieses Mittel sollte auch im Simulator Anwendung finden, um zu dessen Aufgabenangemessenheit und Lernförderlichkeit [ISO06] beizutragen, das heißt: Komplexität - gerade für neue Nutzer - zu reduzieren, sie nicht durch zu viele Optionen zu überfordern und ihnen eine grobe Orientierung zu geben. Sie sollten sich dadurch auf die ersten Schritte konzentrieren und sich den Funktionsumfang nach und nach erschließen können.

Es gibt Funktionen, die zwar von Anfängern noch nicht gebraucht werden, von geübten Nutzern („Power-User“) aber regelmäßig beansprucht werden, um so effizient wie möglich zu arbeiten. Diese Funktionen dürfen Anfänger nicht stören, müssen aber gleichzeitig für Fortgeschrittene schnell erreichbar sein. Um das umzusetzen, musste herausgefunden werden, wie die Nutzer mit dem Simulator arbeiten, welche Ziele sie wie erreichen wollen.

Die Funktionen der zweiten Ebene sollten bei Bedarf durch Mouseover oder Klicks sichtbar bzw. zugänglich gemacht werden. Diese Lösung hatte gegenüber der Verwendung von statisch platzierten Menüs den Vorteil, dass sie immer genau dort untergebracht werden können, wo sie benötigt werden, ohne zu stören.

4.5 Erste Nutzerstudie: Paper-Prototyping

Im frühzeitig durchgeführten Paper-Prototyping wurden die ersten Entwürfe getestet und Feedback zu Verständnis, Erwartungen und Bedürfnissen der Nutzer wie auch zu möglichen Hindernissen eingeholt. Für die Vorbereitung und Durchführung der Tests wurde sich hauptsächlich an [Sny03] orientiert.

Im Folgenden wird kurz das Vorgehen umrissen und anschließend näher auf ausgewählte Aspekte eingegangen.

4.5.1 Vorgehen

Nachdem einige Entwürfe auf Papier gezeichnet, mehrmals überarbeitet und schließlich eine vorläufige, grobe Design-Lösung gefunden worden war, wurde ein Benutzerprofil für die Rekrutierung der Testpersonen erstellt. Es wurden die Ziele der Studie definiert und passende Aufgaben(-typen) festgelegt, die die Probanden mit dem Prototyp lösen sollten. Alle zur Lösung der Aufgaben benötigten Teile des Prototyps wurden angefertigt.

Schließlich wurden die Probanden rekrutiert, Termine für die Tests vereinbart, ein Raum und Equipment (Videokamera, Stativ) organisiert. Dann wurden, über mehrere Tage verteilt, die Tests durchgeführt und für die Nachbereitung und Auswertung auf Video aufgezeichnet. Zwischen den Tests wurden die absolvierten nach- und die anstehenden vorbereitet und dazu Aufgaben und Prototyp überarbeitet.

Benutzerprofile

Da es zum jetzigen Zeitpunkt an realen Benutzern von Software zur Erstellung und Simulation kontextsensitiver Lernanwendungen fehlt, auf deren typische Eigenschaften zurückgegriffen werden könnte, wurde das Profil möglichst unspezifisch angelegt. Es charakterisiert potenzielle Endnutzer des Autorensystems wie folgt:

- Sie sind im Bereich Lehre tätig, im Hochschulbereich oder in der Industrie, und befassen sich mit dem Erstellen von Lehr-Lern-Materialien, auch in elektronischer Form.
- Sie haben nicht zwangsweise einen Hintergrund in Programmierung oder Informatik, sind aber aufgeschlossen für neue Technologien und interessiert am mobilen adaptiven Lernen. Sie sind daher intrinsisch motiviert, solches in Form einer MOTIVATE-Anwendung zu ermöglichen.
- Sie sind (noch) nicht vertraut mit den technischen Details der kontextabhängigen Adaptivität, wie sie im MOTIVATE-Framework umgesetzt ist, können sich aber unter Kontexterfassung etwas vorstellen.
- Sie haben Erfahrung mit Desktop-, Web-, und mobilen Anwendungen und sind vertraut mit den gängigen Office-Anwendungen wie

Textverarbeitungs-, Tabellenkalkulations-, E-Mail- oder Kalender-Programmen.

- Sie sollten die deutsche Sprache beherrschen, da alle Beschreibungen im Autorensystem-Prototyp (noch) auf Deutsch verfasst sind. Alter und Geschlecht spielen keine Rolle.

Diese Eigenschaften werden für Benutzer des gesamten Autorensystems angenommen. Für Benutzer des Simulators gilt daher zusätzlich:

- Sie haben bereits erste Erfahrungen mit dem Autorensystem gesammelt und wissen, wie und wo sie dort Szenarien, Lerneinheiten und Kontextinformationen anlegen und editieren können. Sie kennen also schon die einzelnen Oberflächenkomponenten und Interaktionsmuster und bringen daher spezifische Erwartungen mit.

- Sie sind mit den Begriffen Kontextinformation, Szenario, Lerneinheit und Regeln vertraut, in der Bedeutung, wie sie im Autorensystem verwendet werden.

Ziele der Studie

Neben den Benutzerprofilen wurden die Ziele bzw. Fragen festgelegt, die durch die Tests beantwortet werden sollten und an denen sich die Aufgabenstellungen orientieren würden.

Übergeordnete Fragen (siehe Abschnitt 4.2.2):

1. Inwieweit erfüllt das Design die Usability-Kriterien Selbstbeschreibungsfähigkeit, Erwartungskonformität und Lernförderlichkeit? Inwieweit unterstützt das Design das mentale Modell der Nutzer bzw. die Bildung eines solchen Modells?

2. Sind die angenommenen funktionalen Anforderungen valide und vollständig? Sind Arbeitsschritte und Funktionen angemessen priorisiert?

3. Ist der Simulator intuitiv bedienbar?

Um diese Fragen besser beantworten zu können, wurden sie auf spezifischere Fragen heruntergebrochen:

- Kann der Simulator leicht gefunden und seine Funktionsweise (Arbeitsschritte) erkannt werden?

- Lassen sich Situationen wie von den Nutzern gedacht abbilden?
- Wo gibt es Verständnisprobleme? An welchen Stellen sind Hilfen (Anleitungen, Erklärungen) nötig?
- Sind die Aufgaben bzw. Anwendungsfälle realistisch und nachvollziehbar?
- Welche Use-Cases (Anwendungsfälle) wurden ggf. noch nicht bedacht?
- Welche Funktionalität wird ggf. zusätzlich benötigt oder gewünscht?
- Welche Funktionalität ist am wichtigsten (wird am häufigsten verwendet)?
- Welche Funktionalität ist möglicherweise überflüssig (wird nicht gebraucht oder verwendet)?
- Werden Vorlagen entdeckt, gebraucht und als sinnvoll erachtet? Werden eigene Vorlagen zur Wiederverwendung erstellt?
- Ist die Bedienung effektiv und intuitiv erlernbar?
- Welche Änderungen sind nötig oder wünschenswert?

Rekrutierung

Im nächsten Schritt wurden die Probanden anhand der Benutzerprofile rekrutiert. Es konnten fünf Probanden gewonnen werden. Darunter waren drei Hochschullehrer aus den Bereichen Psychologie, Informatik und Bioinformatik, sowie Teilnehmer des zeitgleich erstmals stattfindenden Seminars „Adaptives Lernen“.[9] Die Letztgenannten waren Studenten aus den Bereichen Informatik und Psychologie bzw. Computational Science und besonders geeignet als Versuchspersonen, da sie

1. im weiteren Verlauf des Semesters selbst das Autorensystem verwenden sollten, um adaptive Lernanwendungen zu erstellen,
2. dabei jedoch noch nicht so vertraut mit dem System waren, dass sie eventuelle Stolpersteine ignorieren würden,

[9] In diesem Seminar war das MOTIVATE-Framework vorgestellt worden, mit dem die Studenten adaptive Lernanwendungen erstellen sollten.

3. den zu simulierenden Adaptierungsmechanismus des MOTIVATE-Frameworks noch nicht im Detail kennengelernt hatten,

sodass sie unvoreingenommen die Usability des Simulators beurteilen konnten.

Szenarien und Aufgaben

Für jede Testperson wurde ein fiktives, aber realistisches Ausgangsszenario konstruiert, in das sie sich aufgrund ihrer persönlichen Erfahrungen leicht hineinversetzen würde. Dazu wurde zunächst eruiert, mit welchen Lernszenarien sich die Probanden bereits persönlich beschäftigt hatten - zum Teil im Rahmen des oben genannten Seminars, zum Teil aus beruflichen Gründen. Durch die Personalisierung der Szenarien und Aufgabenstellungen sollte für alle Probanden ein vergleichbares Maß an Immersion und intrinsischer Motivation während der Tests erzielt werden.

Pro Use-Case (siehe Abschnitt 4.3) wurde eine Aufgabenstellung formuliert, in der ein konkretes beispielhaftes Szenario geschildert und ein zu erreichendes Ziel vorgegeben wurde, aber keine Anleitung zur Lösung der Aufgabe. Es gab insgesamt vier Aufgaben:

1. In Aufgabe 1 sollte als Erstes der Simulator[10] gefunden und anschließend der „Basisfall“ getestet werden: das Anlegen einer einfachen Simulation mit wenigen Übergängen und einer beschränkten Auswahl an Kontextinformationen. Eine zu hohe Komplexität sollte zu Beginn vermieden werden.
   Besonderes Augenmerk lag auf dem Verlauf der Informationen und der Frage: Wird die Klicken-und-ziehen-Funktion erkannt und genutzt? Idealerweise sollten die Vorlagen entdeckt und verwendet werden. Falls mehr Informationen hinzugefügt würden als hinpassten, sollte außerdem der Filter verwendet werden.
   Die Simulation sollte gestartet und die (nicht) erfolgreiche Anpassung erkannt werden.

2. In Aufgabe 2 sollte festgestellt werden, dass die App sich fehlerhaft verhält und in den Editiermodus gewechselt werden (durch Klicken auf die hervorgehobene Lerneinheit).

3. In Aufgabe 3 sollte die Funktion „Deaktivieren“ bzw. „Nicht verfügbar“ entdeckt und verwendet werden.

[10] Der Begriff „Simulator“ wurde nicht verwendet.

4. In Aufgabe 4 sollte die Funktion „Laden“ bzw. „Importieren“ erkannt und benutzt werden. Der Import sollte dazu dienen, die Verwendung des Filters zu testen.

Ablauf

Jeder Test dauerte im Schnitt etwas mehr als 1,5 Stunden und wurde per Videokamera in Ton und Bild aufgezeichnet, wofür zuvor das Einverständnis der Testpersonen eingeholt wurde. Zu Beginn wurden den Probanden einige Fragen zu ihren bisherigen Erfahrungen gestellt und die notwendigen Grundlagen erläutert. Dazu gehörte auch eine kurze Vorstellung des Autorensystems, das die Probanden einmal selbst ausprobieren und so ansatzweise kennenlernen sollten.

Dann wurde ihnen der Paper-Prototyp, das persönliche Szenario und nacheinander die zugehörigen Aufgaben präsentiert. Beim Lösen der Aufgaben waren die Probanden angehalten, „laut zu denken“. Wenn ein Proband längere Zeit nichts sagte, wurde nachgefragt, was er gerade dachte. Während der Tests wurden viele aufschlussreiche Kommentare gegeben, von denen einige erst in der Nachbereitung als besonders erkenntnisreich auffielen. Nach dem Lösen der Aufgaben gaben drei Probanden von sich aus weiteres Feedback: Sie stellten Fragen, sprachen Schwierigkeiten an und machten Verbesserungsvorschläge. Dabei erwähnten die Probanden auch andere Anwendungen wie Kalender-Apps und Video-Player, die sie regelmäßig verwendeten. So konnten noch einmal Erkenntnisse und Anregungen gewonnen werden.

Im Anschluss an jeden Test wurden Notizen und die vollständigen Videoaufzeichnungen durchgesehen, das Gesagte und Erlebte dabei protokolliert und teilweise transkribiert. Positives, Negatives und die wichtigsten Erkenntnissen wurden zusammengefasst und der Prototyp für den jeweils nächsten Test überarbeitet. Dabei wurden Lösungsvorschläge für aufgedeckte Probleme eingearbeitet und einzelne Elemente an das jeweilige Testszenario angepasst.[11]

4.5.2 Ergebnisse

Am Ende der Studie wurde für jeden Test versucht, die eingangs gestellten Forschungsfragen zu beantworten, um Verständnis und Erlernbarkeit, Funktionalität und Bedienbarkeit des Simulators einzuschätzen (siehe oben: „Ziele der Studie“). Die vollständigen, detaillierten Ergebnisse, Antworten und Verbesserungsvorschläge (sowohl der Studie als auch jedes einzelnen

[11] Test 4 und 5 fanden unmittelbar nacheinander statt. Daher konnten zwischendurch keine Änderungen mehr vorgenommen werden.

Tests) können Anhang 1 entnommen werden. In diesem Abschnitt werden die wichtigsten Erkenntnisse aus dem Paper-Prototyping zusammengefasst.

Dazu werden im Folgenden zunächst einige Probleme im Testablauf und Testdesign diskutiert, die bei der Auswertung berücksichtigt werden mussten. Danach werden die Kernergebnisse präsentiert und diskutiert.

Kritische Vorbetrachtungen

Eine mögliche Beeinflussung der Ergebnisse ist durch die inhärente „Low-Fidelity“ des Paper-Prototyps verschuldet. Wie in Abschnitt 4.2.2 angesprochen, wirkten manche Interaktionsformen anders als in einer digitalen Version. Die Interaktion mit dem Prototyp wurde zudem durch die herabgesetzte Geschwindigkeit des „Computers“ stark eingeschränkt. Daneben kam es hin und wieder zu Fehlern, die der Testleiterin in der simultanen Rolle des „Computers“ unterliefen. Zum Beispiel wurde ein Element vergessen rechtzeitig hinzuzufügen oder eine Kontextbedingung in der Simulation übersehen. Diese Fehler wurden aber in der Regel erkannt und korrigiert und führten daher nur kurzzeitig zu leichter Irritation bei den Testpersonen.

In einem Fall jedoch dauerte die Bearbeitung der ersten Aufgabe wegen eines missverständlichen System-Feedbacks[12] so lange, dass der Test nicht vollständig abgeschlossen werden konnte.

Eine Schwäche im Design der Testmaterialien hat außerdem sehr wahrscheinlich den Lernprozess beeinflusst und künstlich beschleunigt: Um zu testen, ob und wie die Probanden Vorlagen verwenden würden, wurden diese passend zum jeweiligen Szenario vorbereitet. Dadurch begannen 4 von 5 Probanden mit einer nicht leeren Vorlage. Hinzu kommt erschwerend, dass 3 dieser Vorlagen die Uhrzeit als Kontextinformation enthielten, sodass daraus ein Ablauf erkennbar wurde. In der Realität wird es allerdings kaum möglich sein, einem Autor, der das erste Mal mit dem System arbeitet, passende Vorlagen anzubieten. Besser wären womöglich unpassende Vorlagen gewesen, um zu sehen, ob die Probanden diese trotzdem zur Orientierung anschauen oder ganz ignorieren würden.

Schließlich muss auch beachtet werden, dass die hier vorgenommene Analyse durch dieselbe Person vorgenommen wurde, die auch den Prototyp entworfen und die Tests durchgeführt hat. Somit können Validität und Objektivität grundsätzlich nicht garantiert werden [LFH10].

[12] „Es wurden keine Lerneinheiten ausgewählt.“ Gemeint war, dass die Anwendung keine Lerneinheit ausgewählt hatte. Die Testperson nahm jedoch die Lesart an, dass *sie* eine Lerneinheit auswählen sollte. Diese Ambiguität blieb lange unentdeckt, da sich alle Handlungen und Äußerungen beider Teilnehmer ins jeweilige mentale Modell fügten.

Diskussion der Ergebnisse

Durch das Involvieren der Testnutzer konnte dennoch frühzeitig eingeschätzt werden, ob das gewählte Design seinen Zweck erfüllt. Die größten Schwierigkeiten beim Verständnis und bei der Bedienung konnten so noch vor der Implementierung aufgedeckt werden. Insbesondere konnte festgestellt werden, dass die Nutzer ein anderes mentales Modell von der Anwendung hatten als die Entwicklerin.

Verständnis: Die größten Verständnisprobleme gab es jeweils zu Beginn: Die Testpersonen waren unsicher, ob und was sie mit dem Simulator tun könnten. Es fehlte an Anleitungen und Erklärungen zu den ersten Schritten und Handlungsoptionen. Solche sind zwingend erforderlich, zumal „echte“ Benutzer keine so konkreten Aufgabenstellungen haben werden wie die Testpersonen.

Eine Hürde waren auch die gewählten Bezeichnungen, insbesondere die für das Simulationsmodell: Da „Szenario“ im Autorensystem schon vergeben war, war nach längerer Suche „Simulation“ gewählt worden. Aus den Tests ging die Erkenntnis hervor, dass das Kontextmodell einen viel engeren Bezug zum erstellten Szenario hat als ursprünglich angenommen: Es wurde erkannt, dass der für die Simulation modellierte Kontext zusammen mit den Lerneinheiten *genau das erstellte das Szenario* bildet.[13] Die Einführung des Begriffs „Simulation“ verklärte diesen Zusammenhang, zumal damit eher eine Emulation der Anwendung assoziiert wurde. Nach der Studie fiel schließlich die Wahl auf die Bezeichnung „Testfall“.

Der Kern der Anwendung, die Zeitleiste, war als solche nicht leicht erkennbar, was das Gesamtverständnis weiter erschwerte. Hier hätte eine Nummerierung der Zeilen (Situationen oder „Schritte“) helfen können.

Die größte Irritation gab es, wenn von der Anwendung erwarteter Kontext in der Simulation fehlte. Als wichtige und sinnvolle Unterstützung wurde daher - neben einer informativen Benachrichtigung - die Vorgabe der für das Szenario relevanten Informationen identifiziert. Das ergab sich vor allem aus der Beobachtung und den Kommentaren der Testpersonen beim Testen der Filter-Funktion, wo sie den Kontext des Szenarios anzeigen lassen wollten. Auch das neu gewonnene Verständnis vom Zusammenhang der Kontextmodelle - in der Anwendungslogik und in der Simulation - führte zu dieser Entscheidung. Fraglich war, ob die Vorgabe der Kontextinformationen auch das Bedürfnis eliminieren würde, das Zustandekommen von

[13] Anfangs war die Annahme von der Funktion des Simulators: Es wird - wegen des ungebundenen situativen Lernens - völlig beliebiger Kontext modelliert und die Reaktion des Systems darauf getestet.

High-Level-Kontext im Simulator zu überprüfen (siehe folgenden Absatz).

Funktionalität: Die Testpersonen wünschten sich vom Simulator zusätzliche Funktionalität. Als besonders wichtig wurde das „Debugging“ der Simulation hervorgehoben: Der Nutzer sollte leichter überprüfen können, *warum* eine Lerneinheit in einer bestimmten Situation (nicht) ausgewählt wird. Dazu wurde ein Abgleich („Matching“) der Kontextinformationen von Simulationsschritt und Lerneinheit durch das System vorgeschlagen.

In einem Fall wurde der Wunsch nach Testen der Kontextinterpretation (Reasoning) im Simulator geäußert, „um wirklich Vertrauen in das System haben zu können“ (Kommentar der Testperson).

Die Möglichkeit, real gemessene Kontextinformationen in den Simulator zu importieren, wurde zwar positiv aufgenommen. Es wurde aber auch der berechtigte Einwand geäußert, dass sich die Abfolge der Messpunkte der Informationen schlecht in die Zeitleiste übertragen ließe.

Das Aus- und Einblenden von Kontextinformationen zum schnelleren Modellieren von Fehlerfällen wurde kaum wahrgenommen. Zum einen war die Bezeichnung der Funktion nicht bzw. missverständlich. Zum anderen sollte im Test nur ein einfacher, einmaliger Sensorausfall modelliert werden, wofür sich noch kein Bedarf nach einer solchen Funktion ergab. Für eine längerfristige und umfassendere Benutzung des Simulators sollte die Option dennoch vorhanden sein, aber nicht irritieren.

Bedienbarkeit: Problematisch waren vor allem einige schlecht gewählte Formulierungen und Platzierungen, die überarbeitet werden mussten.

Ein großes Problem bei der Bedienbarkeit war die unintuitive und dadurch umständliche Navigation zwischen dem Kontext-Tab der Lerneinheiten und dem Simulator. Dieses Problem wurde nach der Studie durch eine Änderung am Autorensystem gelöst: Alle im System verfügbaren Tabs - darunter „Simulator“ - wurden statisch in der Tab-Leiste des Panels platziert. Dadurch wurde eine globale Navigation [NP10] ermöglicht. Der missverständliche Menüpunkt „(Kontext-)Simulation“ konnte danach wieder aus der Hauptnavigation entfernt werden, die eigentlich zum Modellieren der Lernanwendung gedacht war.

Intuitiv bedienbar waren das Starten und Stoppen der Simulation durch den Play/Pause-Knopf sowie das Editieren von Kontext im Popover. Allerdings wurde dort mehr Unterstützung für ein effizienteres Arbeiten gewünscht.

4.6 Konzept für das User-Interface

Nach dem Ende der Paper-Prototyping-Studie wurde anhand aller gewonnenen Erkenntnisse ein in Teilen neues Konzept für das User-Interface (UI) des Simulators erstellt. Das Konzept diente als Grundlage für die Implementierung des ersten digitalen Prototyps [RF13][Sny03] und wird in diesem Abschnitt beschrieben. Die prototypische Realisierung des Konzepts wird im folgenden Kapitel 5 vorgestellt.

4.6.1 Unterstützte Use-Cases

Abbildung 4.3 zeigt alle zu diesem Zeitpunkt identifizierten Use-Cases:

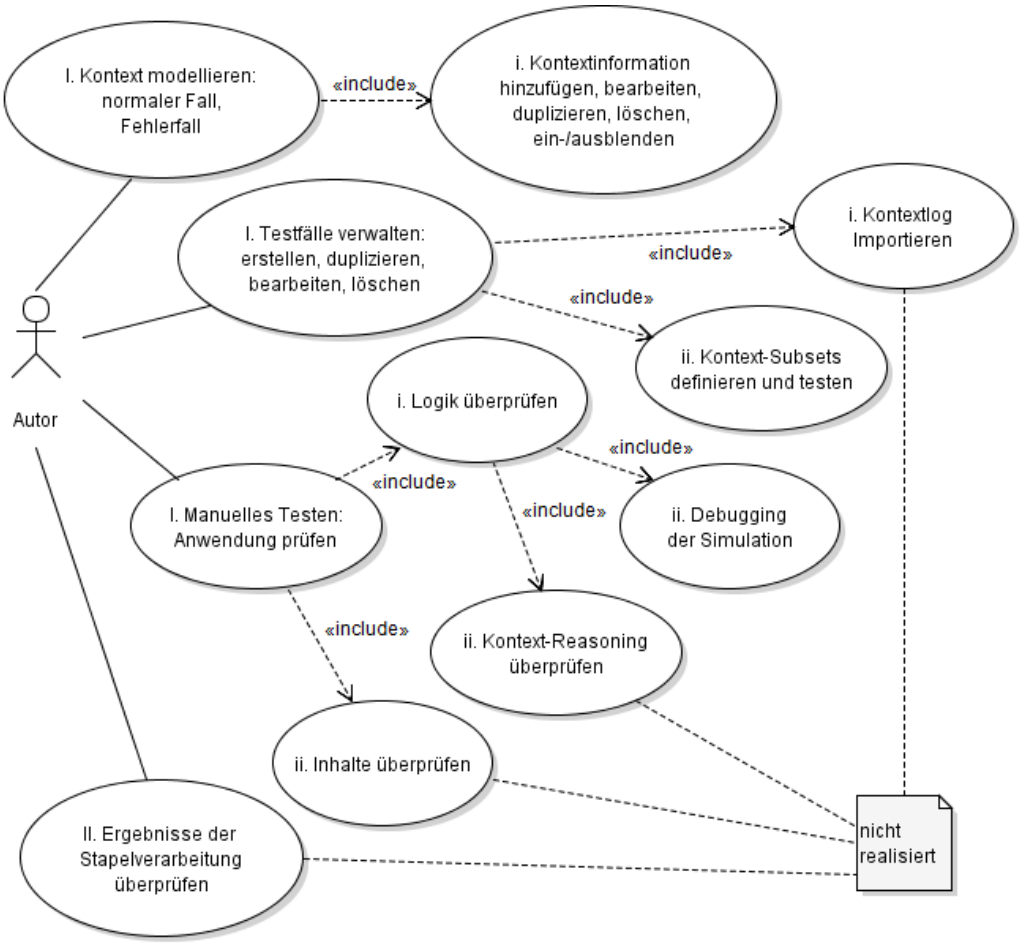


Abbildung 4.3: Modell aller nach dem Paper-Prototyping identifizierten Use-Cases.

Die mit *I.* bzw. *i.* gekennzeichneten Use-Cases sind die, die bereits vor der Studie angenommen wurden (siehe Abschnitt 4.3). Davon wurde das Importieren real gemessener Kontextinformationen als (vorerst) nicht realisierbar eingestuft.

Die mit *II.* bzw. *ii.* gekennzeichneten Use-Cases wurden zusätzlich in der Nutzerstudie identifiziert. Eine Anzeige der Lerninhalte war den Testpersonen zwar wichtig, zu diesem Zeitpunkt aber noch nicht möglich. Weitere Ideen für mögliche Use-Cases bzw. Konkretisierungen der initialen Use-Cases beinhalteten:

- Definieren von Subsets von Kontextinformationen und zugehörigen Testfällen, um die Verfügbarkeit und Berechtigung bestimmter Sensoren zu modellieren.
- Überprüfen („Debugging“) der Kontextmodelle durch ein „Matching“ von Simulationsschritt und Lerneinheit. Dies musste noch manuell und ohne weitere Unterstützung des Systems durchgeführt werden.
- Testen der Kontextinterpretation (Reasoning) im Simulator.
- (Teil-)automatisiertes, kontinuierliches Testen. Die Vision davon beinhaltete die (automatische) Stapelverarbeitung mehrerer Testfälle mit anschließendem (manuellen) Vergleich der dabei generierten Ergebnisberichte.

Die gewünschte Unterstützung für das Debugging, das Testen des Reasoning und das automatisierte Testen konnte im Rahmen dieser Arbeit nicht mehr umgesetzt werden, der Bedarf danach aber festgehalten und in der anschließenden Usability-Studie validiert werden.

Von den realisierbaren Use-Cases wurden dem manuelle Modellieren des Szenariokontextes und Prüfen der Anwendungslogik die höchste Priorität zugewiesen. Dazu gehörten die Funktionen:

- Hinzufügen, Bearbeiten, Löschen und Einstellen des Geltungsbereichs (durch Verschieben und Verlängern/Verkürzen) von Kontextinformationen in der Zeitleiste.
- Abspielen und Pausieren der Simulation und die Benachrichtigung über deren Ergebnis.

Die Verwaltung mehrerer Testfälle wurde ebenfalls als wichtig erachtet, hatte aber weniger hohe Priorität und war somit ein Fall für Progressive

Disclosure (siehe Abschnitt 4.4.3). Dazu zählten das Erstellen, Speichern, Öffnen, Bearbeiten (auch: Umbenennen und Beschreiben) und Löschen von Testfällen.

Das Kopieren von Kontextinformationen und Testfällen sowie die Möglichkeit, Kontextinformationen aus- und einzublenden, wurden als zweitrangig eingestuft. Die Funktionalität dazu wurde dennoch umgesetzt, um die angenommene Priorisierung zu validieren.

4.6.2 UI-Komponenten und Funktionen

Die Benutzerschnittstelle zum Simulator lässt sich grob in zwei Hauptkomponenten unterteilen: In der einen kann der Nutzer interaktiv Testfälle bearbeiten, Kontext modellieren und die Simulation steuern. In der anderen kann er während der Simulation das Verhalten der Adaptierung, das heißt, die Auswahl der Lerneinheiten überprüfen und Nachrichten des Systems rezipieren. In Abbildung 4.4 ist die Aufteilung grob skizziert.

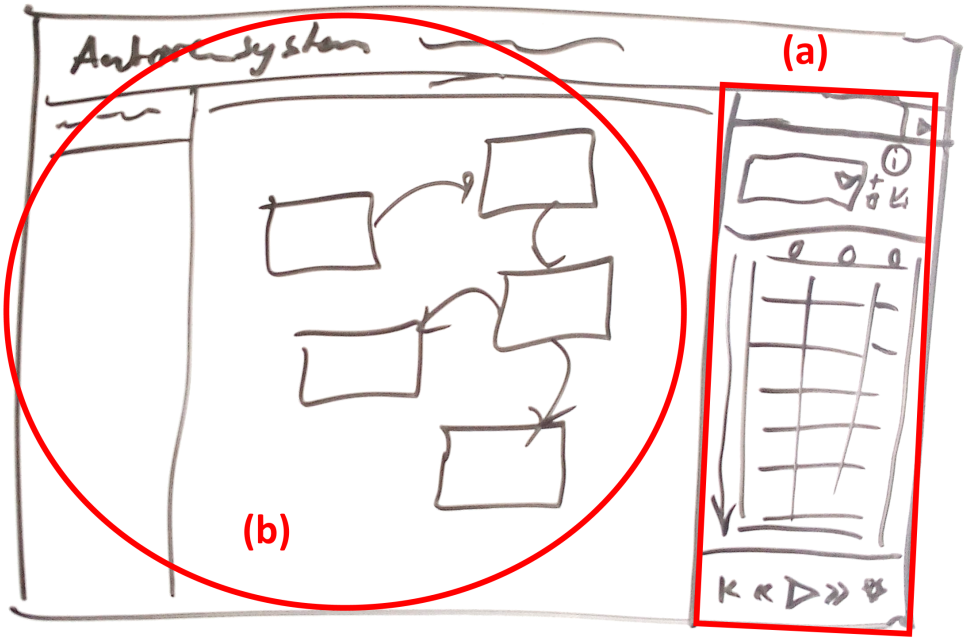


Abbildung 4.4: Skizze der groben Bildschirmaufteilung: (a) Interaktive Komponente, (b) Visualisierungskomponente.

Komponente für Modellierung und Steuerung

Der interaktive Hauptteil des Simulators sollte als Tab im Eigenschaften-Panel des Autorensystems untergebracht werden. Die Hauptansicht des Simulator-Tabs ist vertikal in drei Bereiche aufgeteilt, die in Abbildung 4.5 skizziert sind und im Folgenden beschrieben werden.

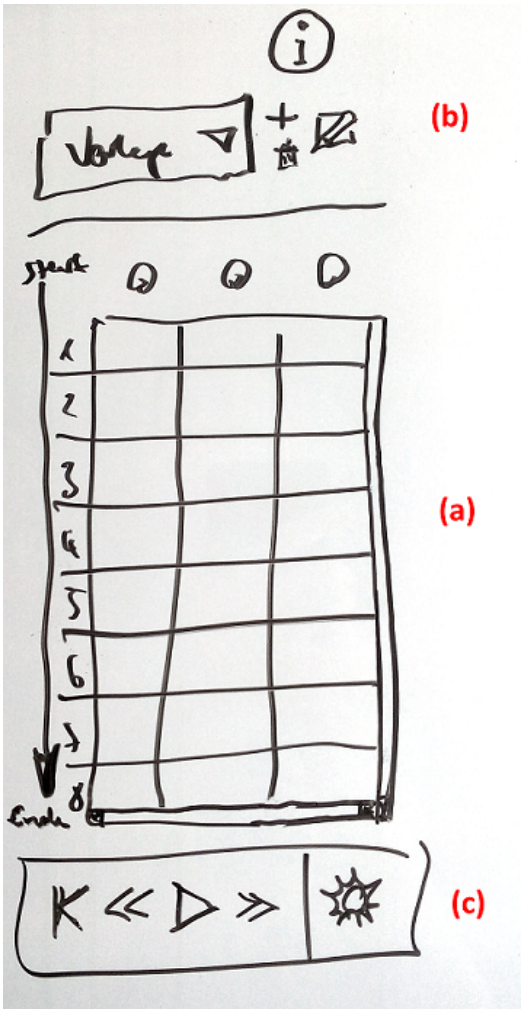


Abbildung 4.5: Skizze des Simulator-Tabs: (a) Zeitleiste, (b) Kopfleiste, (c) Kontrollleiste.

a) Zeitleiste: Zentral befindet sich die Kernkomponente des Simulators: die tabellarische Zeitleiste, die über den meisten Platz verfügt. Darin wird der

Kontext modelliert. Für jede Kontextinformation des aktuell bearbeiteten Szenarios wird eine Spalte angezeigt. Einen festen Platz in der Zeitleiste erhält eine Spalte für das Eintragen abgeschlossener Lerneinheiten. Dadurch, dass die Belegung der Spalten nun vorgegeben wird, können die Kontext-Icons aus den Feldern ausgelagert werden. Sie sind stattdessen statisch in den Spaltenköpfen platziert (in der Skizze kreisförmig über der Tabelle dargestellt). Ein Mouseover auf dem Icon zeigt den Namen der Kontextinformation an. Jede Spalte ist zusätzlich zu Icon und Tooltip in einem helleren Ton der Farbe der Kontextklasse eingefärbt, die dem Anwender aus dem Autorensystem bekannt sein sollte.

Die Felder der Tabelle, die *Zeitfenster*, lassen sich direkt manipulieren: Bei Mouseover über einem Zeitfenster werden Name, ggf. Parameter und der Wert der enthaltenen Kontextinformation in einem Tooltip angezeigt. Dazu wird ein Super-Tooltip verwendet, um den Wert sofort, ohne Verzögerung, und gut lesbar formatiert einsehen zu können.

Durch einfaches Klicken oder Klicken, Halten und nach unten Ziehen der Maus werden ein oder mehrere Zeitfenster markiert. Der Cursor wird dazu bei Mouseover über einem leeren Zeitfenster als Zeiger mit einem „+“ angezeigt. Beim Loslassen der Maustaste öffnet sich an der Stelle ein Popover-Fenster, in dem Wert und ggf. Parameter der ausgewählten Kontextinformation festgelegt werden. Gegenüber einfachen Tooltips erlauben sie mehr Interaktion, bieten aber den gleichen Vorteil, dort platziert zu werden, wo sie benötigt werden. Dadurch wird die kognitive Belastung für den Benutzer reduziert und er kann seine Arbeitsschritte effizienter ausführen. Die Auswahl bzw. Eingabe der Parameter und Kontextwerte ist so gestaltet wie im - den Nutzern des Autorensystems bereits bekannten - Tab zum Editieren der Kontextinformationen der Lerneinheiten. Abbildung 4.6 zeigt zum Vergleich eine Skizze des Popover-Fensters und den erwähnten Kontext-Tab.

Je nach erwartetem Wert wird mit einem Text- oder Nummern-Eingabefeld oder einer Drop-down-Auswahlliste interagiert. Ein Unterschied besteht darin, dass es im Popover logischerweise keine Auswahl von Kontextinformation oder Operator gibt. Außerdem ist im Popover bei Werten aus einer Auswahlliste der erste Wert vorselektiert.[14]

Wird der eingetragene Wert bestätigt, wird überprüft, ob alle erwarteten Eingaben (Wert und ggf. Parameterwerte) getätigt wurden. Diese Prüfung und ggf. die Rückmeldung, dass eine Eingabe fehlt, erfolgen so wie auch im Kontext-Tab: Das Eingabefeld wird dann rot umrandet und ein kleines *alert*-Fenster blockiert den Bildschirm. Sind alle erwarteten Informationen

[14]Im Paper-Prototyping waren Default-Werte gewünscht worden.

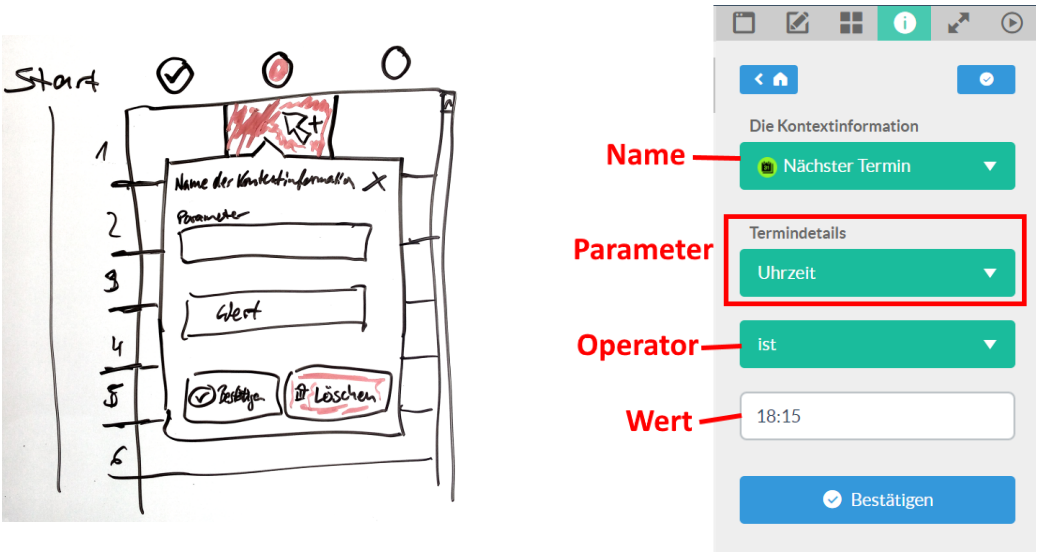


Abbildung 4.6: Links: Skizze des Popover-Fensters. Rechts: Kontext-Tab

vorhanden, schließt sich das Popover und die ausgewählten Zeitfenster werden in genau dem Farbton der Kontextklasse (mit derselben Intensität) eingefärbt. Ein Mouseover auf den Zeitfenstern zeigt dann statt „[Kontextinformation] ist unbekannt“ die eingetragenen Werte an. Zudem werden die im Paper-Prototyping gewünschten Quick-Access-Icons im Zeitfenster untergebracht. Abbildung 4.7 zeigt eine Skizze dieser Optionen (von links nach rechts): Bearbeiten, Löschen, Ausblenden („Wert nicht erfassen“) und Kopieren.

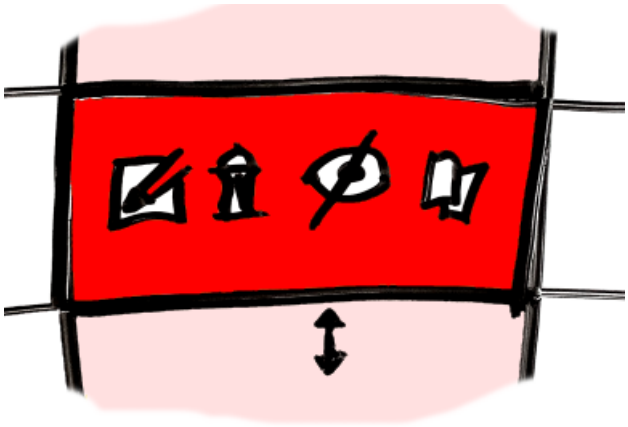


Abbildung 4.7: Skizze eines Zeitfensters mit festgelegtem Kontextwert.

Ein Klick auf das Bearbeiten-Icon öffnet wiederum das Popover. Löschen

leert das Zeitfenster wieder. Bei Klick auf „Wert nicht erfassen“ wird das Zeitfenster als ausgeblendet oder deaktiviert angezeigt (durch ein Overlay oder eine Schraffur) und das Auge-Icon nicht mehr durchgestrichen, sodass die nächste mögliche Aktion „Wert erfassen“ sichtbar wird. Bei Klick auf Kopieren wird das Zeitfenster als kopiert markiert (gestrichelt umrandet). Das Einfügen erfolgt dann durch Klick auf ein freies Feld. Neben den vier Optionen kann das Zeitfenster (bzw. der Gültigkeitszeitraum des Kontextwerts) auch durch Drag-ad-drop verlängert (und wieder verkürzt) und innerhalb der jeweiligen Spalte verschoben werden. Dazu wird der Cursor beim Mouseover über dem Zeitfenster als *move*-Pfeil und über der unteren Kante als *sw-resize*-Pfeil (siehe Skizze in Abbildung 4.7) angezeigt.

In der Spalte, in der die abgeschlossenen Lerneinheiten angegeben werden, verhält es sich etwas anders: Darin stehen Kopieren und das Verlängern nicht zur Verfügung, da eine einmal abgeschlossene Lerneinheit immer abgeschlossen bleibt. Dafür wird im Tooltip zu jedem Zeitfenster eine Liste aller zu diesem Zeitpunkt abgeschlossenen Lerneinheiten angezeigt.

b) Kopfleiste: Über dem Container der Zeitleiste befinden sich ein Drop-down-Menü aller Testfälle, ein Button zum Bearbeiten des ausgewählten Testfalls, Buttons bzw. Icons zum Erstellen und Löschen eines Testfalls sowie eine Hilfe zum Simulator. Die Hilfe soll kurz erklären, wozu der Simulator verwendet werden kann. Sie wird ebenfalls in einem Popover-Fenster angezeigt. Ein Klick auf den Bearbeiten-Button öffnet eine zweite Ansicht des Tabs, in der Name und Beschreibung des Testfalls geändert werden können. Diese Ansicht soll gleich sein mit der des Einstellungen-Tabs für Lerneinheiten: Sie enthält ein Texteingabefeld für den Namen und eines für die Beschreibung sowie oben links einen Zurück-Button. Gespeichert wird jede Änderung automatisch. Eine hinzugefügte Beschreibung wird im Drop-down-Menü der Testfälle in der Hauptansicht bei Mouseover über dem Menüeintrag (Name des Testfalls) als Tooltip angezeigt, um dem Nutzer einen schnellen Überblick über den Inhalt zu geben.

c) Kontrollleiste: Unter der Zeitleiste befindet sich die Kontrollleiste zum Abspielen und Steuern der Simulation sowie ein Button für weitere Einstellungen. Der Benutzer kann zum Starten der Simulation auf „Play“ drücken. Der Button wird dann zum Pausieren-Button, bis pausiert wird oder die Simulation komplett durchgelaufen ist. Dann wird er auf „Play“ zurückgesetzt. Ein Klick auf den Button mit dem Zahnrad-Symbol für Einstellungen öffnet eine weitere Ansicht des Tabs, die der für die Testfall-Eigenschaften ähnlich

ist: Sie enthält oben links einen Zurück-Button und darunter ein Nummer-Eingabefeld mit der Beschriftung „Abspielgeschwindigkeit (in Sekunden pro Situation)“. Die Simulation schreitet pro simulierter Situation nach der unter Optionen angegebenen Zeit (in Sekunden) weiter zur nächsten Situation. Dieser Fortschritt wird durch das Markieren der entsprechenden Zeile in der Zeitleiste (a) angezeigt. Während der Simulation kann zwischen den Schritten vor- und zurückgesprungen werden. Außerdem kann durch Klick auf die Nummer einer Zeile in der Zeitleiste (a) direkt zu einem Simulationsschritt gesprungen werden. Der Button „Zurück zum Start“ hat eine Stopp-Funktion und soll klar machen, dass die Simulation danach beim nächsten Play wieder von vorne beginnt.

Komponente für Ergebnisvisualisierung

Klickt der Nutzer in der Kontrollleiste auf Start, wird er in einem Modal- bzw. Popupfenster über die Regelgenerierung benachrichtigt, da diese einige Sekunden dauern kann. Das Fenster schließt sich automatisch wieder, sobald die Simulation startet.

Die Ergebnisse der Simulation, die Auswahl der Lerneinheiten, werden durch eine Lightbox visualisiert (siehe Abschnitt 4.4.2). Beim Start der Simulation wird die Oberfläche abgedunkelt. Dazu wird ein transparentes dunkles Overlay darüber gelegt. Allein der Simulator-Tab bleibt frei, um dem Nutzer das Steuern der Simulation zu ermöglichen.

Werden in einem Schritt eine oder mehrere Lerneinheiten ausgewählt, werden sie auf das Overlay gehoben, erhalten einen weißen Hintergrund und zusätzlichen breiten farbigen Rand, um zu „leuchten“. Schreitet die Simulation weiter und wählt die Lerneinheiten nicht mehr aus, wird dieses Highlighting wieder aufgeboben.

Wird in einer Simulation keine Lerneinheit ausgewählt, so wird wiederum ein Popup-Fenster mit einer entsprechenden Benachrichtigung angezeigt: „Ergebnis der Simulation: Es konnten keine Lerneinheiten gefunden werden, die zu den Situationen des Szenarios passen.“

5 Implementierung

Das im vorigen Abschnitt 4.6 vorgestellte Konzept wurde prototypisch als Bestandteil des Autorensystems implementiert.[1] Der digitale Prototyp sollte interaktiv bedienbar sein und „echte“ Daten verarbeiten. Das heißt, dass sich die im Autorensystem erstellten Lernszenarien bereits testen lassen sollten. Gleichzeitig wurde wegen der parallelen Weiterentwicklung des Autorensystem-Prototyps eine möglichst lose Kopplung der beiden Software-Komponenten angestrebt.

In diesem Kapitel wird dargelegt, wie die UI-Komponenten und Funktionen des Simulator-Prototyps umgesetzt wurden. Dazu wird zunächst auf ausgewählte Aspekte der Implementierung eingegangen und anschließend werden die Ergebnisse präsentiert.

5.1 Architektur

Bei der Implementierung des Simulators wurde sich am (groben) Model-View-Controller-Muster des Autorensystems orientiert.

Model: speichert und verarbeitet Daten wie die Kontextinformationen, Eigenschaften und Relationen der Lerneinheiten - unabhängig von deren Darstellung in der Oberfläche (View).

View: wird durch HTML, CSS und JavaScript realisiert. HTML und CSS für die (statische) Darstellung, JavaScript für die dynamische Interaktion mit den Elementen der Oberfläche (siehe Abschnitt 5.1.3). Der Kern der Anwendung, die Zeitleiste, wurde als Tabelle implementiert.

Controller: steuern den Datenfluss zwischen Model und View. Benutzeraktionen (User-Events) werden beobachtet und entsprechend reagiert. Zum Beispiel wird ein angefragter Wert aus dem Datenmodell in der Oberfläche angezeigt oder umgekehrt das Datenmodell an Eingaben angepasst.

[1] Der Prototyp inklusive aller benötigten Komponenten ist dieser Arbeit in elektronischer Form beigelegt. In dem Verzeichnis („autorensystem“) findet sich auch eine „README“-Datei mit Anleitungen zu Installation und Start der Anwendung.

Model und Controller sind in JavaScript implementiert, die Daten werden im JSON-Format übertragen und persistiert.

5.1.1 Klassen

Die Abbildungen 5.1 und 5.2 geben einen Überblick über die für die Implementierung des Prototyps relevanten Klassen[2], die im Folgenden kurz vorgestellt werden.

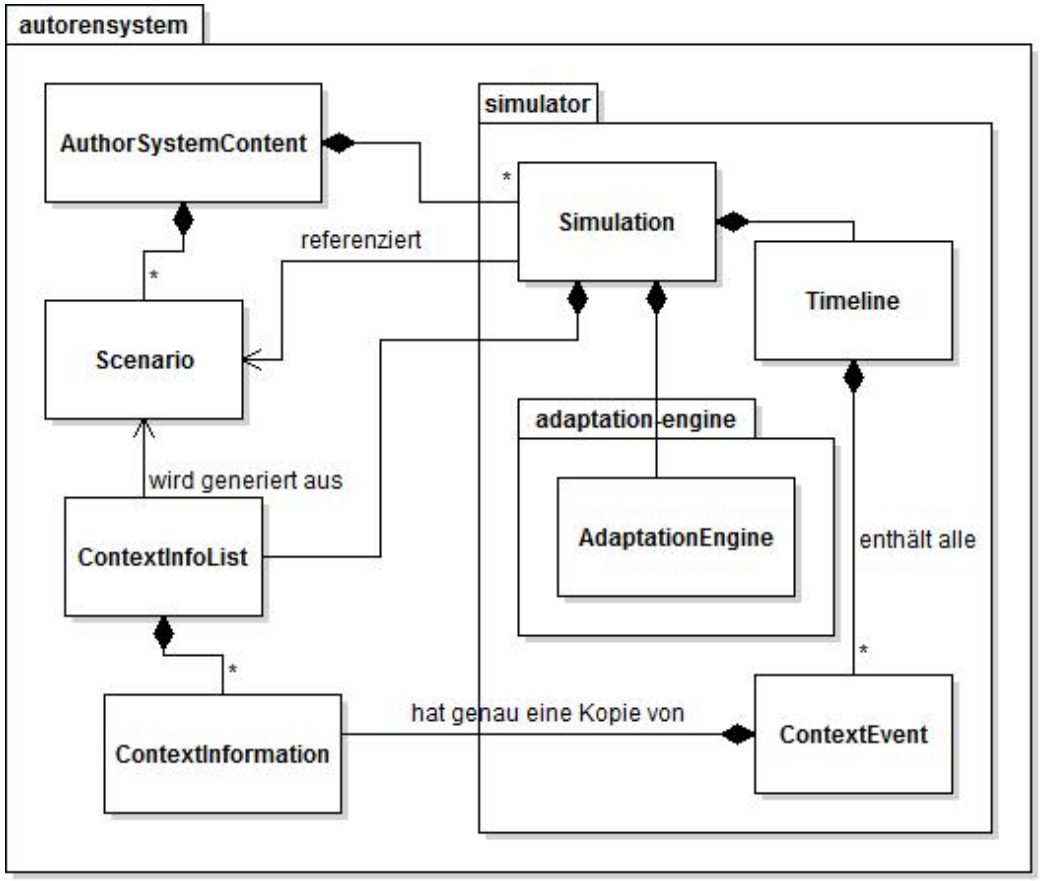


Abbildung 5.1: Klassendiagramm der Modelle für die Simulation: Übersicht.

Einem objektorientierten Ansatz folgend, wurde das Datenmodell für den Simulator in drei Klassen unterteilt realisiert: `Simulation`, `Timeline` und `ContextEvent`.

[2] Da JavaScript eine klassenlose, prototypenbasierte Sprache ist, wurden die „Klassen“ durch ebensolche „Prototypen“ realisiert. Zum besseren Verständnis wird hier aber der aus der objektorientierten Programmierung geläufigere Begriff der „Klassen“ verwendet.

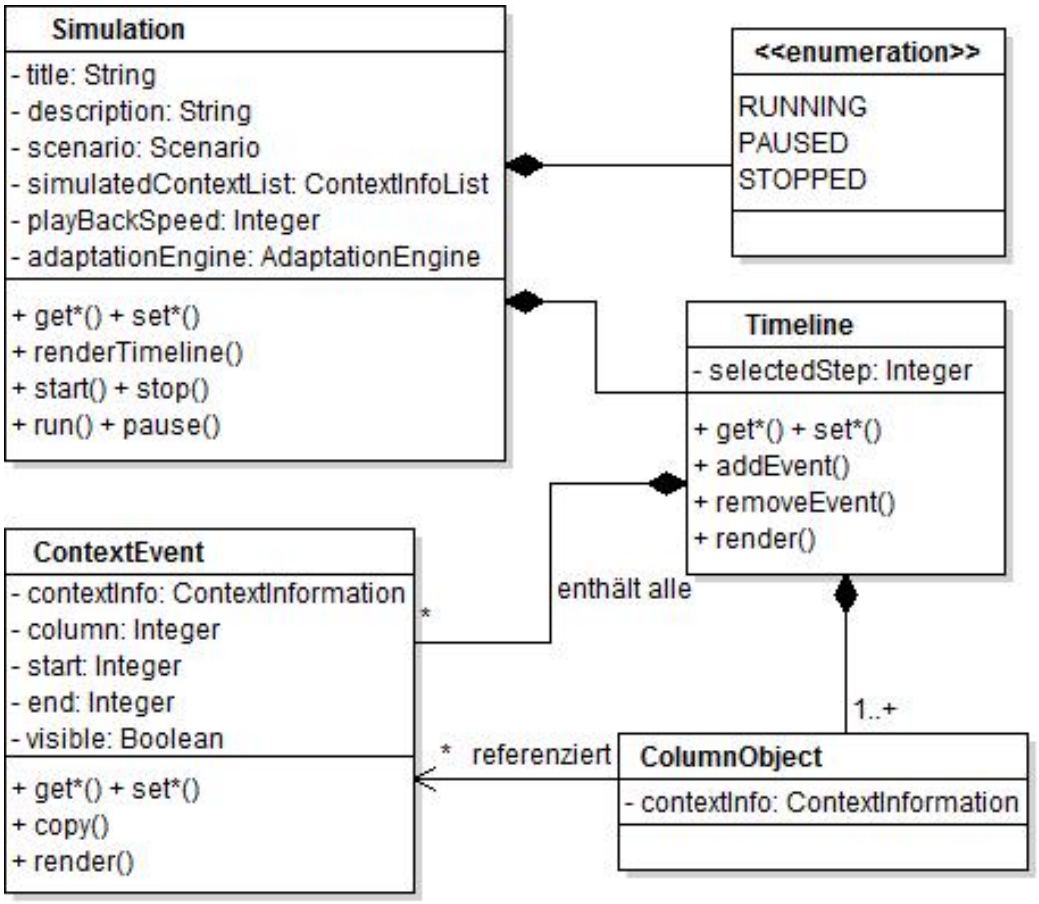


Abbildung 5.2: Klassendiagramm der Modelle für die Simulation: Detailansicht.

- Jede `Simulation` hat - neben Eigenschaften wie Titel, Beschreibung, Abspielgeschwindigkeit und Status - genau eine `Timeline`.

- Die `Timeline` verfügt über alle angelegten `ContextEvents`. Diese sind (anonymen) Objekten zugeordnet, die jeweils eine Kontext-Spalte repräsentieren. Jede Timeline hat mindestens eines dieser Objekte, da die erste Spalte für abgeschlossene Lerneinheiten immer vorhanden ist.

- Ein `ContextEvent` beinhaltet genau eine Kontextinformation (siehe nächsten Absatz) und die Attribute: Start, Ende, Spalte und Sichtbarkeit[3].

[3] „Sichtbar“ heißt hier, dass der Kontextwert in der Simulation „erfasst“ wird.

Zusätzlich wird auf Objekte der Klassen des Autorensystems zurückgegriffen: `AuthorSystemContent`, `Scenario`, `ContextInformation` und `ContextInfoList`. Eine Instanz des `AuthorSystemContent` enthält eine Liste aller Lernszenarien (`Scenario`) und nun auch eine Liste aller Testfälle (`Simulation`). Jede `Simulation` referenziert genau ein `Scenario` und hat genau eine `ContextInfoList` der zu simulierenden Kontextinformationen, die aus dem referenzierten Szenario gefiltert werden. Aus dieser Liste werden die Spalten der `Timeline` erzeugt werden. Jedes `ContextEvent` der `Timeline` hat dann genau eine Kopie einer `ContextInformation` aus dieser Liste (siehe Abschnitt 2.2.3).

5.1.2 Schnittstellen

Der Prototyp des Simulators hat neben dem User-Interface drei Schnittstellen: eine zum Autorensystem, eine zum Server und eine zur Adaptation Engine.

Der Simulator wird aus dem Autorensystem heraus initialisiert und bezieht von diesem auch die Regeln für die Adaptierung während der Simulation. Sein HTML-Gerüst wird über ein GET-Request vom Server geladen.

Das Autorensystem selbst kommuniziert über eine REST-Schnittstelle mit Server und Regelgenerator. Der Regelgenerator wird über ein POST angefragt, dem die Ontologie mitgegeben wird und der die Regeln zurückgibt (siehe auch Abschnitt 2.3.2). Beim Start der Simulation wird die Methode `exportScenario(callback)` aufgerufen, die das Autorensystem zu diesem Zweck bereitstellt. Der Parameter `callback` beinhaltet die gesamte Funktionalität, die zum Ausführen der Simulation benötigt wird: Die Adaptation Engine wird dynamisch geladen und initialisiert mit den empfangenen Regeln. An der Stelle werden auch die Callbacks für die Adaptierung übergeben, das heißt, die Funktionen, die ausgeführt werden sollen, sobald in der Adaptation Engine ein bestimmtes Ereignis eintritt. Hier ist das zum einen die Erfassung einer Kontextinformation sowie zum anderen das „Triggern“ einer Regel, deren Bedingungen erfüllt sind. Der Callback für letztgenanntes Ereignis ermöglicht die Visualisierung der ausgewählten Lerneinheiten.

Pro Simulationsschritt werden der initialisierten Adaptation Engine die im Simulator modellierten Kontextinformationen übergeben, die sie an ContactJS weiterreicht (siehe 2.3.4). Der Callback für die Kontexterfassung bewirkt in dem Moment, dass zuvor übergebene Kontextinformationen gelöscht werden. Nach der Kontextübergabe wird die Regelauswertung durch die Rule Engine einmalig angestoßen. Wird dabei eine Regel erfüllt, wird der entsprechende Callback mit dem UUID der ausgewählten Lerneinheit als Parameter aufgerufen. Darüber wird diese dann im User-Interface als ausgewählt angezeigt.

Abbildung 5.3 zeigt, wie die Kommunikation zwischen dem Simulator und den angeschlossenen Komponenten abläuft, wenn eine Simulation gestartet wird.

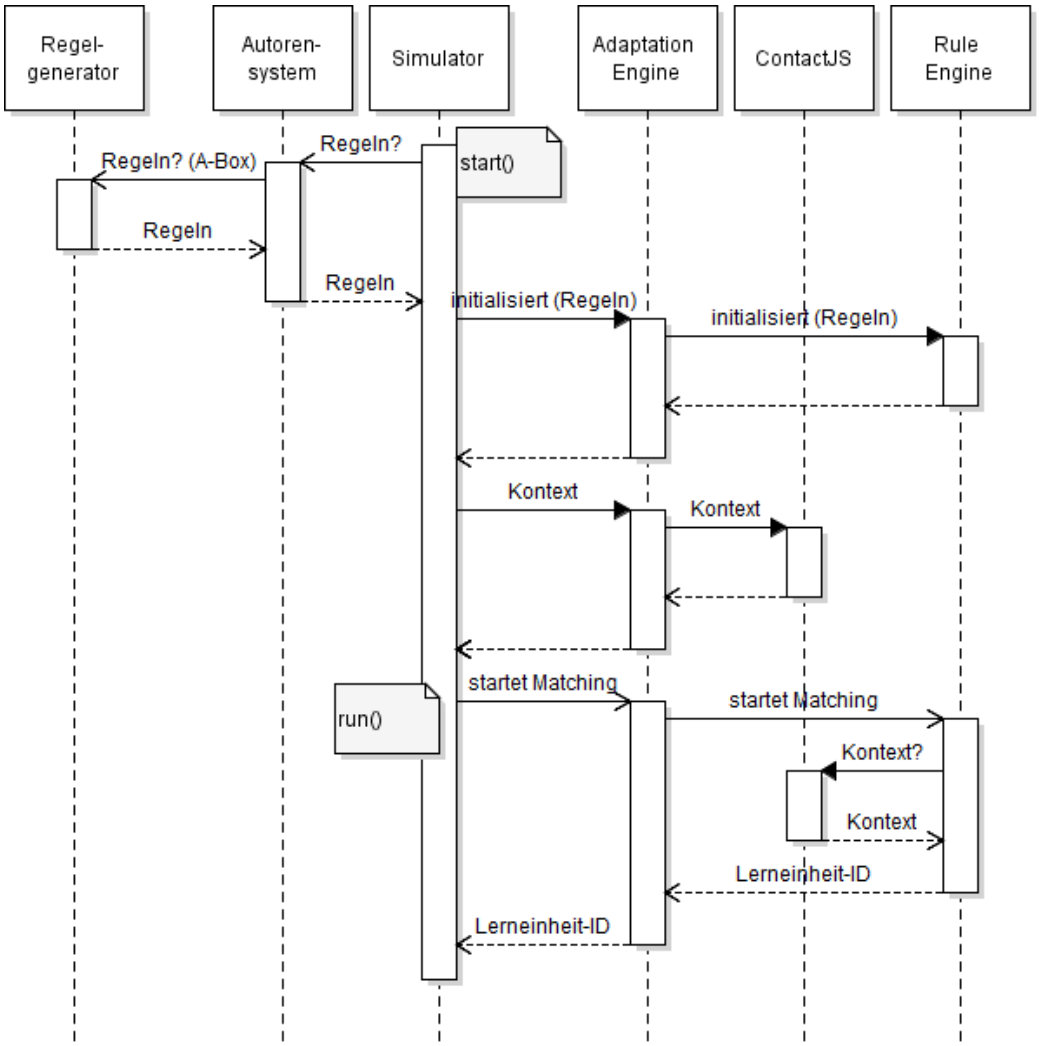


Abbildung 5.3: Sequenzdiagramm der Kommunikation über die Schnittstellen des Simulators.

5.1.3 Verwendete Bibliotheken

Der Autorensystem-Prototyp fußt auf jQuery[4], einem Framework zur Manipulation des HTML-DOM (Document Object Model). Es wird für die Interaktion mit Elementen der Oberfläche verwendet, um auf User-Events wie Klicks oder Mausbewegungen zu reagieren und die Darstellung dynamisch anzupassen. Außerdem ermöglicht jQuery die AJAX-basierte (Asynchronous JavaScript and XML) Kommunikation mit dem Server, das heißt, den Datenaustausch über die REST-Schnittstelle.

Um ein konsistentes „Look and Feel“ mit dem Autorensystem zu erreichen, wurde auf dieselbe Bibliothek zur Darstellung der Oberflächenelemente zurückgegriffen: Flat UI[5] basiert selbst auf Bootstrap[6] und das wiederum auf jQuery. Verschiedene Plugins ermöglichen die Darstellung von bzw. Interaktion mit Super-Tooltips, Icons, Buttons, Popover- und Modal-Dialogfenstern sowie Drop-down-Menüs, die im Simulator-Prototyp zum Einsatz kommen.

Außerdem wurde RequireJS[7] benötigt, ein Framework für das dynamische Laden von JavaScript-Modulen. Die Adaptation Engine wurde mithilfe dieses Frameworks implementiert und in den Simulator eingebunden.

5.2 Benutzerschnittstelle

Die Abbildungen 5.4 bis 5.14 zeigen Screenshots der umgesetzten Komponenten, die im vorigen Kapitel 4 beschrieben wurden.[8]

Die Anordnung und Darstellung einiger Elemente wurden im Zuge der Implementierung aus Gründen der Übersichtlichkeit, Ästhetik und Praktikabilität nochmals leicht modifiziert. So wurde etwa der vertikale Pfeil als Indikator der Ablaufrichtung aufgrund der begrenzten realen Breite des Panels wieder verworfen. Stattdessen wurden mehr Beschriftungen („Wählen Sie einen Testfall“, „Szenario-Zeitleiste“) und eine zusätzliche Hilfe an der Zeitleiste eingeführt. Für die Hilfen wurden statt des „i“-Buttons „?“ verwendet, da angenommen wurde, dass ein „?“ eher auf Hilfe schließen lässt und auch aus dem Autorensystem bekannt war.

Die Optionen zum Verwalten der Testfälle (Erstellen, Löschen, Bearbeiten)

[4] `http://jquery.com/` [Abgerufen am 05.05.2016]
[5] `http://designmodo.github.io/Flat-UI/` [Abgerufen am 05.05.2016]
[6] `http://getbootstrap.com/` [Abgerufen am 05.05.2016]
[7] `http://requirejs.org/` [Abgerufen am 05.05.2016]
[8] Hier wird der letzte Stand des UI-Prototyps präsentiert, in den einige Verbesserungen eingearbeitet worden sind, die sich in der Nutzerstudie als dringend notwendig gezeigt hatten.

wurden schließlich doch um die Option Duplizieren erweitert und ebenfalls aus Platzgründen verlagert: Statt die Funktionsicons neben der Testfall-Auswahl im Kopfbereich zu platzieren, wurden größere, beschriftete Buttons für das Duplizieren und Löschen in der Einstellungen-Ansicht hinzugefügt. Diese Ansicht ist über einen einzigen Button erreichbar.[9] Zum Erstellen eines neuen Testfalls wurde im Drop-down-Menü aller Testfälle ein dedizierter, fester Eintrag „+ Neue Vorlage“ am unteren Ende der Auswahlliste eingefügt.[10]

Außerdem wurden noch zusätzliche Optionen für Zeilen und Spalten implementiert, die ursprünglich nicht Teil des Konzepts waren, aber teils nützlich, teils notwendig schienen. Im Autorensystem-Prototyp sind zum Beispiel einige komplexe Kontextinformationen wie „Nächster Termin“ noch so modelliert, dass für jeden Parameter eine eigene Kontextinformation angelegt werden muss. Um diese Informationen in der Zeitleiste abbilden zu können, ergab sich eine Notwendigkeit, weitere Spalten manuell hinzuzufügen. Dies ist aber wenig intuitiv. Im Konzept sollte jeder komplexen Information eine Spalte zustehen.

Ausgeblendete Kontextinformationen, die in der Simulation nicht erfasst werden sollen, werden diagonal schraffiert angezeigt, ausgewählte Zeilen und Spalten etwas dunkler hervorgehoben. Es wurde darauf geachtet, dass der Cursor immer die jeweils möglichen Aktionen indiziert wie Verschieben, Ziehen oder Klicken.

[9] Der Button zur Navigation zeigte erst das Zahnrad-Symbol für Einstellungen, wurde aber wieder geändert, weil das schon in der Kontrollleiste vorkommt.

[10] Diese Designentscheidung ging auf die Anregung eines Probanden im Paper-Prototyping zurück, der sich beim Erkunden des Autorensystems eine solche Option im Szenario-Menü (linkes Panel) gewünscht hatte.

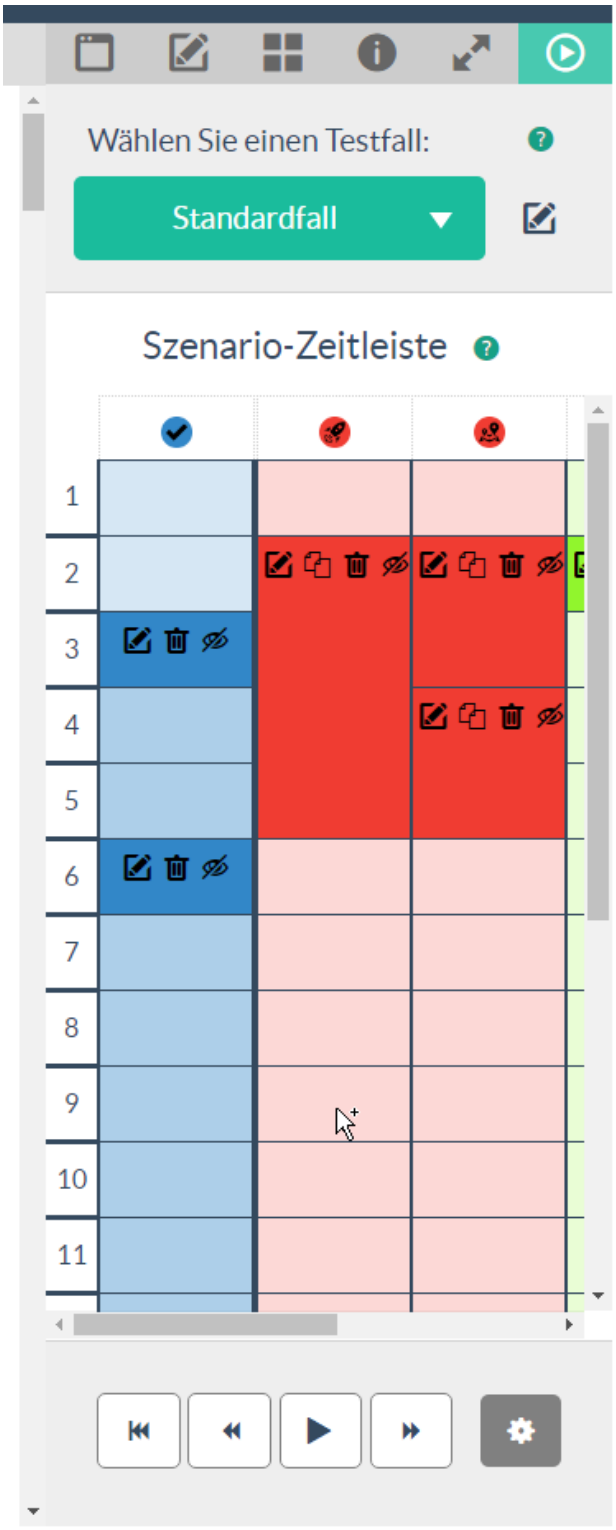


Abbildung 5.4: Simulator-Tab mit Kopfleiste (oben), Zeitleiste (mittig) und Kontrollleiste (unten).

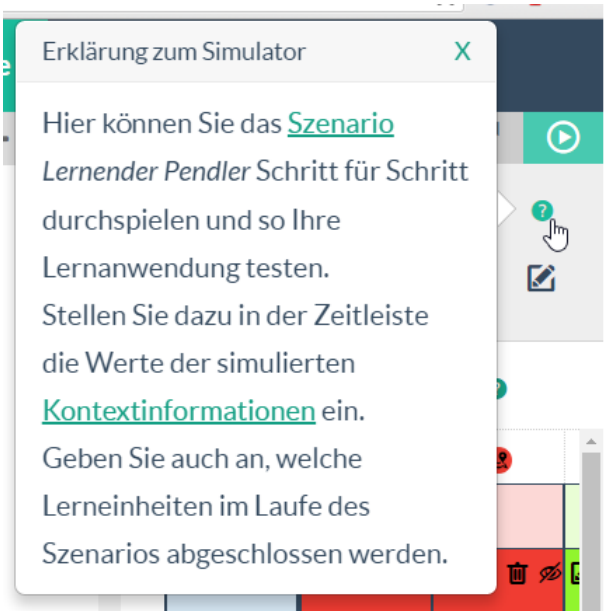


Abbildung 5.5: Hilfe-Popover: Erklärung zum Simulator.

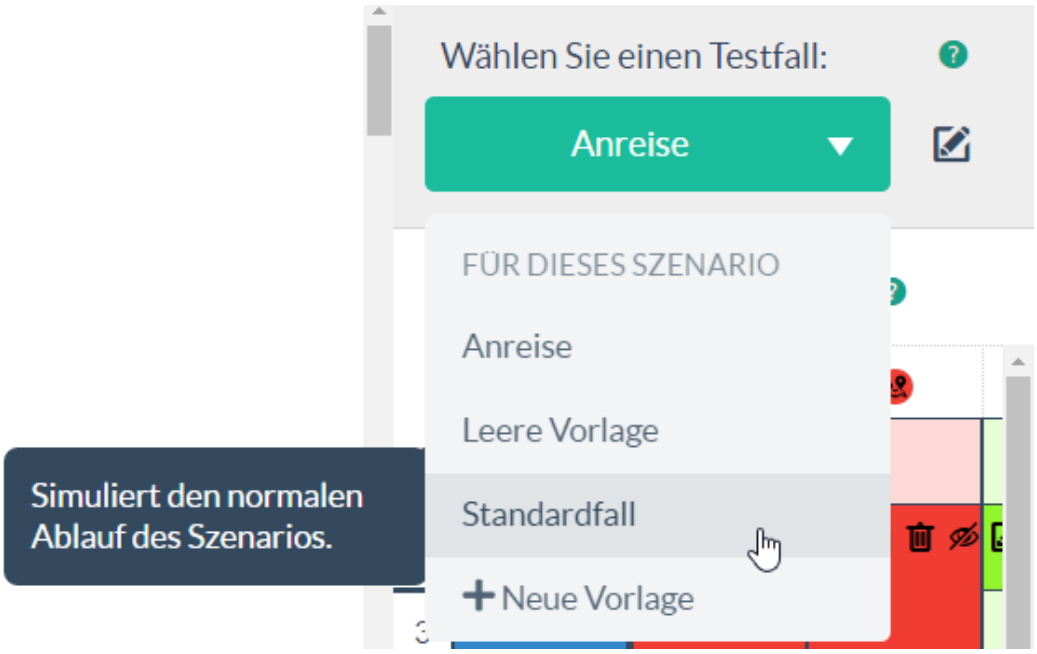


Abbildung 5.6: Drop-down-Menü aller Testfälle mit Erstellen-Option und Beschreibung im Super-Tooltip.

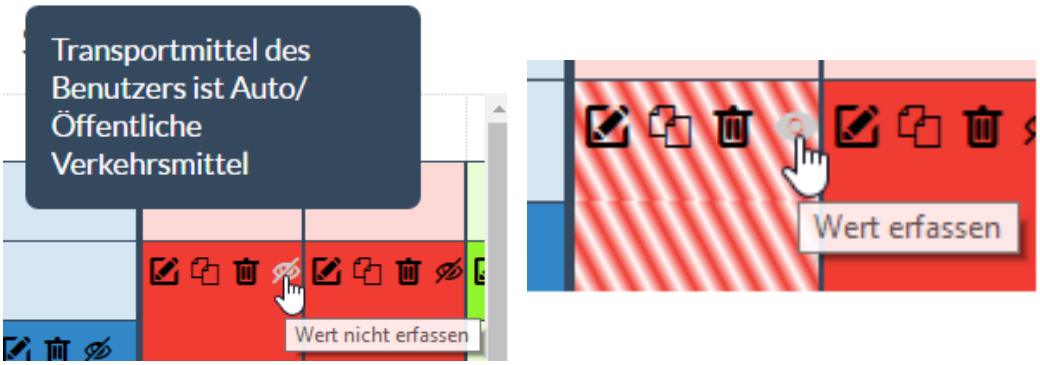


Abbildung 5.7: Links: Super-Tooltip und Ausblenden einer Kontextinformation. Rechts: Einblenden einer Kontextinformation.

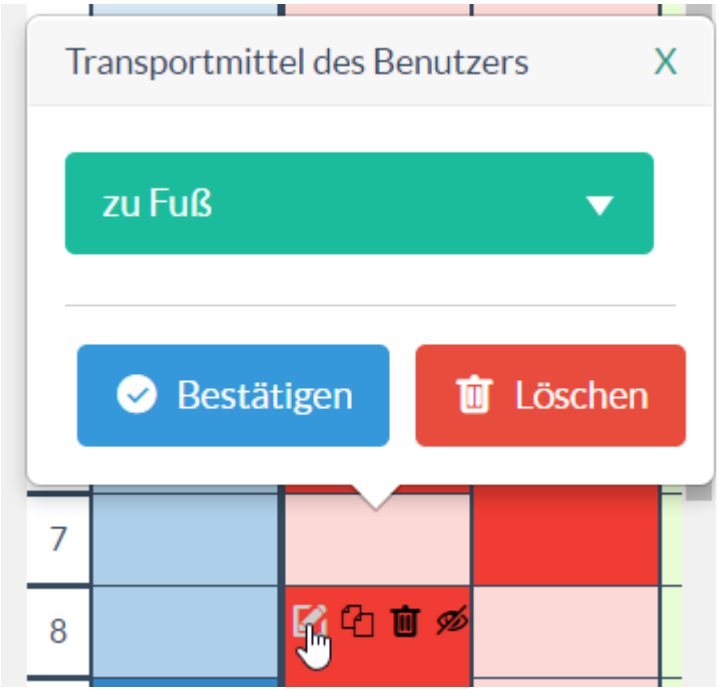


Abbildung 5.8: Popover-Fenster zum Editieren einer Kontextinformation und Quick-Access-Icons.

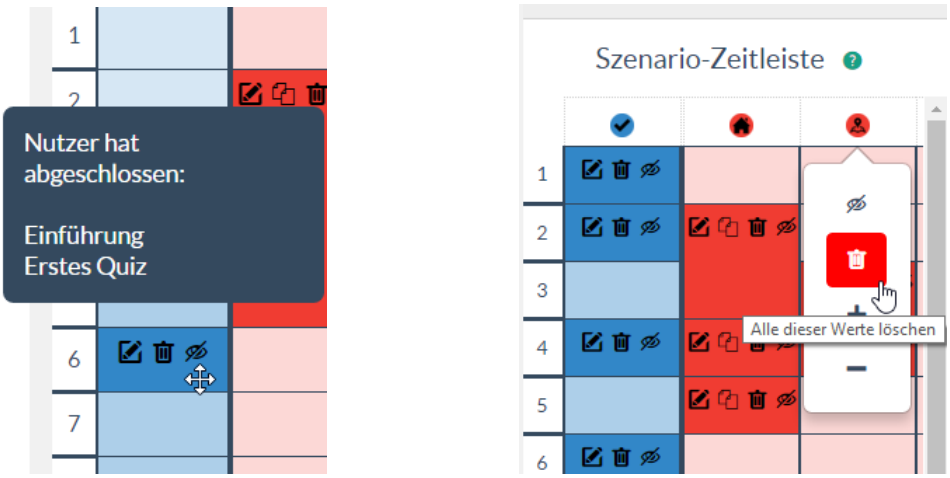
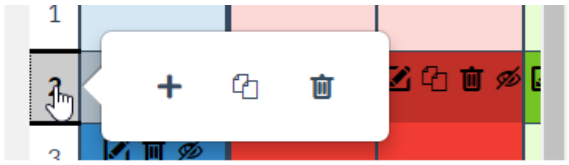
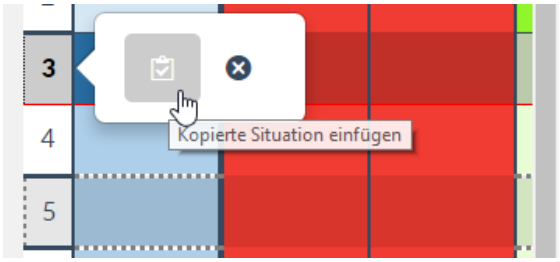


Abbildung 5.9: Links: Super-Tooltip an Zeitfenster der 1. Spalte: Auflistung aller zum ausgewählten Zeitpunkt abgeschlossenen Lerneinheiten. Rechts: Optionen für Spalten der Zeitleiste (von oben nach unten): Ausblenden, Löschen, Spalte hinzufügen, Spalte entfernen.



(a)



(b)

Abbildung 5.10: (a) Optionen für Zeilen der Zeitleiste (von links nach rechts): Zeile einfügen (unterhalb), Zeileninhalt kopieren, Zeile löschen. (b) Optionen für Zeilen der Zeitleiste nach Kopieren einer Zeile (von links nach rechts): Einfügen (unterhalb), Abbrechen.

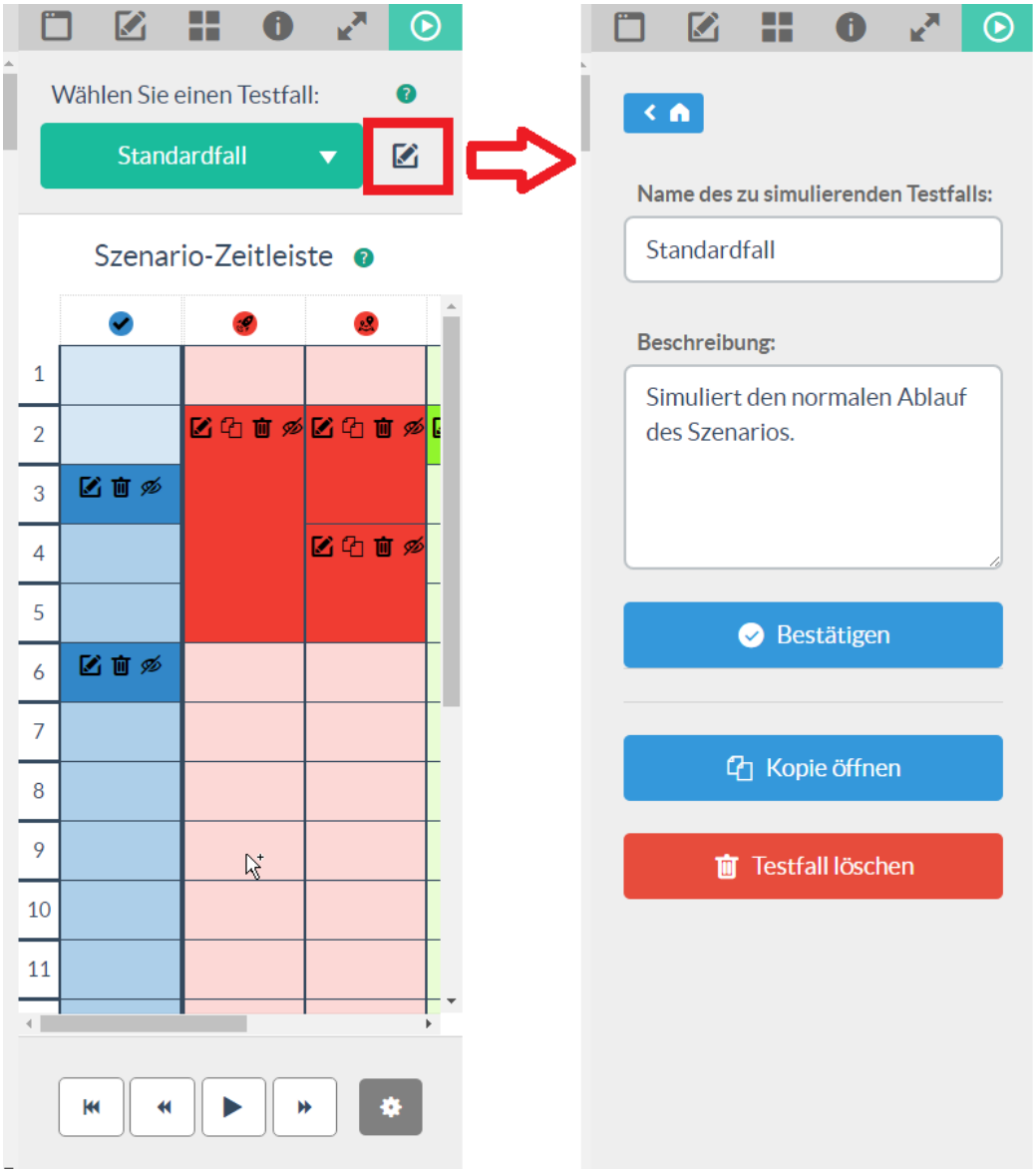


Abbildung 5.11: Navigation zur Ansicht der Eigenschaften eines Testfalls und weiterer Optionen.

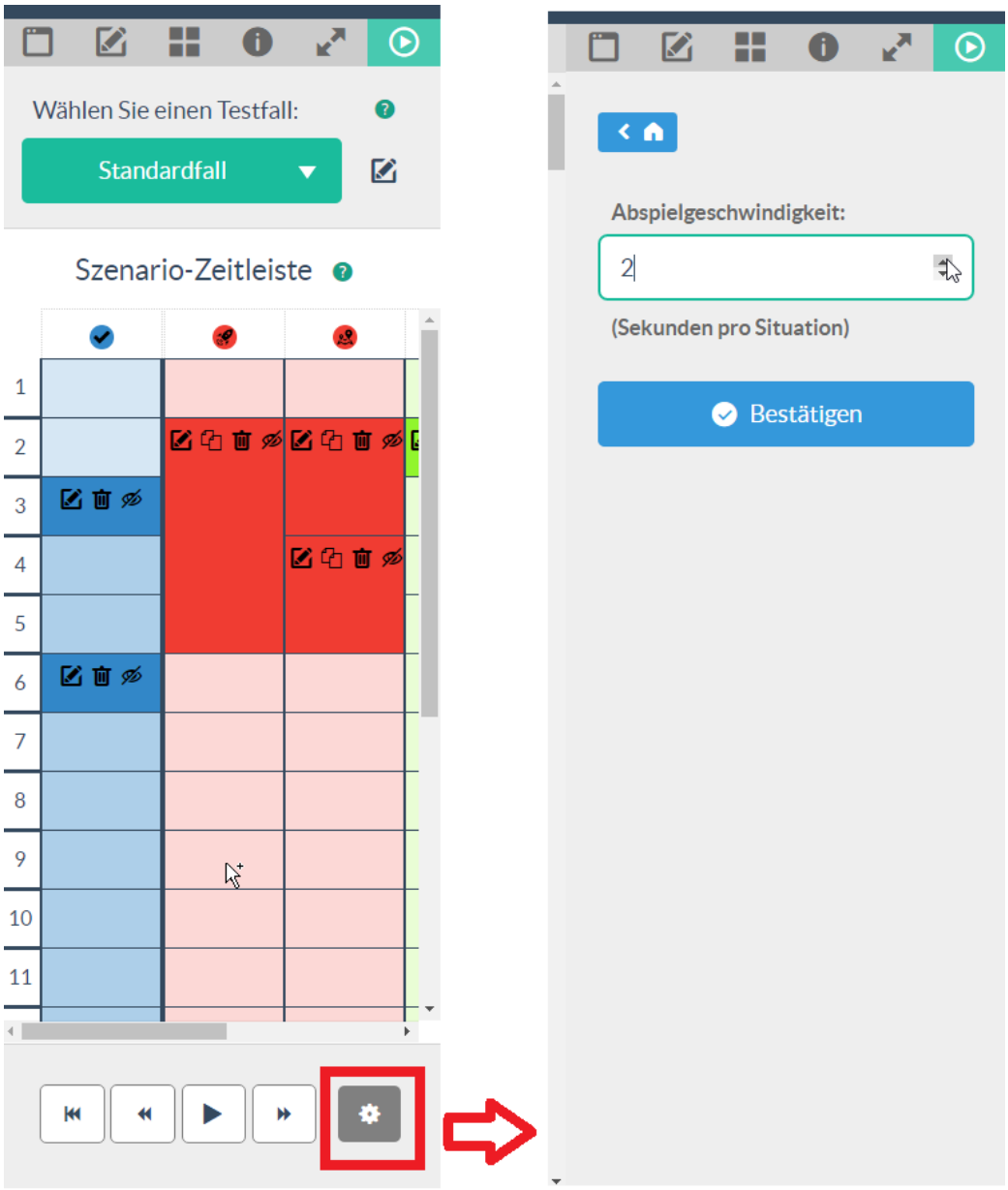


Abbildung 5.12: Navigation zur Ansicht der Optionen der Simulation (vorerst nur Abspielgeschwindigkeit einstellen).

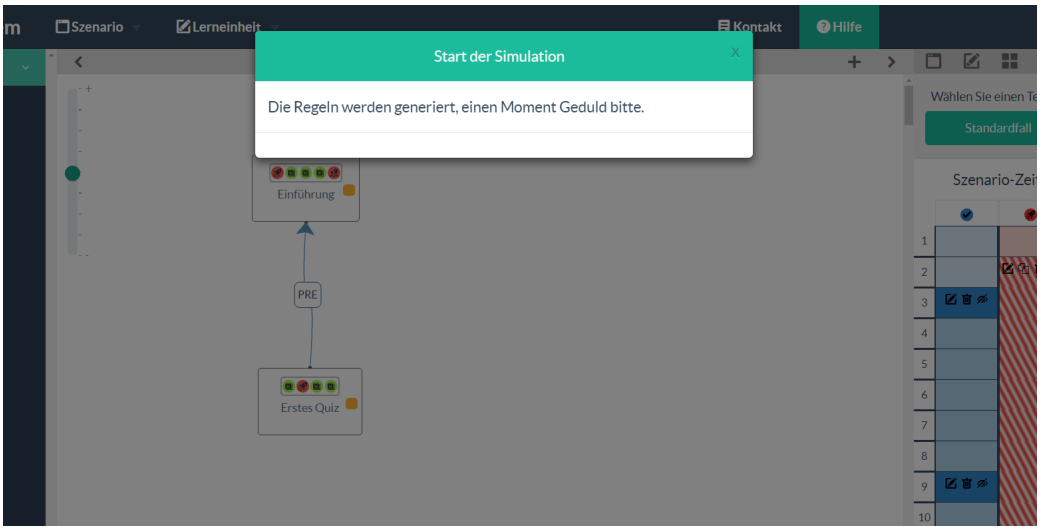


Abbildung 5.13: Benachrichtung zum Start der Simulation.

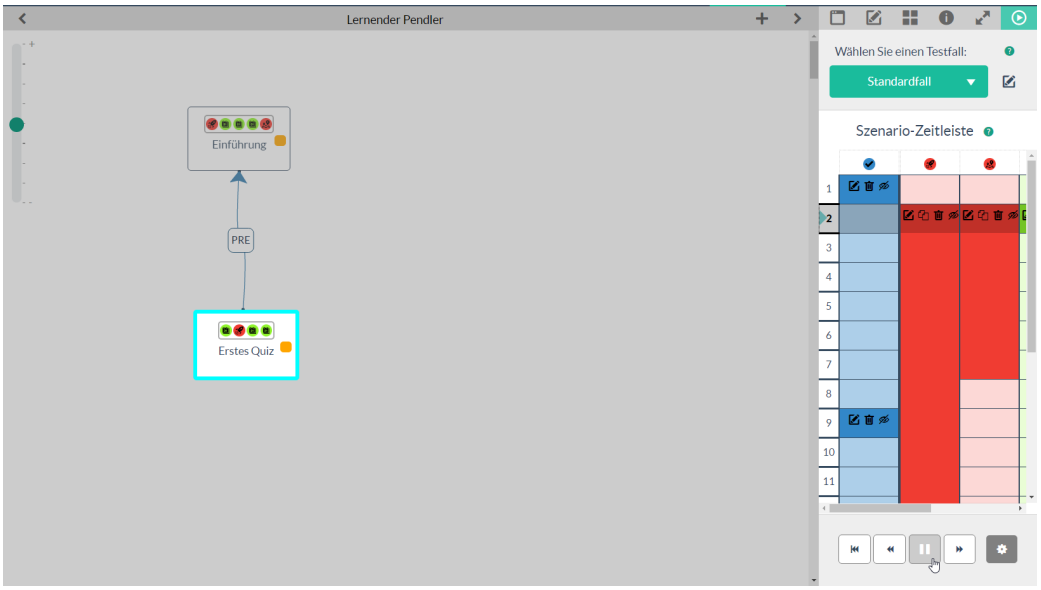


Abbildung 5.14: Lightbox zur Visualisierung der Adaptierung (Auswahl der Lerneinheit) und Highlighting der simulierten Situation (Zeile) in der Zeitleiste.

6 Evaluierung

Im Paper-Prototyping waren erste Designentwürfe für das User-Interface getestet, Anforderungen, Probleme und Lösungsideen ermittelt worden. Darauf basierend wurde ein grobes Konzept für den Simulator und sein User-Interface erstellt und dieses prototypisch implementiert.

In der letzten Phase dieser Arbeit wurde eine zweite Nutzerstudie durchgeführt, diesmal mit dem ersten digitalen, lauffähigen[1] Prototyp. Diesmal ging es um das Feintuning des ermittelten Grobkonzepts. Da es sich um die erste digitale Fassung einer völlig neuartigen Anwendung handelte, wurde auch hier wieder ein qualitativer, teils explorativer Ansatz verfolgt [EY10][Bar10] (siehe auch Abschnitt 4.2.2).

Es sollte die Usability der Funktionalität eingeschätzt werden, die im Paper-Prototyping exploriert und im Rahmen dieser Arbeit implementiert werden konnte. Aber auch die Visualisierung der Adaptierung, die im Paper-Prototyping nur ansatzweise umzusetzen war, sollte nun erstmals getestet werden.

Die anvisierte Einschätzung implizierte, vorhandene Schwachstellen zu identifizieren, Lösungen und auch Erweiterungen zu ermitteln. Probleme, die von mehreren Probanden bestätigt wurden oder offensichtlich waren, wurden falls möglich behoben. Das betraf in erster Linie Schwachstellen in der Implementierung, die nicht Teil des Konzepts waren. Darauffolgende Tests konnten in den meisten Fällen eine Verbesserung bestätigen. Eine Ausnahme wird in Abschnitt 6.2.2 diskutiert.

Die Gesamtheit der aus dieser Studie gewonnenen Ergebnisse markiert den Endstand dieser Arbeit, deren Fragen damit beantwortet werden können. Gleichzeitig dienen sie als Grundlage für künftige Verbesserungen und Erweiterungen des Simulators.

6.1 Vorgehen

Das Vorgehen war ähnlich wie beim Paper-Prototyping: Es wurden die Ziele, Fragen und Aufgaben für die Tests definiert, Probanden rekrutiert, die Tests

[1] Sowohl im Prototyp als auch im Framework selbst fanden sich im Laufe der Studie noch mehrere, teils gravierende Fehler.

durchgeführt, aufgezeichnet und anschließend ausgewertet. Nähere Details und Unterschiede zur ersten Studie werden in den folgenden Abschnitten erläutert.

6.1.1 Benutzerprofile

Für die Rekrutierung von Testnutzern wurden zunächst dieselben Benutzerprofile wie aus der ersten Studie genutzt. Allerdings konnten die gefundenen Testpersonen im Laufe der Studie in Nutzergruppen eingeordnet werden, die in Abschnitt 6.1.3 definiert werden.

6.1.2 Ziele der Studie

Die groben Ziele dieser Studie waren dieselben wie auch schon beim Paper-Prototyping: Verständnis, Funktionalität und Bedienbarkeit sollten evaluiert werden, wobei der Schwerpunkt wiederum auf der Erlernbarkeit des Simulators lag. Vor allem sollte überprüft werden, ob die im Paper-Prototyping identifizierte Unterstützung bei der Kontextmodellierung das Gesamtverständnis befördern würde. Das heißt: Würden die Funktionsweise des Simulators und der Zusammenhang von Szenario, Lerneinheiten und Kontext erkannt werden?

Außerdem konnten nun erstmals Details der Interaktion getestet werden, die sich mit dem Paper-Prototyp nur schwer umsetzen ließen. Dazu gehören Funktionen, die bei Mouseover sichtbar werden, Tooltips, Änderungen des Cursors, bestimmte Manipulationen der Zeitleiste und die Visualisierung der Adaptierung. Neben der Erlernbarkeit und Bedienbarkeit sollte in dieser umfangreicheren Studie auch ein genaueres Bild über die von Nutzern erwartete Funktionalität gewonnen werden. Diese Fragen sollten also durch die Beobachtung und Befragung der Benutzer beantwortet werden:

Verständnis: Inwieweit wird für die Benutzer verständlich, was sie sehen und was sie tun sollen?

- Wird die Zeitleiste als Abfolge von Situationen erkannt?
- Werden die Kontextinformationen aus dem Szenario wiedererkannt?
- Wird die Visualisierung richtig verstanden?
- Wird klar, was getan werden soll und wie?

- Ist das Verständnis abhängig von Erfahrungen mit anderen Anwendungen (IDEs, Kalender, Videoschnitt)?
- Welchen Effekt haben die Hilfstexte (Erklärungen hinter „?“) auf das Verständnis?
- Was ist noch unverständlich? Wo werden noch mehr Hilfen benötigt?

Funktionalität: Inwieweit werden die funktionalen Anforderungen erfüllt?

- Was sind typische Ziele und Abläufe? Was sind die wichtigsten bzw. am häufigsten benutzten Funktionen?
- Welche erwarteten Use-Cases werden noch nicht unterstützt?
- Welche (wichtigen) Funktionen fehlen?
  - Wie wichtig ist Kontext-Reasoning im Simulator?
  - Wie wichtig ist ein Matching zwischen Lerneinheiten und Situationen?
- Welche Funktionen werden nicht benötigt, das heißt, weder verwendet noch gesucht?

Bedienbarkeit: Inwieweit lässt sich der Simulator einfach und intuitiv bedienen?

- Welche Funktionen sind schwer zu entdecken bzw. wiederzufinden?
- Welche Funktionen werden nicht oder schwer verstanden?
- Sind die Funktionen leicht bedienbar? Was wirkt umständlich?
- Wird der Simulator als konsistent, auch mit dem Autorensystem, empfunden?
- Was wird als besonders positiv und „usable“ wahrgenommen?
- Wo sind Verbesserungen nötig?

6.1.3 Rekrutierung

Es wurden wiederum anhand der Benutzerprofile Probanden rekrutiert, darunter auch vier der fünf Paper-Prototyping-Teilnehmer. Insgesamt konnten 13 Probanden für die Studie gewonnen werden.

Den Großteil der Probanden (9 von 13) machten diesmal Teilnehmer des Seminars „Adaptives Lernen“ aus. Diese Probanden hatten inzwischen Erfahrungen mit dem Autorensystem gesammelt und konnten sich konkrete Lernszenarien vorstellen. Es kann angenommen werden, dass sie daher auch intrinsisch motiviert waren, ihre Anwendungen im Simulator zu testen. Insofern entsprachen sie noch mehr dem Benutzerprofil als die anderen Probanden oder auch als die des Paper-Prototyping.

Allerdings hatten 6 dieser Probanden gleichzeitig Programmiererfahrung im Umgang mit dem Context Toolkit (siehe Abschnitt 2.3.4) gesammelt. Somit konnten sie nicht mehr den in Anwendungsprogrammierung unerfahrenen Endnutzer ohne tieferes Verständnis des MOTIVATE-Frameworks repräsentieren.

Die meisten Probanden hatten zudem einen Hintergrund in Informatik, Wirtschaftsinformatik oder Computational Science. Gar keinen Informatik- oder Programmierhintergrund hatten nur zwei Hochschullehrkräfte aus den Bereichen Romanistik und Psychologie, von denen aber eine bereits am Paper-Prototyping teilgenommen hatte. Der Grund für die nicht optimale Nutzerzusammenstellung lag schlicht im Mangel an verfügbaren passenden Testpersonen und Ressourcen (Zeit, finanzielle Mittel).

6.1.4 Szenarien und Aufgaben

Es wurden wieder die Szenarien und die zu lösenden Aufgaben bestimmt. Anders als beim Paper-Prototyping konnte diesmal für die Testaufgabe bei den meisten Probanden auf Lernszenarien zurückgegriffen werden, die diese selbst während des Seminars Adaptives Lernen erstellt hatten. Das ermöglichte eine leichte Identifizierung mit den Aufgaben. Probanden, die nicht am Seminar teilgenommen hatten, wurden zu Beginn ggf. kurz in die Thematik und das Autorensystem eingeführt und legten dabei selbst ein exemplarisches Szenario für die Simulation an.[2]

Ein weiterer Unterschied zum Paper-Prototyping bestand in den Aufgaben. In dieser Studie sollte nur eine, dafür komplexe Aufgabe erfüllt werden: das Testen des (selbst erstellten) Szenarios. Dabei sollte vor allem untersucht

[2] In einem Fall wurde eine einfache Vorlage verwendet, da die Testperson das Autorensystem bereits kannte.

werden, wie die Probanden sich den Simulator erschließen und ihn selbst in der Realität verwenden würden. Die Aufgabenstellung lautete also lediglich: „Teste, wie sich die Anwendung in dem Szenario verhalten würde!“ Darin waren mehrere implizite Teilaufgaben bzw. Schritte enthalten, wie das Finden des Simulators, das Modellieren von Kontext, das Starten der Simulation, das Überprüfen der Adaptierung. Diese wurden aber bewusst nicht als eigenständige Aufgaben formuliert, um den Probanden keine Anleitung vorzugeben und nicht zu viel über die Benutzung zu verraten.

Vor dem eigentlichen Testen der Anwendung sollte als „erstes Szenario“ [Bar10] der erste Eindruck vom Simulator geschildert werden: Die Probanden sollten sagen, wie sie das verstehen, was sie sehen, was sie glaubten, dort tun zu können, und was ihnen unverständlich war.

Nach dem Testen der Anwendung wurden am Ende, je nach verbleibender Zeit, noch weitere Funktionen erkundet, für die zuvor keine Zeit oder kein Bedarf da gewesen war. Das ergab sich in den meisten Fällen auch ohne Nachfrage während des abschließenden Interviews (siehe Abschnitt 6.1.5: „Interview nach dem Test“).

6.1.5 Ablauf

Die Tests folgten alle dem Schema: kurze Einleitung, Fragen zum Hintergrund der Nutzer, Test der Anwendung und Interview mit Fragebogen. In der Länge variierten sie diesmal zwischen 45 Minuten und 2 Stunden, wenn es schwere Fehler im System gab und die Probanden in den Interviews viel Feedback gaben. Während des Testens und der Interviews wurden Ton und Bildschirm aufgezeichnet.[3] Alle Probanden hatten dafür zuvor ihr Einverständnis erklärt.

Interview vor dem Test

Vor Beginn der Tests wurden die Probanden, falls unbekannt, kurz zu ihren bisherigen Erfahrungen mit diesen Themen bzw. Anwendungen befragt:

- Hintergrund: Studienrichtung, berufliche Erfahrungen
- E-Learning: Erstellen von Lernanwendungen oder -inhalten
- Programmierung: Umfang und Verwendung von IDEs
- Regelmäßig genutzte Apps: Office-Anwendungen, Kalender

[3] Von drei Tests existieren wegen technischer Probleme keine bzw. nur ein kleiner Teil der Tonaufnahmen.

- Audio-/Videoschnittprogramme

Diese Informationen sollten dazu dienen, die Probanden mit dem Benutzerprofil zu vergleichen und in mögliche Nutzergruppen einzuordnen. Zudem sollte ein möglicher Effekt von Erfahrungen mit Audio-/Videoschnittprogrammen, Kalendern oder bestimmten IDEs auf das Erlernen des Simulators erkundet werden.

Wie oben bereits erwähnt, waren 9 von 13 Probanden reine Informatiker, sechs davon hatten bereits einschlägige Berufserfahrung. Die übrigen Probanden waren in den Bereichen Bioinformatik, Psychologie und Romanistik tätig. Sechs Probanden hatten berufliche Erfahrungen mit Lehre und E-Learning (außerhalb des oben genannten Seminars). 10 Probanden hatten regelmäßig mit Programmierung zu tun und verwendeten dazu IDEs wie z.B. Eclipse[4] oder WebStorm[5]. Alle Probanden hatten Erfahrungen mit den gängigen Office-Anwendungen. Nur ein Proband gab an, keine Kalender-App zu benutzen. XXX Probanden gaben an, weitreichende Erfahrungen mit Software für die Bearbeitung von Videos, Ton oder Animationen zu haben. Zeitleisten waren aber allen Probanden bekannt.

Lautes Denken

Während der Tests wurde wiederum mit „Lautem Denken“ gearbeitet, wobei die Probanden ihre Schritte kommentieren und Gedanken und Eindrücke laut äußern sollten. Dies hat teilweise schlechter funktioniert als beim Paper-Prototyping, was an der schnelleren Interaktion im digitalen Prototyp gelegen haben mag. Es wurde diesmal deutlich mehr mit dem System interagiert als mit der Testleiterin. Prinzipiell wäre das so auch richtig gewesen [Bar10][RC08], allerdings vergaßen die Probanden dabei oft, ihre Gedanken mitzuteilen. Es musste daher regelmäßig nachgefragt werden, was die Probanden gerade dachten, was sie dann möglicherweise aus dem Konzept brachte. Insgesamt war es schwer, die Gedanken und Handlungen der Probanden während der Tests nachzuvollziehen. Die Aufzeichnungen erwiesen sich als wertvolle Ressource.

[4] `https://eclipse.org/` [Abgerufen am 05.05.2016]
[5] `https://www.jetbrains.com/webstorm/` [Abgerufen am 05.05.2016]

Interview nach dem Test

Im Anschluss an jeden Test wurde anhand eines halb-strukturierten Interviews weiteres Feedback eingeholt.[6] Im Interview wurden die Probanden gebeten, Positives wie Negatives zusammenzufassen. Außerdem wurden ihnen Fragen zu diesen Aspekten gestellt:

- Inwieweit entsprach der Simulator ihren Erwartungen?
- Was hatten sie sich anderes erhofft?
- Wo wären noch mehr Hilfe und Unterstützung nötig?
- Was war am schwersten zu bewältigen?
- Hätten sie nach dem Testen der Lernanwendung mit dem Simulator ein gutes Gefühl, diese freizugeben?[7]

Diese Fragen wurden ergänzt um die System Usability Scale (SUS) [RC08] [Bar10]. Die darin enthaltenen Aussagen gelten trotz ihres kleinen Umfangs als geeignet, die Usability eines Systems einzuschätzen [BKM08]. Hierbei ging es darum, herauszufinden, wie sich jeder Proband zu den einzelnen Aussagen positionieren würde und vor allem, *warum*. Während sie den Fragebogen ausfüllten, kommentierten fast alle Probanden von sich aus einzelne Aussagen und stellten Fragen. So kam auch heraus, dass einige Aussagen sehr unterschiedlich verstanden wurden. [8] Die Ergebnisse der SUS sind daher *per se* nicht allzu aussagekräftig und sollten unter Berücksichtigung des jeweiligen Kontextes betrachtet werden.

Während der Interviews machten die Probanden mehrere Verbesserungsvorschläge, die zunächst einmal alle aufgenommen wurden [Bar10]. In vielen Fällen griffen die Interviewten noch einmal zur Maus, um die angesprochenen Punkte im Simulator zu demonstrieren. Dabei wurden oft noch Funktionen, aber auch Probleme, neu entdeckt.

[6] In zwei Fällen reichte die Zeit der Testpersonen nicht mehr für ein vollständiges Interview. In den ersten zwei Tests waren die Interviews noch weniger strukturiert. Aus organisatorischen Gründen gab es zwei Pilot-Tests, die unmittelbar nacheinander stattfanden, sodass dazwischen keine Änderungen am Testdesign mehr möglich waren.

[7] Mit dieser Frage sollte zusätzlich herausgefunden werden, ob und ggf. was dazu noch fehlte und ob die Simulation richtig verstanden wurde: Ein gutes Gefühl trotz Fehlverhalten hätte ein Indiz für ein falsches Verständnis sein können.

[8] Zum Beispiel wollten anfangs einige das gesamte Autorensystem bewerten. In diesen und den folgenden Interviews wurde daher der Hinweis gegeben, dass sich alle Fragen auf den Simulator bezogen. Manche konnten mit dem Begriff „Inkonsistenzen“ nichts anfangen. Einige versuchten sich die - teilweise schwerwiegenden - Bugs „wegzudenken“, andere nicht.

Nachbereitung

Nach den Tests wurden wiederum die gesammelten Daten - Notizen und Aufzeichnungen - ausgewertet: Die Aufzeichnungen wurden durchgesehen und -gehört und Handlungen, Kommentare und Stellungnahmen der Testteilnehmer sowie mögliche Implikationen protokolliert. Im nächsten Schritt wurden die Protokolle und Teile der Aufzeichnungen nach Auffälligkeiten, Problemen, Lösungsansätzen und weiterem Feedback durchsucht.[9] Pro Test wurden alle Funde den Testfragen (siehe Abschnitt 6.1.2) zugeordnet, die Ergebnisse anschließend gesammelt und dabei quantifiziert.

Die detaillierte Auswertung ermöglichte - wie schon beim Paper-Prototyping (siehe Abschnitt 4.5) - eine Analyse der gefundenen Probleme, ihrer Ursachen und somit möglicher Lösungen. Auch Muster in sowohl Problemen als auch Nutzertypen ließen sich dabei entdecken. Aber nicht nur Schwächen, sondern auch Stärken des evaluierten Konzepts konnten ausgemacht werden.

6.2 Ergebnisse

In den Nutzertests und den sich jeweils anschließenden Interviews konnte sehr viel Feedback generiert werden - sowohl explizit geäußertes als auch solches, das sich implizit aus der Beobachtung der Nutzer während der Tests ergeben hat.

Im Folgenden werden die wichtigsten aus dem Feedback gewonnenen Ergebnisse anhand der Leitfragen der Studie zusammengefasst und diskutiert. Auf eine ausführliche Quantifizierung wird dabei aus Platzgründen verzichtet. Eine vollständige Sammlung der einzelnen Testergebnisse findet sich dazu im Anhang 2.

6.2.1 Kritische Vorbetrachtungen

Wie beim Paper-Prototyping - und wie in jedem Usability-Test [Sny03] - ließ es sich oft nicht vermeiden, dass die Teilnehmer in ihrer Interaktion mit dem Simulator beeinflusst wurden. Aber auch andere Faktoren hatten einen Einfluss auf die Ergebnisse der Studie. Die folgenden Umstände galt es bei der Sammlung und Auswertung der Ergebnisse und der Beantwortung der Testfragen zu berücksichtigen:

[9] Besonders auffällige Probleme - hauptsächlich solche, die durch Fehler in der Implementierung verursacht worden waren - wurden, falls möglich, noch zwischen den Tests behoben (siehe Einleitung zu diesem Kapitel 6).

Charakteristika der Testsituation

Die Testsituation ist für die Probanden nicht natürlich, weil sie beobachtet und aufgezeichnet werden und dabei „laut denken“ sollen [Sny03][Bar10]. Hinzu kommt, dass ihnen eine Aufgabe gestellt wird, die sie versuchen sollen zu lösen. Diese Fokussierung auf die Aufgabenstellung - zusammen mit dem Bewusstsein, beobachtet zu werden - kann mitunter bei den Testpersonen zu einem Abweichen von ihrem natürlichen Verhalten geführt haben. So wurde gleich versucht, die Aufgabe zu lösen, anstatt zuerst die Anwendung in Ruhe zu erkunden und alle gegebenen Hilfen wahrzunehmen. Durch die Einleitung und Aufgabenstellung hatten die Probanden auch bereits eine - zumindest ungefähre - Vorstellung von dem, was sie erwarten würde. Es wäre interessant, herauszufinden, wie Probanden vorgehen würden, wenn die Aufgabenstellung wäre: „Erkunde das System.“

Da sowohl Simulator als auch Autorensystem frühe Prototypen sind, musste den Probanden an einigen Stellen über Hürden des Systems geholfen werden [RC08]. Diese Interaktion zwischen Testperson und Testleiterin veranlasste die Probanden häufig dazu, mehr Fragen zu stellen anstatt laut zu denken.

Dazu kam erschwerend, dass die Testleiterin gleichzeitig die Entwicklerin war und außerdem kaum Erfahrung mit Testmoderation hatte: Obwohl vermieden wurde, Fragen der Testpersonen zu beantworten, konnten nonverbale Signale nicht immer erfolgreich unterdrückt werden. Einige Probanden gaben auch zu, lieber Hilfe bei der Testleiterin gesucht zu haben als sich näher mit dem Simulator auseinanderzusetzen.

Es muss außerdem betont werden, dass die Testsituation einmalig ist und die reale Benutzung nur zu einem Teil widerspiegelt. In dem begrenzten Rahmen der Tests wurde selten die gesamte Funktionalität des Simulators in Anspruch genommen. Auch die Testdaten (Szenarien, Kontextinformationen) unterschieden sich. So kamen manche Probleme in manchen Tests gar nicht zum Vorschein. Die einzelnen Testergebnisse können daher weder für sich stehen noch untereinander ohne Weiteres verglichen werden und sollten prinzipiell im jeweiligen Kontext betrachtet werden.

Charakteristika der Testpersonen

Des Weiteren muss die Aufstellung der Testnutzergruppe(n) als Einflussfaktor berücksichtigt werden: Wie in Abschnitt 6.1.3 angemerkt, verfügte die Mehrheit der Probanden in dieser Nutzerstudie bereits über tiefgehende Kenntnisse von Framework und Context Toolkit.

Sechs der Teilnehmer hatten im Laufe des Seminars unter anderem eigene

Widgets programmiert und hatten somit einen anderen Hintergrund und Blick auf den Simulator. Damit einher gingen möglicherweise einerseits ein schnelleres Verständnis der Anwendung, andererseits auch andere daran gestellte Erwartungen. Ein ähnlicher Effekt muss auch für die vier Testpersonen angenommen werden, die bereits den Paper-Prototyp getestet hatten.

Programmfehler (Bugs)

Schließlich haben Bugs und andere technische Probleme[10] eine Beurteilung der Usability des Konzepts stark beeinträchtigt. Nicht nur die Bedienung, sondern vor allem auch das Verständnis der Anwendung wurden durch Bugs zum großen Teil massiv erschwert. Wenn die Simulation wegen Fehlern im System nicht die erwarteten Ergebnisse angezeigt hat, wurde das aufgebaute mentale Modell, also das Verständnis von der Anwendung, infrage gestellt.

Da es im Simulator selbst immer noch mehrere Bugs gab, vor allem bei der Verarbeitung der absolvierten Lerneinheiten, blieben zwei Bugs im darunterliegenden Framework bis nach dem 10. bzw. 13. Test unentdeckt:[11]

1. Beim Aggregieren der Kontextinformationen in ContactJS wurden diese nicht kumuliert, sondern aktualisiert. Dadurch wurde immer nur eine abgeschlossene Lerneinheit pro Situation erkannt und multiple Relationen nicht korrekt ausgewertet.

2. Bei der Erstellung der A-Box, aus der die Regeln für die Adaptierung generiert werden, wurden ebenfalls Relationen überschrieben.

Nachdem die Bugs im Simulator sukzessive behoben worden waren, konnte retrospektiv festgestellt werden, dass sich sein Fehlverhalten tatsächlich in den meisten Fällen auf diese zwei Bugs zurückführen ließ. Hier wurde die ursprüngliche Anforderung der Korrektheit im Sinne einer 1:1-Abbildung des Systemverhaltens zum Verhängnis für die Evaluation des Simulators. Andererseits konnten so gravierende Fehler im Framework erkannt und zum Teil bereits gebannt werden.

[10] In einem Fall konnte die Simulation mangels Internetzugang nicht gestartet werden, was bei der Testperson die Erwartungshaltung gegenüber dem Simulator veränderte: Sie nahm an, dass der Simulator die ContactJS-Widgets anfragen und Kontext-Reasoning durchführen würde.

[11] Entdeckte Fehler wurden zwischen den Tests behoben. Eine Fehlerbehebung innerhalb der Testsitzung war nicht möglich, da dazu intensives, langwieriges Debugging notwendig war.

6.2.2 Diskussion der Ergebnisse

Ein wichtiger Teil der Fragen zielte auf Verständnis und Erlernbarkeit des Simulators ab. In diesem Zusammenhang konnte die Feststellung gemacht werden, dass sich die Benutzer in ihren Lernstilen bezüglich des Erlernens einer neuen Anwendung unterschieden: Einige lasen gleich zu Beginn wenigstens eine der Hilfen.[12] Andere nahmen diese nicht oder erst bei späteren Problemen in Anspruch und erschlossen sich die Funktionsweise des Simulators stattdessen durch Ausprobieren. Fünf Testnutzer gaben an, dass dieses *Learning by Doing* ihrem Benutzertyp entspreche und sie grundsätzlich ungern Hilfen lesen.[13] Sechs Benutzer fanden allerdings eine einführende Anleitung nötig oder wünschenswert, etwa in Form eines Video-Tutorials oder Wizards anhand eines Beispiel-Szenarios. Es kann daraus geschlussfolgert werden, dass Selbstbeschreibungsfähigkeit und Lernförderlichkeit des Simulators nicht in ausreichendem Maß gegeben waren.

Verständnis

Im Folgenden werden die einzelnen Fragepunkte zum Verständnis des Simulators diskutiert und aufgetretene Probleme und mögliche Lösungen erörtert.

Werden die Kontextinformationen aus dem Szenario wiedererkannt? In der überwiegenden Mehrheit der Fälle traf dies zu. Es wurde auch kommentiert, dass der Zusammenhang von Testfall und Szenario durch die Wiederverwendung der Farben und Icons klar wurde. Auch das automatische Vorfiltern nach den relevanten Kontextinformationen wurde als sinnvoll empfunden. Drei Testnutzer hatten jedoch Probleme, die farbigen Spalten der Zeitleiste mental einzuordnen. In diesen Fällen hatten alle im Lernszenario verwendeten Kontextinformationen dieselbe Klasse, also auch Farbe, und die Icons fehlten. So sahen die Probanden nur eine blaue Spalte und daneben mehrere scheinbar identische Spalten einer anderen Farbe. Es wurde daher eine Beschriftung der (andersfarbigen) Spalten gewünscht, wobei „Kontextinformationen“ als hilfreiche Abgrenzung zu den abgeschlossenen Lerneinheiten vorgeschlagen wurde.

[12] Ein Proband gab im Interview an, dass er „sofort irgendwo einen Hilfe-Button gesucht“ hätte, die vorhandenen aber aufgrund seiner Kenntnis der Autorensystem-Hilfe („Hier finden Sie Hilfe!“) ignoriert habe.

[13] Hier sei angemerkt, dass neben den Bugs im System auch der Testaufbau den Lernprozess für diese Probanden erschwert haben könnte (siehe Abschnitt 6.2.1).

Wird die Zeitleiste als Abfolge von Situationen erkannt? Alle Probanden erkannten, dass die Zeitleiste einen Ablauf darstellte, wobei die Zeilen unterschiedlich benannt wurden: Zumeist wurde von „Schritten“ gesprochen, in einem Fall von „Szenarios“ (Test 9), in einem von „Kombinationen“ (Test 1). Die Bezeichnung „Situation“ schien hingegen nicht für jeden intuitiv. Mehrere Probanden kommentierten die einzelnen modellierten Schritte entsprechend dem vorgestellten Szenario („In Schritt X macht der Nutzer das, danach tut er das...“). In einem Fall wurden die Zeilen anfangs als reale Zeiteinheiten interpretiert. Dabei spielten mehrere Faktoren eine Rolle: Im Szenario wurden die Kontextinformationen „Aktueller Termin“ und „Nächster Termin“ verwendet und die Testperson, die keine der Hilfen gelesen hatte, erwartete zunächst eine Emulation mit „live“ stattfindender Kontexterfassung.

Wird die Visualisierung richtig verstanden? Die Visualisierung der Adaptierung konnte in dieser Studie erstmals getestet werden. Hier offenbarten sich die größten Verständnisprobleme, was nicht zuletzt an den Fehlern im System und dem daraus folgenden unerwarteten Verhalten der Simulation lag. Aber auch unabhängig davon gab es Unklarheiten und Missverständnisse. Mehreren Probanden war nicht klar, was das Aufleuchten der Lerneinheiten bedeutete: ob sie als vorgeschlagen, in Bearbeitung oder abgeschlossen simuliert wurden. Auch die Benachrichtigung am Ende der Simulation konnte diese Frage nicht zufriedenstellend beantworten.

Besonders verwirrend war dabei die Verwendung von *zwei* Farben für das Highlighting. In den ersten zwei Tests war nur eine Farbe verwendet worden, um die jeweils ausgewählten Lerneinheiten hervorzuheben. Das wurde danach aufgrund des Feedbacks von den ersten Probanden geändert[14], was sich in den folgenden Tests jedoch nicht bewährte und nach dem 12. Test wieder rückgängig gemacht wurde. Positiv aufgenommen wurde die Visualisierung des Fortschritts in der Zeitleiste und - in den Fällen, in denen die Adaptierung größtenteils wie erwartet funktionierte - das Zusammenspiel vom Highlighting links (Lerneinheiten) und rechts (Zeitleiste).

Wird klar, was getan werden soll und wie? Die meisten Testnutzer konnten selbstständig herausfinden, wie sie mit dem Simulator arbeiten konnten, auch ohne die Hilfen zu lesen. Diese Nutzer erschlossen sich die Funktionsweise

[14] Die Probanden hatten im Kurs „Adaptives Lernen“ die Lernanwendung bereits auf dem Handy getestet, wo einmal ausgewählte Lerneinheiten fortwährend angezeigt wurden. Beide fanden es irritierend, dass die Simulation sich anders verhielt als die Anwendung und wünschten sich eine Visualisierung der Auswahlhistorie.

vor allem durch Ausprobieren, Mouseover und Klicken. In drei Fällen hatten die Testpersonen anfangs einige Probleme zu erkennen, was sie überhaupt tun konnten und sollten und benötigten einen Hinweis. Der gegebene Hinweis hätte sich in der Hilfe zum Simulator gefunden, wo die Probanden jedoch nicht nachgesehen hatten. Daher wurde am Ende der Studie der Prototyp noch einmal so modifiziert, dass beim ersten Start die Erklärung zum Simulator für wenige Sekunden automatisch eingeblendet wird. Die größte Verunsicherung beim Begreifen der Funktionsweise kam durch das Fehlverhalten der Simulation durch Bugs im Simulator und im Framework zustande (siehe Abschnitt 6.2.1).

Ein weiterer wichtiger Punkt betrifft das Einstellen der abgeschlossenen Lerneinheiten: Für die Anwendung sind diese Informationen nichts anderes als Kontextinformationen. Für Benutzer des Simulators stellte die Interaktion mit der Lernanwendung allerdings eine eigene „Ebene“ oder „Dimension“ (siehe Tests 3, 12 und 13) dar, die näher an den im Autorensystem angelegten und während der Simulation gehighlighteten Lerneinheiten verortet wurde. So wurde zum Beispiel versucht, die Lerneinheiten in die Zeitleiste zu ziehen oder während der Simulation durch Anklicken auszuwählen. Insofern wurde das mentale Modell der Nutzer noch nicht gut genug unterstützt.

Zudem fanden einige Probanden es irritierend, Lerneinheiten als abgeschlossen angeben zu können, wenn diese von der Anwendung noch gar nicht freigeschaltet worden waren. Hier gab es Unterschiede im Verständnis von der Funktionsweise der adaptiven Lernanwendung.

Ist das Verständnis abhängig von Erfahrungen mit anderen Anwendungen (IDEs, Kalender, Videoschnitt? Eine Testperson, die allerdings auch am Paper-Prototyping teilgenommen hatte, bezeichnete die Kontextmodellierung in der Zeitleiste als „Standardproblem“, das ihr von Animationsprogrammen bekannt war. Einige Testnutzer stellten aufgrund der tabellarischen Darstellung der Zeitleiste eine Verbindung zu Tabellenkalkulationsprogrammen wie Excel her. Die Bedienelemente der Kontrollleiste zum Abspielen der Simulation waren jedem vertraut. Davon abgesehen konnte allerdings kein Zusammenhang mit Sicherheit nachgewiesen werden.

Welchen Effekt haben die Hilfstexte (Erklärungen hinter „?“) auf das Verständnis? Wie zu Beginn dieses Abschnitts beschrieben, wurden die Hilfen nicht von jedem in Anspruch genommen. Einige Probanden sagten im Nachhinein, dass die Hilfen teilweise weitergeholfen hätten. Dazu gehörte der Hinweis auf das Festlegen der abgeschlossenen Lerneinheiten.

In der Erklärung zur Zeitleiste wurde vor allem der „Tipp“ als hilfreich empfunden. Mehrere Probanden versuchten nach dem Lesen, eine Kontextinformation „langzuziehen“. Eine Testperson merkte an, dass zwar die Zeilen, aber nicht die Spalten der Zeitleiste erklärt seien und wünschte sich einen entsprechenden Hinweis.

Was ist noch unverständlich? Wo werden noch mehr Hilfen benötigt? Wie in den vorhergegangenen Absätzen erörtert, gab es die größten Verständnisprobleme bei der Visualisierung der Adaptierung. Es mangelte hier an Feedback und Anleitungen im System, da sich die vorhandenen Hilfen bzw. Erklärungen in erster Linie auf das Modellieren der Testfälle bezogen. In einem einführenden Tutorial, beispielsweise auf der begleitenden Projekt-Website, sollte noch ausführlicher erklärt werden, wozu genau die Simulation dient und was visualisiert wird. Zusätzlich könnten im Simulator eine weitere Schnellhilfe oder ein Tooltip (z.B. „Diese Lerneinheit passt zur Situation.“) direkt an der gehighlighteten Lerneinheit unmittelbares Feedback geben.

Sehr oft fiel es den Probanden schwer, das Adaptierungsverhalten nachzuvollziehen, was in den allermeisten Fällen an den Fehlern im System lag (siehe Abschnitt 6.2.1). In einigen, wenigen Fällen verhielt sich die Simulation jedoch korrekt und es fehlten Informationen in der Modellierung, was nicht leicht ersichtlich war. Hier könnte das System noch mehr Unterstützung in Form des bereits im Paper-Prototyping gewünschten „Matchings“ anbieten (siehe Kapitel 4.5).

Auch schwer verständlich war die Bedeutung der abgeschlossenen Lerneinheiten (siehe oben, Absatz „Wird klar, was getan werden soll und wie?“). Ein Hinweis, dass Lerneinheiten auch vom Nutzer konsumiert werden können, selbst wenn sie nie von der Adaptation Engine ausgewählt wurden, wäre hilfreich gewesen. Es war auch trotz verschiedener visueller Hinweise (Liste im Tooltip, Einfärben der Spalte in mittlerer Intensität, Verlängern nicht möglich) nicht leicht ersichtlich, dass eine einmal abgeschlossene Lerneinheit für immer abgeschlossen bleibt.[15] Das Fortbestehen der Gültigkeit dieser einmal angelegten Informationen kann möglicherweise auch dazu beigetragen haben, dass manche Testnutzer dies auch für alle anderen Kontextinformationen annahmen. Sie erwarteten, dass ein eingetragener Wert so lange gilt, bis er durch einen neuen überschrieben wird.

Insgesamt sollte dem Benutzer noch mehr klargemacht werden, was der

[15] Dem könnte eventuell schon begegnet werden, wenn die zu einem Zeitpunkt bereits absolvierten Lerneinheiten aus der Auswahlliste im Popover herausgefiltert würden, was noch nicht implementiert wurde.

Simulator kann und was nicht, zumindest solange seine Erwartungen noch nicht durch mehr Unterstützung des Systems erfüllt werden. So sollte es etwa einen Hinweis geben, dass nur die Auswahl der Lerneinheiten anhand der angegebenen Informationen simuliert wird und keine spezifische App.

Funktionalität

An verschiedenen Stellen offenbarte sich Bedarf an weiteren Funktionen, die den Nutzern die Arbeit erleichtern und ihr Verständnis fördern würden. Drei Testnutzer zeigten sich mit dem gebotenen Funktionsumfang zufrieden und wünschten sich keine weitere Funktionalität. Andere Benutzer hatten mehr Erwartungen an das System. Zum Großteil wurden Bedürfnisse und Wünsche explizit geäußert und konkrete Vorschläge für Erweiterungen und Optimierungen gemacht. Manches ergab sich aber auch implizit aus der Beobachtung der Probanden und den Schwierigkeiten, die sie hatten.

Was sind typische Ziele und Abläufe? Was sind die wichtigsten bzw. am häufigsten benutzten Funktionen? Die grobe Aufgabenstellung, die Anwendung in dem Szenario zu testen, war im Test vorgegeben. Daher wurde in der Mehrheit der Fälle der „Standardablauf“ des imaginierten Szenarios getestet. Einige Testnutzer probierten aber auch das Testen von Fehlerfällen aus.

Ein typisches Ziel war das Überprüfen der Auswahl einer bestimmten Lerneinheit in einer bestimmten Situation, was für mehrere Lerneinheiten wiederholt wurde. Um das Szenario im Simulator abzubilden, wurde häufig zwischen dem Kontext-Tab einer Lerneinheit und dem Simulator-Tab gewechselt. Im Kontext-Tab wurden die in den Regeln definierten Kontextwerte überprüft und zum Teil kopiert, um sie im Simulator einzufügen.

Am wichtigsten bzw. meistbenutzten waren die Vorschau der Kontextwerte im Super-Tooltip, das Hinzufügen, Bearbeiten, Kopieren, Verschieben und Löschen von Werten und das Steuern der Simulation über Play, Pause und wahlweise Vor/Zurück oder Anklicken einer Zeilennummer.

Welche (erwarteten) Use-Cases werden noch nicht unterstützt? Welche (wichtigen) Funktionen fehlen? Wie schon im Paper-Prototyping waren die Testnutzer interessiert an einer Vorschau der erstellten Lerninhalte, um deren Darstellung in der App zu kontrollieren.

Auch das Matching, also die Begründung für die (Nicht-)Auswahl von Lerneinheiten durch einen Abgleich der gegebenen Informationen, sollte durch das System unterstützt werden. Ohne diese Unterstützung mussten

die Nutzer beide Kontextmodelle - in der Anwendungslogik und im Testfall - im Kopf abspeichern und vergleichen, was eine starke kognitive Belastung darstellte. Zumal sich in den Tests schnell herausstellte, dass die Tooltips zur Anzeige der Kontextwerte nicht ausreichten und ein Überblick fehlte. Daher wünschten sich auch mehrere Testnutzer eine Vorschau der Werte in den Zeitfenstern, anstelle der Quick-Access-Icons.[16]

Drei Testnutzer äußerten den Wunsch nach einer Möglichkeit, mehrere Testfälle nacheinander durchlaufen zu lassen und die Ergebnisse zu vergleichen. Zudem wurde erwartet, auch die Kontexterfassung (Interpretation) im Simulator testen zu können und die Logik der eingegebenen Werte durch den Simulator prüfen zu lassen. Der Bedarf nach dieser Funktionalität fiel allerdings geringer aus als nach dem Paper-Prototyping erwartet. Eine mögliche Erklärung dafür wäre, dass in den verwendeten Szenarien selten komplexer Kontext zum Einsatz kam.

Als ein weiterer möglicher Use-Case kam die Idee auf, einen Testfall als Vorlage zur Erstellung eines neuen Szenarios zu verwenden. Um außerdem das mentale Modell der Benutzer besser zu unterstützen (siehe oben, Absatz „Wird klar, was getan werden soll und wie?“), wäre mehr Interaktion mit den Lerneinheiten im Canvas sinnvoll.

In allen Tests wurde sowohl explizit als auch implizit deutlich, dass es im Simulator noch an Unterstützung beim Erstellen der Testfälle mangelt. Einer der häufigsten Arbeitsschritte war das Wechseln zwischen Simulator-Tab und Kontext-Tab zum Abgleich und Kopieren der Kontextinformationen. Eine Testperson bezeichnete das redundante Anlegen der Kontextinformationen treffend als „doppelt gemoppelt“. Verbesserungsvorschläge beinhalteten eine Auswahl der im Szenario angelegten Kontextwerte im Popover und das automatische Generieren des „Standardablaufs“ aus dem Szenario.

Welche Funktionen werden nicht benötigt? Hierzu kann zum jetzigen Zeitpunkt noch keine klare Aussage getroffen werden. Zwar wurden einige Funktionen wie das Einstellen der Abspielgeschwindigkeit oder das Ausblenden von Kontextwerten seltener genutzt als andere. Ob sie aber in Zukunft nicht benötigt werden, wenn das System häufiger benutzt wird, kann nicht beurteilt werden. Es kann nur festgehalten werden, dass - abgesehen von einem Popup-Fenster, das im letzten Test beim Pausieren der Simulation anzeigte, wenn keine Lerneinheiten zur Situation passten - keine Funktion als

[16] Eine weitere Möglichkeit wäre, den Benutzer selbst wählen zu lassen zwischen der Anzeige der Icons und der der Werte. Außerdem könnte eine Übersicht aller Werte einer Situation gegeben werden, etwa bei Mouseover über dem Zeilenkopf.

überflüssig und störend wahrgenommen wurde.

Bedienbarkeit

Insgesamt wurde die Bedienung des Simulators von der Mehrheit der Probanden als einfach und intuitiv empfunden. Die Einstellungen in der Zeitleiste konnten von allen Probanden leicht manipuliert werden: Wert hinzufügen, bearbeiten und löschen war jedem ersichtlich. Die Kombination von Icons und Tooltips war den meisten verständlich. Auswahl und Bearbeiten der Testfälle konnte von fast allen Probanden wie vorgesehen bedient werden. Die Buttons der Abspiel-Kontrollleiste - Play, Pause, Vor, Zurück und Zurück zum Anfang - waren den Probanden bekannt und trafen in der Regel die Erwartungen. Auffällig war, dass der Cursor kaum wahrgenommen wurde, der je Bedienelement die damit verknüpfte Aktionsmöglichkeit (Klick, Verschieben, Ziehen) anzeigte.

Welche Funktionen sind schwer zu entdecken bzw. wiederzufinden? Von den implementierten Funktionen wurden fast alle auf Anhieb gefunden. Eine Ausnahme bilden die Optionen für Zeilen und Spalten, die für die meisten Probanden, die z.B. nach einer Möglichkeit suchten, Zeilen zu löschen oder hinzuzufügen, zu versteckt waren. Probanden, die die Hilfe zur Zeitleiste nicht gelesen hatten, hatten außerdem Probleme, das Verlängern bzw. der Zeitfenster zu entdecken. Dadurch wurden Werte oft umständlich mehrfach kopiert.

Welche Funktionen werden nicht oder schwer verstanden? Die meisten der wahrgenommenen Funktionen wurden so verstanden wie gedacht. Der Button „Zurück zum Start“ wurde allerdings anscheinend leicht missverstanden, da mehrere Probanden sich einen „Stopp“-Knopf wünschten, der dieselbe Funktion hätte. Das im Paper-Prototyping noch sehr unverständliche Ausblenden der Kontextwerte wurde dank des Tooltips „Wert nicht erfassen“ überwiegend richtig verstanden, das Auge-Symbol aber nicht als passend empfunden.

Sind die Funktionen leicht bedienbar? Was wirkt umständlich? Es gab mehrere leichte Fehler in der Darstellung und Interaktion, die im Laufe der Studie sukzessive behoben wurden. Abgesehen davon waren die meisten Funktionen des Simulators für die Mehrheit der Benutzer leicht zu bedienen.

Umständlich war vor allem, dass die Werte der Kontextinformationen nur einzeln bei Mouseover angezeigt wurden und der Super-Tooltip oft nicht

„stabil“ angezeigt wurde. Insgesamt fehlte es an Übersichtlichkeit beim Bearbeiten der Testfälle. In fast jedem Test wurde das starke Bedürfnis geäußert, das Panel, in dem der Simulator-Tab sich befindet, zu vergrößern. Aufgrund der schmalen Breite konnten oft nicht alle Spalten gleichzeitig eingesehen werden, was die Arbeit, das Überprüfen der Modellierung und somit auch das Nachvollziehen der Simulation erschwerte.

Wird der Simulator als konsistent, auch mit dem Autorensystem, empfunden? Die Aussagen 5 und 6 der System Usability Scale (SUS) sowie die Kommentare der Testpersonen geben dazu Aufschluss (siehe Anhang 2.1 und 2.2). Insgesamt wurde der Simulator als konsistent - in sich und mit dem Autorensystem - empfunden. Es wurde von mehreren Probanden bemerkt, dass Elemente wie Kontexticons und -farben wiederverwendet worden seien.

Was wird als besonders positiv und „usable“ wahrgenommen? Die Verwendung der bekannten Farben und Icons und die Visualisierung (wenn sie funktionierte) wurden überwiegend positiv aufgenommen. Vor allem die Manipulation der Zeitleiste, die verschiedenen verfügbaren Optionen und die Abspielkontrollleiste wurden von den meisten Testnutzern als einfach und intuitiv bedienbar empfunden. Auch das Filtern nach den im Szenario relevanten Kontextinformationen wurde für sinnvoll und die Hilfen - falls wahrgenommen - für hilfreich befunden.

Wo sind Verbesserungen nötig? Die wichtigsten Verbesserungsvorschläge wurden in der vorhergegangen Absätzen besprochen. Eine ausführliche Sammlung kann dem Anhang entnommen werden.

7 Fazit

Abschließend sollen die Ergebnisse dieser Arbeit resümiert und ein Fazit gezogen werden. Dazu wird zu den eingangs gestellten Forschungsfragen Bezug genommen und ein Ausblick auf mögliche künftige Arbeiten gegeben.

7.1 Zusammenfassung

In dieser Arbeit wurde ein Konzept für einen Simulator entwickelt, mit dem Benutzer des MOTIVATE-Autorensystems ihre damit erstellten adaptiven Lernanwendungen testen können, um die Adaptivität besser nachvollziehen zu können. Die größte Herausforderung bestand im innovativen Anwendungsbereich, weshalb weder gleichartige Anwendungen noch erfahrene Anwender zurate gezogen werden konnten.

Es sollten Antworten auf diese Forschungsfragen gefunden werden (siehe Kapitel 1):

1. Welche funktionalen und nicht-funktionalen Anforderungen stellen sich aus Nutzerperspektive an einen Simulator für kontextsensitive, adaptive Lernanwendungen, der ein Testen und Verstehen der Anwendungslogik ermöglichen soll?

2. Wie muss das User-Interface (die Benutzungsschnittstelle) gestaltet sein, um eine hohe Usability bei der Arbeit mit dem Simulator zu gewährleisten?

Wobei der Schwerpunkt der zweiten Forschungsfrage auf einer leichten Erlernbarkeit und intuitiven Bedienbarkeit des User-Interface lag. Um dieses Ziel zu erreichen, wurde das erste grobe Konzept in einer Paper-Prototyping-Studie exploriert, die wichtige Erkenntnisse und Einsichten geliefert hat.

Zum einen konnten die anfangs angenommenen funktionalen Anforderungen validiert und durch das Feedback der Teilnehmer ergänzt werden: Zusätzlich zum Modellieren, Simulieren und Verwalten von Testfällen wurde ein Bedarf nach einer Überprüfung der Lerninhaltsanzeige, der Kontextinterpretation, nach automatisiertem Testen und nach mehr Unterstützung beim Nachvollziehen des adaptiven Verhaltens erfasst. Es wurde erkennbar, dass im mentalen Modell der Testnutzer ein enger Zusammenhang zwischen den

Kontextmodellen von Lernszenario und Simulation bestand, der im Simulator abgebildet werden sollte. Diese Erkenntnis hat die Entwicklung des Konzepts entscheidend beeinflusst, welches nach der Studie entsprechend überarbeitet, verfeinert, prototypisch implementiert und erneut mit Benutzern getestet wurde.

In der zweiten, umfassenderen Studie mit dem ersten digitalen Prototyp wurden - neben jeder Menge Programmfehlern - neue Schwierigkeiten beim Verständnis aufgedeckt, aber auch Verbesserungen gegenüber der ersten Studie festgestellt. Vor allem die gewählten Bezeichnungen waren deutlich besser verständlich, es wurden mehr Hilfen durch das System gegeben und die verwendeten Elemente (Farben, Icons, Buttons, usw.) wurden leicht wiedererkannt. Durch die Vorauswahl der in der Simulation benötigten Kontextinformationen wurde die Verbindung zum Szenario schnell hergestellt und die Funktionsweise meist schnell begriffen. Inwiefern die angestrebte Anlehnung an bekannte Anwendungen wie für Tabellenkalkulation, Videobearbeitung oder Kalender die Bedienung für die Nutzer leichter erlernbar machte, ließ sich allerdings schwer einschätzen.

Da sich die Mehrheit der Tests als Suche nach Fehlern im Simulator bzw. MOTIVATE-Framework gestaltete, konnte der Simulator seiner eigentlichen Bestimmung (noch) nicht gerecht werden. Er war konzipiert worden, damit Autoren adaptiver Lernanwendungen die Adaptivität begreifen und nachvollziehen können. Sie sollten die Adaptierungslogik überprüfen und dadurch auch mehr Vertrauen in ihre Lernanwendung gewinnen können. Die meisten Testpersonen hatten jedoch schon ein tieferes Verständnis von der Adaptierungslogik. Für sie bot der Simulator keinen erkennbaren Mehrwert (abgesehen vom Stillen der anfänglichen Neugier). Im Gegenteil: Die Fehler im System führten den Anspruch, das Verständnis der Anwendung zu verbessern, *ad absurdum*. Trotz dieser Einschränkung konnten einige Usability-Probleme aufgedeckt werden, die nicht vom Funktionieren des Frameworks abhingen und Verbesserungen bedürfen.

Die größten Probleme beim Verständnis der Modellierung zeigten sich im Bestimmen der abgeschlossenen Lerneinheiten, die die Benutzeraktionen des Lernenden repräsentierten. Diese Informationen wurden eher an den Lerneinheiten selbst vermutet, mit denen in der Simulation mehr Interaktion erwartet wurde. Die Visualisierung der Auswahl war vielen Nutzern unverständlich und sollte durch mehr unmittelbares Feedback besser erklärt werden. Insgesamt wurde ein Bedarf nach mehr Feedback und Anleitungen deutlich.

Bei der Bedienbarkeit stellte sich der fehlende Überblick über die modellierten Kontextinformationen als größeres Problem heraus. Hier kam oft der Wunsch nach mehr Platz für die Bearbeitungsfläche und einer Vorschau aller

eingetragenen Werte auf, da es schwer fiel, sich alle Werte zu merken.

Neben mehr Unterstützung beim Verständnis, vor allem der Visualisierung, und den im Paper-Prototyping ermittelten Use-Cases, konnten weitere (funktionale) Anforderungen identifiziert werden, die die Nutzer an den Simulator stellten: Das im Autorensystem bereits aufwendig manuell definierte Lernszenario sollte zum Testen im Simulator nicht noch einmal modelliert werden müssen. Ein anderer Vorschlag ging in eine ähnliche Richtung: Demnach sollte ein Testfall als Vorlage für ein neu anzulegendes Lernszenario verwendet werden können. Es kann als Fazit daraus festgehalten werden, dass Autorensystem und Simulator noch besser integriert werden sollten, um dem Nutzer ein einfacheres Testen zu ermöglichen. Der Nutzer sollte sich schließlich mehr auf das Ergebnis als auf die Modellierung der Simulation konzentrieren.

Die in dieser Arbeit entwickelte Lösung stellt einen Vorschlag dar, der die zweite Forschungsfrage beantworten soll und dessen Vor- und Nachteile künftigen Arbeiten als Entscheidungsgrundlage dienen können.

7.2 Ausblick

Eine Vorschau der Inhalte - in den Lerneinheiten-Containern, einem eigenen Fenster oder in einem Emulator - wäre eine mögliche, sinnvolle Erweiterung des Prototyps. Ebenso wie das „Debugging“ der Simulation, also eine Begründung für die Adaptierung durch Anzeige übereinstimmender und fehlender Kontextinformationen. Insgesamt sollte das Überprüfen und Bearbeiten der Kontextmodelle vereinfacht werden. So könnten etwa die an den Lerneinheiten eingestellten Kontextinformationen direkt im Canvas angezeigt und in die Simulation übertragen werden. Das Modellieren der abgeschlossenen Lerneinheiten sollte zudem intuitiver werden.

Mögliche Weiterentwicklungen und Verbesserungen des Simulators sollten durch weitere Nutzerforschung begleitet werden, um herauszufinden, ob die Verbesserungen einen positiven Effekt haben. Besonders interessant und aufschlussreich wären Contextual Inquiry und Usability-Studien mit tatsächlichen Endnutzern des Autorensystems. Solche standen für diese Arbeit noch nicht zur Verfügung. Die bislang einzigen Nutzer des Autorensystems entsprachen nur teilweise der Zielgruppe. Diese „echten“ Nutzer in ihrem realen Nutzungskontext zu beobachten und befragen, wäre jedoch unerlässlich, um tatsächlich valide Forschungsergebnisse zu erhalten.

Literaturverzeichnis

[AFT+15] Amalfitano, Domenico ; Fasolino, Anna R. ; Tramontana, Porfirio ; Ta, Bryan D. ; Memon, Atif M.: MobiGUITAR: Automated Model-Based Testing of Mobile Apps. In: *Software, IEEE* 32 (2015), Nr. 5, S. 53–59

[AFTA13] Amalfitano, Domenico ; Fasolino, Anna R. ; Tramontana, Porfirio ; Amatucci, Nicola: Considering context events in event-based testing of mobile applications. In: *Software Testing, Verification and Validation Workshops (ICSTW), 2013 IEEE Sixth International Conference on* IEEE, 2013, S. 126–133

[Bar10] Barnum, Carol M.: *Usability testing essentials: ready, set... test!* Elsevier, 2010

[BC13] Bruneau, Julien ; Consel, Charles: DiaSim: a simulator for pervasive computing applications. In: *Software: Practice and Experience* 43 (2013), Nr. 8, S. 885–909

[BKM08] Bangor, Aaron ; Kortum, Philip T. ; Miller, James T.: An empirical evaluation of the system usability scale. In: *International Journal of Human–Computer Interaction* 24 (2008), Nr. 6, S. 574–594

[BMF12] Bruck, Peter A. ; Motiwalla, Luvai ; Foerster, Florian: Mobile learning with micro-content: a framework and evaluation. In: *Bled eConference, 25th Bled eConference eDependability: Reliable and Trustworthy eStructures, eProcesses, eOperations and eServices for the Future*, 2012, S. 17–20

[BV03] Barton, John J. ; Vijayaraghavan, Vikram: UBIWISE, a simulator for ubiquitous computing systems design. In: *Hewlett-Packard Laboratories Palo Alto, â AI HPL-2003-93* (2003)

[BVH06] Broens, Tom ; Van Halteren, Aart: Simucontext: Simply simulate context. In: *Autonomic and Autonomous Systems, 2006. ICAS'06. 2006 International Conference on* IEEE, 2006, S. 45–45

[CGO15] Choudhary, Shauvik R. ; Gorla, Alessandra ; Orso, Alessandro: Automated Test Input Generation for Android: Are We There Yet?(E). In: *Automated Software Engineering (ASE), 2015 30th IEEE/ACM International Conference on* IEEE, 2015, S. 429–440

[CSMH12] Cassens, Jörg ; Schmitt, Felix ; Mende, Tobias ; Herczeg, Michael: CASi–A Generic Context Awareness Simulator for Ambient Systems. In: *Ambient Intelligence*. Springer, 2012, S. 421–426

[DAS01] Dey, Anind K. ; Abowd, Gregory D. ; Salber, Daniel: A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-aware Applications. In: *Human-Computer Interaction* 16 (2001), Dezember, Nr. 2, S. 97–166

[Dey01] Dey, Anind K.: Understanding and Using Context. In: *Personal Ubiquitous Computing* 5 (2001), Januar, Nr. 1, S. 4–7

[DSSK06] Dey, Anind K. ; Sohn, Timothy ; Streng, Sara ; Kodama, Justin: iCAP: Interactive prototyping of context-aware applications. In: *Pervasive Computing*. Springer, 2006, S. 254–271

[EY10] Eberhard-Yom, Miriam: *Usability als Erfolgsfaktor: Grundregeln + User Centered Design + Umsetzung*. Cornelsen-Verlag, 2010

[FS88] Felder, Richard M. ; Silverman, Linda K.: Learning and teaching styles in engineering education. In: *Engineering education* 78 (1988), Nr. 7, S. 674–681

[GG14] Griebe, Tobias ; Gruhn, Volker: A model-based approach to test automation for context-aware mobile applications. In: *Proceedings of the 29th Annual ACM Symposium on Applied Computing* ACM, 2014, S. 420–427

[Hei07] Heim, Steven: *The Resonant Interface: HCI Foundations for Interaction Design*. Boston, MA, USA : Addison-Wesley Longman Publishing Co., Inc., 2007

[HM04] Harrer, Andreas ; Martens, Alke: Adaptivität in eLearning-Standards-ein vernachlässigtes Thema? In: *DeLFI*, 2004, S. 163–174

[ISO98] Norm DIN EN ISO 9241-11:1998 1998. *Ergonomische Anforderungen für Bürotätigkeiten mit Bildschirmgeräten - Teil 11: Anforderungen an die Gebrauchstauglichkeit*

[ISO06] Norm DIN EN ISO 9241-110:2006 2006. *Ergonomische Anforderungen der Mensch-System-Interaktion Teil 110: Grundsätze der Dialoggestaltung*

[KH05] Kukulska-Hulme, Agnes: *Mobile learning: A handbook for educators and trainers*. Psychology Press, 2005

[KPPR10] Kakousis, Konstantinos ; Paspallis, Nearchos ; Papadopoulos, George A. ; Ruiz, Pedro A.: Testing self-adaptive applications with simulation of context events. In: *Electronic Communications of the EASST* 28 (2010)

[Kru13] Krug, S.: *Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability*. Pearson Education, 2013

[Lem14] Lemcke, Stefanie: *Evaluierung und Implementierung einer Portierung des Context Toolkit für mobile Web-Technologien*. Potsdam, Deutschland, Universität Potsdam, Masterarbeit, Dezember 2014

[LFH10] Lazar, J. ; Feng, J.H. ; Hochheiser, H.: *Research Methods in Human-Computer Interaction*. John Wiley & Sons, 2010

[LLB+14] Liang, Chieh-Jan M. ; Lane, Nicholas D. ; Brouwers, Niels ; Zhang, Li ; Karlsson, Börje F ; Liu, Hao ; Liu, Yan ; Tang, Jun ; Shan, Xiang ; Chandra, Ranveer u. a.: Caiipa: automated large-scale mobile app testing through contextual fuzzing. In: *Proceedings of the 20th annual international conference on Mobile computing and networking* ACM, 2014, S. 519–530

[MBOL11] McGlinn, Kris ; Brennan, Rob ; O'Sullivan, Declan ; Lewis, Dave: The simcon generator: An interactive context simulator for rapid evaluation of smart building applications using virtual reality. In: *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011 IEEE International Conference on* IEEE, 2011, S. 50–55

[MJZL14] Moebert, T. ; Jank, H. ; Zender, R. ; Lucke, U.: A Generalized Approach for Context-Aware Adaption in Mobile

E-learning Settings. In: *IEEE 14th International Conference on Advanced Learning Technologies, ICALT 2014, Athens, Greece, July 7-10, 2014*, 2014, S. 143–145

[MOG+10] McGlinn, Kris ; O'Neill, Eleanor ; Gibney, Alan ; O'Sullivan, Declan ; Lewis, David: SimCon: A Tool to Support Rapid Evaluation of Smart Building Application Design using Context Simulation and Virtual Reality. In: *J. UCS* 16 (2010), Nr. 15, S. 1992–2018

[Mos12] Moser, Christian: *User Experience Design: mit erlebniszentrierter Softwareentwicklung zu Produkten, die begeistern.* Springer-Verlag, 2012

[MTN13] Machiry, Aravind ; Tahiliani, Rohan ; Naik, Mayur: Dynodroid: An input generation system for Android apps. In: *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering* ACM, 2013, S. 224–234

[NP10] Nielsen, Jakob ; Pernice, Kara: *Eyetracking web usability.* New Riders, 2010

[OCL10] O'Neill, Eleanor ; Conlan, Owen ; Lewis, David: Modeling and simulation to assist context aware system design. In: *Simulation* (2010)

[OKL+05] O'Neil, Eleanor ; Klepal, Martin ; Lewis, David ; O'Donnell, Tony ; O'Sullivan, Declan ; Pesch, Dirk: A testbed for evaluating human interaction with ubiquitous computing environments. In: *Testbeds and Research Infrastructures for the Development of Networks and Communities, 2005. Tridentcom 2005. First International Conference on* IEEE, 2005, S. 60–69

[PSS12] Püschel, Georg ; Seiger, Ronny ; Schlegel, Thomas: Test Modeling for Context-aware Ubiquitous Applications with Feature Petri Nets. In: *Proceedings of the 2nd International Workshop on Model-based Interactive Ubiquitous Systems (MODIQUITOUS)*, 2012, S. 37–40

[RC08] Rubin, Jeffrey ; Chisnell, Dana: *Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests.* 2. Wiley, 2008

[Rey09] Rey, Günter D.: *E-Learning: Theorien, Gestaltungsempfehlungen und Forschung.* Huber Bern, 2009

[RF13] Richter, Michael ; Flückiger, Markus D.: *Usability Engineering kompakt: benutzbare Produkte gezielt entwickeln.* Springer-Verlag, 2013

[RWJM03] Roenneberg, Till ; Wirz-Justice, Anna ; Merrow, Martha: Life between clocks: daily temporal patterns of human chronotypes. In: *Journal of biological rhythms* 18 (2003), Nr. 1, S. 80–90

[SAW94] Schilit, Bill N. ; Adams, Norman ; Want, Roy: Context-Aware Computing Applications. In: *In Proceedings of the Workshop on Mobile Computing Systems and Applications*, IEEE Computer Society, 1994, S. 85–90

[Sny03] Snyder, Carolyn: *Paper prototyping: The fast and easy way to design and refine user interfaces.* Morgan Kaufmann, 2003

[Sta10] Stapelkamp, Torsten: *Interaction-und Interfacedesign: Web-, Game-, Produkt-und Servicedesign Usability und Interface als Corporate Identity.* Springer-Verlag, 2010

[VHH15] Vieira, Vaninha ; Holl, Konstantin ; Hassel, Michael: A context simulator as testing support for mobile apps. In: *Proceedings of the 30th Annual ACM Symposium on Applied Computing* ACM, 2015, S. 535–541

Web-Quellenverzeichnis

[Nie06] Nielsen, Jakob: *Progressive Disclosure*. Online verfügbar: https://www.nngroup.com/articles/progressive-disclosure. [Abgerufen am 29.03.2016], 2006

[Red16a] Redaktion des Dudenverlags: *Duden Online*. Mannheim, Germany : Online verfügbar: http://www.duden.de/node/727052/revisions/1341033/view. [Abgerufen am 25.03.2016], 2016

[Red16b] Redaktion des Dudenverlags: *Duden Online*. Mannheim, Germany : Online verfügbar: http://www.duden.de/node/707326/revisions/1388556/view. [Abgerufen am 25.03.2016], 2016

Appendices

1 Anhang: Ergebnisse der 1. Studie

Hier werden die Ergebnisse aller Tests anhand der Testfragen zusammengefasst. Der Zusammenfassung folgen die vollständigen Ergebnisse für jeden einzelnen Test.

1.1 Paper-Prototyping: Zusammenfassung der Ergebnisse

Verständnis

Kann der Simulator („die Testmöglichkeit“) leicht gefunden und seine Funktionsweise erkannt werden?

- Über die Haupt-Navigationsleiste wurde der Simulator schnell gefunden, der Tab allein aber nicht.
- Allen Testpersonen (TP) war offensichtlich, dass Play die Simulation starten und etwas visualisiert werden würde.
- Dass zuerst Kontext modelliert werden muss, ging möglicherweise schon aus der Aufgabenstellung hervor. Dennoch war den meisten TP zu Beginn nicht klar, was sie tun sollten.
- Nach dem Starten der Simulation und der Erkenntnis, dass und wo Kontextinformationen hinzugefügt werden können, wurde die Funktionsweise gut verstanden.

Lassen sich Situationen wie von den Nutzern gedacht abbilden?

- Kontextinformationen, die eine Situation beschreiben (wie in den Aufgabenstellungen angegeben), wurden wiedererkannt.
- Eine (zeitliche) Abfolge wurde erst durch die Fortschrittsanzeige während der Simulation erkannt. Möglicherweise war nicht erwartet worden, dass eine Abfolge von Situationen modelliert werden kann.
- Es wurden nie gleich *alle* geltenden (in der Aufgabe aufgelisteten) Kontextinformationen angegeben, sondern jeweils eine hinzugefügt und getestet. Das deutet darauf hin, dass nicht klar war, dass Kontext vollständig manuell modelliert werden muss.
- 3 TP versuchten, Lerneinheiten direkt im Canvas auszuwählen, anstatt sie als Kontextinformationen hinzuzufügen.

Wo gibt es Verständnisprobleme? An welchen Stellen sind Hilfen (Anleitungen, Erklärungen) nötig?

- Beim Einstieg: Es war nicht klar, was wo wie getan werden kann. Erwartet wurde eine Simulation bzw. Emulation. Ein Hinweis auf die Möglichkeit, Kontext in der Zeitleiste hinzuzufügen, hätte den Einstieg erleichtert.
- Beim Modellieren: Es war nicht klar, dass alle im Szenario angegebenen Kontextinformationen hinzugefügt werden müssen. Der Zusammenhang von modelliertem Kontext und Kontext an Lerneinheiten sollte klarer werden. Auch wurden bearbeitete Lerneinheiten nicht als Kontextinformationen verstanden.
- In der Simulation: Es war schwer nachvollziehbar, warum - wider Erwarten - keine Lerneinheiten ausgewählt wurden. Die ersten 2 TP fanden, dass die Nachricht, dass keine Lerneinheit ausgewählt wurde, hilfreich gewesen wäre. In den folgenden Tests zeigte sich, dass die Nachricht allein nicht ausreichte, sondern auch eine Begründung benötigt wurde.
- Begriffen, Bezeichner und Feedback waren unverständlich oder missverständlich formuliert.

Funktionalität

Sind die Aufgaben bzw. Anwendungsfälle realistisch und nachvollziehbar?

- Überwiegend ja, die meisten TP konnten sich gut in die Szenarien hineinversetzen.
- 1 TP gab an, „niemals“ für jedes Gerät und jede Information einzelne Simulationen anzulegen.

  1 TP „graust[e] es vor“ dem Import von (im Sekundentakt) gemessenen Kontextinformationen.

Welche Use-Cases (Anwendungsfälle) wurden ggf. noch nicht bedacht? Welche Funktionalität wird dazu benötigt oder gewünscht?

- 1 TP: Reasoning (Interpretation und Zerlegung von High-Level-Kontext) im Simulator zu testen.

- 2 TP: „Debugging“ der Simulation (Vergleich von Lerneinheiten und Situationen).
- 1 TP: Automatisches Testen für längerfristiges Arbeiten mit dem Autorensystem.
- 1 TP: Definieren von Subsets von vorhandenen und fehlenden Sensoren (Kontextinformationen) und dazu vordefinierte Testcases.
- 2 TP: Anzeige (Vorschau) der eigentlichen Inhalte.

Welche Funktionalität ist am wichtigsten bzw. wird am häufigsten verwendet?

- Kontext (schnell) modellieren: Kontextinformationen hinzufügen, ansehen, bearbeiten, löschen, kopieren, verschieben, verlängern/verkürzen, ausblenden. Vorlagen anlegen, öffnen, bearbeiten.
- Wechsel zwischen Lerneinheiten und Simulator zum Überprüfen der Modellierung(en).
- Testen: Play/Replay, Pause und Überprüfen der Auswahl.

Welche Funktionen wurden nicht gebraucht?

- Einige Optionen wurden nicht in Anspruch genommen: Kontext aus einer anderen Vorlage in die geöffnete importieren (möglicherweise missverstanden), Vorlagen löschen (in den Aufgaben nicht nötig), „Debug“-Modus (Tests 4 und 5), „Speichern & Neu“ (Tests 2 und 3).
- Der Filter wurde zwar ausprobiert, aber seine Funktion war eine andere als von den TP erhofft: Gesucht wurde nach einer Unterstützung beim Modellieren des Szenarios. Das Filtern nach Kontextklassen erschien dagegen wenig sinnvoll.

Werden Vorlagen entdeckt, gebraucht und als sinnvoll erachtet? Werden eigene Vorlagen zur Wiederverwendung erstellt?

- Im 1. Test wurde keine Vorlage verwendet (evtl. nicht entdeckt). Nach Änderungen wurde jedes Mal eine Vorlage verwendet (siehe dazu Abschnitt „Kritische Vorbetrachtungen“).

- Vorlagen wurden zwar wiederverwendet, aber nur weil die Navigation zurück zum Simulator unklar war. Dass keine Vorlagen für eine spätere Wiederverwendung erstellt wurden, liegt aber vermutlich auch an der einmaligen Testsituation.

Bedienbarkeit

Sind die Funktionen intuitiv bedienbar?

1. Kontext modellieren:

- Die Interaktion mit den Kontext-„Kästchen“ und Popover-Menüs war, nachdem sie einmal entdeckt war, intuitiv.
- Die Ausblenden-Funktion wurde nicht verstanden.

2. Kontextmodelle verwalten:

- Im 1. Test war „Neue Simulation“ missverständlich und Speichern nicht ersichtlich. Vorlagen wurden nicht angesehen.
- In den folgenden Tests waren nach Änderungen die Optionen und Popup-Dialoge bis auf Kleinigkeiten intuitiv verständlich.

3. Simulation steuern:

- Play, Reload, Pause und die Anzeige des Ablaufs waren jedem sofort verständlich.
- Die Steuerung der Geschwindigkeit war in Test 1 und 2 weder klar noch benötigt und wurde danach entfernt, in Test 3 aber wieder erwartet.

Insgesamt:

- In 1. Test gab es zu wenig Kontrolle über Dialogsteuerung (Bestätigen/Schließen). Nach entsprechenden Änderungen wurden Zurück- und Bestätigen-Buttons intuitiv verwendet. Speichern per Default war nicht ersichtlich.
- Die Navigation (Öffnen und Schließen des Simulator-Tabs) wirkte unintuitiv und dadurch umständlich.

- In 3 Fällen war das Scrollen problematisch.
- Icons waren größtenteils verständlich, aber Beschriftungen nicht ausreichend und oft unverständlich oder missverständlich.

Notwendige Änderungen:

- Die Ablaufrichtung sollte deutlich werden: durch Spalten für Kontextinformationen und wenigstens Zeilenbeschriftung. Der Ablauf sollte nicht fließend, sondern springend visualisiert werden, um keinen falschen Ablauf innerhalb einer Zeile zu suggerieren.
- Bezeichnungen und Begrifflichkeiten sollten verständlicher werden für: simulierten Kontext („Szenario“ erwartet statt „Simulation“), Kontextinformationen ausblenden, Geschwindigkeit regeln.

  Besonders Feedback-Nachrichten müssen sorgfältig und eindeutig formuliert, Erklärungen und Anleitungen als solche erkennbar werden.
- Mehr Unterstützung beim Modellieren von Kontext:

  Ein Quick-Access für die häufigsten Aktionen und eine Schnellansicht der Kontextwerte wurden erwartet.

  Erwartete und fehlende Kontextinformationen sollten deutlich gemacht werden. Es sollte nach Kontextinformationen gefiltert werden können, die im Szenario vorkommen.

  1 TP wünschte sich, Kontext aus Lerneinheiten direkt in Simulationsschritte übertragen zu können.
- Die Navigation zwischen Tabs sollte intuitiver und einfacher sein.

Test 1 (01.12.15)	
Frage	Antwort
Durchgeführte Änderungen	Keine
Kann der Simulator (die „Testmöglichkeit") leicht gefunden werden?	Nein. Gesucht wurde zuerst an den Lerneinheiten, dann im Szenario-Menü.
Kann seine Funktionsweise erkannt werden?	Play war aber offensichtlich, aber nicht Modellieren von Kontext → erst nach Erkenntnis, dass Kontext hinzugefügt werden kann (Drücken auf „+"-Button → Popover).
Lassen sich Situationen wie von den Nutzern gedacht abbilden?	• Erst nachdem Ablaufrichtung und Hinzufügen von Kontextinformationen entdeckt wurden. • Zeitinformation gesucht, aber nicht gefunden/erkannt, Verlauf oben→unten undeutlich. • Es wurde immer nur eine Zeile gefüllt und eine neue Simulation angelegt. • Bearbeitete Lerneinheiten wurden eher an den Lerneinheiten als im Simulator-Tab erwartet. • Anhaltendes Highlighting (pro Zeile) der Lerneinheiten „gefällt mir".
Wo gibt es Verständnisprobleme / Überraschungen?	• Einstieg: unklar, dass und wie zuerst Kontext modelliert wird. • Modellierung: unklar, dass Lerneinheiten als Kontextinfos (rechts) ausgewählt werden können. • Simulation: wenn Lerneinheiten wider Erwarten nicht ausgewählt wurden und nichts passierte, außer dass der Balken durchlief.
An welchen Stellen sind Hilfen (Anleitungen, Erklärungen) nötig?	• Hinweis, dass und wie mit Modellieren begonnen werden kann: „Durch Klicken in [Bereich] neue Informationen hinzufügen." + eindeutige Bezeichnung/Beschriftung für Bereich. • Hinweis, dass abgeschlossene Lerneinheiten als Kontextinfos angegeben werden können. • Feedback über Adaption: „Ich fänd's intuitiv, wenn eine Meldung käme, dass keine Lerneinheit angezeigt würde." • Matching zwischen simuliertem Kontext und Lerneinheit(en) unterstützen.
Sind die Aufgaben bzw. Anwendungsfälle realistisch und nachvollziehbar?	Ja, gut vorstellbar.
Welche Use-Cases (Anwendungsfälle) wurden ggf. noch nicht bedacht? Welche Funktionalität wird dazu benötigt oder gewünscht?	Keine.
Welche Funktionalität ist am wichtigsten bzw. wird am häufigsten verwendet?	• Kontext hinzufügen und bearbeiten. • Play/Reload/Pause. • Kontext in Lerneinheiten überprüfen. • Neue Simulation anlegen.
Welche Funktionen werden nicht gebraucht?	• Simulation zum Bearbeiten öffnen, Löschen. • Beschreibung hinzufügen. • Geschwindigkeit einstellen.
Werden Vorlagen entdeckt, gebraucht und als sinnvoll erachtet?	Nicht verwendet (evtl. nicht entdeckt).
Werden eigene Vorlagen zur Wiederverwendung erstellt?	Nein.
Sind die Funktionen intuitiv bedienbar? Kontext modellieren	• Ja: Klicken auf „+" zum Hinzufügen, anklicken der Kästchen zum Öffnen & Bearbeiten. • Nein: Bestätigen/Schließen fehlt. Filter und Deaktivieren nicht verständlich. Bearbeitete Lerneinheiten direkt an

	Lerneinheiten angeben (statt/zusätzlich zu Modellierung im Tab).
Kontextmodelle verwalten	• Ja: Name und Beschreibung ändern. • Nein: „Neue Simulation" nicht eindeutig. Speichern-Buttons / Feedback zu Speichern fehlen. Bearbeiten-Button evtl. nicht eindeutig: „Was passiert hier?" • „Zum Bearbeiten öffnen" nicht benutzt. • Bedeutungen unklar: „Neue Simulation", „Hochladen".
Simulation steuern	• Ja: Play, Reload, Pause und Ablauf durch Indikator klar. • Nein: „er könnte stoppen", wenn über alle Informationen drüber. „+/-" nicht gleich klar.
Insgesamt	Erwartet Bestätigen- / Schließen-Buttons. Kontrolle über Dialogsteuerung und Speichern. (Default: Nicht speichern.)
Welche Änderungen sind nötig oder wünschenswert?	• Spalten („Raster") für Kontextinformationen → Kontext ordnen und Ablaufrichtung verdeutlichen. • Simulator „ist ein Tool und gehört in eine Toolbar" (in globale Navigationsleiste). • Speichern und Schließen steuern. • „Neue Simulation", „Zeige" (Filter), „Deaktivieren", „Hochladen", "+/-" umbenennen. • Lerneinheiten pro Situation durch Klicken auf Unit im Canvas auswählen. • Bei Deaktivieren: Option „Für alle". • Scrollen „etwas nervig" → Panel bei Bedarf vergrößern (nach links/rechts ziehen, nicht nur ein-/ausblenden)

Test 2 (02.12.15)	
Frage	Antwort
Durchgeführte Änderungen	• Menüpunkt „Simulation” in Navigationsleiste. • Optionen unter „Simulation”: „Erstellen, Öffnen, Löschen, Laden”. („Laden” ersetzt „Hochladen”.) • Pro Option öffnet sich Popup-Dialog (analog zu anderen Menüpunkten: „Szenario”, „Lerneinheit”). • Statt “+Optionen”-Button in Simulator-Tab „Speichern & Neu”. • In Kontext-Popover: „Bestätigen”-Button und „X” zum Schließen. „Deaktivieren” umbenannt in „Nicht verfügbar”. • Zusätzliche Spalten. • In Simulation: Balken stoppt vor leeren Zeilen und geht zurück zum Anfang.
Kann der Simulator (die „Testmöglichkeit”) leicht gefunden werden?	Ja.
Kann seine Funktionsweise erkannt werden?	Ja. (Anm.: Wählt zuerst „Öffnen”.)
Lassen sich Situationen wie von den Nutzern gedacht abbilden?	• Ja. (Anm.: Evtl. zu gut passende Vorlage gegeben.) ○ Aber: Chronologische Abfolge „wird nicht suggeriert” → zeitliche Abfolge in Vorlage „Glücksfall”! • Nicht offensichtlich, dass Informationen fehlen → andere Annahme über Framework: nicht erfasste Informationen auf Standardwerte gesetzt.
Wo gibt es Verständnisprobleme / Überraschungen?	• Menüpunkt + Option: „kein Szenario, sondern Simulation auswählen, sehr verwirrend”. → Begrifflichkeit: „Simulation” als Menge von Kontextinformationen. • Zusammenhang von Kontextinformationen in Lerneinheiten und in Simulation: „braucht man ein bisschen Zeit für”. • Bei nicht erfolgter Auswahl ohne Feedback („verwirrend”): Nutzer muss abwägen, ob Fehler in Simulation oder Lerneinheiten. • „Nicht verfügbar” und „Zeige” (Filter) unverständlich.
An welchen Stellen sind Hilfen (Anleitungen, Erklärungen) nötig?	• Feedback in Simulation: „Nachricht, dass nichts passiert, hätte mir schon deutlich geholfen.” • Zusammenhang von Lerneinheiten-Kontext und simuliertem Kontext verdeutlichen.
Sind die Aufgaben bzw. Anwendungsfälle realistisch und nachvollziehbar?	• Größtenteils ja. • Zum Simulieren der App auf verschiedenen Geräten mit verschiedenen Sensoren/Berechtigungen: zu aufwendig, für jedes Gerät und jede Information einzelne Simulationen anzulegen („würde ich niemals machen”).

Welche Use-Cases (Anwendungsfälle) wurden ggf. noch nicht bedacht? Welche Funktionalität wird dazu benötigt oder gewünscht?	• Subsets an verfügbaren und berechtigten Sensoren erfragen und Verhalten testen (vordefinierte Testcases).
	• Übertragen von Kontext zwischen Lerneinheiten und Simulation (von Lerneinheit in neue Zeile).
	• Debugging der Simulation:
	  1. Tooltips an Lerneinheiten: welche Informationen fehlen ggü. Zeile? (Matching)
	  2. Matching der Kontextinformationen („Kästchen und Knöppel”) visualisieren.
	• TP würde („wenn ich faul bin”) <u>eine</u> Simulation mit allen Informationen und Zeiten anlegen und nach Bedarf ein-/ausblenden (aber nur einzelne Info-Kästchen).

Welche Funktionalität ist am wichtigsten bzw. wird am häufigsten verwendet?	• Modellierung schnell erstellen können: Hinzufügen, Vorschau, Bearbeiten, Löschen, Ausblenden, Verschieben, Verlängern/-kürzen, Kopieren. • Wechsel zwischen Vorlagen. • Simulation ausgewählter Situationen (Zeilen).

	• Überprüfen der Lerneinheiten (Kontextregeln). • Bei mehr Informationen: Filter.
Welche Funktionen werden nicht gebraucht?	• „Importieren" von Vorlagen in Vorlagen. (Anm.: Funktionalität an falscher Stelle untergebracht, daher möglicherweise missverstanden.) • Geschwindigkeit der Simulation steuern (da kein Ablauf erwartet, sondern Auswahl einzelner Zeilen). • „Speichern & Neu" (wollte an der Stelle „Speichern & Schließen").
Werden Vorlagen entdeckt, gebraucht und als sinnvoll erachtet?	Ja. Importieren von Kontextlog als hilfreich/nützlich empfunden.
Werden eigene Vorlagen zur Wiederverwendung erstellt?	Nein. (Aber würde laut eigener Aussage so vorgehen.)
Sind die Funktionen intuitiv bedienbar? Kontext modellieren	• „Nicht verfügbar" und „Importieren" unverständlich. • Erwartet Mouseover, verschieben, verlängern, kopieren. • Hinzufügen zu umständlich: erwartet Default-Werte und Quick-Access.
Kontextmodelle verwalten	• Optionen und Popup-Dialoge verständlich. • „Laden" schlecht gewählt, da synonym zu „Öffnen".
Simulation steuern	• Nein: erwartet keinen Ablauf, sondern: Zeile anklicken + Play. • Reload sollte in Play integriert sein statt separat: automatisch vor jedem Play.
Insgesamt	• Erwartet Rechtsklick (an Kontext-Kästchen und Lerneinheiten). • Icons gut, Beschriftungen weniger.
Welche Änderungen sind nötig oder wünschenswert?	• Begrifflichkeiten überarbeiten: „Simulation" → „Kontext/Szenario/...", „Nicht verfügbar" → „Ausblenden/Ignorieren/…", „Zeige" (Filter), „Laden" → „Importieren/Generieren" . • „Schließen" statt „Speichern und Neu". • Reload in Play integrieren. Vor-/Zurückspulen statt „+/-". • Ganze Zeilen in Simulation auswählen statt Fortschrittsbalken („steppen" statt „fließen"). • Importieren-Funktion aus Kontextinformation-Popover rausnehmen. • Kontextinformationen aus Lerneinheiten in Simulation übertragen. • Default-Werte, Mouseover, Verschieben, Quick-Access: Duplizieren, Löschen, Ausblenden (Auge). • Dropdown-Menü aller Simulationen (statt nur Namensanzeige). • Möglichkeit, nach relevanten Kontextinformationen (nur Szenario) zu filtern. • Unterstützung bei Editieren der Lerneineiten bei entdecktem Fehlverhalten.

Test 3 (04.12.15)	
Frage	Antwort
Durchgeführte Änderungen	• „Simulation" → „Kontextsimulation", „Zeige" → „Filtern nach", „Laden" → „Importieren", „Nicht verfügbar" → „Ausblenden". • „Importieren" raus aus Kontext-Popover. • Bei Nicht-Auswahl in Simulation: Nachricht, dass keine Lerneinheit ausgewählt wurde. • Reload-Button entfernt (Reload vor jedem Play). • Geschwindigkeitskontrolle („+/-") entfernt (da oft missverstanden und nie benutzt). • Fortschrittsanzeige springend statt fließend (auf Zeilen zeigen statt Balken verschieben). Anm.: *Besonderheit des Szenarios: Schlecht modellierter Kontext „Termin findet statt" (high-level) + Parameter als separate Informationen (low-level) sorgte für Irritation.*
Kann der Simulator (die „Testmöglichkeit") leicht gefunden werden?	Ja. („Klingt für mich jetzt sehr nach Emulation" → „würde ich sofort auf '[Kontext-]Simulation' gehen".)
Kann seine Funktionsweise erkannt werden?	• Ja, aber erst nach Hinweis: „Zum Hinzufügen einer Kontextinformation hier klicken oder klicken und ziehen." • Hinterher: „Es kam auch das raus, was ich erwartet habe"
Lassen sich Situationen wie von den Nutzern gedacht abbilden?	• Prinzipiell ja. (Anm.: Evtl. zu gut passende Vorlage gegeben.) • Aber: Zeitliche Abfolge nicht ersichtlich, Indikatoren gesucht, aber nicht gefunden. • High-Level-Kontext allein sorgt für „schlechtes Gefühl", würde „weiter unten bei den rohen Daten [anfassen]" und Interpretation durch System (Reasoning) erwarten („beruhigender").
Wo gibt es Verständnisprobleme / Überraschungen?	• Zu Beginn unklar, was zu tun ist. Emulator erwartet. • Timeline nicht als solche erkannt, erst durch Fortschrittsanzeige nach Play. • Bei Nicht-Auswahl nicht offensichtlich, dass Informationen fehlen. • Bei Zustandekommen von High-Level-Kontext („Termin findet statt") ohne Low-Level-Kontext (Datum, Zeit, Ort, ...). • „Ausblenden" anders verstanden, als: wird nicht angezeigt, aber bleibt im Hintergrund bestehen.
An welchen Stellen sind Hilfen (Anleitungen, Erklärungen) nötig?	• Beginn: Erklärung, was zu tun ist und wie Kontext-Modell aufgebaut ist (Zeilen = Situationen). • Zu High-Level-Kontext: größtmögliche Freiheit in Simulation; kein Reasoning.
Sind die Aufgaben bzw. Anwendungsfälle realistisch und nachvollziehbar?	• Größtenteils ja. • Kontext importieren schwer abbildbar auf Timeline („Es graust mir vor der Zeitauflösung / Menge" → Messen im Sekundentakt).
Welche Use-Cases (Anwendungsfälle) wurden ggf. noch nicht bedacht? Welche Funktionalität wird dazu benötigt oder gewünscht?	• Automatisches Testen für „Continuous Integration": Stapelverarbeitung mehrerer Simulationen. • Zweck des Autorensystems: Grundlage für Kommunikation mit Domänenexperten („größte Stärke von diesen visuellen Dingen"). • Debugging im Pause-Modus: Mouseover an Lerneinheiten. • Zustandekommen von High-Level-Kontext (Reasoning = Interpretation) testen: 1. High-level-Kontext zerlegen 2. Low-level-Kontext interpretieren

Welche Funktionalität ist am wichtigsten bzw. wird am häufigsten verwendet?	• Kontext ansehen, hinzufügen, bearbeiten, löschen, (verlängern). • Abgleich mit Lerneinheiten. • Play/Pause. • Simulation erstellen, bearbeiten.
Welche Funktionen werden nicht gebraucht?	• Ausblenden (mögl. wegen Missverständnis). • Speichern & Neu (zum Erstellen Top-Menü verwendet).
Werden Vorlagen entdeckt, gebraucht und als sinnvoll erachtet?	• Prinzipiell ja. • Aber: s. Anm.; mangelhaft modellierter Kontext.
Werden eigene Vorlagen zur Wiederverwendung erstellt?	• Nein. • Aber: erstellte wird geöffnet nach Navigation zu Lerneinheit-Tab.
Sind die Funktionen intuitiv bedienbar? Kontext modellieren	• Konsistenz („habe mich zurechtgefunden") durch: gleiche Symbole, Farben, „Menüs immer nach der gleichen Strategie aufgebaut". • Hinzufügen und Öffnen durch Klicken intuitiv. Verschieben und langziehen („an der Unterkante packen"), Löschen in Popover erwartet. • „Ausblenden" missverstanden.
Kontextmodelle verwalten	• Text in Erstellen-Dialog („als Vorlage") leicht missverständlich. • Button „Speichern & Neu" übersehen.
Simulation steuern	• Play/Pause intuitiv. • Schneller/langsamer erwartet.
Insgesamt	• Konsistenz („habe mich zurechtgefunden") durch: gleiche Farben, Symbole, „Menüs immer nach der gleichen Strategie aufgebaut". • Mouseover: Tooltips (Fließtexte) und passender Cursor erwartet. Icons alleine reichen nicht. • Ganze Sätze für gut befunden. • Annahme: Speichern per Default. • Navigation zum Simulator unklar, dadurch umständlich: Toolbar → Öffnen („Ach du Scheiße").
Welche Änderungen sind nötig oder wünschenswert?	• Vor-Auswahl von Kontextwerten (Default-Werte). • Vorschau/Überblick über Werte (wenigstens Tooltip). • Indikatoren für Timeline: wenigstens Nummerierung der Schritte. • Filtern nach „Sachen, die vorkommen": nur für Szenario relevanter Kontext. • Ablaufgeschwindigkeit einstellen evtl. sinnvoll („von mehreren Tools dieser Art" bekannt). • Formulierungen überarbeiten: in Erstellen-Dialog, „Ausblenden". • Navigation zwischen Tabs erklären/erleichtern.

Test 4 (09.12.15)	
Frage	Antwort
Durchgeführte Änderungen	• „Speichern & Neu" → „Optionen". • Hinter Optionen: „Kontextinformationen aus [Auswahlliste] importieren" + „Bestätigen", „Diese Kontextinformationen [Auswahlliste] als nicht verfügbar simulieren", „Simulation im Modus [Standard/Debug] starten". • Stopp-Button rechts neben Play. Davor „?"-Button. Anm. 1: *Wegen langer Einführung in Thema und missverständlichen Feedbacks nur 1. Aufgabe des Tests geschafft.* Anm. 2: *TP evtl. nicht Teil der Zielgruppe („falsche Generation", nutzt privat keinen Computer).*
Kann der Simulator (die „Testmöglichkeit") leicht gefunden werden?	Ja, nach Hinweis: Kontext der Lerneinheiten ändern würde Verhalten der App ändern.
Kann seine Funktionsweise erkannt werden?	• Unsicher. → s. Anm. • Nennt Kontextinformationen „Filterkriterien" (mögl. richtig erkannt?), sagt, Modellierung sei „schon okay".
Lassen sich Situationen wie von den Nutzern gedacht abbilden?	Unsicher. → s. Anm.
Wo gibt es Verständnisprobleme / Überraschungen?	• Probleme „auf die Meta-Ebene zu kommen". • Kann sich unter „Kontextsimulation" (ggü. „Simulation") nicht viel vorstellen. • Abfolge nicht ersichtlich. • Unklar, „was ich machen kann und machen will". • Feedback-Nachricht „Es wurden keine Lerneinheiten ausgewählt." missverstanden als Handlungshinweis („Dann müssen wir erst mal Lerneinheiten auswählen.").
An welchen Stellen sind Hilfen (Anleitungen, Erklärungen) nötig?	• Einstieg: Was kann und muss getan werden? • Simulation: Was wird simuliert? • Bei Nicht-Auswahl: Hinweis, dass (und ggf. welche) Informationen fehlen.
Sind die Aufgaben bzw. Anwendungsfälle realistisch und nachvollziehbar?	Unklar. (s. Anm. 1)
Welche Use-Cases (Anwendungsfälle) wurden ggf. noch nicht bedacht? Welche Funktionalität wird dazu benötigt oder gewünscht?	• Keine Angaben. (s. Anm. 1) • Aber prinzipiell mehr interessiert an Inhalten.
Welche Funktionalität ist am wichtigsten bzw. wird am häufigsten verwendet?	• Vergleich der modellierten Kontextinformationen: Lerneinheiten - Simulation. • Simulation starten. (s. Anm. 1)
Welche Funktionen werden nicht gebraucht?	• Kontextinformationen importieren, „als nicht verfügbar simulieren" (in „Optionen"). (s. Anm. 1)
Werden Vorlagen entdeckt, gebraucht und als sinnvoll erachtet?	Ja.
Werden eigene Vorlagen zur Wiederverwendung erstellt?	Nein. (s. Anm. 1)
Sind die Funktionen intuitiv bedienbar? Kontext modellieren	Popover-Fenster öffnen durch Anklicken der Kästchen, Bestätigen intuitiv.
Kontextmodelle verwalten	Dialog-Popoup-Fenster, Vorlagenauswahl und -beschreibung intuitiv.
Simulation steuern	Play-Button intuitiv.

	Starten in Optionen bei Modus-Auswahl vermutet. (Anm.: Kann an „low fidelity" liegen.)
Insgesamt	• Zurück-Button intuitiv. • Navigation unintuitiv („Wie bin ich hier hingekommen? Was war davor?"). • Öffnen/Schließen von Dialogen schlecht steuerbar („Wo geh ich zurück? Wie krieg ich das zum Verschwinden?"). • Bestätigen in Optionen fehlt. • Interaktionsmuster ungewohnt (z.B. Drag-and-drop).
Welche Änderungen sind nötig oder wünschenswert?	• Bezeichnung „Kontextsimulation" → „Testfall" besser. • Änderung der Benachrichtigung, dass *das System* keine Lerneinheit ausgewählt hat. • Hilfe weiter oben platzieren. (Wizard/Tutorial wäre auch sinnvoll.)

Test 5 (09.12.15)	
Frage	Antwort
Durchgeführte Änderungen	Keine weiteren.
Kann der Simulator (die „Testmöglichkeit") leicht gefunden werden?	Ja.
Kann seine Funktionsweise erkannt werden?	• Ja. (Richtige Annahme, aber unsicher.) • Benachrichtigung, dass keine Lerneinheit ausgewählt wurde, richtig verstanden („Also hat's nicht funktioniert."). • „Aufblinken" der Lerneinheiten als Auswahl verstanden („erst mal Ok").
Lassen sich Situationen wie von den Nutzern gedacht abbilden?	Ja. (Nach Erkenntnis, dass Felder angeklickt werden können.) Anm.: *Vorlage mögl. wieder zu gut passend vorbereitet!*
Wo gibt es Verständnisprobleme / Überraschungen?	• Am Anfang nicht klar, was gemacht werden kann. • In Simulation: keine Auswahl (Anm.: Scrollbalken nicht gesehen, d.h. dass freie Spalten vorhanden.) → Nicht klar, warum („Liegt's an Einstellungen oder an Simulation?") und was getan werden muss. → Unsicher, ob erst Lerneinheit ausgewählt werden muss → s. TP4.
An welchen Stellen sind Hilfen (Anleitungen, Erklärungen) nötig?	• Bei Nicht-Auswahl in Simulation zumindest eindeutiges Feedback. • Feedback zu Vollständigkeit der Simulation, d.h. fehlende Kontextinformationen.
Sind die Aufgaben bzw. Anwendungsfälle realistisch und nachvollziehbar?	Ja. (Aber würde App auch auf Gerät testen.)
Welche Use-Cases (Anwendungsfälle) wurden ggf. noch nicht bedacht? Welche Funktionalität wird dazu benötigt oder gewünscht?	Inhalte ansehen → wenigstens Vorschau.
Welche Funktionalität ist am wichtigsten bzw. wird am häufigsten verwendet?	• Kontextwerte ansehen, hinzufügen, bearbeiten, löschen, langziehen, ausblenden. • Kontexteinstellungen in Lerneinheiten und Simulation vergleichen. • Play/Replay.
Welche Funktionen werden nicht gebraucht?	• Kontext importieren. • Simulationsmodus (Debug).
Werden Vorlagen entdeckt, gebraucht und als sinnvoll erachtet?	Ja.
Werden eigene Vorlagen zur Wiederverwendung erstellt?	• Nicht mit der Absicht, aber um zurück zu kommen nach Lerneinheiten-Tab.
Sind die Funktionen intuitiv bedienbar? Kontext modellieren	• Felder anklicken, Popover zum Ansehen, bearbeiten und hinzufügen, Felder langziehen intuitiv. • (Möglichkeit, Kontext hinzuzufügen nicht gleich gefunden, weil freie Felder verdeckt waren und Scrollmöglichkeit nicht gesehen wurde. → Paper-Prototyp-Besonderheit!) • Filter verstanden. • Löschen nicht gefunden. • „Optionen"-Button (unten) gar nicht wahrgenommen. • „Kontextinformationen als nicht verfügbar simulieren" verständlich, Auge-Symbol auch gut, aber nicht in Optionen erwartet. • Findet Sortierung in Spalten gut.
Kontextmodelle verwalten	• Erstellen von Vorlage, Öffnen, Importieren → Optionen und Popup-Dialoge intuitiv verständlich. • Automatisches Speichern nicht ersichtlich.

Simulation steuern	• Play/Replay intuitiv. Erwartet Play statt Replay nach Änderungen am Modell. • Fortschrittsindikator = kleiner Pfeil an Zeilenrand, der weiterspringt, verständlich („Aha… spielt das durch”).
Insgesamt	• Öffnen/schließen nicht konsistent. • Navigation zurück zu Simulator nicht klar. • Speichern nicht ersichtlich.
Welche Änderungen sind nötig oder wünschenswert?	• Name der Simulation durch Anklicken bearbeiten (statt hinter Einstellungen-Button). • Löschen durch „Hochschieben” (Gegenstück zu Langziehen) oder in Popover erwartet. • Zeile einfügen (am linken Rand, wie bei Excel). • Zeile markieren und verschieben. • Löschen und Ausblenden als Optionen an jedem Kästchen (s. TP2 → Quick-Access) oder in Popover.

2 Ergebnisse der 2. Studie

Im Folgenden findet sich die vollständige Sammlung der Ergebnisse aller Tests in tabellarischer Form.

Anmerkungen:

- Für einige, aber nicht alle Tests sind Übersichten der Usability-Probleme beigefügt, wenn die Probleme zu umfangreich waren.
- Die Frage nach nicht benötigten Funktionen stellte sich als für diese Studie ungeeignet und nicht zu beantworten heraus. Denn das alleinige Nichtbenutzen einer Funktion in der beschränkten, einmaligen Testsituation lässt noch keine Schlüsse auf den tatsächlichen Bedarf zu. Es hätten hier allenfalls Bemerkungen der Testpersonen festgehalten werden können, wenn sie eine Funktion als überflüssig und störend empfänden. Dies war aber nie der Fall.
- Die Frage nach typischen Zielen und Abläufen und häufig benutzten Funktionen wurde nicht für alle Tests explizit beantwortet, da fast alle Testpersonen dieselben Schritte verfolgten, die sich zumeist aus der Aufgabenstellung ergaben. Vermerkt wurden daher hauptsächlich Auffälligkeiten im Sinne vom Abweichen von den „üblichen“ Arbeitsschritten.

Als Letztes sind die Antworten aller Probanden auf die Fragen (bzw. Aussagen) der System Usability Scale (SUS) angehängt.

Hintergrund	
Beruf/Fachrichtung	Informatik (Student, Berufserfahrung)
Lehre/E-Learning	nur Adaptives Lernen
Programmiererfahrung	ja, mehrere Software-Entwicklungs-Tools
Office-Erfahrung	ja (MS Excel, E-Mail, Google Calendar)
Schnittsoftware-Erfahrung	keine

Anmerkungen

1.Pilottest
- Technische Probleme: nur Notizen und Screen-, keine Tonaufzeichnung vorhanden.
- Test fand früher statt als vorgesehen, war noch nicht gut vorbereitet:
- Interview weniger strukturiert.
- Bugs, die sich z.T. durch bessere Vorbereitung hätten vermeiden lassen.

Änderungen

keine (Pilottest)

Aufgetretene Bugs / Probleme

schwer:
- Regeln nicht neu generiert, dadurch kein Export und fehlerhaftes, unerwartetes Verhalten.
- Mehrere in 1. Spalte „Abgeschlossene Lerneinheiten": nach Drag-and-Drop Werte werden überschrieben; nach Start der Simulation Auswahl weg, dafür Input-Feld.
- Bugs im Framework (erst am Ende der Studie entdeckt).
- Ältere Version der Adaptation Engine: „CONTAINS"-Operator nicht umgesetzt.

leicht:
- Nutzen der vor/zurück-Tasten pausiert Simulation nicht, dadurch ungewünschtes Springen bei vor, zurück erst bei mehrmaligem Klicken.
- Spalten zu schmal, daher Kopieren-Icon nicht erreichbar.
- Fehlerhafte Darstellung von Bootstrap-Tooltips.
- „Overload" durch zu viele und schlecht dargestellte Tooltips: unübersichtlich, Funktionsverlust, d.h. kein informativer Mehrwert mehr.

Sonstige Bemerkungen / Auffälligkeiten

Erste Schritte:
- Erkundung der Oberfläche durch Mouseover.
- Lässt Simulation komplett durchlaufen (kein Ergebnis, keine Nachricht).
- Klick in Zeile zur Auswahl in Simulation.
- Öffnet und liest **Hilfe zur Zeitleiste.**
- Fügt Informationen hinzu, lässt dabei jeweils mehrere Zeilen Abstand.

Ergebnisse

Testfragen	**Beobachtungen und Kommentare**	**Interview und SUS**
Verständnis		
Wird die Zeitleiste als Abfolge von Situationen erkannt?	• Reihenfolge erkannt. • Fährt gleich zu Beginn mit Maus über Zeilenköpfe, sieht Tooltips: „Situation [1..N]" • Nennt Situationen • „Kombinationen" → mentales Modell von „Situation": keine Momentaufnahme, sondern auch Ablauf möglich.	
Werden die Kontextinformationen aus dem Szenario wiedererkannt?	• Fragt erst, warum es nicht mehr gibt. Wird dann klar (ohne Hilfe) und nicht mehr benötigt.	Zusammenhang von Kontext an Lerneinheiten und in Testszenario klar durch Vorauswahl der relevanten Kontextinformationen, „klar" und

		„übersichtlich" durch Spalten und Farben.
Wird die Visualisierung richtig verstanden?	Teils: Auswahl ja.	
Wird klar, was getan werden soll und wie?	Ja. (s.o.: Erste Schritte)	
Haben die Hilfstexte (Erklärungen hinter „?") einen Effekt auf das Verständnis? Welchen?	• Erklärung zur Zeitleiste benötigt und hilfreich (verstanden und auch Tipp umgesetzt). • Erklärung zum Simulator hilfreich (wegen Hinweis zu abgeschlossenen Lerneinheiten).	
Was ist unverständlich? Wo werden noch mehr Hilfen benötigt?	Fehlverhalten der Simulation. Aber klar, dass es Fehler im System gibt, nicht in der Logik.	• In Simulation: Irritation wegen abweichenden Verhaltens ggü. App: Einmal ausgewählte Lerneinheiten werden nicht ausgeblendet, sondern bleiben aktiviert. • „Abgeschlossene Lerneinheit" nicht eindeutig: abgeschlossen durch Nutzer oder System? • Workshop oder Videoanleitung nötig, dann gut nutzbar.
Funktionalität		
Was sind typische Ziele und Abläufe? Was sind die wichtigsten bzw. am häufigsten benutzten Funktionen?	• Szenario wird im Simulator nachgebildet, Orientierung an Lerneinheiten: • Kontextinformationen "nachschlagen" und kopieren • Kontext und abgeschlossene Lerneinheiten in Simulation einstellen • Tooltips lesen • Simulation starten • Überprüfung der Auswahl bestimmter Lerneinheiten • Vergleich Lerneinheit - Zeile	
Welche Use-Cases werden noch nicht unterstützt?		Darstellung/Abspielen der Inhalte überprüfen (in Fenster oder Emulator).
Welche (wichtigen) Funktionen fehlen?		• Historie der Auswahl (als Annäherung an Verhalten der App) • Liste in Report am Ende oder/und andersfarbiges Highlighting im Graph. • Relationen hervorheben (passend zu abgeschlossenen Lerneinheiten) • Plausibilitätsprüfung bei abgeschlossenen Lerneinheiten • Filtern nach Lerneinheiten, die bereits ausgewählt wurden, statt einfach Lerneinheiten zu bestimmen

Welche Funktionen werden nicht benötigt?	-	
Bedienbarkeit		
Ist der Simulator einfach und intuitiv bedienbar?	siehe Tabelle "Bedienbarkeit"	siehe SUS
Wird der Simulator als konsistent, auch mit dem Autorensystem, empfunden?		siehe SUS
Welche Verbesserungen sind nötig oder gewünscht?	• Notification irritierend, wenn keine neuen Regeln generiert werden müssen. • Erwartet Ladebalken, "Warten"-Kreis o.ä. bei Regelgenerierung (statt nur 1 Nachricht) • Verwunderung, dass Kontext-Events bei Kopieren und Einfügen einer Zeile gespalten werden, hätte Merging erwartet • Zu viele Zeilen (80): Abwarten bis Simulationsende dauert zu lang • Abspielknöpfe Vor/Zurück sollten auch ohne Play funktionieren.	• Icons besser im Popover, "Löschen" besser nicht in neuem Popover • Flexible Breite des Panels gewünscht (Aufziehen) • Lerneinheiten sollten sortiert (alphabetisch), durchsuchbar und nach Szenariologik gefiltert sein (nur die, die noch nicht abgeschlossen sind) • Simulation sollte pausieren bei Vor/Zurück.
Gesamteinschätzung		
Wurden Erwartungen getroffen? *Welche (anderen) Erwartungen gab es?*		nicht gefragt (s. Anm.)
Gibt das Testen mit dem Simulator ein gutes Gefühl?		nicht gefragt (s. Anm.)
Positives		• Möglichkeit der Simulation ggü. Test auf Handy. • „Oft Jammern auf hohem Niveau"
Was war am schwersten?		nicht gefragt (s. Anm.)
Kommentare zur SUS		
• Annahme, TP erstellt eine Lernanwendung mit dem Autorensystem • Kein Transkript vorhanden (s. Anm.)		

Funktion	Anmerkungen
Vorlagen: Auswahl + Neue erstellen	leicht bedient
Vorlagen: Eigenschaften (Name, Beschreibung)	etwas umständlich: Bestätigen erwartet
Vorlagen: Löschen, Kopie erstellen	nur löschen genutzt
Kontext hinzufügen: 1 Feld	anfangs zufällig gefunden - leicht
Kontext hinzufügen: > 1 Feld	in 1. Spalte vergeblich versucht, dann auch verstanden, warum das nicht geht bei anderen leicht
Kontext editieren in Popover	sehr umständlich: Bug in Abgeschlossenen Lerneinheiten: immer wieder Input-Feld statt Auswahlliste; alphabetische Sortierung der Lerneinheiten gewünscht

Test 1 (11.03.2016)

Quick-Access: Tooltip	sehr umständlich: schlecht verfügbar (zu schmaler Bereich und ausgeblendet, wenn Maus über Icon) und leicht fehlerhaft dargestellt (Bug in Bootstrap)
Quick-Access-Optionen (Icons)	fehlerhafte Darstellung, weil Spalten zu schmal (falscher Default eingestellt); insgesamt etwas umständlich, weil alles sehr klein
Quick-Access: Bearbeiten	s.o.
Quick-Access: Löschen	leicht
Quick-Access: Kopieren	wegen Bug (Spaltenbreite) nicht anzuklicken
Quick-Access: Ausblenden	zuerst versehentlich genutzt, wollte kopieren; später genutzt - verstanden
Quick-Access: Verschieben	nicht genutzt
Quick-Access: Länge ändern	genutzt - leicht bedienbar
Spaltenoptionen	entdeckt, nachdem auch Zeilenoptionen gefunden wurden
Spalte (Werte) löschen	erwartet, komplette Spalte zu löschen, nicht nur Werte
Spalte (Werte) nicht erfassen	genutzt, aber nicht benötigt
Spalte hinzufügen	nicht vorhanden
Zeilenoptionen	Label sehr schmal (geändert); Tooltip "Situation [1..N]" wird angezeigt alle einmal getestet
Zeile kopieren und einfügen	mehr Feedback besser: welche ist kopiert?
Steuerung: Zeile auswählen	durch Klick in Zeile (statt auf Label) versucht
Steuerung: Vor/Zurück	Simulation pausiert nicht; Steppen sollte auch gehen, nachdem Simulation durchgelaufen ist
Optionen: Geschwindigkeit	nicht genutzt

Hintergrund	
Beruf/Fachrichtung	Informatik (Student, Berufserfahrung)
Lehre/E-Learning	nur Adaptives Lernen (mehr interessiert an Kontexterfassung als an E-Learning)
Programmiererfahrung	ja, mehrere Software-Entwicklungs-Tools
Office-Erfahrung	ja (aber eher Programmierer als Anwender)
Schnittsoftware-Erfahrung	keine

Anmerkungen

2. Pilottest
- Technische Probleme: nur Notizen und Screen-, keine Tonaufzeichnung vorhanden
- Test fand früher statt als vorgesehen, war noch nicht gut vorbereitet:
  - Interview weniger strukturiert
  - Bugs, die sich z.T. durch bessere Vorbereitung hätten vermeiden lassen
  - Probleme mit Szenario: TP legt 2. Szenario an (noch keine Unterstützung)

Änderungen

keine (Pilottest)

Aufgetretene Bugs / Probleme

siehe Test 1.

Sonstige Bemerkungen / Auffälligkeiten

- Nutzertyp: immer alles ausprobieren, mit allem rumspielen („Kann ja nichts kaputt gehen")
- Erste Schritte: Hilfe zum Simulator gelesen. Mouseover über jedes Feld der 1. Zeile. Klick in 1. Feld, schließt Popover wieder. Benennt erst Vorlage um.

Ergebnisse

Testfragen	**Beobachtungen und Kommentare**	**Interview und SUS**
Verständnis		
Wird die Zeitleiste als Abfolge von Situationen erkannt?	Ja.	Hat auch verstanden, dass Reihenfolge prinzipiell beliebig ist: Dass Schritte nicht Zeiten sind „muss gelernt werden". Schritte als Orientierungshilfe.
Werden die Kontextinformationen aus dem Szenario wiedererkannt?	Ja.	Zusammenhang zwischen Kontext links und rechts gut verständlich.
Wird die Visualisierung richtig verstanden?	Ja.	
Wird klar, was getan werden soll und wie?	Ja.	
Haben die Hilfstexte (Erklärungen hinter „?") einen Effekt auf das Verständnis? Welchen?	• Effekt unklar. Hilfe zum Simulator gleich zu Beginn gelesen, alles verstanden • Hilfe zur Zeitleiste erst später gelesen, nachdem Simulation nicht richtig funktionierte. Tipp hilfreich, Beispiel zufällig passend zum Szenario.	

Was ist unverständlich? Wo werden noch mehr Hilfen benötigt?	Fehlverhalten der Simulation. Aber klar, dass es Fehler im System gibt, nicht in der Logik.	• In Simulation: Irritation wegen abweichenden Verhaltens ggü. App: Einmal ausgewählte Lerneinheiten werden nicht ausgeblendet, sondern bleiben aktiviert • "Abgeschlossene Lerneinheit" nicht eindeutig: ○ 1. Abgeschlossen durch Nutzer oder System? ○ 2. Gültigkeit unklar: Ab wann? Warum nicht "langziehbar" wie andere Infos? (Inhärent "lebenslange" Gültigkeit nicht ersichtlich, Tooltip nicht wahrgenommen)
Funktionalität		
Was sind typische Ziele und Abläufe? Was sind die wichtigsten bzw. am häufigsten benutzten Funktionen?	• Auswahl der Lerneinheiten prüfen • Kontext in Lerneinheiten ansehen (bei nicht selbst erstelltem Szenario) • Kontext und abgeschlossene Lerneinheiten in Simulation einstellen • Tooltips lesen • Simulation starten, Auswahl überprüfen • Legt viele neue unbenannte Vorlagen an.	
Welche Use-Cases werden noch nicht unterstützt?		Darstellung/Abspielen der Inhalte überprüfen (in Fenster oder Emulator).
Welche (wichtigen) Funktionen fehlen?		• Historie der Auswahl (als Annäherung an Verhalten der App) • Liste in Report am Ende oder/und andersfarbiges Highlighting im Graph • „Nice to have": Testfälle automatisch nacheinander ablaufen lassen.
Welche Funktionen werden nicht benötigt?	-	
Bedienbarkeit		
Ist der Simulator einfach und intuitiv bedienbar?	siehe Tabelle "Bedienbarkeit"	siehe SUS
Wird der Simulator als konsistent, auch mit dem Autorensystem, empfunden?		siehe SUS

Welche Verbesserungen sind nötig oder gewünscht?	• Zu viele Zeilen (80): Abwarten bis Simulationsende dauert zu lang. Simulation sollte leere Zeilen überspringen • Abspielknöpfe Vor/Zurück sollten auch ohne Play funktionieren • Durchgängig geltende Informationen sollten verbunden werden (merging und splicing) • „Testfall kopieren" - Kopie sollte sich gleich öffnen (s. Google Docs) • Panel und Spalten zu schmal (s.a. Bug).	• „Lerneinheit abgeschlossen" klarer formulieren: Nutzer hat zu diesem Zeitpunkt absolviert • Zeilen-/Spalten-Optionen nicht gefunden. Vorschlag: Optionen-Popover schon bei Mouseover anzeigen statt erst bei Klick.
Gesamteinschätzung		
Wurden Erwartungen getroffen? *Welche (anderen) Erwartungen gab es?*		nicht gefragt (s. Anm.)
Gibt das Testen mit dem Simulator ein gutes Gefühl?		nicht gefragt (s. Anm.)
Positives		• "klar definiert": Übersicht, Zeitleiste, Kontext • Möglichkeit der Simulation ggü. Test auf Handy.
Was war am schwersten?		nicht gefragt (s. Anm.)
Kommentare zur SUS		
• Annahme, TP erstellt eine Lernanwendung mit dem Autorensystem • Kein Transkript vorhanden (s. Anm.)		

Funktion	Anmerkungen
Vorlagen: Auswahl + Neue erstellen	Bug in Vorlagenauswahl: falsche Zuordnung; unübersichtlich bei mehreren "Leere Vorlage"
Vorlagen: Eigenschaften (Name, Beschreibung)	Bestätigen erwartet
Vorlagen: Löschen, Kopie erstellen	"Testfall kopieren" - sollte gleich geöffnet werden
Kontext hinzufügen: > 1 Feld	in 1. Spalte versucht, wo es nicht geht; markieren aber möglich (falsches Signal); da, wo es ging, leicht
Kontext editieren in Popover	sehr umständlich: Bug in Abgeschlossenen Lerneinheiten: immer wieder Input-Feld statt Auswahlliste
Quick-Access: Tooltip	leicht - liest vor allem Tooltips an leeren Feldern
Quick-Access-Optionen (Icons)	
Quick-Access: Löschen	Bug in Abgeschlossenen Lerneinheiten: löscht eins, alle weg - sonst leicht
Quick-Access: Kopieren	einmal versucht, aber immer noch Anzeigebug, daher nicht bedienbar
Quick-Access: Ausblenden	nicht genutzt
Quick-Access: Verschieben	genutzt
Quick-Access: Länge ändern	nach Lesen der 2. Hilfe ausprobiert - leicht
Spaltenoptionen	erst nach dem Test entdeckt, nachdem auf Zeilenoptionen hingewiesen wurde

Test 2 (11.03.2016)

Zeilenoptionen	Nicht gefunden. Label sehr schmal (geändert); Tooltip "Situation [1..N]" wird angezeigt. Hinweis nach Test.
Steuerung: Vor/Zurück	Simulation pausiert nicht; Steppen sollte auch gehen, nachdem Simulation durchgelaufen ist
Optionen: Geschwindigkeit	nicht genutzt

Hintergrund	
Beruf/Fachrichtung	Informatik
Lehre/E-Learning	beides
Programmiererfahrung	ja, verschiedene Software-Entwicklungs-Tools
Office-Erfahrung	ja (MS Office, E-Mail, Kalender)
Schnittsoftware-Erfahrung	ja, Animationsprogramme (Macromedia, Adobe)

Anmerkungen

- Teilnehmer der Paper-Prototyping-Studie
- Szenario aus Paper-Prototyping wiederverwendet und vorbereitet, aber eine Kontextinformation fehlte in Modellierung
- Bugs im System, daher keine Adaptierung → Wechsel zu Demo-Szenario

Änderungen

- In Pilottests aufgetretene Bugs (die identifizierten) behoben
- Zwei Farben für Highlighting: aktuelle (cyan) und Historie (darkcyan)
- Anzeige der Benachrichtung am Ende der Simulation: keine Lerneinheiten gefunden
- "Testfall kopieren" umbenannt in "Kopie öffnen" und öffnet Kopie in Hauptansicht
- Bestätigen-Knopf in Testfall-Einstellungen hinzugefügt
- Beschränkung der Bootstrap-Tooltips auf Kontextinformationen (Felder), um visuellen "Lärm" zu reduzierten und Aufmerksamkeit auf Kontextwerte zu lenken
- Kopierte Situation eindeutig als kopiert gekennzeichnet (gestrichelte Border)
  Anzahl der Zeilen reduziert: von 80 auf 20.
- "Abgeschlossene Lerneinheiten" umbenannt in "Absolvierte Lerneinheiten" und nicht mehr "langziehbar" (Hinzufügen nur noch durch Klicken), dafür Einfärben der unteren Zellen in mittlerer Intensität, um fortbestehende Gültigkeit anzudeuten
- Highlighting der Zeilen/Spalten bei Mouseover über Label, außerdem breiteres Zeilenlabel, um (dezent) auf Optionen hinzuweisen.

Aufgetretene Bugs / Probleme

schwer:

- Regeln werden nicht neu generiert
- Bugs im Framework (erst am Ende der Studie entdeckt)
- Ältere Version der Adaptation Engine: “CONTAINS”-Operator nicht umgesetzt
- Kontextparameter werden nicht richtg an Adaptation Engine übergeben (Formatfehler), dadurch kein Matching

leicht (aber Verwirrung stiftend)

- In Simulation: bei Zurücksteppen bleibt History geighlightet.

Sonstige Bemerkungen / Auffälligkeiten

- Windows-Nutzer: erwartet Rechtsklick
- Würde keine Systemhilfen in Anspruch nehmen
- Erwartung/Annahme, man könne die Simulation durch Interaktion mit Lerneinheiten steuern.

Ergebnisse

Testfragen	**Beobachtungen und Kommentare**	**Interview und SUS**
Verständnis		
Wird die Zeitleiste als Abfolge von Situationen erkannt?	Nicht auf den ersten Blick, erst nach Simulation „Simulationsschritte” erkannt. Zuerst für jede Situation neue Vorlage angelegt.	
Werden die Kontextinformationen aus dem Szenario wiedererkannt?	Ja. (s. Anm.)	

Wird die Visualisierung richtig verstanden?	Teils: • Highlighting von „Sachen", die zu Schritten passen • Zwei Farben unklar	
Wird klar, was getan werden soll und wie?	• Richtig verstanden, aber unsicher • Kontextinformationen gesehen: „Das muss ich intellektuell erst mal verarbeiten, was das für mich konkret bedeutet." • Verständnis und Erwartung, dass beliebige Lerneinheiten als abgeschlossen bestimmbar sind.	
Haben die Hilfstexte (Erklärungen hinter „?") einen Effekt auf das Verständnis? Welchen?	• Beide zu Anfang gelesen, Effekt unklar • Hilfe zum Simulator: „Alles klar" • Hilfe zur Zeitleiste: „Da bin ich gespannt, wie das geht." (Tipp)	
Was ist unverständlich? Wo werden noch mehr Hilfen benötigt?	• 1. Spalte: „Die zwei Ebenen verschwimmen" • In Simulation: Verwunderung über immer aktive Lerneinheit (aber Annahme richtig: keine Bedingung).	• 1. Spalte (Absolvierte Lerneinheiten): „am Anfang gar nicht gerafft" • Sollte stärker abgehoben sein, „weil die auch immer da ist": Doppel-Border, oder über andere Spalten „Kontextinformationen" schreiben, über 1. Nichts • Bei Simulationsschritten: Matching rechts und links fehlt • „Knoten-Highlighting" verwirrend: Was bedeuten die zwei Farben? (Vermutung: zeigt Sequenz an.) • Einführungsvideo / Workshop zwingend notwendig.
Funktionalität		
Was sind typische Ziele und Abläufe? Was sind die wichtigsten bzw. am häufigsten benutzten Funktionen?	• Ziel: Standardfall des Szenarios testen. (Anm.: War durch Aufgabe vorgegeben.) • Vorlagenauswahl ansehen, neue anlegen und umbenennen • Wechsel zwischen Lerneinheiten-Kontext-Tab und Simulator: Kontrollieren und Kopieren von Werten • Kontext modellieren und überprüfen: Tooltips lesen, Kontext hinzufügen, löschen, kopieren • Simulation abspielen und überprüfen: Play und Pause, Vor- und Zurückspringen.	

Welche Use-Cases werden noch nicht unterstützt?	• Testfall als Vorlage für sukzessive Erstellung eines neuen Szenarios • (Noch) nicht relevanter Kontext sollte vorhanden sein (ausgegraut) zur Orientierung.	
Welche (wichtigen) Funktionen fehlen?	Manuelles Löschen von Kontext-Spalten, ergo auch Hinzufügen erwartet.	• Sehr wichtig, auch für Verständnis: Begründung für (Nicht-) Auswahl, Unterstützung, um „das Rechte mit dem Linken zu verbinden" • Mouseover an Lerneinheiten: Matching („kleine Infobox: Das trifft zu, weil XYZ zutrifft") • Wichtig: Konsistenzprüfung - Schließen sich Werte aus? Ist Kontext logisch?
Welche Funktionen werden nicht benötigt?	-	
Bedienbarkeit		
Ist der Simulator einfach und intuitiv bedienbar?	siehe Tabelle "Bedienbarkeit"	siehe SUS
Wird der Simulator als konsistent, auch mit dem Autorensystem, empfunden?		siehe SUS
Welche Verbesserungen sind nötig oder gewünscht?	Herausfiltern nicht vorkommender Informationen bei Verwendung einer anderen Vorlage „uncool" und nicht nachvollziehbar.	• Stoppen "frickelig" (Pause bei Anwahl einer Zeile in Simulation) • Sichtbarkeit, Überblick ("alles zu klein"): 1. Panel vergrößern. 2. Zu viele Zeilen in der Zeitleiste (20): sollte ab 10 weiß bleiben, mit Hinzufügen-Option. 3. Kontextwerte "verkürzt" in Kästchen, Icons dafür in Popover verlagern ("viel besser") • Mehr Unterstützung bei Kontext: Date/Time Picker etc.
Gesamteinschätzung		
Wurden Erwartungen getroffen? *Welche (anderen) Erwartungen gab es?*		• „Standardproblem", kennt Zeitraster von Adobe Dreamweaver u.a. Macromedia-Produkten • Hochkant-Format nicht verwirrend („nur um 90° gedreht") • Nichts anderes erhofft.
*Gibt das Testen mit dem Simulator ein gutes Gefühl?**		Ja, aber würde zum jetzigen Zeitpunkt auch noch manuell testen („Vertrauensfrage")

Positives	• Testfälle umbenennen („sehr nice") • Tooltip („Sehr sehr gut") • Kopieren • Herausfiltern nicht mehr verwendeter Kontextinformationen des Szenarios („Gute Arbeit, hat mir gefallen") • Zeilenindikator („Der Pfeil gefällt mir auch.") • Bezeichnungen und Hilfstexte gut • Möglichkeit, beliebige Lerneinheiten als abgeschlossen anzugeben.	• Farben • Kontrollleiste („sinnvoll" und symmetrisch) • „der Detailgrad" („viel Mühe reingesteckt") • „Kontexte anlegen war einfach" • Zeitleiste „übersichtlich" gestaltet • „Das mit den Bedienelementen fand ich intuitiv" (Icons)
Was war am schwersten?		Größte Schwierigkeit: nicht zu wissen, in welchem Modell der Fehler liegt.
Kommentare zur SUS		
• Annahme, TP erstellt eine Lernanwendung mit dem Autorensystem • Annahme, es hätte so funktioniert wie in Demo-Szenario • 4. Technischer Support „oder ich brauch viel Zeit. Ich brauche ein Tutorial-Video oder so was." Würde auch danach noch mal Hilfe brauchen • „Das ist jetzt auf die Simulation bezogen, ne? Nicht auf das gesamte Ding. Oder worauf ist das bezogen?" • „Es sind ja jetzt nur die Lerneinheiten und die Kontextinformationen, also ohne Content" • Technischer Support für Modellierung: „Es gibt ziemlich viele Kontexte." Zweifel, ob Kontext falsch modelliert ist (Lerneinheiten oder Simulation) • „Wenn ich komplexere Szenarien mache, brauch ich schon Unterstützung" • 7. „die meisten Leute…komplett überfordert: Der Anwendungskontext ist schon sehr spezifisch ... schon auf das Wesentliche reduziert, aber immer noch zu viel, zu viele Kontexte" und Lerneinheiten-Graph zu groß. • 10. Annahme, dass mit „eine Menge Dinge lernen" (letzter Punkt) Bedienelemente gemeint sind („Da hast du Glück").		

Funktion	Anmerkungen
Vorlagen: Eigenschaften (Name, Beschreibung)	"sehr nice"
Vorlagen: Löschen, Kopie erstellen	Bug: Vorlagen weg
Kontext hinzufügen: > 1 Feld	nicht genutzt
Kontext editieren in Popover	Bug: Popover schließt sich beim Markieren (behoben) Mehr Unterstützung gewünscht (Timepicker, Datepicker)
Quick-Access: Tooltip	Tooltip weg, wenn Maus über Icon (behoben) Feld zu klein
Quick-Access-Optionen (Icons)	"intuitiv", aber besser in Popover stattdessen Kontextwerte
Quick-Access: Ausblenden	Bug in 1. Spalte: 1 ausblenden blendet alle aus
Spaltenoptionen	
Spalte (Werte) löschen	erwartet, ganze Spalten löschen zu können
Spalte (Werte) nicht erfassen	nicht genutzt
Spalte hinzufügen	nicht vorhanden
Zeilenoptionen	Löschen gesucht, bei 3. Versuch gefunden: 1. Versuch: Rechtsklick, 2. Versuch: Entf ("Ich sehe, du hast dir richtig Gedanken gemacht.")

Test 3 (17.03.2016)

Steuerung: Zeile auswählen	kein automatisches Pausieren (behoben)
Steuerung: Vor/Zurück	kaum genutzt, eher Auswahl durch Anklicken
Optionen: Geschwindigkeit	nicht genutzt

Hintergrund	
Beruf/Fachrichtung	Informatik (Student)
Lehre/E-Learning	E-Learning, Adaptives Lernen
Programmiererfahrung	ja, verschiedene Software-Entwicklungs-Tools
Office-Erfahrung	ja
Schnittsoftware-Erfahrung	nein, "Zeitleistenmäßiges schon länger nicht mehr", aber bekannt aus Simulationen (Spiele)

Anmerkungen

- Teilnehmer der Paper-Prototyping-Studie. Erinnerte sich noch sehr gut. („diese tolle Zeittabelle, an die ich mich noch erinnere.") Kannte Framework aus Seminar
- Bugs im System, daher keine Adaptierung → Wechsel zu Demo-Szenario.

Änderungen

- Bugfix: Regelgenerierung und Kontextparameterübergabe an Adaptation Engine überarbeitet
- Doppelte Border zwischen 1. Spalte und übrigen hinzugefügt zur besseren Abgrenzung.

Aufgetretene Bugs / Probleme

schwer:

- Weiterer Bug in Kontextparameterübergabe an Adaptation Engine
- Bugs im Framework (erst am Ende der Studie entdeckt)
- Ältere Version der Adaptation Engine: „CONTAINS"-Operator nicht umgesetzt

leicht: siehe Test 3

Sonstige Bemerkungen / Auffälligkeiten

- Sonderfall: „Wert unbekannt" hat andere Bedeutung als dass kein Wert geliefert („nicht erfasst") wird
- Erste Schritte
  - Hält Maus über erstes Feld und liest Tooltip
  - „Das erste, was ich mir denke ist: Die blaue Spalte muss wohl wichtig sein. Die ist so abgehoben, aber wahrscheinlich sind die thematisch zugeordnet, weil ich glaube, dass ich die Farben wiedererkenne."
  - „Und jetzt fang ich einfach mal an, mit dem Mouseover rumzuspielen."
  - Erwartet mehr Interaktion mit den Lerneinheiten: Reinziehen in Zeitleiste.

Ergebnisse

Testfragen	**Beobachtungen und Kommentare**	**Interview und SUS**
Verständnis		
Wird die Zeitleiste als Abfolge von Situationen erkannt?	• Ja, alles sofort richtig erkannt und verstanden • Zeitliche Abfolge an Ziffern erkannt. (Erinnert sich aber auch an Paper-Prototyping.) s. Anm.	
Werden die Kontextinformationen aus dem Szenario wiedererkannt?	Ja, durch Farben und Tooltips. s. a. Anm.	
Wird die Visualisierung richtig verstanden?	• Ja, Erwartungen getroffen • Aber zwei Farben nur richtig verstanden wegen Kenntnis der App.	
Wird klar, was getan werden soll und wie?	Ja.	
Haben die Hilfstexte (Erklärungen hinter „?") einen Effekt auf das Verständnis? Welchen?	Nein.	

Was ist unverständlich? Wo werden noch mehr Hilfen benötigt?	• Fehlverhalten der Simulation wegen Fehlern im System • Handlungsspielraum nicht klar: Erklärung, dass Lerneinheiten auch angesehen werden können, wenn nicht ausgewählt. („anderes Verständnis von [...] App")	10 - 15-minütiges Video für Einstieg in Thematik und Autorensystem.
Funktionalität		
Was sind typische Ziele und Abläufe? Was sind die wichtigsten bzw. am häufigsten benutzten Funktionen?	• Szenario ausgedacht, konkrete Erwartungen, was passieren wird (Auswahl) • Tooltips lesen • Kontextwerte anlegen und ggf. verlängern (nacheinander von links nach rechts, 1. Spalte zum Schluss). Testen • Szenarien: 1. Standardfall. 2. Prüfen, ob Lerneinheit noch zu späterem Zeitpunkt gelesen werden kann, nachdem Bedingungen nicht mehr gelten.	Simulator für Verifizierung und Nachvollziehen (Fokus auf Kantenbeziehungen).
Welche Use-Cases werden noch nicht unterstützt?		• Stapelverarbeitung und kontinuierliches, automatisiertes Testen (Testgetriebene Entwicklung) • Soll-Ist-Abgleich 1. Einstellen, welche Einheiten pro Szenario (nicht) ausgewählt werden sollen 2. Abschlussbericht: Zustand eingetreten oder nicht
Welche (wichtigen) Funktionen fehlen?	• Filteroption in Abgeschlossene Lerneinheiten-Popover (Checkbox): nur freigeschaltete Lerneinheiten	Kontextwertanzeige auch an Lerneinheiten bei Mouseover.
Welche Funktionen werden nicht benötigt?	-	
Bedienbarkeit		
Ist der Simulator einfach und intuitiv bedienbar?	siehe Tabelle "Bedienbarkeit"	siehe SUS
Wird der Simulator als konsistent, auch mit dem Autorensystem, empfunden?		siehe SUS
Welche Verbesserungen sind nötig oder gewünscht?	• Überblick fehlt: alle Spalten sollten gleichzeitig sichtbar sein. • Alphabetische Sortierung der Lerneinheiten • Einstellen, dass Wert für gesamten Testlauf gelten soll.	• Mehr Schrift, weniger Symbole: Kontextwerte direkt als Text in Zellen (erste zwei Zeilen). Icons erst bei Mouseover • Automatisches Hinzufügen von (mehreren) Zeilen am Tabellenende (wie bei Excel) bei Scrollen und bei Verschieben von Werten • Oder: Plus-Button am Ende. Klick auf Zeilennummer nicht erwartet

		• Filteroption in „Absolvierte Lerneinheit"-Popover (Checkbox): nur freigeschaltete Leineinheiten • Besser unterschiedliche Symbole für Eigenschaften und Optionen (Zweimal Zahnrad-Symbol)
Gesamteinschätzung		
Wurden Erwartungen getroffen? *Welche (anderen) Erwartungen gab es?*		„Noch zwei drei Sachen mehr gewünscht":
Gibt das Testen mit dem Simulator ein gutes Gefühl?		Wenn Ist und Soll übereinstimmen
Positives	Visualisierung des letzten Zustands noch nach Durchlauf.	• Visualisierung: Zusammenspiel von Fortschritt in Zeitleiste und Lightboxing „intuitiv" • Farben: anschaulich und geordnet • Vorfiltern nach relevanten Kontextinformationen „gar nicht aufgefallen, das war so intuitiv" • Intuitive Manipulation der Werte • Durchsteppen um zu „gucken, woran's hakt" • Insgesamt erwartungskonform.
Was war am schwersten?	Bug in 1. Spalte: Überschiebene Lerneinheiten, muss alles löschen und noch mal von vorne hinzufügen.	• Unsicherheit, welches Modell fehlerhaft ist • Kognitive Belastung, weil 1. Werte nur bei Mouseover angezeigt werden → besser direkt als Text in Zellen 2. bei Fehlverhalten Situationen umständlich abgeglichen und gemerkt werden müssen • „Zustandsveränderungen im Kopf synchron [zu] halten", „ständig hin und her klicken"

Kommentare zur SUS

- „Ich vermute mal, das geht jetzt von der Annahme aus, dass ich regelmäßig mit diesem Kontext-Framework arbeite."
- „Mit 'das System' meinst du deinen Test-Prototypen, oder?"
- Technischer Support: „hier und da vielleicht für Fragen"
- Inkonsistenzen: „konsequent durchgezogen, sowohl im Guten als auch im Schlechten"
- 7. „...die meisten Leute...": Video wäre schon nett, Workshop nicht notwendig. „So wie es gerade ist [...] Autorensystem kriegst du ohne Einführung nicht hin."
- 9. „zwischen 2 und 3", „Nicht sicher, ob es ein Fehler oder mein eigenes Versagen war. Ich hatte nicht das Gefühl, dass ich was kaputt machen könnte, aber ich hab ständig das System hinterfragt, wenn es darum ging, dass ich vielleicht einen Fehler gemacht hab."
- 10. „Nö. Eigentlich nicht. Ich bin sicher, da hat mir meine Erfahrung geholfen, dass ich schon damit schon mal gearbeitet hab und dass ich mit dem Autorensystem gearbeitet hab."

Test 4 (18.03.2016)

Funktion	Anmerkungen
Vorlagen: Eigenschaften (Name, Beschreibung)	„Einstellungsrädchen im ersten Moment komplett missachtet"
Vorlagen: Löschen, Kopie erstellen	nicht genutzt
Kontext hinzufügen: > 1 Feld	nicht genutzt: immer nur 1 Feld anlegen + langziehen
Kontext editieren in Popover	alphabetische Sortierung der Lerneinheiten gewünscht; kleiner Bug: Popover schließt unerwartet
Quick-Access: Tooltip	wird manchmal nicht angezeigt
Quick-Access-Optionen (Icons)	
Quick-Access: Kopieren	nicht genutzt (stattdessen langziehen)
Quick-Access: Ausblenden	„so was wie auskommentieren" Auge eher für Darstellung bekannt, aber Tooltip klar.
Quick-Access: Verschieben	am Tabellenende muss erst verkürzt werden, bevor weiter verschoben werden kann
Quick-Access: Länge ändern	„ganz ganz winziger Bereich"
Spaltenoptionen	nicht entdeckt/gesucht
Zeilenoptionen	erst nach Hinweis gefunden
Zeile hinzufügen	Bug: kein Einfügen am Ende „einzelne Zeilen hinzufügen ist Mist"
Zeile kopieren und einfügen	Einfügen unintuitiv, weil anders als bei Zellen kopieren
Steuerung: Zeile auswählen	nicht genutzt
Steuerung: Vor/Zurück	erst im Interview probiert - leicht
Steuerung: Zurück zum Start	leicht, aber hätte „Stopp" erwartet: "ich muss das Ding manuell zurücksetzen"
Optionen: Geschwindigkeit	nicht genutzt

Hintergrund	
Beruf/Fachrichtung	Psychologie (Schwerpunkt Pädagogik)
Lehre/E-Learning	Lehre
Programmiererfahrung	keine
Office-Erfahrung	ja (nutzt auch Kalender-App)
Schnittsoftware-Erfahrung	k.A:

Anmerkungen

- Teilnehmer der Paper-Prototyping-Studie („alles vergessen")
- Szenario aus Paper-Prototyping wiederverwendet und vorbereitet. Besonderheit im Szenario: Kontextinformation „Nächster Termin" mit mehreren Parametern. Für jeden Parameter muss eigene Kontextinformation (Spalte) hinzugefügt werden
- Bugs im System, daher keine Adaptierung → Wechsel zu Demo-Szenario
- Technische Probleme: Nur 4 Minuten der Bildschirmaufzeichnung vorhanden (Ton vollständig).

Änderungen

keine

Aufgetretene Bugs / Probleme

siehe Test 4

Sonstige Bemerkungen / Auffälligkeiten

- „Ich bin jemand, der viel ausprobiert und rumspielt und selbstständig entdeckt"
- Anfang: Erkundung durch Mouseover.

Ergebnisse

Testfragen	**Beobachtungen und Kommentare**	**Interview und SUS**
Verständnis		
Wird die Zeitleiste als Abfolge von Situationen erkannt?	Ja. (s. Anm.)	
Werden die Kontextinformationen aus dem Szenario wiedererkannt?	Ja. (s. Anm.)	
Wird die Visualisierung richtig verstanden?	Ja. (s. Anm.)	
Wird klar, was getan werden soll und wie?	Ja. (Funktionsweise, Ablauf komplett erklärt.)	
Haben die Hilfstexte (Erklärungen hinter „?") einen Effekt auf das Verständnis? Welchen?	Nicht gelesen.	
Was ist unverständlich? Wo werden noch mehr Hilfen benötigt?	Problem mit parametrierten Kontextinformationen: neue Spalte anlegen als Lösung nicht offensichtlich; Hinweis auf Modell der Kontextinformationen und darauf, dass es in den Spalten noch Optionen gibt, hat geholfen.	• Problem mit Kontextinformation „Nächster Termin": „Parameter bei der Person [zu] sehen." • Einführung anhand von Beispielszenario, Schritt-für-Schritt-Wizard • Hilfen zu Grundlagen nötig (Workshop / Einführungsvideo)
Funktionalität		
Was sind typische Ziele und Abläufe? Was sind die wichtigsten bzw. am häufigsten benutzten Funktionen?	Standardablauf testen: Erwartete Kontextwerte in 1. Zeile angelegt -> Simulation gestartet.	Testen auf gröbste Fehler in Logik

Welche Use-Cases werden noch nicht unterstützt?	Szenario-Modell als Ausgang für Logik-Erstellung	Idee (für Anfänger): System leitet Nutzer Schritt für Schritt durch Modellierung des Szenarios in Zeitleiste: 1. Kontextwert, 2. Kontextwert usw. “Komplexität weiter rausgenommen”
Welche (wichtigen) Funktionen fehlen?	weitere, beliebige Kontextinformationen (Spalten) hinzufügen für o.g. Use-Case	• Auswahl im Szenario vorhandener Werte: Autovervollständigung/Auswahlliste • „Stopp-Bedingung”: Breakpoints + vor/zurück-Springen-Buttons (für komplexere Szenarien).
Welche Funktionen werden nicht benötigt?	-	
Bedienbarkeit		
Ist der Simulator einfach und intuitiv bedienbar?	siehe Tabelle „Bedienbarkeit"	siehe SUS
Wird der Simulator als konsistent, auch mit dem Autorensystem, empfunden?		siehe SUS
Welche Verbesserungen sind nötig oder gewünscht?		• Spalten-Optionen nicht offensichtlich: Plus, Punkte oder Pfeilchen in Spaltenkopf • Stopp- besser als Zurück-Button (von sich aus nicht benutzt) • Leere Zeilen am Ende, Autostopp zu spät. Simulation „könnte auch vorher abbrechen” • Panel-Größe flexibel • Mehr Unterstützung bei Eingaben.
Gesamteinschätzung		
Wurden Erwartungen getroffen? *Welche (anderen) Erwartungen gab es?*		Ja: keine konkreten Erwartungen, aber „begreifbar" und „nachvollziehbar"
Gibt das Testen mit dem Simulator ein gutes Gefühl?		Ja, um Fehlern in Programmlogik vorzubeugen. Aber trotzdem auf Handy testen.
Positives	Automatisches Hinzufügen von Spalten für neue Kontextinformationen im Szenario	• Insgesamt „gut und eingängig gemacht” • Punkte aus Paper-Prototyping umgesetzt, z.B. „runterziehen" • Mouseover „immer da" • „aufgeräumt, übersichtlich, nicht überladen, einfach gestaltet, aus meiner Sicht ansprechend” • Einfache Editier- und Manipulierbarkeit • Kurze Erklärung neben verschiedenen Optionen.
Was war am schwersten?	Neue Spalte für neuen Kontextparameter hinzufügen: erkennen und finden	• Fehler (Bugs) • „Grundlegende Kritik hab ich da jetzt eigentlich nicht.”
Kommentare zur SUS		
• Annahme, TP erstellt eine Lernanwendung mit dem Autorensystem • 3. „einfach zu nutzen - Es hat ein komplexes Aufgabenziel, deshalb würde ich zu 3 tendieren. Es ist per se nicht einfach zu nutzen. Wenn ich da an einen Laien denken, der würde da Schwierigkeiten haben einzusteigen. Dadurch dass die Hilfe nicht unabdingbar ist für das		

Verständnis, außer ich habe schon gutes technisches Vorwissen, gebe ich deshalb die 3. Aber das werdet ihr nicht großartig ändern können. Ich finde, das ist schon sehr gut gelöst. Der Punkt ist einfach der Komplexität geschuldet."

- 8. „Wir sind nur auf den Simulator beschränkt? Das musst du mir ja sagen, weil ich immer wieder aushole."
- 9. „Man kann sich am Anfang nicht super sicher fühlen weil es ist ja was komplett Neues gewesen und da probiert man erst mal aus. Ich war mir nicht sicher bei dem was ich tue, dass das auch das ist, was zielführend ist. Das gehört auch zu meiner Vorgehensstrategie [...]"

Funktion	Anmerkungen
Kontext editieren in Popover	mehr Unterstützung gewünscht: Datum/Zeit, Werte zur Auswahl
Quick-Access: Verschieben	entdeckt nach Hinweis, dass Reihenfolge geändert werden kann
Quick-Access: Länge ändern	gut gefallen
Spaltenoptionen	nicht erwartet, schwer gefunden
Spalte (Werte) löschen	erwartet „-" für Löschen ganzer Spalte
Spalte (Werte) nicht erfassen	nicht genutzt
Spalte hinzufügen	im Szenario zwangsweise benötigt, erst durch Hinweise vermutet („In den einzelnen Spalten hast du noch Funktionen")
Zeilenoptionen	nicht genutzt
Steuerung: Zurück zum Start	Nicht genutzt. Fände Stopp-Button besser
Optionen: Geschwindigkeit	nicht genutzt

Hintergrund	
Beruf/Fachrichtung	Bioinformatik
Lehre/E-Learning	Quereinstieg Lehramt, Moodle, Adaptives Lernen
Programmiererfahrung	viel, auch IDE (R, Python, kein Java, JS)
Office-Erfahrung	Ja, auch Kalender-App
Schnittsoftware-Erfahrung	Nein

Anmerkungen

- Besonderheiten des Szenarios:
  - Keine Icons für Kontextinformationen vorhanden. Verwendete Kontextinformationen haben dieselbe Klasse (Farbe). Kontextwerte heißen wie Lerneinheiten
  - Beispiel-Vorlage vorhanden.

Änderungen

- Icon für Testfalleigenschaften: Editieren statt Zahnrad
- Bugs in Parameterübergabe in Adaptation Engine behoben.

Aufgetretene Bugs / Probleme

Erste halbe Stunde ohne Internetzugang: Simulation funktionierte nicht wegen Interpreter.

Sonstige Bemerkungen / Auffälligkeiten

Ergebnisse

Testfragen	**Beobachtungen und Kommentare**	**Interview und SUS**
Verständnis		
Wird die Zeitleiste als Abfolge von Situationen erkannt?	Zeitfenster erkannt.	
Werden die Kontextinformationen aus dem Szenario wiedererkannt?	Ja, aber falsch zugeordnet.	
Wird die Visualisierung richtig verstanden?	Ja.	
Wird klar, was getan werden soll und wie?	Teils: Kontextinformationen bearbeiten und Starten.	Mehr erwartet.
Haben die Hilfstexte (Erklärungen hinter „?“) einen Effekt auf das Verständnis? Welchen?	• 1. Hilfe ignoriert/übersehen • 2. Hilfe gleich zu Beginn und später noch zweimal gelesen. Zeitfenster erkannt.	
Was ist unverständlich? Wo werden noch mehr Hilfen benötigt?	• Farben der Spalten nicht erkannt und nicht ausreichend • siehe Anmerkungen: Besonderheiten des Szenarios • Nicht klar, dass abgeschlossene Lerneinheiten angegeben werden müssen. (Zuerst ja, später nicht mehr.)	„Wozu muss ich denn den Wert kopieren?”
Funktionalität		
Was sind typische Ziele und Abläufe? Was sind die wichtigsten bzw. am häufigsten benutzten Funktionen?	• Testen, „ob alles funktioniert." • Fokus auf Auswahl bestimmter Lerneinheiten • Überblick über angelegte Werte	

	• Werte hinzufügen und bearbeiten • Play	
Welche Use-Cases werden noch nicht unterstützt?		• Erwartet, dass Inhalte angezeigt werden • Erwartet, Kontexterfassung (Interpretation) testen zu können.
Welche (wichtigen) Funktionen fehlen?		
Welche Funktionen werden nicht benötigt?	-	
Bedienbarkeit		
Ist der Simulator einfach und intuitiv bedienbar?		insgesamt als zu umständlich und nicht intuitiv empfunden
Wird der Simulator als konsistent, auch mit dem Autorensystem, empfunden?		
Welche Verbesserungen sind nötig oder gewünscht?	• Tooltips etwas „schwierig" • Stopp-Button • 1. Hilfe besser sichtbar machen: Erklärung, was und was nicht mit Simulator möglich ist.	• mehr Beschriftungen: wenigstens „Kontextinformationen" über Spalten • Werte in Feldern und Name der Kontextinformation über Spalte.
Gesamteinschätzung		
Wurden Erwartungen getroffen? Welche (anderen) Erwartungen gab es?		Nein, mehr erhofft.
Gibt das Testen mit dem Simulator ein gutes Gefühl?		
Positives	Wert löschen, verschieben, ausblenden klar	„Dass man's nicht jedes Mal aufs Handy laden muss, ums überhaupt zu testen"
Was war am schwersten?	Spalten erkennen.	Insgesamt zu unintuitiv für zu wenig Funktionalität.

Kommentare zur SUS

- 1. „Wenn ich das Autorensystem benutze, würde ich das Testsystem schon auch benutzen."
- 2. „Also so komplex fand ich's jetzt nicht"
- 3. „Wie definiert man denn einfach?" - Dass wenn du vorhast etwas zu tun, du es in wenigen Schritten umsetzen kannst (ohne Umstände) – „Also ich fand, es könnte noch einfacher sein. Ich fand's noch nicht so richtig intuitiv. Ich finde, für das was es kann… Also es macht ja eigentlich gar nicht so viel. Also für das Angebot, das ich bekomme, da fand ich's zu umständlich. Das müsste man irgendwie …. ja, intuitiver machen."
- 4. „Ob ich technischen Support brauche, hängt davon ab, wieviel Zeit ich investieren möchte, um es selber zu lernen, das System. Ich frag dich lieber, anstatt selber zu probieren" - Bist du denn der Typ, der bei Software gleich rumprobiert? – „Ja. Also eigentlich lese ich nie 'ne Anleitung"
- 6. konsistent: ja (dass passiert, was erwartet wurde mit der Funktionalität, die da ist)
- 7. „Andere, die schlauer sind als ich, lernen das bestimmt schneller"

Hintergrund	
Beruf/Fachrichtung	Informatik
Lehre/E-Learning	Hochschullehre, E-Learning, Adaptives Lernen
Programmiererfahrung	ja, Software-Entwicklungs-Tools (IDE)
Office-Erfahrung	ja, MS Office, E-Mail, Kalender
Schnittsoftware-Erfahrung	ja, Camtasia, Magix

Anmerkungen

- Framework und Autorensystem bekannt. Ideen für Simulator bekannt aus Vortrag
- Vorgefertigtes Beispiel-Szenario verwendet. Besonderheiten:
  - Verwendete Kontextinformationen haben dieselbe Klasse (Farbe)
  - Kontextwerte heißen wie Lerneinheiten
- Beispiel-Vorlage vorhanden

Änderungen

Keine.

Aufgetretene Bugs / Probleme

- Bug in Kontextfeld: doppeltes Klicken fügt zwei Events in ein Feld ein
- Bugs im Framework (erst am Ende der Studie entdeckt)

Sonstige Bemerkungen / Auffälligkeiten

„Zumal es ja eh zum Ausprobieren einladen soll."

Ergebnisse

Testfragen	**Beobachtungen und Kommentare**	**Interview und SUS**
Verständnis		
Wird die Zeitleiste als Abfolge von Situationen erkannt?	Ja. (s. Anm.)	
Werden die Kontextinformationen aus dem Szenario wiedererkannt?	Ja. (s. Anm.)	
Wird die Visualisierung richtig verstanden?	Nein.	
Wird klar, was getan werden soll und wie?	Ja. (s. Anm.)	
Haben die Hilfstexte (Erklärungen hinter „?") einen Effekt auf das Verständnis? Welchen?	Nicht gelesen.	
Was ist unverständlich? Wo werden noch mehr Hilfen benötigt?	• Unklar, ob Standort oder Lerneinheit aktiv (s. Anm.) • Nicht offensichtlich, dass noch nicht ausgewählte Lerneinheiten als abgeschlossen bestimmt werden können.	Unterscheidung zwischen aktivierten und abgeschlossenen Lerneinheiten unklar.
Funktionalität		
Was sind typische Ziele und Abläufe? Was sind die wichtigsten bzw. am häufigsten benutzten Funktionen?	• 1. Kompletter Ablauf modelliert, 2. Simulation • „Falsches" Modell (wg. unintuitiver Pfeilrichtungen) trotzdem testen.	
Welche Use-Cases werden noch nicht unterstützt?		Anzeige der Inhalte

Welche (wichtigen) Funktionen fehlen?	Implizit: mehr Feedback zu Auswahlverhalten (Matching): Schraffur an ausgeblendetem Feld nicht genug.	Idee: Icons an Lerneinheit während Simulation: Häkchen für abgeschlossen (konsistent mit 1. Spalte), Auge o.ä. für angesehen.
Welche Funktionen werden nicht benötigt?	-	
Bedienbarkeit		
Ist der Simulator einfach und intuitiv bedienbar?		
Wird der Simulator als konsistent, auch mit dem Autorensystem, empfunden?		
Welche Verbesserungen sind nötig oder gewünscht?	• Neu hinzugefügte Spalte bei neuer Kontextinformation in Regeln nicht gleich ersichtlich, wenn gescrollt werden muss. -> *evtl. Hinweis* nötig • Inkonsistenz in Optionen: kein Bestätigen wie in Einstellungen	Pfeil (von oben nach unten) statt Zeilennummern.
Gesamteinschätzung		
Wurden Erwartungen getroffen? *Welche (anderen) Erwartungen gab es?*		• Keine Erwartungen • „fehlt nix", „tut genau das, was es tun soll": effektiv simulieren
Gibt das Testen mit dem Simulator ein gutes Gefühl?	„Alles in allem scheint es zu funktionieren. Ich würd's also freigeben. Wir behaupten mal, ich komme zu Schloss Charlottenhof"	Ja. (Aber kein besseres Verständnis der Adaptierung, weil bereits verstanden.)
Positives	Zeilen- und Spaltenoptionen (demonstriert, nicht selbst gefunden).	• „Simulation mit der Zeitleiste" • Icons und Verschieben, bekannte Muster (Symbolik, Mauszeiger) • Vor/zurück-Steppen • Tooltips.
Was war am schwersten?		Bugs: Lerneinheiten nicht erreichbar

Kommentare zur SUS

- 1. zuerst 3 gewählt, „Da ich selten sowas anlege, eigentlich eher weniger"
- 2. „Unnötig komplex nicht. Es ist komplex."
- 3. „Einfach zu nutzen - das ist es nicht. Man muss sich schon ein bisschen damit vertraut machen: was Kontext ist und so weiter…" - Du meinst, wenn du das Vorwissen noch nicht hast? – „Genau. Es ist nur einfach zu nutzen, wenn du weißt, was du da tust, mit welchen Paradigmen du dich rumschlägst" - Angenommen, du wärst schon vertraut mit der Materie? - (ändert von 3 auf 2) „Muss mich trotzdem mit einigen Sachen vertraut machen: klarmachen, was ne Zeitachse ist… und so weiter"
- 4. „Eigentlich schon gut. Ich denke, dass man es so nutzen könnte"
- 7. „Aber das liegt nicht am System, das liegt an dem Thema. Also du musst Lehrender sein, du musst dich mit Kontext auskennen. Es kommen so viele Sachen zusammen"
- 8. „Die Bedienung ist nicht umständlich, das Thema ist umständlich"
- 10. „Das liegt nicht am System, das liegt am Paradigma."

Hintergrund	
Beruf/Fachrichtung	Informatik, Wirtschaftsinformatik (Student)
Lehre/E-Learning	nur Adaptives Lernen
Programmiererfahrung	ja, Software-Entwicklungs-Tools (IDE)
Office-Erfahrung	ja, Visio, Excel, E-Mail, OpenOffice - kein Kalender
Schnittsoftware-Erfahrung	ja, Windows Moviemaker

Anmerkungen

- Vorgefertigtes Beispiel-Szenario verwendet. Besonderheiten:
  - Verwendete Kontextinformationen haben dieselbe Klasse (Farbe), aber Icons vorhanden.
  - Kontextwerte heißen wie Lerneinheiten
  - Beispiel-Vorlage vorhanden
- Adaptives Lernen abgebrochen (Zeitgründe) - Idee für Lernszenario, aber nicht mehr umgesetzt.
- Keine Erfahrung mit Kontexterfassung
- Testablauf: Möglicherweise Situation falsch eingeschätzt, zu viele Hilfen zu früh gegeben. Besser gewesen wäre, auf Hilfen hinzuweisen.

Änderungen

Keine.

Aufgetretene Bugs / Probleme

- Zweimal Klick auf Feld: 2 Events in 1 Feld. (s. Test 7)
- Bugs im Framework (ContactJS)
- kleiner Bug in Popover: Markieren und Maus außerhalb loslassen schließt Popover.

Sonstige Bemerkungen / Auffälligkeiten

- Ohne Vorkenntnisse des Autorensystems „hätte ich sofort irgendwo einen Hilfe-Button gesucht."
- „Ich bin so einer, der liest sich lieber immer erst mal alles durch und dann wird noch probiert."
- Steuert Simulation bevorzugt durch Klick auf Zeilennummer.

Ergebnisse

Testfragen	**Beobachtungen und Kommentare**	**Interview und SUS**
Verständnis		
Wird die Zeitleiste als Abfolge von Situationen erkannt?	Ja.	
Werden die Kontextinformationen aus dem Szenario wiedererkannt?	Ja.	
Wird die Visualisierung richtig verstanden?	Ja.	
Wird klar, was getan werden soll und wie?	Nein.	Wozu die Tabelle?

Haben die Hilfstexte (Erklärungen hinter „?") einen Effekt auf das Verständnis? Welchen?		• Absichtlich ignoriert, weil wegen Erfahrung mit Autorensystem („Hier finden Sie Hilfe!") kein Inhalt erwartet
		• Hätten geholfen.
		  ○ Hilfe zum Simulator: 1. Satz „hatte ich mir schon gedacht", 2. Satz „erklärt schon son bisschen die Funktionsweise."
		  ○ 2. Hilfe zur Zeitleiste: „Achso? Ja, doch, das hilft."

Was ist unverständlich? Wo werden noch mehr Hilfen benötigt?	• Einstieg: Was tun? Wie beginnen? • Hinweis: Dem Simulator vorgeben, was passieren soll (1. Hilfe). • Farben der Spalten: blau verstanden, rot unklar (s. Anm.: Wert = Lerneinheit)	• Tabelle nicht selbsterklärend: Kontext modellieren nicht ersichtlich. • Erklärung zu Tabelle und Highlighting benötigt.
Funktionalität		
Was sind typische Ziele und Abläufe? Was sind die wichtigsten bzw. am häufigsten benutzten Funktionen?	• 1. Standardszenario, 2. Fehlverhalten des Users testen • Simulieren: Start, Schritt auswählen • Kontextwert eintragen • Simulieren	(Normalerweise immer zuerst Hilfe suchen.)
Welche Use-Cases werden noch nicht unterstützt?		Inhaltsvorschau
Welche (wichtigen) Funktionen fehlen?	• Bereits absolvierte Lerneinheiten aus Auswahlliste herausfiltern • Mehr Unterstützung bei Eingaben	• Feedback zu nicht-erfüllten Bedingungen: Fenster, Infobox oder rotes Highlighting der Relationen im Graph • Einführung (Wizard besser als Video).
Welche Funktionen werden nicht benötigt?	-	
Bedienbarkeit		
Ist der Simulator einfach und intuitiv bedienbar?	siehe Tabelle „Bedienbarkeit"	siehe SUS
Wird der Simulator als konsistent, auch mit dem Autorensystem, empfunden?		siehe SUS
Welche Verbesserungen sind nötig oder gewünscht?	Geschwindigkeit zu schnell, Optionen-Button nicht wahrgenommen.	• Testfalleigenschaften (Name, Beschreibung) direkt bei Vorlage im oberen Bereich (Vorschlag: kurz aufschieben). Edit-Button nicht wahrgenommen, „+" vermutet (wie für Lerneinheiten) • Namen der Kontextinformationen statt Symbolen • Buttons und Drop-down-Menü besser sichtbar machen (Problem mit Flat Design des Autorensystems)
Gesamteinschätzung		
Wurden Erwartungen getroffen? *Welche (anderen) Erwartungen gab es?*		• Simulator nicht als solchen erkannt, Emulator-Fenster erwartet • „Erwartet, dass der Simulator ein bisschen das Denken abnimmt." • Drag-and-drop von Kontextinformationen-Bausteinen in Zeitleiste (wie bei Visio).
Gibt das Testen mit dem Simulator ein gutes Gefühl?		Wenn es funktioniert.

Test 8 (23.03.2016)

Positives	Tooltip 1. Spalte: Auflistung aller abgeschlossenen Lerneinheiten	• Testmöglichkeit an sich • Möglichkeit, auch Fehlerfälle zu testen • hilfreiche Hilfen (Erklärungen) • Ablauf + Graph
Was war am schwersten?		• Einstieg: Was wie tun? (ohne Hilfen) • Simulation (wegen Bugs) • Insgesamt zu wenig Feedback.
Kommentare zur SUS		
• Anmerkungen vorne weg: bezogen nur auf Simulator und Voraussetzung, dass mit Autorensystem gearbeitet werden würde • 1. „WENN ich so was machen würde" • 2. „unnötig komplex": unübersichtlich; „an manchen Stellen ja" • 3. „einfach zu nutzen": „Noch nicht. Mit den ganzen Sachen, die wir angesprochen haben, dann ja." • 4. Technischer Support: „Spätestens wenn ein Wizard drin ist, dann geht's ja auch so…. Wenn ich mir sehr viel Zeit nehme und die Hilfen lese, dann brauch ich auch den Support nicht. Ich brauch bloß die Zeit und dann schaff ich's auch irgendwann… irgendwann" (Annahme: ohne Bugs; „mit technischer Support Mitarbeiter gemeint") • 5. „gut integriert" – „da hatte ich teilweise meine Probleme, wo was ist; hätte ich mir woanders offensichtlicher vorstellen können" • 7. „die meisten Leute": Jetziger Standpunkt – „durch das 'schnell' würde ich sagen: nee" • 9. „also sicher hab ich mich garantiert nicht gefühlt" • 10. „Da hat manchmal nicht ein Durchlauf gereicht, ums zu verstehen"		

Funktion	Anmerkungen
Vorlagen: Auswahl + Neue erstellen	Problem mit Flat Design: erst für Label gehalten; bei vielen Testfällen langes Scrollen nötig
Vorlagen: Eigenschaften (Name, Beschreibung)	gesucht, aber Knopf zu unauffällig, nicht wahrgenommen - Plus vermutet
Vorlagen: Löschen, Kopie erstellen	nicht genutzt
Kontext hinzufügen: 1 Feld	Bug: 2. Klick fügt 2. Event in dasselbe Feld ein
Kontext hinzufügen: > 1 Feld	Langziehen gesucht, bei Anlegen nicht
Kontext editieren in Popover	kleiner Bug (unerwartetes Schließen, danach behoben)
Quick-Access: Tooltip	Auflistung in 1. Spalte für gut befunden
Quick-Access-Optionen (Icons)	
Quick-Access: Löschen	(zu) leicht - versehentlich Eintrag gelöscht
Quick-Access: Kopieren	nicht genutzt
Quick-Access: Ausblenden	nicht genutzt
Quick-Access: Verschieben	leicht - Cursor wahrgenommen
Quick-Access: Länge ändern	gesucht, aber erst nach Lesen der Hilfe gefunden - nur Verschieben-Multipfeil gesehen
Spaltenoptionen	nicht genutzt
Zeilenoptionen	gefunden, aber nicht genutzt
Steuerung: Zeile auswählen	am meisten damit gearbeitet
Steuerung: Vor/Zurück	nicht genutzt
Optionen: Geschwindigkeit	danach gesucht, aber Knopf zu unauffällig, nicht wahrgenommen

Hintergrund		
Beruf/Fachrichtung	Informatik (Studentin)	
Lehre/E-Learning	nur Adaptives Lernen	
Programmiererfahrung	ja, Software-Entwicklungs-Tools (IDE), wenig JS	
Office-Erfahrung	ja, viel (u.a. beruflich)	
Schnittsoftware-Erfahrung	kaum (wenig Windows Moviemaker)	
Anmerkungen		
• Szenario aus Test 4 wiederverwendet (Icons fehlen) - nicht aktuelle Version, d.h. Fall: fremde App testen.		
Änderungen		
"Bestätigen"-Button in Optionen hinzugefügt (konsistent mit Eigenschaften)		
Aufgetretene Bugs / Probleme		
• Bug im Framework - der Testperson bekannt aus Testen der App: „Spiegelt total die Realität wider." • Leicht*:* Zeilen am Tabellenende einfügen funktioniert nicht • >1 Feld eintragen (klicken und ziehen) öffnet Popover nicht.		
Sonstige Bemerkungen / Auffälligkeiten		
Funktionsweise schnell verstanden durch Ausprobieren.		
Ergebnisse		
Testfragen	**Beobachtungen und Kommentare**	**Interview und SUS**
Verständnis		
Wird die Zeitleiste als Abfolge von Situationen erkannt?	Ja. („Situationen" werden „Szenarios" genannt, meint aber das Gleiche.)	
Werden die Kontextinformationen aus dem Szenario wiedererkannt?	Ja.	
Wird die Visualisierung richtig verstanden?	Ja.	
Wird klar, was getan werden soll und wie?	Ja.	
Haben die Hilfstexte (Erklärungen hinter „?“) einen Effekt auf das Verständnis? Welchen?	Für Verständnis nicht benötigt, evtl. für mehr Sicherheit.	Nach Test: Option >1 Feld eintragen entdeckt durch Tipp.
Was ist unverständlich? Wo werden noch mehr Hilfen benötigt?	Am Anfang nicht ersichtlich, dass und warum nichts ausgewählt wird.	Hilfe zum Einstieg: Wizard („Damit wär ich noch schneller gewesen")
Funktionalität		
Was sind typische Ziele und Abläufe? Was sind die wichtigsten bzw. am häufigsten benutzten Funktionen?	• Bestimmte Lerneinheiten nacheinander testen • Kontextregeln in Lerneinheit ansehen • Mouseover - Tooltips lesen • Situation von links nach rechts einstellen • Simulation (1. Mal ganzer Durchlauf, danach Pause + Zeilenauswahl).	
Welche Use-Cases werden noch nicht unterstützt?		• Darstellung der Inhalte prüfen • Testfälle (Ergebnisse) vergleichen.

Welche (wichtigen) Funktionen fehlen?	Feedback zu Nicht-Auswahl (Matching)	Inhalte bei Klick auf Lerneinheit in Emulator Feedback bei Nicht-Auswahl.
Welche Funktionen werden nicht benötigt?	-	(siehe Kommentar zur SUS)
Bedienbarkeit		
Ist der Simulator einfach und intuitiv bedienbar?	Auch Ausblenden und Tipp (>1 anlegen durch drag-and-drop) richtig verstanden	s.u.: Positives
Wird der Simulator als konsistent, auch mit dem Autorensystem, empfunden?		
Welche Verbesserungen sind nötig oder gewünscht?	Hilfe zum Simulator nicht leicht wahrzunehmen. („Szenario-Zeitleiste" zieht Aufmerksamkeit auf sich)	• Leere Zeilen nicht durchlaufen, stattdessen abbrechen und Bericht ausgeben • Panel skaliert vergrößern: Lerneinheiten sollen sichtbar bleiben • Zeilenoptionen besser sichtbar machen, "+" am unteren Ende vermutet • Tooltips besser sichtbar machen: „…" statt Icons und Zeilen höher • Hilfe an Kontrollleiste (Mouseover zu langsam).
Gesamteinschätzung		
Wurden Erwartungen getroffen? *Welche (anderen) Erwartungen gab es?*		Tabelle überrascht; hätte Emulator erwartet.
Gibt das Testen mit dem Simulator ein gutes Gefühl?	Ja, funktioniert wie gedacht und bekannt.	
Positives		• Simulationsmöglichkeit an sich • Zusammenspiel links-rechts (passendes Aufleuchten → „direkter Vergleich") • "einrichten total easy": einfügen, bearbeiten, ... • Zeilenoptionen („wenn man weiß, dass sie da sind") • Mehrere Situationen statt nur einer • Icons intuitiv • Zurückgehen („sinnvoll") • Neue Vorlage + Bearbeiten intuitiv • Kopie öffnen („Das finde ich auch schön. Dann kann ich nur minimale Sachen noch bearbeiten").
Was war am schwersten?		Der Einstieg: unsicher, ob Annahmen richtig sind.

Kommentare zur SUS

- 1. „wenn ich's brauche, ja" - Ausgehend von der Annahme, dass du dann auch mit dem Autorensystem arbeitest. – „ja"
- 2. „ist doch nicht komplex. Du hast ja nur die Funktionen, die man wirklich braucht, drin."
- 3. „'einfach zu nutzen' - wenn man reingekommen ist. Also Startschwierigkeiten hatte ich ein bisschen, aber wenn man dann drin ist, ist es total easy."

- 4. „nö”
- 5. „‘...gut integriert sind’ - Jetzt in dem Autorensystem mit integriert, oder wie?” - Ja.
- 6. „Hast du Inkonsistenzen drin? Inkonsistenz ist ja, wenn man mit gleichen Sachen verschiedene Ergebnisse erzielt, oder so… mit gleichen Handlungen… Nein. Nein, nein, nein, keine Inkonsistenzen”
- 7. „Das Beherrschenlernen geht fix. Wenn du dich hinsetzt und das auch vorher jemandem erk- du hast uns ja jetzt als Testlauf reinrennen lassen - wenn du aber jemandem sagst: Du machst das hier, du machst das hier, du machst das da - viel Spaß”
- 8. „‘sehr sicher gefühlt’ - am Anfang nicht, am Anfang war ich sehr nervös, weil ich dachte, ich raff’s nicht. Aber dann hab ich - jetzt fühl ich mich sicher. Jetzt könnt ich’s auch wem anders erklären.”
- 9. „Nein. Musste ich nicht. So.“(wählt 5) - Auch nicht zu der Gesamtthematik (Kontext usw.)? – „Ach so, meinst du, im Allgemeinen? Kontextinformationen… OK, vielleicht musste ich ein paar Dinge lernen.” (wählt 4) „Ja, ich hatte den Kurs vorher. Der Kurs war das Lernen dazu. Da warn ein paar Vorträge...” - Aber wenn du schon so weit bist, dass du mit dem Autorensystem arbeitest? – „Dann braucht man keine Vorkenntnisse mehr, glaub ich nicht. Also ich stell‘s mir schwierig vor, wenn jemand komplett neu im Autorensystem ist und dann den Simulator auch noch gleich dazu kriegt, in der Form jetzt wie du sagst: ‘mach mal!’ - das stell ich mir schon schwierig vor. Aber auch einfach weil das Autorensystem an sich nicht immer 100% userfreundlich ist. Es ist halt noch early access, ne? Aber das ist ja normal... Ja, ich musste was lernen, den Kurs. Das war schon gut als Vorwissen. Aber viel würde ich jetzt nicht sagen.”

Hintergrund	
Beruf/Fachrichtung	Lehramt Informatik, Russisch
Lehre/E-Learning	beides; viel E-Learning (auch Contenterstellung), Adaptives Lernen
Programmiererfahrung	ja, aber wenig und selten, IDEs verwendet
Office-Erfahrung	sehr viel, auch Google Calendar
Schnittsoftware-Erfahrung	ja, mehrere für Video-Schnitt

Anmerkungen

- Adaptives Lernen abgebrochen (Zeitgründe) - Idee für Lernszenario, aber nicht mehr umgesetzt.
- Keine Erfahrung mit Kontexterfassung
- Wegen Bugs im Framework keine Adaptierung -> Unterbrechung und Wechsel zu Beispiel-Szenario aus Test 7 und 8.

Änderungen

- leichte Bugs (siehe Test 9) behoben
- ContactJS auf neueste Version aktualisiert (verarbeitet „CONTAINS"-Operator)
- Tooltips „stabilisiert": werden bei Mouseover über Icons nicht mehr ausgeblendet, sondern bleiben angezeigt
- Zeilen reduziert: von 20 auf 10
- Benachrichtigung in Simulation wird angezeigt, wenn Nutzer Pause drückt und keine Auswahl erfolgt ist
- Hintergrund wird bei Start der Simulation gedimmt, unabhängig von Auswahl - so wie es auch ursprünglich gedacht war.

Aufgetretene Bugs / Probleme

schwer: Bugs im Framework (ContactJS) - danach erstmals entdeckt.

Sonstige Bemerkungen / Auffälligkeiten

- Typ 2Learning by Doing". Alles durch Ausprobieren (Tooltips und Klicken) erkundet, keine Hilfe gelesen. Aber: „Wenn ich zu Hause wäre, alleine, ohne irgendwas, dann hätte ich die auf jeden Fall angeklickt."
- Nach Entdecken der Zeilenoptionen: „Ist wie bei Excel"

Ergebnisse

Testfragen	**Beobachtungen und Kommentare**	**Interview und SUS**
Verständnis		
Wird die Zeitleiste als Abfolge von Situationen erkannt?	Ja, aber anfangs Schritte als Zeiteinheiten (Stunden oder Tage) und Zeitleiste als Log.	
Werden die Kontextinformationen aus dem Szenario wiedererkannt?	Ja.	
Wird die Visualisierung richtig verstanden?	Ja.	
Wird klar, was getan werden soll und wie?	Nicht sofort. Hatte andere Vorstellung (s.u.)	„Ich wusste ungefähr, was es ist und dann ging's schon los mit dem Klicken."
Haben die Hilfstexte (Erklärungen hinter „?") einen Effekt auf das Verständnis? Welchen?	Nicht gelesen und nicht benötigt.	Nach Test, liest Tipp: „Also hätte ich vielleicht auch lesen sollen."

Was ist unverständlich? Wo werden noch mehr Hilfen benötigt?		„am Anfang irritierend, weil ich hier dieses Häkchen hab [1. Spalte oben], erst mal dachte, dass das ne Zeit ist [Zeitleiste -> Einheiten] und ich hab immer noch nicht herausgefunden, ob die hier [Zeile] nebeneinander stehen dürfen oder nicht" (s. Anm.: keine Adaptierung wegen Bugs, vermutlich deshalb falsche Annahme)
Funktionalität		
Was sind typische Ziele und Abläufe? Was sind die wichtigsten bzw. am häufigsten benutzten Funktionen?	Erste Aktionen: Tooltips lesen. Play. Kontext an Lerneinheit ändern. (s.u.)	
Welche Use-Cases werden noch nicht unterstützt?		
Welche (wichtigen) Funktionen fehlen?	Feedback zu Auswahl (Begründung: Matching)	„Ist eigentlich alles, was man braucht, schon drin, würd' ich sagen."
Welche Funktionen werden nicht benötigt?	-	
Bedienbarkeit		
Ist der Simulator einfach und intuitiv bedienbar?		s.u.: Positives
Wird der Simulator als konsistent, auch mit dem Autorensystem, empfunden?		
Welche Verbesserungen sind nötig oder gewünscht?	• Einleitung zu Beginn zeigen, da Hilfen nicht gelesen wurden, aber geholfen hätten • evtl. Hilfe zu Ausblenden: „Vielleicht nicht abspeichern oder so in der App?" • Hinweis auf Verschieben: erst später entdeckt, vorher kopieren + löschen	Stopp-Button, um nicht alle leere Zeilen zu durchlaufen.
Gesamteinschätzung		
Wurden Erwartungen getroffen? Welche (anderen) Erwartungen gab es?	• Emulator mit „live" Kontexterfassung erwartet • "Ich dachte nicht, dass man reintippen darf. Ich dachte, das ist schon alles vorgegeben und festgelegt."	„Ich hatte nicht so große Erwartungen. Ich konnte mir nicht viel drunter vorstellen."
Gibt das Testen mit dem Simulator ein gutes Gefühl?		„Wär' ich natürlich viel sicherer." Würde trotzdem App testen. Simulator „für die ersten Schritte"
Positives		• Highlighting ästhetisch (Umrandung) • Passendes Aufleuchten zu Situationen • Tooltips an Zeitfenstern + Klicken („gefällt mir") • Tastenkombination [Abspielen] „jedem bekannt". • Zurück-Button mit Häuschen. Hilfen vorhanden

		• Tab in Tab-Bar „total klar”.
Was war am schwersten?	• Funktionsweise erkennen (ohne Hilfe) • Fehlverhalten wegen Bugs.	„Nö, nix."
Kommentare zur SUS		
• 1. „Später” • 2. „so komplex ist es gar nicht” • 4. „Ja, wenn ich alleine wäre” • 5. „Inkonsistenzen” unklar • 8. wählt 4 – „Also bloß von dem Simulator? Da muss man ja nur auf Play und Bearbeiten und … nee.” Wählt 5. „Das ist so ein gutes Ding” • 10. „Nee, ist ja nur klicken. Was soll ich denn lernen? Klicken lernen?” - Frage nach Hintergrund lernen – „Wenn ich dieses Seminar nicht hätte, dann wär‘ das für mich ne ganz andere Sache. Aber dadurch, dass ich das kenne, dass ich ja Szenario und Lerneinheit und die ganzen Sachen hatte, brauchte ich eigentlich nicht viel Neues lernen, weil ich wusste, um was es geht”		

Hintergrund	
Beruf/Fachrichtung	Psychologie, Informatik (Studentin)
Lehre/E-Learning	Adaptives Lernen, Lehrerfahrung
Programmiererfahrung	ja, auch mehrere IDEs
Office-Erfahrung	ja, auch Google Calendar
Schnittsoftware-Erfahrung	ja, Moviemaker, Adobe Premiere

Anmerkungen

- 1. Teilnehmerin der Paper-Prototyping-Studie. (größte Veränderung ggü. Paper-Prototype)
- Besonderheiten des Szenarios:
  - Keine Icons (und keine Übersetzungen der Kontextnamen) vorhanden.
  - Nur Kontextinformationen einer Klasse = eine Farbe.
- Beispiel-Vorlage vorhanden
- Nur 5 Minuten der Audioaufnahme erhalten (Fehler beim Verschieben).

Änderungen

Aufgetretene Bugs / Probleme

- Tooltip verschwindet manchmal nicht
- Pausierte Simulation nicht gestoppt bei Öffnen von neuem Testfall
- Bug bei Ändern von Testfalleigenschaften: Speichern/Zurück überschreibt Eigenschaften mit denen der zuletzt geöffneten.

Sonstige Bemerkungen / Auffälligkeiten

Einladen zum Spielen „entspricht meinem Nutzertyp" → „klicken ist immer gut"

Ergebnisse

Testfragen	**Beobachtungen und Kommentare**	**Interview und SUS**
Verständnis		
Wird die Zeitleiste als Abfolge von Situationen erkannt?	Ja. (s. Anm.)	
Werden die Kontextinformationen aus dem Szenario wiedererkannt?	Nicht gleich (s. Anm.)	
Wird die Visualisierung richtig verstanden?	Teils. (s. Hilfen)	
Wird klar, was getan werden soll und wie?	Ja (s. Anm.)	
Haben die Hilfstexte (Erklärungen hinter „?") einen Effekt auf das Verständnis? Welchen?	• 2. Hilfe zu Beginn gelesen: „Die Tipps erscheinen mir sinnvoll" • 1. Hilfe („?") gesehen, aber nicht gelesen.	
Was ist unverständlich? Wo werden noch mehr Hilfen benötigt?	• Liste der abgeschlossenen Lerneinheiten in Tooltip erst spät entdeckt → danach Annahme, dass alle Werte immer gelten bis sie geändert werden • Spalten nicht gleich klar (ohne Icons).	• Hilfen zu zwei Farben in Visualisierung (Vermutung: Reihenfolge pro Situation), besser: nur eine Farbe • Erklärung zu Spalten: „Achsenbeschriftung" fehlt (Icons würden reichen, wenn vorhanden).
Funktionalität		
Was sind typische Ziele und Abläufe? Was sind die wichtigsten bzw. am		

häufigsten benutzten Funktionen?		
Welche Use-Cases werden noch nicht unterstützt?		Keine.
Welche (wichtigen) Funktionen fehlen?		Keine.
Welche Funktionen werden nicht benötigt?	-	
Bedienbarkeit		
Ist der Simulator einfach und intuitiv bedienbar?		siehe SUS
Wird der Simulator als konsistent, auch mit dem Autorensystem, empfunden?		siehe SUS
Welche Verbesserungen sind nötig oder gewünscht?	• Zeilenoptionen: Einfügen nach Kopieren nicht intuitiv • 1. Hilfe besser sichtbar machen. Evtl. wegen Nähe zu „Wählen Sie einen Testfall" ignoriert.	• Änderungen der Werte sichtbar machen: Wert-Vorschau (Schrift statt Icons) • „Scrollen nervt" - Impuls zum Aufziehen des Panels • Stopp-Button • Andere Icons statt Auge (Nicht erfassen) und Zahnrad (Geschwindigkeit) • Modi bei Zeilen: Optionen sollten immer verfügbar sein.
Gesamteinschätzung		
Wurden Erwartungen getroffen? *Welche (anderen) Erwartungen gab es?*		• Erwartungen getroffen (auch wegen Paper-Prototyping) • Keine Erwartungen/Wünsche darüber hinaus (auch nicht von anderen Emulatoren)
Gibt das Testen mit dem Simulator ein gutes Gefühl?		(s. Anm.)
Positives		• Konstellationen (Abfolge) pro Szenario. • Benötigte Kontextinformationen schon da • „Sinnvolle" Funktionen (schnell ändern, kopieren, löschen, ausblenden) • Vor-/Zurückspringen. • „sehr durchdacht" • Lädt zum Spielen ein „entspricht meinem Nutzertyp" • Zeitleiste-Hilfe: Tipps sinnvoll.
Was war am schwersten?		Erinnern der Kontextwerte (Überblick fehlt)
Kommentare zur SUS		
Nicht kommentiert.		

Hintergrund	
Beruf/Fachrichtung	Romanistik
Lehre/E-Learning	Hochschullehre, viel E-Learning (Aufzeichnungen, Filme, Adobe Connect)
Programmiererfahrung	keine
Office-Erfahrung	ja (Apple-Produkte)
Schnittsoftware-Erfahrung	ja, iMovie, Camtasia („hässlich")

Anmerkungen

- Einführung in Thema + Autorensystem gegeben, findet Projekt spannend
- Kein vorbereitetes, sondern eigenes Szenario ausgedacht, in Autorensystem angelegt und getestet
- Besonderheit: Kontextinformation „Momentane Lerneinheit" benutzt (Hinweis evtl. missverstanden) -> Dummy ohne Umsetzung -> keine Adaptierung.

Änderungen

Keine.

Aufgetretene Bugs / Probleme

schwer: Bug im Framework

Sonstige Bemerkungen / Auffälligkeiten

Versucht als Erstes, Lerneinheit in die Tabelle zu ziehen → „das sind hier zwei verschiedene Ebenen." (s. Kommentar von TP 3)

Ergebnisse

Testfragen	**Beobachtungen und Kommentare**	**Interview und SUS**
Verständnis		
Wird die Zeitleiste als Abfolge von Situationen erkannt?	(Ja.) Vorstellung für jeden Schritt nacheinander kommentiert.	
Werden die Kontextinformationen aus dem Szenario wiedererkannt?	Ja, auch abgeschlossene Lerneinheiten: „also ich hab diese Pfeile auch hier gegeben"	
Wird die Visualisierung richtig verstanden?	Ja, versteht Highlighting als von der Anwendung ausgewählt und dem Studenten zum Lernen vorgeschlagen.	
Wird klar, was getan werden soll und wie?	• Erster Eindruck: „wüsste gar nicht, was ich machen muss." • Nächster Schritt nach Anlegen von Testfall unklar. („vielleicht Play")	
Haben die Hilfstexte (Erklärungen hinter „?") einen Effekt auf das Verständnis? Welchen?	Hilfen erst am Ende gelesen (2. zuerst): „Da steht 'mit den gültigen Kontextinformationen' - wenn dann diese Regeln hier übereinstimmen, [...] nur dann kannst du das machen. Alles andere steht nicht zur Wahl"	

Was ist unverständlich? Wo werden noch mehr Hilfen benötigt?	• Unklar, ob Werte nochmal definiert werden müssen. (s. Erwartungen u. fehlende Funktionen) • Versucht, Lerneinheit in Tabelle zu ziehen → „das sind hier zwei verschiedene Ebenen" (s. Test 3) • „Zeitleiste" nicht klar • Versteht nicht, warum Lerneinheiten ohne Kontext ausgewählt werden, andere mit Kontext nicht.	
Funktionalität		
Was sind typische Ziele und Abläufe? Was sind die wichtigsten bzw. am häufigsten benutzten Funktionen?	• Legt nacheinander Situationen für alle Lerneinheiten an • Hat neue Idee: legt Lerneinheit an und Situation im Testfall • Play (Hinweis nachdem nächster Schritt unklar)	
Welche Use-Cases werden noch nicht unterstützt?		Inhaltsanzeige
Welche (wichtigen) Funktionen fehlen?	Besserer Überblick über Kontextwerte. *Matching = Begründung der (Nicht-) Auswahl* (s. Hilfen und Test 3)	Szenario wie gedacht im Simulator abbilden: Informationen automatisch aus Lerneinheiten laden
Welche Funktionen werden nicht benötigt?	-	
Bedienbarkeit		
Ist der Simulator einfach und intuitiv bedienbar?		
Wird der Simulator als konsistent, auch mit dem Autorensystem, empfunden?		
Welche Verbesserungen sind nötig oder gewünscht?	• Besserer Überblick über Kontextwerte • Scrollen -> neu eingefügte Kontextspalte schwer erkennbar, möchte Canvas verkleinern.	
Gesamteinschätzung		
Wurden Erwartungen getroffen? *Welche (anderen) Erwartungen gab es?*		• Schwer zu beantworten: „Ich weiß gar nicht, wie das funktioniert und was das bedeutet. Es erscheint mir so viel Fummelei und so viel Information, von der ich nicht weiß, wofür." • Sollte alles einfacher sein.
Gibt das Testen mit dem Simulator ein gutes Gefühl?		Nach Testen (Adaptierung nicht wie erwartet): Könnte sich gut vorstellen, die App an die Studenten zu geben. (*evtl. auch Frage missverstanden).*
Positives	Zu Hinzufügen zu >1 Zeitfenstern: „ah, sehr gut"	(Autorensystem: Anlegen des Lerneinheiten-Graphs)

Was war am schwersten?		• „doppelt gemoppelt… bin ich ständig nur am Klicken - zu viel Aufwand für etwas, was schon am Anfang gewusst ist” • „Dann musst du hier noch mal alle Werte organisieren. Das lohnt sich nicht.”

Kommentare zur SUS

- 1. „Also wenn das so ist, dass ich da 5 Stunden sitze… Also ich hab die Zeit nicht”
- 2. „Ja, das schon. Also diese ganzen Werte, die man doppelt angibt”
- 3. „Das nicht unbedingt” (findet Anlegen der Lerneinheiten, Relationen schön)
- 4. „Ja, stimmt.”
- 5. „Das weiß ich nicht.”
- 6. „Das kann ich nicht beurteilen.”
- 7. „Das weiß ich nicht.”
- 8. „Ähm…. ja. Sag ich mal so.”
- 9. „Nein.”
- 10. (Kein Kommentar.)

Funktion	Anmerkungen
Kontext hinzufügen: 1 Feld	Klickt mehrmals auf Spaltenkopf zum Hinzufügen.
Kontext hinzufügen: > 1 Feld	2ah, sehr gut"
Quick-Access: Tooltip	evtl. schlecht wahrgenommen („weil ich weiß jetzt nicht, was das ist”)
Quick-Access-Optionen (Icons)	
Quick-Access: Bearbeiten	spät wahrgenommen/erkannt: „Kann man Sachen verändern nachher?”
Quick-Access: Kopieren	automatisches Duplizieren (Einfügen) erwartet; Probleme, Icon zu treffen
Quick-Access: Ausblenden	nicht genutzt
Quick-Access: Verschieben	leicht
Quick-Access: Länge ändern	zuerst in 1. Spalte versucht; nach Demo: „muss man treffen”
Spaltenoptionen	
Spalte (Werte) löschen	ganze Spalte löschen erwartet

Hintergrund	
Beruf/Fachrichtung	Informatik
Lehre/E-Learning	Lehre, viel E-Learning
Programmiererfahrung	ja, keine IDEs
Office-Erfahrung	ja, viel (MS)
Schnittsoftware-Erfahrung	ja

Anmerkungen

- Interview aus zeitl. Gründen nicht vollständig durchgeführt. (Feedback und SUS, aber keine Fragen.)
- Projekt und Framework bekannt. Idee für Zeitleiste möglicherweise aus Vortrag, Abbildungen (Screenshots) aus Paper bekannt (Beschreibungen lt. eigener Aussage nicht).
- Kein vorbereitetes, sondern eigenes Szenario ausgedacht, in Autorensystem angelegt und getestet.

Änderungen

- Hilfe zum Simulator wird automatisch beim 1. Start des Simulators angezeigt, „?" etwas größer dargestellt (nach dem Test wieder zurückgesetzt) und ein Stück weiter weg vom Rand für bessere Wahrnehmung
- Spaltenbeschreibung und Hinweis auf Spalten-/Zeilenoptionen in Hilfe zur Zeitleiste hinzugefügt. (Wurde im Test aber nicht angesehen.)
- Highlighting der Auswahlhistorie entfernt: wieder nur noch einfarbiges Highlighting der Auswahl. (Bericht = Liste am Ende unangerührt.)
- Modi für Klick auf Zeilennummern (in Simulation Auswahl, sonst Zeilenoptionen) wurden entfernt: Klick auf Nummer öffnet immer nur Optionen-Popover
- Bereich zum Verlängern (unterer Rand) vergrößert
- Bugs aus Test 11 (Überschreiben von Testfalleigenschaften) behoben
- Automatisches Speichern des Gesamtzustands (Autorensysteminhalt) bei Verlassen/Neuladen der Seite hinzugefügt.

Aufgetretene Bugs / Probleme

- Bug in ContactJS. Reihenfolge der abgeschlossenen Lerneinheiten wird dadurch relevant (sollte eigentlich nicht) -> wurde danach (behelfsmäßig) angepasst
- Erstmals Bug in A-Box-Generierung entdeckt: ließ Auswahl einer Lerneinheit mit 2 PRE zu, wenn nur 1 erfüllt war. („Ich find das [links: Lerneinheiten-Graph] logisch, ich find das [rechts: Zeitleiste] logisch, aber wie es zusammen spielt, finde ich nicht logisch.") Wurde danach behoben
- Anzeige-Bug in Tooltip in 1. Spalte: Reihenfolge durcheinander -> danach behoben
- Leicht: Bug bei Verlängern: geht manchmal nur bei 2. Versuch.

Sonstige Bemerkungen / Auffälligkeiten

- „Bei Excel gibt's ja so was" (Ecke unten rechts zum Verlängern) --> s. Test 10
- „Ich mach immer mit der rechten Maustaste" --> s. Test 3
- Möchte „interaktiv den Verlauf der Simulation steuern" durch Anklicken der Lerneinheiten. --> s. Tests 3, 4, 7
- „Also jetzt dominieren die Icons, also die Aktivitäten […]" --> *Wahl zwischen Anzeige wäre sinnvoll: Werte oder Icons*

Ergebnisse

Testfragen	**Beobachtungen und Kommentare**	**Interview und SUS**
Verständnis		
Wird die Zeitleiste als Abfolge von Situationen erkannt?	Ja. (Dreidimensionalität erwartet, s.u. & Anm.)	
Werden die Kontextinformationen aus dem Szenario wiedererkannt?	Ja, erklärt Verständnis: erste Spalte aus Lerneinheiten, Rest aus Kontext generiert	
Wird die Visualisierung richtig verstanden?	Nein. (s.u.: Hilfen)	

Wird klar, was getan werden soll und wie?	Ja. Erwartet, Kontextwerte einstellen zu können und „hier unten durchspielen".	
Haben die Hilfstexte (Erklärungen hinter „?“) einen Effekt auf das Verständnis? Welchen?	• Effekte unklar: 1. Hilfe zu Beginn, 2. Hilfe gar nicht gelesen • mögl. Missverständnis wg. „Anwendung durchspielen"	
Was ist unverständlich? Wo werden noch mehr Hilfen benötigt?	• Highlighting unklar: Lerneinheit in Bearbeitung oder abgeschlossen? Nicht als Auswahl/Vorschlag durch System verstanden. (Erklärung fehlt in 1. Hilfe.) • Anderes Verständnis von App-Logik: „dass, wenn man eine Information nicht hat, man davon ausgeht, dass sie so sein kann." • Verhalten der App nicht verstanden (unerwartet wegen Bugs im Framework).	Nicht klar, dass Kontextwerte nur in festgelegten Zeitfenstern gelten und danach nicht mehr. Tooltip „unbekannt" erst spät gesehen. „[...] muss ich das überall hinkopieren?” (Hilfe zur Zeitleiste nicht gelesen.)
Funktionalität		
Was sind typische Ziele und Abläufe? Was sind die wichtigsten bzw. am häufigsten benutzten Funktionen?	*Wechsel zu Lerneinheiten-Kontext-Tab bei Modellieren nicht nötig, weil Szenario erst unmittelbar zuvor angelegt. „Mein Plan ist jetzt, erst mal einmal durchlaufen lassen, um einen Eindruck zu kriegen und dann schrittweise durchspielen, um zu sehen, was da passiert.”*	
Welche Use-Cases werden noch nicht unterstützt?	Möchte „interaktiv den Verlauf der Simulation steuern" durch Anklicken der Lerneinheiten.	
Welche (wichtigen) Funktionen fehlen?	• *Kontextwertanzeige auch an Lerneinheiten bei Mouseover.* („Was hab ich gesagt?") • Matching bei Mouseover in Simulationsmodus („Ich bin jetzt neugierig, woran er festgemacht hat, dass er hier reingeht und nicht da reingeht” - zeigt mit Maus auf Lerneinheiten) • Simulation starten bei Vor. (Sonst: „Passiert nix. Jetzt bin ich hilflos” - immer Play vor Step umständlich, bei 10 Zeilen schnell durchgelaufen)	
Welche Funktionen werden nicht benötigt?	Ungefragtes Popup bei jeder Pause nervt („jaja, halt die Backen")	
Bedienbarkeit		
Ist der Simulator einfach und intuitiv bedienbar?	• s. u.: Verbesserungen • Kontext anlegen zuerst in Spaltenköpfen und Vorlagen-Auswahl gesucht (s. Test 12)	

	• Verlängern an Ecke statt an Kante erwartet (Excel) und nach oben ziehen versucht	
Wird der Simulator als konsistent, auch mit dem Autorensystem, empfunden?		s. u.: Positives
Welche Verbesserungen sind nötig oder gewünscht?	• Panel größer/kleiner machen • Erklärung zu Gültigkeitsdauer der Werte (s.o.: Hilfen) --> Hilfe zur Zeitleiste prominenter oder Option in Popover • Leere Zeilen überspringen. (Durchlaufen bis Ende „irritiert mich.") • Werte in Feldern. („für die Übersicht gewünscht, dass es dran steht") • Langziehen vereinfachen (Cursor allein nicht wahrgenommen, Bereich zu schmal, Bug (s.o.))	
Gesamteinschätzung		
Wurden Erwartungen getroffen? *Welche (anderen) Erwartungen gab es?*	Aufleuchten: „Ach DA machst du das [...] Ich hab mich gefragt, wie du's machst und da ist es gut"	(keine konkreten) Erwartungen: 3-dimensionaler Raum (Zeit, Kontext, Anwendung -> s. Anm.), „schwierig, auf den Monitor zu bringen" - "Im Prinzip hast du das Einzige gemacht, was du machen konntest, nämlich die Dimensionen entkoppelt"
Gibt das Testen mit dem Simulator ein gutes Gefühl?		Nicht gefragt (s. Anm.)
Positives	• Feststellung, dass nur Kontextinfos aus Szenario vorhanden: „was ja auch sinnvoll ist". • „Gut ist, dass der Popupdialog [Popover] mitwandert mit dem Feld. Dadurch hat man den Zusammenhang"	• „gut in das Vorhandene eingefügt": • „Look and Feel getroffen", „Logik von dem Simulator getroffen", • „arbeitest gut mit dem zusammen, was man als Autor für die Lerneinheit vorneweg eh schon gemacht hat" --> Vgl. Test 12
Was war am schwersten?	„Ich find das [links: Lerneinheiten-Graph] logisch, ich find das [rechts: Zeitleiste] logisch, aber wie es zusammen spielt, finde ich nicht logisch." (Bugs)	
Kommentare zur SUS		
• 2. „komplex, aber nicht unnötig" • 3. wegen Bugs • 4. Techn. Support: „manchmal" • 6. „kann ich schwer einschätzen" • 8. umständlich: „manches", z.B. das Runterziehen (etwas fehlerhaft)		

Ergebnisse der System Usability Scale (SUS)

Aussage	1 = Stimme voll zu	2	3	4	5 = Stimme gar nicht zu
Ich kann mir sehr gut vorstellen, das System regelmäßig zu nutzen.	**1, 2, 4, 8, 9, 11, 13**	3, 5, 6, 10	7, 12		
Ich empfinde das System als unnötig komplex.	12			**1, 3, 4, 5, 6, 8, 13**	2, 7, 9, 10, 11
Ich empfinde das System als einfach zu nutzen.	4, 10, 11	**1, 2, 5, 7, 9, 13**	6	3, 8, 12	
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	10, 12	3	1, 6	**4, 7, 8, 13**	**2, 5, 9, 11**
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.	**4, 7, 9, 10, 11, 13**	1, 2, 3, 5, 6	8, 12		
Ich finde, dass es im System zu viele Inkonsistenzen gibt.				2, 12, 13	**1, 3, 4, 5, 6, 7, 8, 9, 10, 11**
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen	9, 10	**1, 4, 5, 6, 11**	2, 8, 12, 13	3, 7	
Ich empfinde die Bedienung als sehr umständlich.		12	8	**1, 3, 5, 6, 7, 13**	2, 4, 9, 10, 11
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.	7, 10, 11	**1, 2, 4, 5, 9, 13**	3, 6	8	12
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	7, 12	5, 8, 11	2	**1, 3, 4, 6, 9, 13**	10