Die visuelle Programmiersprache Puck – Entwicklung, Erprobung, Reflexion

Lutz Kohl
Fakultät für Mathematik und Informatik
Friedrich-Schiller-Universität Jena
lutz.kohl@gmail.com

Abstract: In diesem Artikel werden wesentliche Erkenntnisse aus der Dissertation des Autors vorgestellt [Ko09]. Puck ist eine visuelle Programmiersprache mit der grundlegende Konzepte imperativer Programmierung erlernt werden können. Bei einer Reflexion zeigte sich, dass existierende Empfehlungen zur Entwicklung interaktiver Lernumgebungen bei der Entwicklung von Puck nachvollzogen werden konnten. 40 Lehrkräfte die Puck im Unterricht erprobten und anschließend einen Fragebogen beantworteten, empfanden das Verringern der Syntaxfehler durch Puck als günstig für den Anfangsunterricht. Weitere Ergebnisse der Befragung werden vorgestellt.

1 Einleitung

Im Informatikunterricht werden verschiedene visuelle (Software-)Werkzeuge eingesetzt. Mit einigen von ihnen wird versucht, die Einführung in die Programmierung zu erleichtern, und zwar durch didaktisch reduzierte Entwicklungsumgebungen, durch das Vermeiden von Syntaxfehlern sowie durch das Bereitstellen motivierender Aufgaben. Puck ist ein solches visuelles Werkzeug zur Einführung in die imperative Programmierung. Die drei genannten Ansätze wurden auch bei der Entwicklung von Puck aufgegriffen. Durch die Einschränkung auf 16 Bausteine und zwei Datentypen bzw. durch die einfach gehaltene Entwicklungsumgebung wird einer Überforderung von Schülerinnen und Schülern entgegengewirkt. Durch die Programmierung mit Bausteinen, die nur in syntaktisch korrekter Weise kombiniert werden können, werden Syntaxfehler verhindert. Durch die Möglichkeit, neben textuellen auch visuelle und akustische Ausgaben zu erstellen, können neben mathematischen auch viele andere motivierende Probleme und Aufgaben bearbeitet werden. Nach der Entwicklung der visuellen Programmiersprache wurde das System an verschiedenen Stellen erweitert und verbessert, Aufgaben und Materialien wurden bereitgestellt und Puck wurde mit einer größeren Anzahl von Lehrkräften im Unterricht erprobt. Im Abschnitt 2 wird über die Entwicklungen an der visuellen Programmiersprache reflektiert, indem überprüft wird, inwieweit die von Arnold und Hartmann vorgeschlagenen pragmatischen Empfehlungen zur Entwicklung von interaktiven Lernumgebungen auch im Entwicklungsprozess von Puck nachvollzogen werden können [AH07]. Im Abschnitt 3 werden die Ergebnisse einer Erprobung des Puck-Systems im Rahmen einer Untersuchung zu kompetenz-orientiertem Informatikunterricht präsentiert. Abschließend werden im Abschnitt 4 Schlussfolgerungen bezüglich der genannten Empfehlungen und Puck gezogen.

2 Einordnung der Entwicklungsarbeiten an Puck

Zehn Empfehlungen zur Entwicklung interaktiver Lernumgebungen wurden von Arnold und Hartmann vorgestellt und am Beispiel des Systems InfoTraffic veranschaulicht [AH07]. Eine empirische Erprobung von InfoTraffic erfolgte nicht ([Ar07] S. 74 ff.). Somit ist derzeit auch keine Aussage darüber möglich, ob das Umsetzen dieser Empfehlungen zu einer im Unterricht erfolgreich eingesetzten interaktiven Lernumgebung führt. Um dem zu begegnen, soll im Folgenden für jede der zehn Empfehlungen reflektiert werden, inwieweit diese auch bei der Entwicklung des Puck-Systems von Bedeutung waren. Anschließend wird im Abschnitt 3 dargestellt, inwieweit sich Puck im Unterricht bewährt hat.

1. Braucht es dazu überhaupt den Computer? Für Arnold und Hartmann macht es keinen Sinn, Lernumgebungen für Themen zu entwickeln, die genau so gut oder besser ohne Computer vermittelt werden können [AH07]. Speziell interessant für interaktive Lernumgebungen sind ihrer Meinung nach deshalb gerade abstrakte und schwierige Themen, bei denen im Unterricht durch Individualisierung den unterschiedlichen Kenntnissen und Lerntempi Rechnung getragen werden kann.

Programmieren kann als abstraktes und schwieriges Thema bezeichnet werden, auch unterschiedliche Vorkenntnisse und Lerntempi sind im Programmierunterricht nicht ungewöhnlich ([Ko09] S. 73 ff.). Programme, als Ergebnisse der Programmierung werden von Computern ausgeführt. Deshalb ist eine Einführung in die Programmierung mit Computern, bei der Schülerinnen und Schüler individuell an Programmen arbeiten können, durchaus sinnvoll.

2. Ist das Unterrichtsthema auch in 10 Jahren noch relevant? Arnold und Hartmann stellen dar, dass der Aufwand für das Entwickeln interaktiver Lernsoftware im Allgemeinen so groß ist, dass er sich nur für längerfristig relevante Themen rechtfertigt [AH07].

Die Einführung in die „Algorithmik“ wird bereits seit Mitte der 1970er Jahre im Informatikunterricht vermittelt und ist auch heute noch aktuell ([Ba96] S. 113; [Hu04] S. 51; [Th99] S. 18 ff.). Somit ist dieser Bereich in dem sich ständig wandelnden Fach Informatik einer der kontinuierlichsten. Aufgrund der Einordnung von „Algorithmisierung“ als fundamentale Idee der Informatik kann davon ausgegangen werden, dass dieser Inhalt auch in 10 Jahren noch relevant sein wird ([SS04] S. 89 f.).

3. Use-Cases: Ist Interaktivität möglich? Arnold und Hartmann sind der Meinung, dass ein hoher Grad an Interaktivität ausschlaggebend für die Qualität einer computergestützten Lernumgebung ist [AH07]. Da sich nicht jedes Thema für eine Mensch-Maschinen-Interaktion eignet, schlagen sie vor, bereits in der Spezifikationsphase eines Projektes Musteraufgaben zusammenzutragen, welche mithilfe der geplanten Lernumgebung bearbeitet werden sollen.

Aufgrund dieser Musteraufgaben kann ihrer Meinung nach abgeschätzt werden, ob das System zu einer wirklichen Interaktion mit den Lernenden führt und ob verschiedene kognitive Stufen angesprochen werden.

In der Taxonomie von Schulmeister kann die visuelle Programmiersprache Puck auf Level V („the ultimate level of interactivity“) eingeordnet werden, da mit Puck neue Programme erstellt werden können [Sc03]. Da Algorithmen zum größten Teil bereits computergestützt unterrichtet wurden, stellte sich beim Entwickeln des Puck-Systems die Frage nach der Eignung dieses Themas für die Mensch-Maschinen-Interaktion nicht. Beispiele für Aufgaben, die zur Einführung in die Programmierung genutzt werden können, standen schon vor der Entwicklung von Puck in großer Anzahl zur Verfügung (z. B. [Fo89]). Insbesondere bei der Überlegung, die Datentypen auf Integer und Boolean zu beschränken, wurde geprüft, ob dies die Anzahl der zu bearbeitenden Aufgaben nicht zu sehr einschränken würde. Durch die zusätzlich entwickelten Bausteine Color, Rectangle/Ellipse, Sleep, Sound und Random wurde es ermöglicht, eine Vielzahl von weiteren Aufgaben umzusetzen wie z. B. das in Abbildung 1 dargestellte Bildschirmschonerprogramm. Abbildung 2 zeigt, dass bereits während der Entwicklung von Puck mit konkreten Aufgaben gearbeitet wurde.

4. Paper Based Prototyping Arnold und Hartmann argumentieren, dass es sich bei Lernumgebungen, bei denen der Benutzeroberfläche eine besonders hohe Bedeutung zukommt, lohnt, so lange wie möglich nur auf Papier zu arbeiten und Klebeband, Post-Its, Schere sowie Flipcharts zu nutzen [AH07]. Für das schnelle Erstellen und Variieren von Programmoberflächen empfehlen sie Präsentationswerkzeuge.

Auch bei der Entwicklung des Puck-Systems wurden viele Entwürfe – insbesondere für die Darstellung der Bausteine und deren Anordnung im Arbeitsbereich – auf dem Papier entwickelt. Auch wenn einzelne Design-Entscheidungen nach der Entwicklung des ersten Prototyps verbessert wurden, so kann doch festgestellt werden, dass der Einsatz von Paper-Based-Prototyping zu einer erheblichen Arbeitserleichterung führte. Als einfaches Werkzeug wurden neben den oben genannten auch Folienausdrucke zusammen mit einem Tageslichtprojektor zur Präsentation und Diskussion in einem Seminar von Lehramtsstudenten eingesetzt. Dadurch konnte frühzeitig die Drag and Drop-Funktionalität diskutiert werden. Abbildung 2 zeigt Entwürfe für erste Puck-Bausteine, die mit einem Zeichenprogramm erstellt, auf Folie ausgedruckt, ausgeschnitten und zu einem Programm zusammengefügt wurden, das den größten gemeinsamen Teiler zweier positiver Zahlen berechnet.

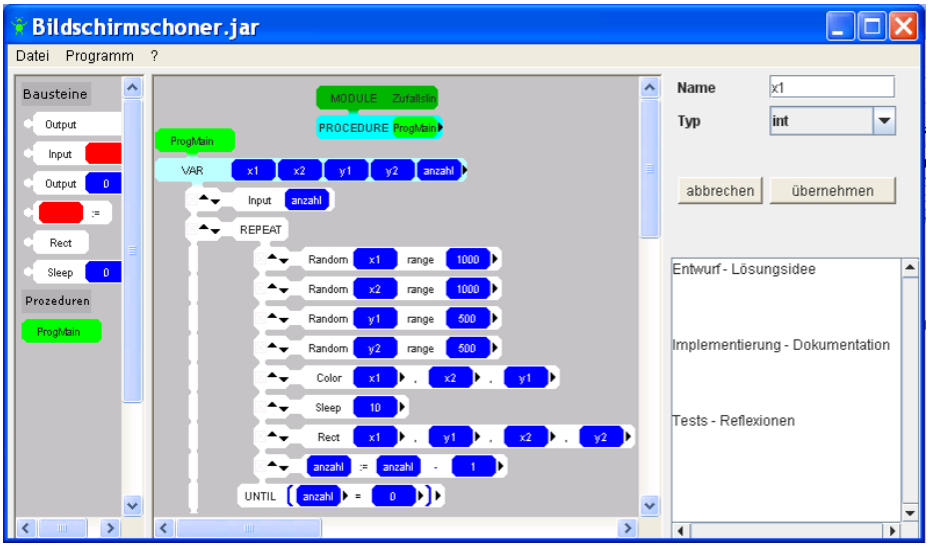

Abbildung 1: Ein aktuelles Puck-Programm, das einen Bildschirmschoner realisiert

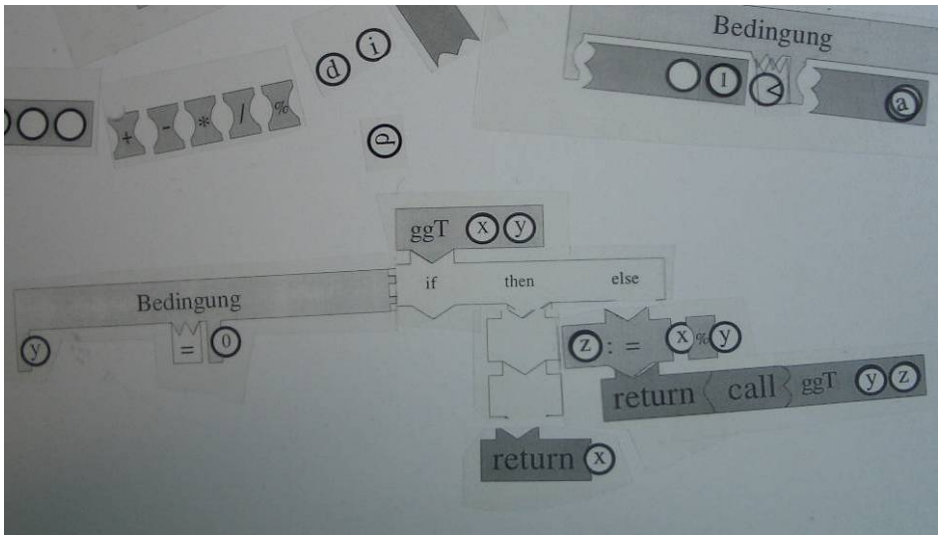

Abbildung 2: Entwürfe für erste Puck-Bausteine auf Folie gedruckt, ausgeschnitten und zu einem Programm zusammengefügt, das den größten gemeinsamen Teiler zweier positiver Zahlen berechnet

5. Rapid Prototyping Arnold und Hartmann empfehlen, einen ersten Prototyp direkt nach dem Festlegen der Benutzungsschnittstelle und der Use-Cases zu implementieren, diesen möglichst schnell einzelnen ausgewählten Testpersonen zu zeigen und ihn anschließend experimentell zu erproben [AH07].

Der in Abbildung 2 dargestellte erste Prototyp des Puck-Systems wurde einer Arbeitsgruppe von Thüringer Informatiklehrern bereits im Jahr 2004, also während der Entwicklung, präsentiert. Nach dem Fertigstellen der ersten Version von Puck erfolgten Tests an zwei Jenaer Schulen im Frühjahr 2005. Dabei konnten viele Anregungen für Verbesserungen und Weiterentwicklungen gesammelt werden. Ein Beispiel sind die Bausteine Color und Rect/Ellipse, die aufgrund der Anforderungen der Lehrpersonen entwickelt wurden.

6. Technische Anforderungen: so einfach wie nur möglich Arnold und Hartmann fordern bei der Entwicklung interaktiver Lernumgebungen Einfachheit und den Verzicht auf unnötige Funktionen, außerdem sollten die Produkte auf verschiedenen Plattformen einsetzbar sein, um den großen Entwicklungsaufwand zu rechtfertigen [AH07].

Bei Weiterentwicklungen an Puck wurde stets versucht, das System so einfach wie möglich zu lassen oder es sogar noch zu vereinfachen. Experimente mit den Datentypen Double und Character wurden aufgrund der Erhöhung der Komplexität der visuellen Darstellung abgebrochen. Das Puck-System wurde in Java implementiert und benötigt lediglich die Java-Laufzeitumgebung (JRE), die für verschiedene Plattformen zur Verfügung steht und auf vielen Computern bereits installiert ist. Das gesamte System befindet sich in einer Datei die auf den Betriebssystemen Windows und Linux ohne Installation direkt gestartet werden kann. Auf spezielle Effekte, die moderne Grafikkarten oder schnelle Prozessoren erfordern, wurde bei der Entwicklung von Puck verzichtet.

7. Frühzeitige Erprobung Arnold und Hartmann empfehlen, erste Tests mit ganzen Schulklassen so früh wie möglich durchzuführen, denn diese Tests zwingen dazu, konkrete Aufgabenstellungen und begleitendes Unterrichtsmaterial auszuarbeiten, und decken in der Regel eine ganze Reihe von Schwachstellen auf, denn Schülerinnen und Schüler denken und handeln oft anders, als Lehrpersonen sich das vorstellen [AH07].

Wie oben bereits dargestellt, wurden schon wenige Monate nach der Entwicklung der ersten Puck-Version erste Tests mit konkreten Aufgabenstellungen in zwei Schulklassen durchgeführt. Die Erfahrungen dieser und verschiedener weiterer Tests zeigten, dass der Empfehlung von Arnold und Hartmann vollkommen zugestimmt werden kann: Durch den Einsatz im Unterricht konnten Schwachstellen aufgedeckt und beseitigt werden. Bestehende Aufgabenstellungen zur Einführung in die Programmierung konnten bei der Arbeit mit Puck teilweise übernommen werden. Weiterhin wurden frühzeitig Aufgaben und begleitendes Unterrichtsmaterial entwickelt (siehe nächste Seite).

8. Sparsamkeit bei der Benutzerschnittstelle Die Benutzeroberfläche einer Lernumgebung soll im Idealfall selbsterklärend sein, deshalb fordern Arnold und Hartmann diese nach den ersten Tests einem Reviewprozess zu unterwerfen, bei dem für jeden Button, jede Beschriftung usw. kritisch hinterfragt wird, ob dieses Element wirklich benötigt wird und ob es an der richtigen Stelle platziert ist [AH07].

Nach den ersten Tests des Puck-Systems fand der von Arnold und Hartmann geforderte Review- und Verbesserungsprozess statt. Dabei wurden z. B. das Vereinbaren von globalen Variablen sowie die Möglichkeit, innerhalb einer Prozedur Unterprozeduren zu deklarieren, entfernt, denn davon wurde im Unterricht kein Gebrauch gemacht. In der ersten Puck-Version war das Ausführen der Programme noch recht umständlich. Zu einem visuell erstellten Programm musste das zugehörige textuelle Programm gespeichert, mit einem externen Editor geöffnet und dort kompiliert sowie ausgeführt werden. Mit der Zielstellung, im Einführungsunterricht ausschließlich Puck zu verwenden, wurde das System so weiterentwickelt, dass es nun möglich ist, die visuell entwickelten Programme direkt auszuführen. Durch diese Auflösung der Abhängigkeit von einer textuellen Programmiersprache wurde die Benutzung des Systems vereinfacht, Menüeinträge konnten entfernt werden.

9. Verbreitung der Lernumgebung Arnold und Hartmann fordern, um überhaupt eine Wirkung der Lernumgebung im Unterricht zu erreichen, diese Schülerinnen und Schülern, Lehrkräften und anderen Benutzern zur Verfügung zu stellen [AH07]. Dabei sollten ihrer Meinung nach verschiedene Verteilerkanäle bedient und didaktische Begleitmaterialien bereitgestellt werden.

Zur Verbreitung des Puck-Systems wurden folgende Verteilerkanäle genutzt:

- Puck wurde Lehrkräften in regionalen Fortbildungen präsentiert.
- Puck wurde in Fachzeitschriften der Didaktik der Informatik vorgestellt.
- Puck wurde auf Fachtagungen der Didaktik der Informatik und auf Messen präsentiert.
- Für das Puck-System wurde eine umfangreiche Webseite entwickelt: www.ipuck.de [Zugriffsdatum: 21.05.2009]
- Über die in Abschnitt 3 vorgestellte Untersuchung wurden mehr als 500 Schulen in Deutschland, Österreich und der Schweiz per E-Mail informiert. In dieser E-Mail wurde auch auf das Puck-System hingewiesen.

Um die Verbreitung von Puck zu ermöglichen, wurden folgende didaktische Begleitmaterialien entwickelt und im Internet sowie teilweise in gedruckter Form zur Verfügung gestellt:

- eine Dokumentation bzw. Hilfe-Seiten zum Puck-System
- eine Einführung in Puck mit fünf Beispielen
- eine Unterrichtseinheit zur Einführung in die Programmierung unter Verwendung von Puck
- Arbeitsmaterial zur Untersuchung „Kompetenzorientierter Informatikunterricht mit der visuellen Programmiersprache Puck“ inkl. einer umfangreichen Aufgabensammlung [Ko08]

10. Sicherstellung von Unterhalt und Kontinuität Um den Unterhalt einer Lernumgebung zu garantieren, muss nach Arnold und Hartmann sichergestellt werden, dass nach der Veröffentlichung etwaige Fehler behoben, evtl. Erweiterungen entwickelt und Anpassungen an neue Software-Versionen bereitgestellt werden [AH07]. Weiter argumentieren sie, dass Lehrkräfte für die Einarbeitung in eine neue Lernumgebung sowie für die Aufbereitung des Unterrichtsstoffes Zeit nur dann investieren, wenn abzusehen ist, dass das eingesetzte System über längere Zeit nutzbar bleibt, was insbesondere bei Systemen, die im Hochschulumfeld entstehen, nicht immer sichergestellt ist. Bei der Umsetzung der an die Entwickler herangetragenen Wünsche nach Erweiterungen einer Software empfehlen Arnold und Hartmann vorsichtig zu sein, denn häufige Überarbeitungen und Erweiterungen verunsichern die Benutzer und führen dazu, dass bereitgestellte Unterrichtsmaterialien aktualisiert werden müssen. Außerdem misst sich die Qualität von Lernumgebungen ihrer Meinung nach nicht an der Menge von Features und die Erweiterung des Programmcodes erschwert die Wartung. Deshalb empfehlen Arnold und Hartmann das Motto „Weniger ist mehr“ [AH07].

Das Puck-System wird seit der ersten Veröffentlichung im Jahr 2004 von der Abteilung Didaktik der Fakultät für Mathematik und Informatik an der Friedrich-Schiller-Universität Jena betreut. Fehler wurden behoben, Erweiterungen entwickelt und Anpassungen an neue Softwareversionen, wie der Übergang von Java 1.4 auf Java 5.0, wurden vorgenommen. Seit der Präsentation von Version 2.4 im Sommer 2006 wurden – trotz intensiven Einsatzes im Unterricht – keine größeren Probleme bei der Arbeit mit dem Puck-System festgestellt. Bei der Entwicklung wurde stets nach dem oben genannten Motto „Weniger ist mehr“ gehandelt und so wurde Puck nicht voreilig um weitere Datentypen oder Bausteine erweitert. Puck wurde von Anfang an unter der GNU General Public License entwickelt [FSF91]. Damit wurden die Voraussetzungen dafür geschaffen, dass das System in der Zukunft durch eine Gruppe von interessierten Nutzern weiter betreut werden kann.

Ergebnis der Analyse
Es konnte festgestellt werden, dass alle zehn von Arnold und Hartmann vorgestellten Empfehlungen zur Entwicklung von interaktiven Lernumgebungen auch bei der Entwicklung der visuellen Programmiersprache Puck nachvollzogen werden konnten. Im folgenden Abschnitt soll dargestellt werden, inwieweit sich Puck für den Einsatz im Unterricht eignet. Kann durch die Untersuchungsergebnisse gezeigt werden, dass das Puck-System, bei dessen Entwicklung die Empfehlungen von Arnold und Hartmann nachvollzogen werden konnten, eine im Unterricht überwiegend erfolgreich eingesetzte interaktive Lernumgebung ist, so wäre dies ein Beleg für die Sinnhaftigkeit der genannten Empfehlungen, der dazu führen könnte, dass neue interaktive Lernumgebungen zielgerichteter entwickelt werden können.

3 Ergebnisse einer Erprobung der visuellen Programmiersprache Puck im Unterricht

Zur Erprobung der visuellen Programmiersprache Puck, eines Kompetenzmodells und zugehöriger Aufgaben wurden Informatiklehrkräfte aus Deutschland, Österreich und der Schweiz aufgerufen, sich im ersten Halbjahr des Schuljahres 2007/2008 an einer Untersuchung zu beteiligen. 84 Lehrerinnen und Lehrern meldeten sich für die Untersuchung in den Klassenstufen 8-10 an. Ihnen wurden Kompetenzmodell, Aufgaben und Puck zur Erprobung im Unterricht zur Verfügung gestellt ([Ko08]; [Ko09] S. 149ff.). 40 Lehrpersonen füllten am Ende der Untersuchung einen Online-Fragebogen aus. Die Ergebnisse der Befragung in Bezug auf Puck werden nachfolgend präsentiert.

Verringern von Syntaxfehlern durch Puck

- Alle befragten Lehrpersonen gaben an, dass das Verringern der Syntaxfehler durch Puck günstig für den Anfangsunterricht war.
- Die Hälfte der Lehrkräfte hatte durch die Arbeit mit der visuellen Programmiersprache mehr Zeit für die individuelle Förderung einzelner Lernender.
- 90,0 % der Lehrpersonen stimmten der Aussage zu, dass auch leistungsschwächere Schülerinnen und Schüler durch die Arbeit mit dem Puck-System Erfolge erzielen konnten.

Insgesamt werden die Angaben der Lehrpersonen so interpretiert, dass durch das Verhindern von Syntaxfehlern in Puck und die damit einhergehende bessere Betreuung durch die Lehrkraft auch leistungsschwächere Lernende bessere Ergebnisse im Informatikunterricht erzielen können. Dies ist vor allem dann wichtig, wenn Mindeststandards umgesetzt und somit alle Schülerinnen und Schüler in Deutschland Kompetenzen informatischer Bildung erwerben sollen, gleich, welcher Herkunft sie sind, welchen sozialen Hintergrund sie haben und welche möglichen Behinderungen sie aufweisen ([GI08] S. 3).

Motivation durch die Arbeit mit Puck

- Lediglich 15,0 % der Lehrpersonen, stimmten der Aussage zu, dass sich für viele Schülerinnen und Schüler die Arbeit mit dem Puck-System als schwierig erwies. 80,0 % der Lehrkräfte waren gegenteiliger Ansicht.
- Wie Abbildung 3 zeigt, waren 77,5 % der Lehrerinnen und Lehrer der Meinung, dass das Puck-System Schülerinnen und Schüler motiviert, sich in den Informatikunterricht aktiv einzubringen.
- Der Einsatz der visuellen Programmiersprache Puck motivierte 90,0 % der befragten Lehrkräfte für die Unterrichtsvorbereitung und -durchführung.
- Der Aussage, dass der Einarbeitungsaufwand in das Puck-System für Informatiklehrkräfte sehr hoch ist, stimmte lediglich ein Lehrer (2,5 %) zu.
- 87,5 % der Lehrkräfte wollten Puck im nächsten Schuljahr wieder einsetzen.

Ein einfaches motivierendes Programmiersystem im Unterricht ist nicht nur für Lehrende und Lernende attraktiv, es kann auch dazu beitragen, dem Fachkräftemangel in der Informatik entgegenzuwirken, indem es das Interesse der Schülerinnen und Schüler für diese Disziplin bereits in der Schule weckt. Die Antworten der Lehrkräfte geben Anlass zur Annahme, dass Puck ein solches motivierendes Werkzeug ist. Da die Lehrkräfte diejenigen sind, die über einen Einsatz im Unterricht bestimmen, sind deren Einschätzungen wichtig für die Verbreitung eines Werkzeuges.

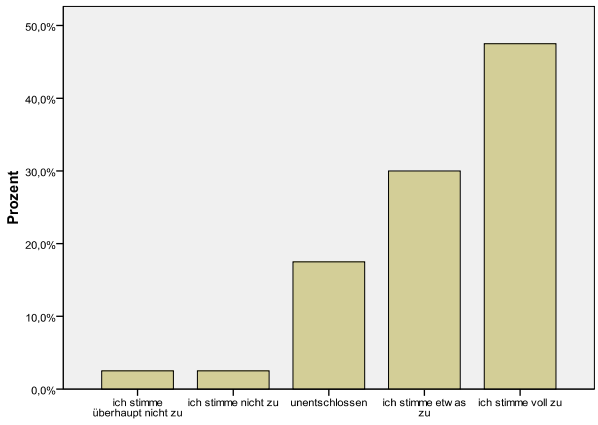

Abbildung 3: Einschätzung der Lehrpersonen zu der Aussage „Das Puck-System motiviert die Schülerinnen und Schüler, sich in den Informatikunterricht aktiv einzubringen."

Unterrichtsmaterialien zu Puck

- Der Aussage, dass es nicht genügend geeignete Unterrichtsmaterialien für den Einsatz des Puck-Systems gibt, stimmten 40,0 % der Lehrpersonen zu, 52,5 % der Lehrkräfte hielten die vorhanden Materialien für ausreichend, 7,5 % waren unentschlossen.

Die Forderung nach weiteren Unterrichtsmaterialien kann bei einer größeren Anzahl an Lehrkräften festgestellt werden. Fraglich ist, ob alle diese Lehrenden auch alle vorhandenen Materialien kannten. Um diese besser zu verbreiten wurden sie während der Untersuchung auf der Internetseite www.ipuck.de bereitgestellt. Weiterhin wurde es auf dieser Webseite ermöglicht, Materialien und Programme zur visuellen Programmiersprache auszutauschen. Zusätzlich könnten weitere Unterrichtsmaterialien entwickelt werden.

Frühstmöglicher Einsatz von Puck

- Die Frage, ab welcher Jahrgangsstufe Puck frühestens eingesetzt werden kann, wurde differenziert beantwortet. 60,0 % der Lehrpersonen halten einen Einsatz ab Jahrgangsstufe 8 bzw. 9 für sinnvoll. 35,0 % könnten sich vorstellen, Puck bereits in der 7. Jahrgangsstufe oder sogar noch früher einzusetzen.

- Ein Lehrer, der die bereitgestellten Materialien in der 7. Klasse eingesetzt hatte, gab an, dass das Prinzip und das System Puck sehr gut ankamen.

Eine Erprobung des Systems in den Jahrgangsstufen 5 bis 7 wäre somit ein Ansatz für weitere Forschungsarbeiten.

Anmerkungen zu Puck

Fünf Lehrpersonen machten explizit positive Anmerkungen zu Puck. Weiterhin formulierten die Lehrkräfte viele zusätzliche Funktionswünsche. Die meisten der genannten Vorschläge wie z. B. die Möglichkeit Bausteine zu kopieren, das Generieren von Struktogrammen sowie das Umsetzen von Funktionen, Real- und String-Variablen, waren schon bei der Weiterentwicklung von Puck diskutiert und unter Beachtung der Forderung nach Einfachheit und dem Verzicht auf unnötige Funktionen nicht umgesetzt worden (vgl. Abschnitt 2). Dem Vorschlag einiger Lehrkräfte entsprechend wurde im Anschluss an die Untersuchung eine Linux-Version von Puck implementiert. Weitere Anregungen sollten bei der zukünftigen Weiterentwicklung von Puck genauer analysiert werden. Trotz der verschiedenen Vorschläge waren 72,5 % der Lehrkräfte der Ansicht, dass die Funktionalität von Puck für eine Einführung in die Programmierung ausreicht. Dies spricht dafür, das Grundkonzept der visuellen Programmiersprache beizubehalten.

Vorbereitung auf textuelles Programmieren durch Puck

- Lediglich fünf Prozent der Lehrpersonen gaben an, dass den Schülerinnen und Schülern durch die Arbeit mit dem Puck-System ein falsches Bild von Programmierung vermittelt wird. Sie begründeten diese Einschätzung nicht. 70,0 % der Lehrkräfte waren gegenteiliger Meinung.
- 60,0 % der Lehrkräfte waren der Auffassung, dass die Schülerinnen und Schüler durch den Unterricht mit dem Puck-System auf die Arbeit mit einer textuellen Programmiersprache gut vorbereitet worden sind.

Die Lehrkräfte gaben hier nur eine persönliche Einschätzung ab, konnten sich aber nicht sicher sein, ob diese Einschätzung auch wirklich zutrifft, da sie nicht auf entsprechende Erfahrung mit einem Übergang zu einer textuellen Programmiersprache zurückgreifen konnten. Trotzdem weisen die Einschätzungen der Lehrpersonen zu den beiden zuletzt analysierten Aussagen darauf hin, dass es durch die Arbeit mit Kontrollstrukturen, Anweisungen und Prozeduren in Puck möglich ist, wesentliche strukturelle Grundlagen für das textuelle Programmieren zu legen.

Im Januar 2009 wurden die beteiligten Lehrpersonen noch einmal bezüglich ihrer Erfahrungen zum Übergang von Puck zu einer textuellen Sprache befragt. Vier Lehrkräfte antworteten auf die Anfrage. Sie berichteten, dass es keine Probleme beim Übergang gab. Drei Lehrpersonen hatten in der 9. bzw. 10. Klassenstufe die textuellen Sprache JavaScript im Anschluss an die Arbeit mit Puck eingeführt, eine Lehrkraft hatte im Anschluss an die Arbeit mit Puck die Programmiersprache Oberon in der 11. Klassenstufe eingesetzt. Alle vier Lehrpersonen gaben an, dass sich die Erfahrungen der Schülerinnen und Schüler mit Puck lediglich positiv und in keiner Weise negativ auf den Unterricht mit der textuellen Sprache auswirkten.

Im Einzelnen gaben die Lehrpersonen Folgendes zur Auswirkung der Erfahrungen mit Puck beim Erlernen einer textuellen Programmiersprache an:[1]

Lehrkraft 1: Die algorithmische Denkweise wurde nicht durch die großen Schwierigkeiten, die die Schüler mit der Syntax von JavaScript haben, überdeckt. Die Schüler waren besser in der Lage, die Ebenen Programmstruktur/Algorithmus und Syntax auseinanderzuhalten und in ihrer Denk- und Argumentationsweise von einer Ebene in die andere zu wechseln.

Lehrkraft 2: Die Schüler verstehen nach meinen Beobachtungen die Strukturen der neu erlernten Programmiersprache (Variablenzuweisung, Bedingungen, Schleifen, etc.) schneller, weil sie diese von Puck schon kennen. Ich sehe keine negativen Auswirkungen. Die Schüler müssen sich mit dem Erlernen der Schreibweisen genauso konzentrieren wie es diejenigen, die kein Puck hatten, auch tun müssen.

Lehrkraft 3: Der Begriff des Algorithmus und seine Merkmale sowie Kontrollstrukturen für die Steuerung von Programmabläufen waren nach der Arbeit mit Puck bereits bekannt. Wiedererkennungseffekte traten auf. Neue Datentypen wurden gut akzeptiert, da sie die Programmier-/Lösungsmöglichkeiten erweiterten.

Lehrkraft 4: Es gab keinerlei Probleme beim Umstieg von Puck zu der textuellen Sprache. Das Verständnis der Algorithmenstrukturen konnte erreicht werden. Durch das Generieren des Oberon-Quelltextes aus der Puck-Lösung wurden von den Schülern Zusammenhänge erkannt. Jetzt können sich die Schüler auf Besonderheiten der Syntax, Fehlermeldungen, Einbindung von Import-Modulen und neue Operationen konzentrieren.

4 Schlussfolgerungen und Fazit

Im Abschnitt 2 wurde gezeigt, dass die von Arnold und Hartmann gegebenen Empfehlungen zur Entwicklung von interaktiven Lernumgebungen auch im Entwicklungsprozess von Puck nachvollzogen werden konnten. Im Abschnitt 3 wurden die weitgehend positiven Ergebnisse der Erprobung von Puck im Rahmen einer Untersuchung vorgestellt. Insgesamt konnten die Empfehlungen von Arnold und Hartmann somit an einer im Unterricht erfolgreich eingesetzten interaktiven Lernumgebung (bzw. visuellen Programmiersprache) nachvollzogen werden. Dies wird als Beleg für die Sinnhaftigkeit der Empfehlungen interpretiert, der dazu führen sollte, dass diese auch bei der Entwicklung weiterer interaktiver Lernumgebung beachtet werden.

Die generell positiven Einschätzungen der 40 Lehrerinnen und Lehrer, die am Ende der Untersuchung den Fragebogen beantworteten, gaben keinen Anlass grundlegende Veränderungen an Puck vorzunehmen. Die Programmiersprache eignet sich offensichtlich für eine Einführung in die Programmierung in den Klassenstufen 8-10. Die Angaben von vier Lehrpersonen deuten darauf hin, dass ein Übergang von Puck zu textuellen Programmiersprachen möglich ist und positive Effekte haben kann.

[1] Die stichpunktartigen Angaben der Lehrkräfte 3 und 4 wurden hier zu Sätzen vervollständigt.

Literaturverzeichnis

[AH07] Arnold, Ruedi ; Hartmann, Werner: Pragmatische Empfehlungen zur Entwicklung von interaktiven Lernumgebungen. S. 171–182. Gesellschaft für Informatik e.V., 2007.

[Ar07] Arnold, Ruedi: Interactive Learning Environments for Mathematical Topics. PhD thesis, ETH Zürich, 2007.

[Ba96] Baumann, Rüdeger: Didaktik der Informatik. Klett, 1996.

[Fo89] Fothe, Michael: 80 Programme in Turbo-Pascal. Akademie der Wissenschaften der DDR Berlin, 2. Auflage, 1989.

[Fo02] Fothe, Michael: Problemlösen mit Python. Thüringer Institut für Lehrerfortbildung, Lehrplanentwicklung und Medien, 2002.

[FSF91] Free Software Foundation: Gnu General Public License. Internetdokument, 1991. – URL [Zugriffsdatum: 21.05.2009]: http://www.gnu.org/licenses/old-licenses/gpl-2.0.txt

[GI08] GI: Grundsätze und Standards für die Informatik in der Schule – Bildungsstandards Informatik für die Sekundarstufe I. In: LOG IN (2008), Nr. 150/151, Beilage.

[Hu04] Hubwieser, Peter: Didaktik der Informatik Grundlagen, Konzepte, Beispiele. Springer Verlag, 2. Auflage, 2004.

[Ko08] Kohl, Lutz: Arbeitsmaterial zur Untersuchung "Kompetenzorientierter Informatikunterricht mit der visuellen Programmiersprache Puck". In: Jenaer Schriften zur Mathematik und Informatik 4 (2008). – URL [Zugriffsdatum: 21.05.2009]: http://www.minet.uni-jena.de/preprints/kohl_08/KohlArbeitsmaterialUntersuchung.pdf

[Ko09] Kohl, Lutz: Kompetenzorientierter Informatikunterricht in der Sekundarstufe I unter Verwendung der visuellen Programmiersprache Puck. Dissertation, Jena, 2009. – URL http://www.db-thueringen.de/servlets/DerivateServlet/Derivate-17565/Dissertation.pdf [Zugriffsdatum: 27.05.2009]

[SS04] Schubert, Sigrid ; Schwill, Andreas: Didaktik der Informatik. Spektrum Akademischer Verlag, Berlin, 2004.

[Sc03] Schulmeister, Rolf: Taxonomy of Multimedia Component Interactivity A Contribution to the Current Metadata Debate. Studies in Communication Sciences. 3 (1) S. 61–80, 2003.

[Th99] Thüringer Kultusministerium: Lehrplan für das Gymnasium Informatik. Internetdokument, 1999. – URL [Zugriffsdatum: 21.05.2009]: http://www.thillm.de/thillm/pdf/lehrplan/gy/gy_lp_if.pdf