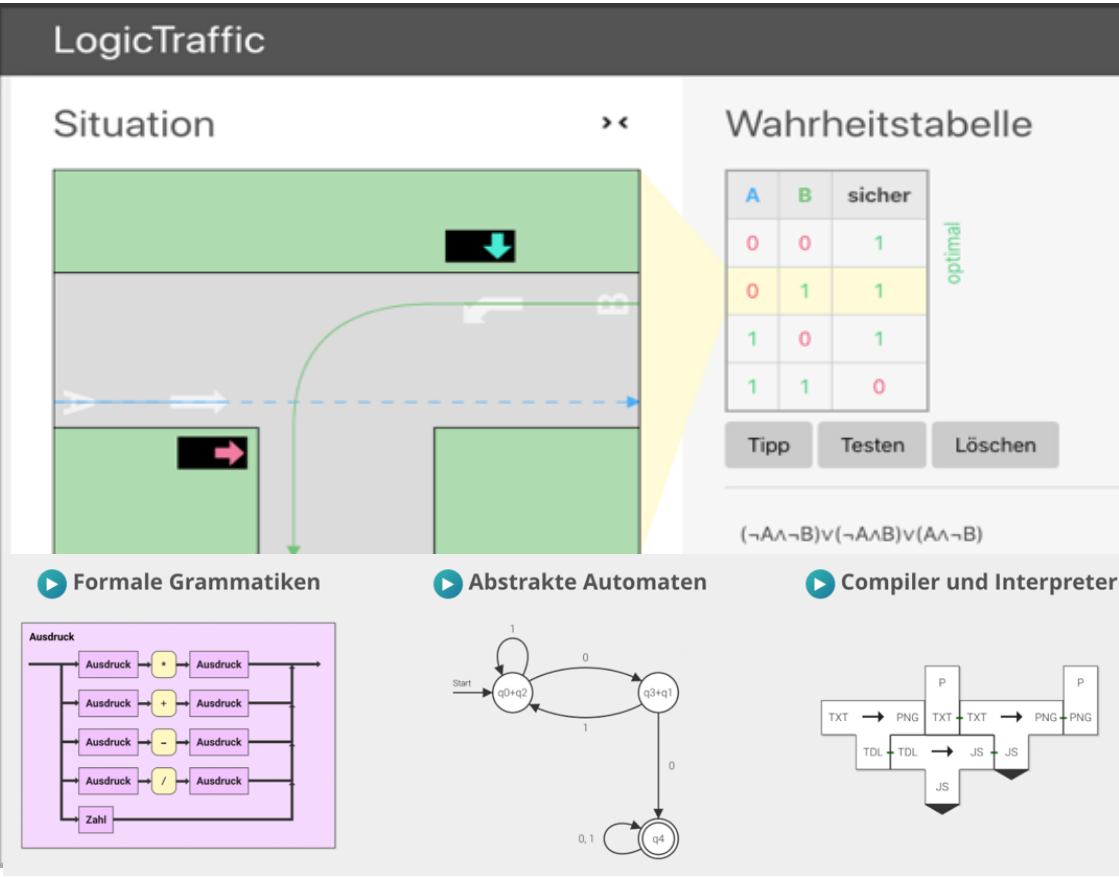
pädagogische hochschule schwyz



Entwicklung und langfristige Wartung interaktiver Lernumgebungen für den Informatikunterricht

Arbeitsbericht April 2021

Michael Hielscher, Ruedi Arnold & Werner Hartmann

Entwicklung und langfristige Wartung interaktiver Lernumgebungen für den Informatikunterricht

Arbeitsbericht April 2021

Michael Hielscher[1], Ruedi Arnold[2] & Werner Hartmann[3]

Abstract: Computergestützte, interaktive Lernumgebungen finden sich heute in vielen Unterrichtsfächern. Der Informatikunterricht ist per se prädestiniert für den Einsatz solcher Lernumgebungen. Stark verbreitet und viel genutzt sind Lernumgebungen für den Einstieg ins Programmieren und zur Visualisierung von Algorithmen. Daneben finden sich Lernumgebungen zu Datenbanken, Kryptologie, Netzwerk-Protokollen, der Simulation von Mikrocontrollern oder Themen aus der theoretischen Informatik. Diese bedingen themenspezifische Entwicklungen und können nicht mit gängigen Autorenwerkzeugen erstellt werden. Aufgrund des meist beträchtlichen Entwicklungsaufwandes entstehen solche Lernumgebungen oft im Rahmen von Hochschulprojekten und die mittelfristige Wartung ist nicht immer sichergestellt. Wechsel bei den zur Entwicklung verwendeten Technologien sowie bei den Endgeräten und Betriebssystemen gefährden zusätzlich die längerfristige Verfügbarkeit. Anhand von zehn Lernumgebungen, die in den letzten rund 20 Jahren in unserem Umfeld entstanden sind, untersuchen wir, welche Eigenschaften von Lernumgebungen zu einer längerfristigen Verfügbarkeit beitragen.

Keywords: Entwicklung interaktiver Lernumgebungen, nachhaltige Wartung von Lernsoftware, Informatikunterricht

1 Einleitung

Eine interaktive Lernumgebung ermöglicht den Dialog zwischen den Lernenden und dem Lernstoff und ermöglicht es den Lernenden mit der Lernumgebung zu interagieren, einen Lerninhalt zu erkunden und besser zu verstehen. Soll eine Lernumgebung höhere Stufen der Interaktivität - zum Beispiel gemäß der Taxonomie der Interaktivität von Multimedia von Schulmeister [Sc02] - ansprechen, kann die Lernumgebung in der Regel nicht mehr mit gängigen Autorenwerkzeugen entwickelt werden, sondern bedingt eine eigenständige, spezifisch auf den Lerninhalt ausgerichtete Softwareentwicklung und Programmierung.

Fachspezifische, computergestützte interaktive Lernumgebungen werden in der Regel von interdisziplinären Teams entwickelt, zusammengesetzt aus Vertretern der Fachdisziplin und Softwareentwicklern. Beim Informatikunterricht sind die Vertreter der Fachdisziplin meist auch gleich die Entwickler, mit ein Grund, warum es für die Informatik besonders viele Lernumgebungen gibt. Bereits 1996 machten sich Carlson, Guzdial et al.

[1] michael.hielscher@phsz.ch, Pädagogische Hochschule Schwyz, Institut für Medien und Schule
[2] ruedi.arnold@hslu.ch, Hochschule Luzern, Departement Informatik
[3] werner.hartmann@phsz.ch, Pädagogische Hochschule Schwyz, Institut für Medien und Schule

grundlegende Gedanken zur Gestaltung interaktiver, webbasierter Lernumgebungen [Ca96]. Dazu gibt es zahlreiche Empfehlungen, die sich auf Erkenntnisse aus dem Themenbereich "Multimedia Learning" (vgl. dazu etwa das Standardwerk von Richard Mayer [Ma09]) oder der Visualisierung von Algorithmen und Datenstrukturen (vgl. dazu etwa [DFS02]) abstützen. Als Methode für derartige Forschungs- und Entwicklungsprojekte bietet sich Design-Based Research an, also ein iteratives Vorgehen mit abwechselnden Phasen der Entwicklung und Erprobung bzw. Evaluation (vgl. dazu etwa [Re05]). Diese Ausführungen beziehen sich alle primär auf didaktische Aspekte bei der Gestaltung von Lernumgebungen.

Hingegen finden sich kaum Empfehlungen zum Vorgehen bei der Software-Entwicklung, der Bereitstellung und langfristigen Wartung und Weiterentwicklung von Projekten, die in der Schulpraxis eingesetzt werden sollen. Arnold und Hartmann [AH07] führen aus, dass Lehrpersonen abwägen, ob sich der Aufwand für die didaktische Aufbereitung eines Themas unter Einbezug einer Lernumgebung lohnt, sollte diese nach ein paar Jahren nicht mehr verfügbar sein. Für Lehrpersonen ist die Verfügbarkeit zentral, da die Einarbeitung, die Entwicklung begleitender Unterrichtsmaterialien und das Sammeln von Erfahrungen beim Einsatz im Unterricht viel Zeit benötigt. Der längerfristigen Verfügbarkeit und der Wartung muss deshalb schon zu Beginn der Entwicklung einer Lernumgebung Beachtung geschenkt werden.

2 Analyse von Lernumgebungen bezüglich ihrer längerfristigen Verfügbarkeit

Im Rahmen dieser Arbeit wurden zehn Lernumgebungen zu Informatikthemen mit Blick auf ihre längerfristige Verfügbarkeit und dabei angetroffenen Herausforderungen untersucht (siehe Tabelle 1). Die ausgewählten Lernumgebungen stammen alle aus unserem engeren Umfeld. Die Auswahl ist damit zwar nicht repräsentativ, erlaubt uns aber auch nicht-technische Aspekte miteinzubeziehen. Ein Teil dieser Lernumgebungen wurde seit der ersten Veröffentlichung mehrfach überarbeitet bis hin zu kompletten Neuentwicklungen, ein Teil ist heute aufgrund technischer Einschränkungen im Unterricht nicht mehr einsetzbar. Wie lange eine Lernumgebung für Schulen verfügbar ist, hängt von verschiedenen Faktoren ab. Es spielen sowohl rein technische Aspekte als auch personelle, finanzielle und institutionelle Rahmenbedingungen eine Rolle.

2.1 Risiko obsoleter Technologien minimieren

Das Angebot auf iLearnIT.ch wurde zu großen Teilen mit Adobe-Flash, die Lernumgebung zur Huffman-Kodierung als Java-Applet entwickelt. Beide Technologien werden von aktuellen Browsern nicht mehr unterstützt. Zum Zeitpunkt der Entwicklung einer Lernumgebung ist schwierig vorauszusehen, welche Technologien längerfristig Bestand haben werden. Auch verwendete Bibliotheken und Frameworks können unter Umständen nicht weiterentwickelt werden. Bei der Problematik obsoleter Technologien

handelt es sich um ein generelles Phänomen in der Softwareentwicklung. Empfehlenswert ist es dennoch, Technologien zu verwenden, die auf breiter Basis auch in der Softwareindustrie genutzt werden und damit eine in der Regel größere Halbwertszeit garantieren als Nischentechnologien und/oder Hochschul-Eigenentwicklungen. Zudem sollte die Anzahl eingesetzter Technologien so gering wie möglich gehalten werden.

Lernumgebung	Thema	Technologie		Veröffentlichung	
		1. Version	2. Version	1. Version	letzte Anpassung
AtoCC / FLACI[1]	Formale Sprachen, abstrakte Automaten, kf-Grammatiken und Compilerbau	Win32 Anwendung	JavaScript, HTML, CSS	2004	2020
Exorciser[2]	Reguläre Sprachen, kf-Grammatiken, Markov-Algorithmen	Java	-	2004	2006
GraphBench[2]	NP-Vollständigkeit und Reduktionen	Java	-	2005	2005
iLearnIT[3]	Fundamentale Ideen der Informatik	Flash	HTML	2007	2018
Kara[2]	Einstieg ins Programmieren	Java	Java	1998	2016
Kompression[2]	Huffman-Kodierung	Java Applet	-	2001	2004
LogicTraffic[4]	Aussagenlogik	Java	JavaScript, HTML, CSS	2007	2020
Parser4Kids[2]	Einführung in Parsing	Java	-	2008	2006
ProgrammingWiki[5]	Einstieg ins Programmieren, Datenbanken etc.	PHP, MediaWiki	-	2008	2020
Soekia[6]	Funktionsweise von Internetsuchmaschinen	Java, Lucene	JavaScript, HTML, CSS	2003	2018

[1]atocc.de flaci.com [2]swisseduc.ch [3]ilearnit.ch [4]logictraffic.ch [5]programmingwiki.de [6]soekia.ch

Tabelle 1: Untersuchte Lernumgebungen für Informatik

2.2 Abhängigkeit von Software-Bibliotheken vermeiden

Die Lernumgebung Soekia verwendete in der ersten Version Java als Technologie und setzte auf Apache Lucene für die Simulation einer Suchmaschine. Bei Bibliotheken wie Lucene können größere Versionssprünge eine einfache Aktualisierung der Lernumgebung verhindern. Zudem wird bei Lernumgebungen oft nur ein geringer Teil der Funktionspalette umfangreicher Bibliotheken benötigt und die Gesamtkomplexität der Lernumgebung wird unnötig erhöht. Ähnlich verhält es sich mit komplexen Entwicklungsumgebungen mit mehrstufigen Übersetzungsprozessen (z.B. TypeScript, Polyfills, SASS, Webpack), die sich bei umfangreichen Industrieprojekten anbieten, die Überarbeitung einer Lernumgebung aber erschweren. Die Abhängigkeit von eingesetzten Frameworks und Tools wurde von Ward Cunningham treffend mit der "technical debt"-Metapher [Cu92] beschrieben. Jedes Framework erhöht den Lern- und

Einarbeitungsaufwand für Entwickler und kann die längerfristige Wartung erschweren. Bei der Entwicklung von Lernumgebungen empfiehlt es sich deshalb, wenn immer möglich auf die Verwendung und Einbindung umfangreicher Bibliotheken oder externer Dienste zu verzichten.

2.3 Modulare Umgebungen entwickeln

Je nach Themenbereich lassen sich Lernumgebungen mehr oder weniger modular aufbauen. Die Lernumgebung AtoCC/FLACI zum Beispiel umfasst voneinander unabhängige Teilmodule zu formalen Sprachen, abstrakten Automaten sowie zum Compilerbau (siehe Abb. 1). Diese Modularisierung ermöglicht die weitgehend unabhängige Aktualisierung einzelner Module und nicht mehr funktionsfähige Module können einfach entfernt werden. Das modulare Vorgehen hat sich bei umfangreicheren Lernumgebungen wie InfoTraffic und Kara bewährt.

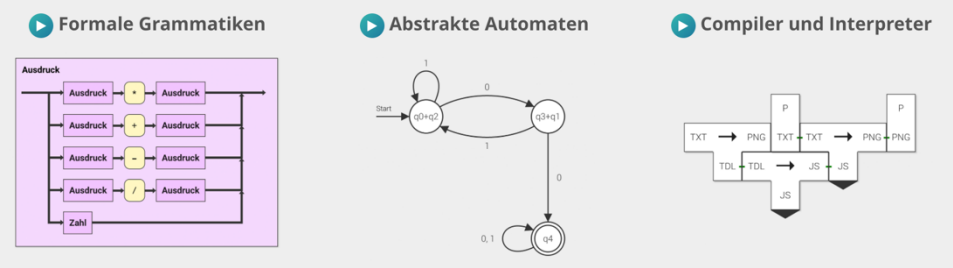


Abb. 1: Lernumgebung FLACI mit verschiedenen, vernetzten Modulen

2.4 Begleitmaterialien versionsunabhängig gestalten

Die Lernumgebung Kara adressiert verschiedene Themen der Programmierung. Die auf Java basierende Lernumgebung wird seit fast zwei Jahrzehnten in vielen Schulen genutzt. Dazu trägt das umfangreiche Angebot an Begleitmaterialien, von Tutorials über Übungsaufgaben samt Lösungen bis hin zu ganzen Lehrgängen bei. Für die Verbreitung an Schulen spielen begleitende Unterrichtsmaterialien eine wichtige Rolle, sowohl Hintergrundinformationen und methodisch-didaktische Hinweise für die Lehrpersonen als auch Materialien, die sich direkt an die Schülerinnen und Schüler richten. Umfangreiche Begleitmaterialien sind aber ein zweischneidiges Schwert: Sie tragen zur Verbreitung der Lernumgebung bei, was sich wiederum positiv auf die Motivation zur Weiterentwicklung auswirkt. Gleichzeitig resultiert ein höherer Wartungsaufwand bei Anpassungen der Software und hier insbesondere der graphischen Benutzerschnittstelle. Wichtig ist deshalb, dass die Begleitmaterialien möglichst produktunabhängig gestaltet werden, zum Beispiel ohne oder mit sparsamer Einbindung von Screenshots.

2.5 Lokale Installation vermeiden

Die Nutzung im Unterricht erfordert eine möglichst einfache Nutzung der Lernumgebung. Müssen erst aufwändige Softwareinstallationen vorgenommen werden, erschwert das die verbreitete Nutzung. AtoCC wurde als Win32-Desktop Anwendung entwickelt. Lokal wurden weitere Softwarewerkzeuge wie Java Development Kit, Scheme, .NET Framework oder Ghostscript eingebunden. Diese Abhängigkeiten führten immer wieder zu Problemen und Supportaufwand. Lernumgebungen sollten wann immer möglich auf lokale Installationen verzichten. Überarbeitungen und Fehlerbeseitigungen können bei Webapplikationen zeitnah eingepflegt werden. Lernende können webbasierte Lernumgebungen zudem auch einfacher zuhause auf ihren privaten Endgeräten nutzen, siehe z.B. Abb. 2.

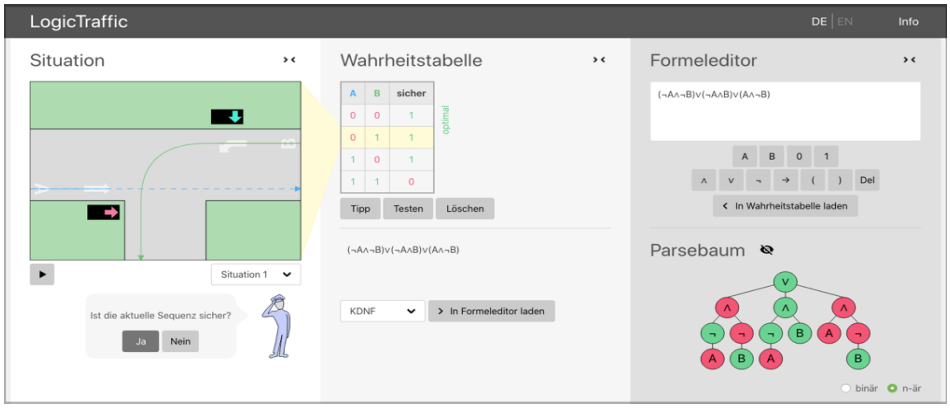


Abb. 2: Webbasierte Version der Lernumgebung LogicTraffic

2.6 Attraktivität der Weiterentwicklung bedenken

Bei den von uns betrachteten Lernumgebungen Exorciser, GraphBench und Parser4Kids handelt es sich um Java-Entwicklungen, die nur noch bedingt auf Endgeräten einsetzbar sind und überarbeitet werden müssten. Allen drei Lernumgebungen gemeinsam ist, dass sie nur eine sehr kleine Zielgruppe adressieren. GraphBench als Beispiel behandelt NP-vollständige Probleme und Reduktionen, ein anspruchsvolles Thema aus der Komplexitätstheorie auf Stufe Hochschule. Die Lernumgebung entstand im Rahmen einer Dissertation. Ein Re-Engineering der Java-Applikation in eine Webapplikation wäre mit großem Aufwand verbunden, der Anreiz für eine Überarbeitung ist aufgrund der kleinen Zielgruppe gering. Bereits vor Beginn der Entwicklung einer Lernumgebung sollte abgewogen werden, ob die Zielgruppe genügend groß ist. Eine Veröffentlichung als Open Source kann Überarbeitungen zwar ermöglichen. Die Erfahrungen bei den untersuchten Lernumgebungen zeigen aber, dass kaum substanzielle Beiträge durch Drittpersonen erfolgten.

3 Fazit

Unsere gesammelten Erfahrungen bei der Entwicklung und Pflege verschiedenster Lernumgebungen über mehrere Jahrzehnte hinweg zeigen, dass insbesondere technisch überschaubare Umgebungen mit stetiger Nutzung und Zielgruppe die größten Überlebenschancen besitzen.

Auf der technischen Ebene ist die Lernumgebung so einfach wie möglich zu halten. Der Funktionsumfang soll sich gemäß dem Paretoprinzip auf die Funktionen beschränken, die im Unterricht relevant sind. Auf den ersten Blick attraktiv erscheinende Features, die im Unterricht jedoch nur selten genutzt werden, werden besser weggelassen.

Neben den technischen Aspekten und Empfehlungen muss immer auch der Faktor "Mensch" berücksichtigt werden. Die Erfahrung zeigt, dass die längerfristige Verfügbarkeit einer Lernumgebung entscheidend von den Entwicklerinnen und Entwicklern und der Verankerung in einer Institution (z.B. Hochschule, Schule, Trägerverein) abhängt. Nutzt die Hochschule die Lernumgebung selbst in ihrem Unterricht oder im Rahmen von Weiterbildungsangeboten, trägt das zur Motivation bei, die Lernumgebung über einen längeren Zeitraum zu warten.

Literatur

[AH07] Arnold, Ruedi; Hartmann, Werner: Pragmatische Empfehlungen zur Entwicklung von interaktiven Lernumgebungen. Didaktik der Informatik in Theorie und Praxis–INFOS 2007–12. GI-Fachtagung Informatik und Schule, 2007.

[Ca96] Carlson, David; Guzdial, Mark; Kehoe, Colleen; Shah, Viren; Stasko, John: WWW Interactive Learning Environments for Computer Science Education. Association for Computing Machinery, New York, NY, USA, 1996.

[Cu92] Cunningham, Ward: The WyCash portfolio management system. ACM SIGPLAN OOPS Messenger, 4(2):29–30, 1992.

[DFS02] Demetrescu, Camil; Finocchi, Irene; Stasko, John T: Specifying algorithm visualizations: Interesting events or state mapping? In: Software Visualization, S. 16–30. Springer, 2002.

[Ma09] Mayer, Richard E.: Multimedia Learning. Cambridge University Press, NY, 2009.

[Re05] Reinmann, Gabi: Innovation ohne Forschung? Ein Plädoyer für den Design-Based Research- Ansatz in der Lehr-Lernforschung. Unterrichtswissenschaft, 33(1):52–69, 2005.

[Sc02] Schulmeister, Rolf: Taxonomie der Interaktivität von Multimedia - Ein Beitrag zur aktuellen Metadaten-Diskussion. it-Information Technology, 44(4):193–199, 2002.