B. Batinic, A. Koller & H. Sikora (Hrsg.). (2007). E-Learning, digitale Medien und lebenslanges Lernen. (Schriftenreihe E-Learning), Linz: Trauner. 3 - 11.

Entwicklungsgeschichtliche Analogien zwischen Programmiersprachen und Lernplattformen

Jörg R. MÜHLBACHER und Andreas PUTZINGER

Zusammenfassung

Dieser Beitrag versucht entscheidende Kriterien zu identifizieren, um Investitionen in Lernplattformen und Lernmaterialien langfristig zu sichern und somit „stranded and short-term investments“ zu vermeiden. Zur Beantwortung werden Analogieschlüsse aus der Entwicklungsgeschichte der Programmiersprachen gezogen. Abschließend wird ein Ausblick auf Funktionen gegeben, welche nach Meinung der Autoren prägend für die nächste Generation von Lernplattformen sein werden.

1 Einleitung

In einer 2005 veröffentlichten Studie (vgl. BAUMGARTNER ET AL., 2003), die vom Österreichischen Bundesministerium für Bildung, Wissenschaft und Kultur in Auftrag gegeben worden war, wurden alleine im europäischen Raum rund 400 erhältliche Systeme für E-Learning erhoben, wovon 115 dem engeren Bereich der Learning Management Systemen (LMS) zuzuordnen sind.

Dieser beobachtete Trend zur Diversität erlaubt einen Vergleich mit der Entwicklungsgeschichte der Programmiersprachen, welche im folgenden Kapitel kurz zusammengefasst wird. Angelehnt an diese werden anschließend Analogien zu Lernplattformen identifiziert, welche die Basis für die gegebenen Empfehlungen bilden.

2 1001 Programmiersprachen

Die Entwicklung und Marktdurchdringung bei Programmiersprachen war und ist auch heute noch vor allem durch einige Faktoren gekennzeichnet, welche nachfolgend genannt und im Anschluss diskutiert werden:

- Meilensteine in der Konzeption (umgesetzte Paradigmen)
- heftigste akademische Diskussionen über relativ geringfügige syntaktische Variationen innerhalb von Sprachfamilien und damit verbundene, unzählige, meist kurzlebige Neuentwicklungen
- politische Argumente
- Marktposition der Anbieter

In der Vergangenheit war die Einführung eines neuen Programmierparadigmas meist mit völlig neuen und inkompatiblen Programmiersprachen verbunden. Betrachtet man die Entwicklungsgeschichte imperativer Sprachen war ein früher Meilenstein sicherlich der

Übergang von Assemblercode zu höheren Sprachen, die damals vor allem durch FORTRAN und C vertreten waren. Dieser Schritt ermöglichte erstmals eine Abstraktion der Software von der verwendeten Hardware und stellte somit die Basis für die Entwicklung portabler Programme nach dem bekannten Motto „write once – deploy anywhere“ dar. Die Konzepte der Blockstrukturierung und der Gültigkeitsbereiche (engl. Scopes) für Variable brachte erstmals Algol 60 (BACKUS ET AL., o.J.). 1972 erkannte man, umgesetzt mit Vertretern der PASCAL-Familie (vgl. JENSEN & WIRTH, 1991), dass viele Unzulänglichkeiten bisheriger Software a priori durch strengere Typbindungen und Prüfungen zur Compilezeit vermieden werden können. In Modula-2 wurde das Konzept der Datenkapselung eingeführt (vgl. WIRTH, 1989 sowie MÜHLBACHER & HÖRMANSEDER, 1987), was wiederum Voraussetzung und Bestandteil objektorientierter Sprachen ist, als deren erster Vertreter Smalltalk (vgl. GOLDBERG 1983) zu nennen ist. Die Entwicklung ist natürlich längst nicht beendet. Auch in der heutigen Zeit wird mehr denn je an Konzeptverbesserungen und weiteren Abstraktionsebenen geforscht, die aber eher mit Spracherweiterungen, denn mit Ersetzungen werben und somit meist nahtlos in einen bestehenden Code adoptiert werden können. Als ein Beispiel sei hier das relativ neue Paradigma der Aspektorientierung[1] erwähnt, welches beispielsweise in Java mit AspectJ[2] oder in C# mit Aspect#[3] umgesetzt werden kann, um nur zwei Beispielimplementierungen zu nennen.

Die Diskussion um die „beste“ Programmiersprache scheint sich mittlerweile etwas gelegt zu haben. Dies ist sicherlich auch Umgebungen wie Microsoft.NET oder Java zu verdanken. Die Programmiersprache per se tritt hierbei immer mehr in den Hintergrund, da durch Einführung von Laufzeitsystemen und der damit verbundene Übersetzung in Zwischensprachen, wie in die Common Intermediate Language (CIL)[4] oder auch in JAVA Bytecode (vgl. LINDHOLM & YELLIN, 1999), von der ursprünglich verwendeten Programmiersprache abstrahiert wird. Dadurch herrscht implizite Kompatibilität zwischen Bibliotheken desselben Systems über alle Hardwareplattformen und Sprachen hinweg. Weiters ist zu beobachten, dass das Kriterium der konkreten, eventuell schon beherrschten Syntax einer Sprache als Entscheidungsgrundlage für ein Projekt immer unwichtiger wird und vom Kriterium der Verfügbarkeit einer breiten Palette umfassender Bibliotheken und Frameworks für ein Laufzeitsystem abgelöst wird. Dies führt dazu, dass verstärkt eine Fokussierung auf die zu erstellende Anwendung stattfindet und das syntaktische Umfeld als naives Kriterium aus Entscheidungsprozessen verdrängt wird.

Generell ist festzustellen, dass zwischen den „Fundamentalschritten“ hitzige Kontroversen tobten. Zahlreiche universitäre Einrichtungen sowie auch kommerziell angetriebene Firmen entwickelten unterschiedliche Ausprägungen diverser Konzepte, was in einer Vielzahl an Programmiersprachen resultierte. Diese setzten allesamt zwar sehr ähnliche Konzepte um, waren jedoch meist inkompatibel zueinander. Die Unterscheidung und somit der Rechtfertigungsgrund für eine eigene Sprache sind und waren oft wenige spezielle Punkte, wie beispielsweise die Berufung auf die syntaktisch vereinfachte Umsetzung eines Konzeptes.

---

[1] Web: HTTP://DE.WIKIPEDIA.ORG/WIKI/ASPEKTORIENTIERTE_PROGRAMMIERUNG

[2] Web: HTTP://WWW.ECLIPSE.ORG/ASPECTJ/

[3] Web: HTTP://ASPECTSHARP.SOURCEFORGE.NET/

[4] Web: HTTP://WWW.ECMA-INTERNATIONAL.ORG/PUBLICATIONS/STANDARDS/ECMA-335.HTM

Mit zunehmender Globalisierung werden auch politische Argumente für oder gegen eine Sprache immer schlagender, sei dies auf einzelne Firmen oder auch auf ganze Nationen bezogen. Als Beispiel seien hier die nicht immer sachlichen Argumente gegen PL/I[5] in den 70er-Jahren genannt, die vermuten ließen, dass diese von einer gegen IBM gerichteten Strömung initiiert wurden. Als anderes Beispiel soll hier auch die Sprache Ada[6] genannt werden, gegen welche a priori eine gewisse Skepsis bestand, da sie vom amerikanischen Verteidigungsministerium (Department of Defense) vorangetrieben wurde und außerhalb der USA wenig dauerhaften Anklang fand. Auch heute können noch ähnliche Diskussionen mitverfolgt werden, wenn es um die Frage „.NET oder Java?“ geht, da man bei Verwendung von Microsoft.NET implizit auf Microsoft-Produkte aufsetzt, wobei sich dieser Umstand mit der Akzeptanz und dem Einsatz des MONO- sowie des Rotor-Projekts ändern kann.

Was die Marktposition der Anbieter betrifft, so sind ihr viele Produkte zum Opfer gefallen: Zum Teil, weil der finanzielle Atem für kontinuierliche Weiterentwicklung und Wartung fehlt, zum Teil aber auch, weil wegen Uneinigkeiten zueinander inkompatible Dialekte entstanden sind.

Es ist ebenfalls zu beobachten, dass Entwicklungsgeschichte und Marktdurchdringung von Betriebssystemen ähnlich verlaufen sind, zahlenmäßig in einem kleineren Umfang, aber inhaltlich durchaus vergleichbar.

Die berechtigte Frage ist daher: Warum soll die Entwicklung bei Lernplattformen grundsätzlich anders verlaufen?

3 Streit um die beste Plattform

In Analogie zu den Programmiersprachen, wo bereits eine Marktbereinigung aus verschiedenen Gründen erfolgt ist, kann dies im Segment der E-Learning-Plattformen gerade beobachtet werden. Tummelten sich, wie in der einführenden Studie zitiert (vgl. BAUMGARTNER ET AL., 2003), vor einigen Jahren noch sehr viele Plattformen am Markt, scheint das Angebot heute eher zu stagnieren oder sogar abzuflauen.

Wenn bei Neueinführung einer E-Learning-Plattform auf ein freies Projekt gesetzt werden soll und an der Institution keine eigene Plattform entwickelt wird, so ist zu beobachten, dass die Wahl sehr oft auf einige wenige fällt. Als Beispiele genannt seien hier Moodle[7], Ilias[8], WeLearn[9] (in Österreich) oder auch das international von vielen Universitäten getragene Sakai-Projekt[10].

---

5 Web: HTTP://WWW-306.IBM.COM/SOFTWARE/AWDTOOLS/PLI/

6 Web: HTTP://WWW.OPEN-STD.ORG/JTC1/SC22/WG9/

7 Web: HTTP://WWW.MOODLE.ORG

8 Web: HTTP://WWW.ILIAS.DE

9 Web: HTTP://WWW.WELEARN.AT

10 Web: HTTP://WWW.SAKAIPROJECT.ORG

Der kommerzielle Bereich erweckt den Eindruck, dass die Produkte der Firma Blackboard Inc.[11] stärker an Verbreitung gewinnen. Dies lässt sich sicherlich nicht zuletzt durch die Übernahme der Konkurrenten Prometheus und WebCT begründen, deren Funktionen langfristig in die Hauptproduktlinie übernommen werden[12]. Weiters wurde 2006 vom US-Patentamt einem Patentantrag von Blackboard Inc. stattgegeben (Patent US 6 988 138[13]), der nicht nur rein technische Details enthält, sondern auch Begriffsbestimmungen und ganze Prozessabläufe im E-Learning umfasst. Dies löste weltweit einen Proteststurm aus, weil dadurch u. a. natürlich auch Open-Source-Produkte Gefahr laufen, gegen Patentrechte zu verstoßen. Ob ein durch das Software Freedom Law Center eingebrachter Einspruch[14] erfolgreich sein wird, bleibt abzuwarten. Die Hauptargumentationslinie der Gegner will beweisen, dass von Blackboard Inc. versucht wurde, einerseits Trivialitäten zu patentieren, andererseits aber auch Techniken, die bereits den heutigen Stand der Technik repräsentieren und somit nicht patentierbar sind. In der Zwischenzeit wurde aber bereits eine erste Klage eingereicht (Blackboard Inc. vs. Desire2Learn Inc).

Bei der Frage nach der zu verwendenden Lernplattform empfiehlt sich die Rückbesinnung auf die ursprüngliche Überlegung, weshalb eine Plattform überhaupt benötigt wird und welche Rolle diese im Lernprozess spielt. Dies kann wiederum mit einer Analogie auf Programmiersprachen beantwortet werden: Genauso, wie eine konkrete Programmiersprache ein Mittel zum Zweck für die Erstellung einer Applikation ist, dient die Lernplattform hauptsächlich als Umfeld für die Vermittlung und Erarbeitung von Wissen mittels Lernmaterialien. Somit ist es nicht die Lernplattform selbst, welche die Qualität von E-Learning maßgeblich entscheidet, sondern vielmehr sind die Lehr- und Lernmaterialien und deren Aufbereitung ausschlaggebend. Der Plattform obliegt die Aufgabe, den Lern- und Lehrprozess optimal zu unterstützen und dabei vor allem eine möglichst intuitive, flexible und auch standardkonforme Umgebung für Kurse zu schaffen, sodass verschiedene pädagogische Modelle bestmöglich abgewickelt werden können und individuelle Lernarrangements organisatorisch ermöglicht werden.

In Helmut Qualtingers „Der Halbwilde“ heißt es sinngemäß, dass *„ich zwar nicht weiß, wohin ich genau hinfahren möchte, ich mit diesem Motorrad aber schneller dort bin“*. Die darin enthaltene Aussage ist doppelzüngig: Vordergründig macht sich Qualtinger darüber lustig, dass man das eigentliche Ziel verloren habe und sich auf ein technisches Vehikel überbordend konzentriert, anderseits wird doch zugegeben, dass technische Voraussetzungen stimmig sein sollten.

Diese Aussage kann durchaus auch auf Lernplattformen angewandt werden: Auch wenn man zum Entscheidungszeitpunkt für eine Lernplattform noch nicht erahnen kann, welche konkreten Anforderungen im späteren Betrieb gestellt werden und welche künftigen Entwicklungen sich im E-Learning-Bereich etablieren werden, müssen die Auswahlkriterien für die Plattform darauf abzielen, die oben genannten Unsicherheitsfaktoren zu berücksichtigen. Bedenkt man, dass die – auch kostenmäßig – dominanten Werte in einer E-Learning-Umgebung einer Institution die Kursmaterialien sind, liegt auf der Hand, dass

---

[11] Web: HTTP://WWW.BLACKBOARD.COM

[12] Web: HTTP://WWW.BLACKBOARD.COM/WEBCT

[13] Web: HTTP://PATFT.USPTO.GOV/NETAHTML/PTO/SRCHNUM.HTM, Suche nach 6,988,138

[14] Web: HTTP://WWW.SOFTWAREFREEDOM.ORG/NEWS/20061130A.HTML

es absolute Pflicht sein muss, in der Kurserstellung auf Standards zu setzen. Beispiele hierfür sind SCORM[15], CPS[16], QTI[17] und LOM[18], welche nach dem momentanen Stand der Technik unbedingt seitens der Plattform unterstützt werden sollten. Nur durch diese Standardkonformität kann gewährleistet werden, dass bei einem eventuellen späteren Wechsel der Plattform die wichtigsten Assets eines Lernarrangements, nämlich die angesprochenen Lerninhalte, weiter verwendet werden können.

Ein weiteres wichtiges Funktionsmerkmal einer Plattform ist die Möglichkeit, Inhalte aus dem System exportieren zu können. Als Beispiel sei hier genannt, dass man im System angelegte Kalender in einem standardisierten Kalender-Format, wie etwa iCal, lokal auf dem Arbeitsplatz speichern kann.

Weiters soll bedacht werden, dass Plattformen, welche definierte Schnittstellen für PlugIns bieten, eventuell auch ohne Versionsaufstieg (und somit ohne den damit verbundenen Kosten und ohne dem daraus entstehenden Aufwand) mit neuen Entwicklungen mithalten können.

4 Gestaltung des Contents

Nicht nur bei der Wahl der Lernplattform ist das Argument der ökonomischen Zukunftssicherheit ausschlaggebend, sondern auch bei der Erstellung von Lerninhalten. Es ist durchaus zu empfehlen, auch bei Lernmaterialien auf Standardkonformität zu setzen, da dies im Weiteren Plattformunabhängigkeit bedeutet. Auf Standards aufgesetzte Materialien können zu einem späteren Zeitpunkt, im besten Fall ohne Veränderung, auch in anderen Plattformen verwendet werden. Erreicht kann dies dadurch werden, indem auf plattformproprietär erstellte Inhalte verzichtet (z. B. mit WYSIWYG-Objekten innerhalb der Plattform) und jegliches relevantes Lernmaterial unabhängig, standardkonform und außerhalb der Plattform mit individuell präferierten Werkzeugen erstellt wird. Auf diese Weise wird gewährleistet, dass mit dem monetär aufwändigen Prozess der Lernpaketerstellung nicht nur auf eine singuläre Plattform abgezielt wird, sondern das Material zu einem späteren Zeitpunkt auf einer Vielzahl von Plattformen lauffähig sein wird.

Das Einhalten von Standards ist aber nicht nur deshalb notwendig, um zu einem gewissen Zeitpunkt möglichst flexibel und unabhängig zu sein, sondern auch, um bei künftigen Erweiterungen auf der sicheren Seite zu sein. Natürlich werden auch Standards bzw. Defacto-Normen weiterentwickelt und neu etabliert. Lernobjekte, die in einem alten Standard entwickelt wurden, können meist in den neuen konvertiert werden. Für künftige Plattformen ist es wesentlich einfacher, einen automatisierten Migrationsprozess oder zumindest Kompatibilität anzubieten, wenn zuvor auf (mittlerweile veraltete) Standards gesetzt wurde. Die technische Lebensdauer von Lernpaketen wird durch die Einhaltung von Standards deshalb quasi automatisch erhöht.

---

[15] Web: HTTP://WWW.ADLNET.GOV/SCORM/INDEX.CFM

[16] Web: HTTP://WWW.IMSGLOBAL.ORG/CONTENT/PACKAGING/INDEX.HTML

[17] Web: HTTP://WWW.IMSGLOBAL.ORG/QUESTION/INDEX.HTML

[18] Web: HTTP://LTSC.IEEE.ORG/WG12/

Den Autoren ist durchaus bewusst, dass die Gestaltungsfreiheit für Lernobjekte auf den kleinsten gemeinsamen Nenner, nämlich auf den Standard selbst, beschränkt ist, und somit von auf den ersten Blick oft praktisch anmutenden Funktionen einer speziellen Plattform explizit abgeraten wird. Bei näherer Überlegung ist dies aber in Hinblick auf ungewisse Entwicklungen und der Vermeidung von kurzlebigen Investments notwendig und gerechtfertigt.

Eine weitere Empfehlung ist, Lerneinheiten im Umfang möglichst klein zu halten, sodass man von Mini- oder sogar Mikroeinheiten sprechen kann. Diese kleinen Einheiten können dann in mehrere größere eingebunden werden (1:n-Beziehung). Der Vorteil, der sich daraus ergibt besteht darin, dass Inhalte nur an einer einzigen Stelle gewartet werden müssen und automatisch in allen verwendeten Paketen aktualisiert werden. In der Terminologie der „Content Packaging Specification“ (CPS) von IMS[19] spricht man hierbei vom Konzept der „Submanifeste“. Somit können einzelne Teile schnell und einfach wieder verwendet werden, wobei ihre Wartung zentral erfolgt („Single-Source“-Prinzip). Zur Verwaltung eignen sich sogenannte „Content Repositories“, welche einerseits Funktionen zur Verwaltung von Lerninhalten und andererseits auch Funktionen zum späteren Wiederauffinden von bereits erstellen Modulen bieten.

Es genügt aber nicht, nur bei der Zusammenstellung des Lernmaterials auf E-Learning-Standards zu setzen. Auch bei verwendeten Dokumentformaten müssen Überlegungen bezüglich Standards eine Rolle spielen. Sehr beliebt für Lernmaterialien sind das „Portable Document Format“ (PDF) sowie Flash (SWF), beide von der Firma Adobe. Diese Formate haben die Eigenschaft, dass Inhalte nur betrachtet, meist aber nicht geändert werden können. Es existieren somit zwei unabhängige Dateien mit demselben Inhalt, einmal das bearbeitbare Dokument (z. B. in Microsoft Word verfasst) und weiters die daraus generierte Datei für das Lernpaket, z. B. ein PDF-Dokument. Dies birgt die organisatorische Gefahr, dass, wenn künftig etwas an den Inhalten des PDFs geändert werden soll, das Originaldokument nicht mehr gefunden wird. Deshalb ist es notwendig, dass auch die zugehörigen bearbeitbaren Dateien, welche meist nicht Teil des Lernpaketes sind, strukturiert abgelegt werden.

Weiters ist offensichtlich, aber nur bedingt als indirekte Empfehlung zu betrachten, dass je standardisierter, strukturierter und technisch einfacher das Lernmaterial gehalten wird, desto einfacher sich künftig die Wartung desselben gestaltet und desto größer die künftige erwartete Kompatibilität sein wird.

5 Ausblick auf künftige Erweiterungen

Ein weiterer Trend zeichnet sich bereits jetzt ab. Es ist unumstritten, dass neue Formen des Lernens und Lehrens häufiger werden: Zwar gilt Blended Learning zur Zeit als eine der effektivsten Lernarrangements, doch ist, vor allem bei der Vorbereitung auf eine Kursteilnahme oder für „Massenschulungen“, ein Vordringen von E-Learning ohne Präsenzphasen zu beobachten. Bei diesem Ansatz wird der persönliche Kontakt zu

[19] Web: HTTP://WWW.IMSGLOBAL.ORG

physischen Lehrpersonen geringer oder entfällt ganz. Genau diesem Faktum muss eine E-Learning-Plattform in Zukunft Rechnung tragen.

Mit abnehmendem Kontakt zu physischen Lehrenden wird die menschliche Komponente im Unterrichtsprozess reduziert. Genau diese Komponente ist es aber, die sich an die Bedürfnisse von Lernenden anpasst und individuell auf diese eingehen kann. Diese Aufgabe des „an den Lerndenen anpassen und individuell unterstützen“ muss in Zukunft bestmöglich von der Lernplattform selbst übernommen werden, nicht nur auf Ebene von einzelnen Benutzern sondern beispielsweise auch auf Ebene von Arbeitsgruppen, sodass auch in diesen beste Ergebnisse erzielt werden können (Ermitteln der optimalen Zusammensetzung einer Gruppe, etc.). Jetzige Plattformen sind zwar adaptierbar, d. h., dass sie auf ein bevorzugtes Lernarrangement zugeschnitten, also adaptiert, werden können. Sie sind aber nicht adaptiv, d. h., sie können nicht auf individuelles Lernverhalten eingehen. Durch Einsatz adaptiver Techniken werden sie sich an individuelle Lerntypen sowie Lernstile automatisch anpassen können, womit ein konstruktives Lernmodell unterstützt wird. Die Autoren sind der Überzeugung, dass adaptive Funktionen einen wichtigen Meilenstein in der Entwicklung der nächsten Generation von E-Learning-Systemen darstellen werden. Diese These wird gestützt durch zahlreiche Publikationen (z. B. CRISTEA ET. AL., 2006) oder auch durch das Stattfinden von SummerSchools zu dem Thema „Personalized E-Learning“[20].

Nicht nur die Lernplattformen müssen für die Unterstützung von Adaptivität geändert werden. Gleichzeitig müssen natürlich auch Lernmaterialien für diese Zwecke angepasst werden. Zusätzliche Metadaten und für verschiedene Lerntypen adaptierte und somit inhaltsredundante Materialien werden einen wichtigen Teil der Lernpakete von morgen ausmachen. Die Plattform wird implizit durch alleinige Bedienung des Systems und durch Beobachtung der einzelnen Benutzer sowie auch explizit durch notwendiges Feedback der Lernenden besser auf die Bedürfnisse jedes einzelnen eingehen können.

Als erste Prototypen für adaptive Lernplattform sind NetCoach (vgl. WEBER AT AL., 2001) und im erweiterten Sinne auch AHA (vgl. DE BRA ET AL., 2006) zu nennen.

6 Fazit

Die im Zusammenhang mit der Entwicklung von Programmiersprachen und -systemen genannten Beobachtungen lassen einige Empfehlungen für das Entwickeln von E-Learning-Material zu:

1. Analog dazu, dass eine Programmiersprache eine gute Basis für die zu entwickelnde Applikation liefern muss, ist ein Kriterium bei einer E-Learning-Plattform, dass sie eine bestmögliche und zukunftssichere Umgebung für Lerninhalte bieten soll, die an die gewünschten Lernszenarien flexibel angepasst werden kann. Die Rolle der Plattform selbst soll nach Meinung der Autoren nicht überbewertet werden. Wichtiger als die Plattform ist sicherlich der Lerninhalt, welcher immer bestmöglich nach neuesten Erkenntnissen der Didaktik aufbereitet werden soll. In technischer Hinsicht gilt oft

[20] Web: HTTP://WWW.NCIRL.IE/RESEARCH_&_INNOVATION/REALT/SUMMER_SCHOOL_2006

„einfacher ist besser“, da sehr aufwändiger Inhalt oft schwierig zu warten ist und nicht gewährleistet werden kann, dass dieser für jeden Lernenden jetzt und auch später technisch überhaupt verwendet werden kann. Weiters soll auf Dokumentformate gesetzt werden, von denen man annehmen kann, dass sie auch in einigen Jahren noch verbreitet sein werden.

2. Die Forderung nach guter Dokumentation gilt immer. In der beobachteten Praxis ist leider oft festzustellen, dass Dokumentationen von Entwicklern als lästige Pflicht empfunden werden. Nicht unähnlich ist die Situation bei E-Learning-Material, insbesondere die Metadaten betreffend. Eine wohldokumentierte Klassenbibliothek entspricht einer geordneten und mit Metadaten versehenen Bibliothek von Lernmodulen.

3. Software, vor allem solche, die mit den Benutzern einen hohen Interaktionsgrad hat, muss einem Usability-Test unterzogen werden. Ein Großteil der „Legacy-Software“, obwohl aus syntaktischer und semantischer Sicht noch geeignet, muss deshalb umgeschrieben werden, weil sie den mittlerweile gestiegenen Anforderungen in den GUIs („Graphical User Interface“) nicht mehr gerecht wird. Die Situation bei E-Learning ist nicht viel anders. Etwas erschwerend kommt allerdings hinzu, dass die Art der Interaktivität stark vom Lernmodell und dem gewählten bzw. geplanten Lernarrangement abhängt. Bei einer Blended-Learning-Situation kann der Vortragende bzw. ein Coach zusätzliche Kommunikationskomponenten vor Ort einbringen und damit allenfalls GUI-Schwächen oder mangelnde Interaktionsfähigkeit eines Lernpaketes abfedern. Daher muss vor einer Portierung, falls diese unabdingbar erscheint, die Interaktivität der Module in einem Usability-Test geprüft werden, um den verlorenen Aufwand gering halten zu können.

4. Software, die eine klare Unterteilung in Module mit wohldefinierten Abhängigkeiten aufweist, ist einem Redesign besser zugänglich. Bei Lernsoftware betrifft eine Überarbeitung neben rein technischen Aspekten vor allem auch die Lernpfade, die aus den Abhängigkeiten der einzelnen Module („Learning Objects“) abgeleitet werden.

5. Software, die im Sourcecode Sprachdialekte verwendet, ist kaum zu portieren. Dies kommt insbesondere dann zum Tragen, wenn automatische Portierungshilfen verwendet werden sollen oder könnten. So wie in der Programmiersprache C das ANSI C ein standardisierter und kompatibler gemeinsamer Nenner ist, ist es auch bei E-Learning wichtig, die Einhaltung von akkreditierten oder De-facto-Standards zu forcieren. Dadurch wird eine weitgehende Trennung von Lernmaterialien und Plattformen ermöglicht, was einen späteren eventuellen Wechsel des Systems wesentlich erleichtern wird. Weiters können unabhängige Werkzeuge zur Erstellung der Materialen verwendet werden.

   Es wurde darauf hingewiesen, dass bei Einhaltung von Standards und bei Vermeidung von plattformproprietären Funktionen viel eher gewährleistet werden kann, sowie dass das erstellte Lernmaterial über einen längeren Zeitraum genutzt und von Lehrenden gewartet werden kann. Beherzigt man Standards auf verschiedenen Ebenen, kann man davon ausgehen, dass bereits erstellte Materialien mit neuen Bedürfnissen migriert werden können und der bereits investierte Aufwand somit nicht verloren ist.

Zusammenfassend kann folgende Binsenweisheit aus dem Bereich Software-Engineering auch für Entwicklung, Weiterentwicklung und Portierung von E-Learningsoftware unter-

strichen werden: Auf Design, Verwendung von Standards und gute Dokumentation kommt es an. In manchen Fällen ist es allerdings ökonomischer, ein Redesign und eine Neuimplementierung erst dann in Betracht zu ziehen, wenn die Anwender dies auch explizit fordern.

Die verwendete Plattform spielt dabei vorläufig nur eine nachrangige Rolle, solange Standards eingehalten werden. Es kann auch als gesichert angenommen werden, dass adaptive Funktionen eine wesentliche Neuerung in Plattformen darstellen werden.

Literatur

Backus, J. W. et al. (o. J.). Revised Report on the Algorithmic Language Algol 60. Verfügbar unter: HTTP://WWW.MASSWERK.AT/ALGOL60/REPORT.HTM [Stand 27.5.2007].

Baumgartner, P., Häfele & H., Maier-Häfele, K. (2003). *Evaluation von Lernplattformen: Verfahren, Ergebnisse und Empfehlungen* (Version 1.3). Verfügbar unter: HTTP://WWW.BILDUNG.AT/STATISCH/BMBWK/LERNPLATTFORMEN_EVALUATION_UND_ERGEBNISSE_1_BIS_3.PDF [Stand 27.5.2007].

Cristea, A.I., Wentzler, A., Heuvelman, E., De Bra, P. M. E. (2006). Adapting SME learning environments for adaptivity. In *Proceedings Sixth IEEE International Conference on Advanced Learning Technologies (ICALT, Kerkrade, The Netherlands, July 5-7, 2006)* (pp. 130-132). IEEE.

De Bra, P., Smits, D. & Stash, N. (2006). *The Design of AHA!* Proceedings of the ACM Hypertext Conference, Odense, Denmark, August 23-25, 2006 (pp. 133). Verfügbar unter: HTTP://AHA.WIN.TUE.NL/AHADESIGN/ [Stand 27.5.2007].

Jensen, K. & Wirth, N. (1991). *Pascal User Manual and Report*. ISO Pascal Standard. Springer-Verlag, 4th ed.

Goldberg, A. (1983). *Smalltalk-80. The Interactive Programming Environment*. Addison-Wesley.

Mühlbacher, J. R. & Hörmanseder, R. (1987). *Modula-2 auf DOS*. Hanser Verlag.

Lindholm, T., Yellin, F. (1999). *The Java(TM) Virtual Machine Specification*. Prentice Hall PTR.

Weber, G., Kuhl, H.-C. & Weibelzahl, S. (2001). Developing adaptive internet based courses with the authoring system NetCoach. In *Proc. of the 8th International Conference on User Modeling (UM2001), Workshop on Adaptive Hypertext and Hypermedia,* Sonthofen, Germany, pp. 35-48.