10/2003

Das Multimedia-Magazin für Österreichs Schulen

www.cd-austria.at

CD Austria

Sonderheft des bm:bwk

Schulinformatik in Österreich, quo vadis?

✦ *Handreichung für den Unterricht*

Herausgegeben von Prof. Mag. Peter Micheuz

bm:bwk

Hefte können, solange der Vorrat reicht, kostenlos beim Verlag nachbestellt werden: *redaktion@cd-austria.at*

bm:bwk

CD Austria

Zum Geleit

Aus Sicht der Auftraggeber erscheint das zielgerichtete Aufgreifen der inhaltlichen Diskussion zum Stellenwert der Schulinformatik unter Einbezug der Gegebenheiten in den benachbarten Ländern Deutschland und Schweiz für zweckdienlich, als gerade in Zeiten des dynamischen Wandels die Suche nach bleibenden bzw. bestimmenden Bildungswerten im Fachbereich Informatik der Orientierung dienlich sein kann. Genauso wenig wie die in breiten Bildungskreisen geforderte und in Österreich curricular verankerte Computernutzung in den Schulen mit dem eigentlichen Fach- und Wissensanspruch der Informatik gleichgesetzt werden kann, so wenig relevant sind im Grunde genommen auch Modetrends wie E-Learning oder die von der Wirtschaft an das Bildungswesen herangetragenen Zertifizierungen für die eigentlichen Themen einer Kerninformatik.

Herr Prof. Mag. Peter Micheuz (Gymnasium Völkermarkt) ist an der Universität Klagenfurt für die fachdidaktischen Veranstaltungen des Lehramtsstudiums Informatik zuständig und wurde mit der Redaktion und Herausgabe eines weiteren Sonderheftes der Computerfachzeitschrift CD Austria unter dem Titel „Schulinformatik in Österreich – quo vadis?“ beauftragt. Die vorliegende Auswahl der Beiträge namhafter Experten sowie Wissenschaftler aus dem In- und Ausland repräsentiert nicht nur einen inhaltlichen Querschnitt zur gestellten Frage, sondern zeigt auch auf, dass die Schulinformatik nach wie vor als eigenständiges Fach positioniert und ihr allgemein bildender Wert in turbulenten Zeiten wie den unsrigen erhalten werden muss.

Der Leser/Die Leserin erhält in den Autorenbeiträgen ebenso Aufschluss über die Abgrenzung des Fachbereiches Informatik zu möglichen „Konkurrenzdisziplinen“ wie auch Informationen über die Situation der Informatik bei den deutschsprachigen Nachbarn. Mag. Micheuz vertritt stellvertretend für die Autoren den Standpunkt, dass unverzichtbare Basismodule, ja „fundamentale Ideen der Informatik“ im Sinne der Diktion von Andreas Schwill einen zeitlosen Anspruch sicherstellen können und müssen.

Als Vertreter des Bildungsministeriums bedanken wir uns herzlich für die konzeptionelle und redaktionelle Betreuungsarbeit durch Herrn Prof. Mag. Peter Micheuz, ferner bei allen Autoren für die Erstellung der kompetenten Beiträge, sowie für die erneut unter Beweis gestellte hervorragende Zusammenarbeit mit dem CDA-Verlag bei Herrn Geschäftsführer Harald Gutzelnig. Mögen alle Leserinnen und Lesern viel Freude mit dem vorliegenden CDA-Sonderheft haben und es auch für didaktische Zwecke nutzen können.

Wien, im September 2003

MR Dr. Rudolf Apflauer
Leiter IT-Gesamtkoordination
im bm:bwk

MR Mag. Dr. Anton Reiter
bm:bwk, Sektion IV

Inhaltsverzeichnis

Vorwort des Herausgebers

Zweifelsohne suggeriert die durchaus nicht rhetorisch angelegte Frage „Schulinformatik in Österreich, quo vadis?“ ein wenig den Wunsch nach Orientierung im Schulfach Informatik. Ein Fach, das es mit dieser Bezeichnung genau genommen nur an Gymnasien und bisweilen an höheren technischen Bundeslehranstalten gibt.

Den Königsweg der Schulinformatik gibt es nicht und er wird auch im vorliegenden Sonderheft nicht vorgezeichnet. Ebensowenig werden Sie Patentrezepte für die Durchführung einzelner Unterrichtsstunden finden. Ich kann aber garantieren, dass Sie – angeleitet durch die Vielfalt der Beiträge - Ihren Bildungsauftrag im Fach Informatik oder in anderen informatiknahen Schulfächern zu Ihrem Vorteil reflektieren werden. Gehen müssen Sie die oft unsicheren Wege der Schulinformatik aber schon selber. Es wäre jedoch kein Zufall, wenn Sie nach Lektüre dieser CDA-Sonderausgabe etwas an Trittsicherheit gewinnen würden.

Schulautonome Entwicklungen haben in den vergangenen Jahren nahezu in allen Schultypen eine veritable Inflation von computerbezogenen, schulinformatischen Unterrichtsgegenständen ausgelöst. Dies ist nicht weiter verwunderlich, da vor allem berufsbildende höhere Schulen auf den wachsenden Bedarf an IT-Arbeitskräften durch eine verstärkte informatikbezogene Ausbildung reagiert haben. Das Werben um SchülerInnen unter dem Deckmantel von Notebookklassen, E-Learning-Angeboten und weiteren Spezialangeboten lässt bisweilen den Verdacht einer „Fassadeninformatik“ aufkommen, in der die gut gemeinten, aber manchmal zu hoch gesteckten Ziele nicht erfüllt werden (können).

Von den Ausprägungsformen informationstechnologischen Unterrichts, beispielsweise die Wirtschaftsinformatik und Office Management in den HAKs, Technische Informatik und Programmierkurse in den HTBLAs und Mediendesign in den HLWs, werden Sie auf den folgenden Seiten schon aus Gründen des knappen Platzangebotes kaum etwas erfahren. Die spezifisch informatische Berufsausbildung tritt in dieser Sonderausgabe zugunsten von Überlegungen hinsichtlich grundlegender informatischer Bildung in den Hintergrund.

Ich lade Sie ein, mit mir einen gedanklichen Streifzug durch das junge, dynamische, und noch lange nicht gefestigte Gebiet der Schulinformatik zu unternehmen, in dem sowohl prominente Vertreter aus der Fachdidaktik als auch Schulpraktiker zu Wort kommen werden. Sie können dann ihren eigenen Standpunkt mit der derzeitigen Situation der Schulinformatik vergleichen.

Die folgenden Beiträge sollen Ihnen vor allem für das Fach Informatik, aber auch für alle aus ihr abgeleiteten Gegenstände in diversen Schultypen eine Grundorientierung geben.

Mir liegt viel daran, die permanente Diskussion über kurzlebiges Produktwissen im Gegensatz zu tiefer gehendem und langlebigem Grundlagen- und Konzeptwissen insoweit zu moderieren, als es sich dabei um eine Dialektik handelt, der sich keine Informatiklehrkraft entziehen kann und – wie sie mehrfach in dieser Sonderausgabe erfahren werden – die auch Sie für einen guten Informatikunterricht nutzbringend einsetzen können.

Ich kann nicht beurteilen, ob die Schulinformatik in Österreichs Bildungspolitik besser aufgehoben ist als in anderen vergleichbaren Staaten. Eine diesbezügliche Studie oder objektive Überprüfung in Zeiten von PISA steht noch aus. Jedenfalls ist in den letzten Jahren mit Unterstützung des zuständigen Ministeriums – aber vor allem dank vieler unermüdlicher Autodidakten, Avantgardisten und Einzelkämpfer an den Schulen, denen ich an dieser Stelle meine Anerkennung und Bewunderung aussprechen möchte – aus Österreichs schulinformatischer Landschaft eine bunt blühende Wiese geworden. Gelegentliches Unkraut in Form von unqualifiziertem Informatik-Unterricht soll den insgesamt positiven Gesamteindruck nicht schmälern.

Ebensowenig soll an dieser Stelle verschwiegen werden, dass durch die im kommenden Schuljahr in Kraft tretende Stundenkürzung an vielen Schulen - besonders aber an den AHS durch die Abwahl des Wahlpflichtgegenstandes Informatik und insgesamt durch die Behinderung von Schulentwicklungskonzepten mit verstärktem IT-Angebot - der Schulinformatik ein Bärendienst erwiesen worden ist. Aber das lässt sich korrigieren. Es soll die (schulinformatische) Wiese nicht austrocknen, sondern intensiver kultiviert werden, damit unsere künftigen SchülerInnengenerationen auf eine dynamische Informations- und Wissensgesellschaft optimal vorbereitet werden.

Mag. Peter Micheuz

Muss es gleich ein Pflichtfach sein?

Die Konzeption des neuen Pflichtfachs Informatik an Bayerns Gymnasien

Im Februar 2000 verkündete der bayerische Ministerpräsident Dr. Edmund Stoiber, dass es an Bayerns Gymnasien ab 2003 ein Pflichtfach Informatik für alle Schülerinnen und Schüler geben würde. Das zweistündige Fach wird in den 6. Jahrgangsstufen (Altersstufe: ca. 11 Jahre) aller Gymnasialzweige sowie in den 9., 10. und 11. Jahrgangsstufen des naturwissenschaftlich-technologischen Zweiges (an ca. 50% aller Gymnasien angeboten) eingerichtet werden. Darüber hinaus wird es nun endlich auch reguläre Grund- und Leistungskurse in Informatik (und damit auch ein Abiturfach) geben.

Prof. Dr. Peter Hubwieser
Technische Universität München
Fakultät für Informatik

Natürlich traf der bayerische Ministerrat diese folgenschwere Entscheidung nicht aus dem Stand, sondern auf der Grundlage jahrelanger intensiver Vorarbeiten, die im Wesentlichen von der Technischen Universität München in enger Kooperation mit dem Bayerischen Staatsministerium für Unterricht und Kultus geleistet wurden.

In diesem Artikel soll die Entstehung und die Konzeption dieses neuen Faches kurz skizziert werden. Eine vollständige Darstellung der Lerninhalte würde den Rahmen dieses Beitrags sprengen. Auf meinen Webseiten[1] finden sich viele Unterrichtsbeispiele und Verweise auf konkrete Ausgestaltungen. Die Lehrpläne finden sich auf den Seite[2] des bayerischen Staatsinstituts für Schulpädagogik und Bildungsforschung.

1. Die Entstehung des neuen Faches

Der Informatikunterricht hat in Bayern eine lange und wechselvolle Geschichte hinter sich. Von den ersten Ansätzen in den 60er Jahren über das breite Konzept der fächerintegrierten „informationstechnischen Grundbildung" bis zur Einführung eines Kollegstufengrundkurses „Informatik" Anfang der 80er Jahre wurde zwar viel erreicht, der eigentliche Durchbruch zu einem vollwertigen, regulären Pflichtfach gelang zumindest an unseren ca. 400 Gymnasien jedoch nicht. Dafür hätte man einerseits das argumentative Sperrfeuer vieler Interessengruppen gegen die Informatik im Unterricht unterlaufen und andererseits zusätzliche, starke Argumente gegen die übermächtige Lobby der etablierten Fächer ins Feld führen müssen. Hier seien nur einige der gebetsmühlenhaft wiederholten Argumente gegen ein Fach Informatik aufgeführt:

1. Die Informatik kann keine langlebigen Unterrichtsinhalte vorweisen.
2. Die Bedienung von Anwendungssystemen lernen die Kinder von selbst!
3. Informatik muss im Rahmen anderer Fächer unterrichtet werden, denn: computerunterstütztes Lernen soll sich schließlich nicht nur auf ein Fach beschränken.
4. Wozu müssen alle Schülerinnen und Schüler programmieren lernen? Programmierkurse sind nicht allgemeinbildend!
5. Es gibt keine ausgebildeten Informatiklehrer.

Wer ein Fach Informatik einrichten wollte, musste diese Argumente entkräften. In aller Kürze gelang uns das folgendermaßen (in der Reihenfolge der obigen Argumente):

1. Hier lässt sich durch einen kleinen Blick auf den Kanon der universitären Inhalte leicht das Gegenteil beweisen. Man muss dabei allerdings darauf achten, nicht in die Fallen von rein produktspezifischer Ausbildung (z.B. „Word 2000-Kurs") oder starker Syntaxlastigkeit (bei Programmiersprachen) zu verfallen.
2. Dies gelang durch eine Abgrenzung: Informatikunterricht beschäftigt sich stets mit (zumindest in gewisser Weise) abstrakten Systemen. Konkrete Systeme dienen nur als Mittel bzw. Beispiel zur Veranschaulichung allgemeiner Konzepte und stellen kein eigenständiges Lernziel dar. Wir behandeln z.B. die Dokumentenstruktur von Texten, wofür als Beispiel sowohl das Format von Microsoft Word als auch das von Star-Office dienen kann. Die Menüstruktur eines Werkzeugs kann dagegen niemals der eigentliche Unterrichtsgegenstand sein.
3. Auch hier half eine Abgrenzung: Computerunterstütztes Lernen thematisiert im Gegensatz zum Informatikunterricht keine informatischen Fachkonzepte und kann daher parallel zu diesem in allen anderen Fächern stattfinden. Leider ist noch nicht zu allen Verantwortlichen durchgedrungen, dass sich die beiden Möglichkeiten („Computereinsatz in allen Fächern" bzw. „dedizierter Informatikunterricht") nicht gegenseitig ausschließen, sondern im Gegenteil sehr gut ergänzen.
4. Dieses Argument konnten wir durch die Instrumentalisierung des Programmierens unterlaufen: Programmieren und das Erlernen einer Programmiersprache ist im Informatikunterricht nie Selbstzweck. Programme dienen stets der Implementierung von Modellen, um diese zu simulieren, zu validieren oder

[1] *http://ddi.in.tum.de*
[2] *http://www.isb.bayern.de*

zu veranschaulichen, kurz: „kein Programm ohne Modell!“. Die allgemeinbildende Wirkung von Modellierung ist dagegen seit langem weitgehend anerkannt (Stichwort: Hilfe zur Strukturierung komplexer Systeme, von der Steuererklärung über Reisebuchungen und Eisenbahnverkehr bis zum Börsenhandel).

5. Das letzte Argument packten wir direkt bei den Hörnern: wir führten an der TU München und der FAU Erlangen zwischen 1995 und 1998 drei Schnellkurse durch, in dem wir ca. 100 Lehrkräfte zum Staatsexamen in Informatik führten. Danach gab es zumindest eine ausreichende Anzahl von Informatiklehrern, um eine fachlich qualifizierte Diskussion über das neue Fach führen zu können.

Diese unsere Argumentationslinie beruhte letztlich auf dem didaktischen Konzept, das wir für das neue Pflichtfach erarbeitet hatten. Eine ausführliche Darstellung des Konzeptes findet sich in [Hubwieser 2001]. Durchgehend gemeinsam ist den Konzeptionen aller Jahrgangsstufen, dass wir gleichzeitig Lernerfolge auf drei Schichten erzielen wollen:

1. **Systeme strukturieren:** Mit Hilfe spezieller grafischer Modellierungstechniken (z.B. Datenfluss-, Objekt-, Automatendiagramme) soll die Fähigkeit zur Strukturierung und Beschreibung komplexer Systeme ausgeprägt werden. Im Anfangsunterricht werden ausschließlich Dokumente und automatisierte Abläufe beschrieben (Texte, Grafiken, Programme), später auch andere Systeme aus den Erfahrungswelt (Banken, Firmen, Steuererklärungen, etc.)

2. **Werkzeuge beherrschen:** Die o.g. Modelle werden möglichst sofort nach ihrer Erstellung implementiert bzw. realisiert und damit ausgiebig simuliert und getestet. Dazu werden Hard- und Softwaressysteme (Grafik-, Text- und Hypertextsysteme, Tabellenkalkulationen, relationale Datenbanksysteme, Programmiersprachen, usw.) benutzt, deren Bedienung so in sehr zielgerichteter, aufgabenbezogener Weise ausgiebig eingeübt wird.

3. **Technik verstehen:** Während der Realisierung ergibt sich an vielen Stellen Gelegenheit zu Tiefblicken auf technische Konzepte und Strukturen (Speicher, Prozessoren, Rechnerstrukturen, Betriebssysteme, etc.).

Entscheidend für den Erfolg des Unterrichtskonzeptes ist eine möglichst feingliedrige Verzahnung dieser Ebenen (idealerweise im Rahmen von Unterrichtsprojekten), so dass beinahe ständig auf allen drei Ebenen gleichzeitig gelernt wird.

2. Anfangsunterricht

Der informatische Anfangsunterricht muss sich vor allem gegenüber der reinen Bedienerschulung, wie sie beispielsweise an den Volkshochschulen oder auch in weiten Strecken des ECDL betrieben wird, abgrenzen (siehe Argumente 1 und 2 in Abschnitt 1). Einerseits muss der Unterricht dabei in altersgemäßer Weise sehr handlungsorientiert sein, andererseits müssen die vermittelten Konzepte so langlebig sein, dass sie auch 13 Jahre nach dem Erlernen beim voraussichtlichen Berufseintritt der Schülerinnen und Schüler noch relevant sind. Außerdem müssen sie eine gewissen Übertragbarkeit und allgemein bildende Wirkung besitzen, um mit den Inhalten klassischer Fächer wie Mathematik. Physik, Geografie oder Deutsch konkurrieren zu können. Reine Technikfertigkeiten erfüllen diese Kriterien keinesfalls. Dennoch muss in diesem Jahr auch ein solides begriffliches Fundament für den Computereinsatz in allen anderen Fächern gelegt werden (siehe auch [Hubwieser 2002]).

Diesen Anforderungen begegnen wir mit einer Mischung aus einer an Kreativität orientierten Anwendung von Standardsoftware bzw. speziellen Programmiersystemen (virtuelle Roboter) mit objektorientierter bzw. algorithmischer Modellierung der erzeugten Dokumente und Programme.

Strukturen	Werkzeuge	Aktivitäten
Objekte, Attribute, Klassen, Methoden,	Vektor- und Rastergrafiksysteme	Information grafisch darstellen: Zeichnen und Malen
Beziehungen zwischen Objekten	Textsysteme	Information textuell darstellen: Schreiben, Dichten, Verzieren
Rekursive Beziehungen zwischen Objekten, Bäume	Dateimanager	Information hierarchisieren
Verweise, geflechtartige Strukturen	Hypertextsysteme	Information vernetzen
Automatische Abläufe, Algorithmen	Robotersysteme	Informationsverarbeitung automatisieren

Die Tabelle auf Seite 5 stellt die strukturelle Entwicklung der Lernkonzepte den dazu verwendeten Werkzeugen gegenüber.

Motiviert durch Einsparung von Schreibarbeit wird eine kleine Pseudosprache eingeführt, anhand derer die Schülerinnen und Schüler bereits hier eine typische Arbeitsweise der Informatik kennen lernen: den Entwurf von formalen (künstlichen) Sprachen:

Baum.Füllfarbe = rot

Dieser Anfangsunterricht wird seit dem Schuljahr 2000/01 jährlich in ca. 40 Klassen mit sehr großem Erfolg getestet.

3. Informatische Mittelstufe

In der Mittelstufe (Jahrgangsstufen 9-11) spielt die Modellierung allgemeiner Systeme die Hauptrolle. Dazu lernen die Schülerinnen und Schüler im Lauf dieser drei Jahre eine Reihe von Modellierungstechniken, die jeweils bestimmte Aspekte von Systemen beschreiben, kennen und anwenden. Im Gegensatz zur professionellen Softwareentwicklung kann man jedoch nicht zuerst alle diese Techniken einführen, um danach das erste Programm zu schreiben. Zur Aufrechterhaltung der Motivation sowie zur Einübung der Techniken müssen sich im Gegenteil Modellierung und Programmierung möglichst kleinschrittig abwechseln. Daher haben wir jeder Modellierungstechnik eine bestimmte Implementierungsplattform zur Seite gestellt, mit der bereits das erste Modell realisiert und simuliert werden kann. Im Einzelnen ergibt sich daraus der Ablauf wie in Tabelle unten ersichtlich.

Ganz neu im gesamten Lehrplan der bayerischen Gymnasien ist das große Projekt in der Jahrgangsstufe 11 (Umfang ca. ein halbes Schuljahr), bei dem Aspekte des Projektmanagements im Lehrplan vorgeschrieben werden. Es dient also nicht hauptsächlich zur Vermittlung von anderen Lerninhalten (wie die bisher üblichen Unterrichtsprojekte, die als didaktische Methode verstanden werden), sondern zur Übung in Projektplanung, -management und -durchführung unter besonderer Berücksichtigung der Erfordernisse von Softwareprojekten (Versionsverwaltung, Schnittstellen, Qualitätssicherung etc.). Demzufolge sollte die Projektleitung auch (zumindest überwiegend) von den Schülerinnen und Schülern übernommen werden.

4. Informatik in der Oberstufe

In den neuen 3-stündigen Grund- und 5-stündigen Leistungskursen der 12. und 13. Jahrgangsstufen finden sich Themen, die einerseits im Hinblick auf die Bildungsziele der Informatik am Gymnasium behandelt werden müssen, andererseits in den vorausgegangenen Jahren aus unterschiedlichen Gründen (mangelnde Voraussetzungen etc.) nicht bearbeitet werden konnten. Die in den vorausgegangenen Jahren erlernten und eingeüb-

Jgst.	Modellierung	Beschriebene Systemstrukturen	Implementierungsplattform
9	Funktionale Modelle (Datenflussdiagramme)	Grobstruktur: Komponenten (Black Box), Schnittstellen und Datenflüsse	Tabellenkalkulation (funktionale Darstellung von Termen)
	Datenmodellierung (ER-Modelle)	Statische Feinstruktur: Klassen, Objekte, Attribute und Beziehungen	Relationale Datenbank
10	Zustands-Übergangsmodelle, Algorithmen	Zeitliches Verhalten eines einzelnen Systems: Ablaufstrukturen	Imperative (zuweisungsorientierte) Programmiersprache
	Funktionale Modelle (Datenflussdiagramme), Rekursion	Grobstruktur: Komponenten (Glass Box), Schnittstellen und Datenflüsse	Funktionen und Prozeduren in einer imperativen Programmiersprache
11	Objektorientierte Modellierung	Statische und dynamische Feinstruktur: Klassen, Objekte, Attribute und Beziehungen, Methoden, Interaktionen, Klassenhierarchien	Objektorientierte Programmiersprache
	Kombination aller Techniken in einem großen Projekt	Alle wesentlichen Systemaspekte	Kombination mehrerer Plattformen (z.B: Datenbank und Programmiersprache)

Jgst.	
12	Rekursive Datenstrukturen: Listen, Bäume
	Analyse von Algorithmen: Effizienz, Grenzen der Berechenbarkeit
13	Formale Sprachen: Notation, Syntax, Semantik, Erkennung durch Automaten, Mustererkennung
	Arbeitsweise von Rechenanlagen: Registermaschine, Bussysteme, Systemnahe Programmierung,
	Grundlegende Konzepte von Betriebssystemen: Prozess- und Speicherverwaltung, E/A-Prozesse
	Kommunikation und Synchronisation von Prozessen: Rechnernetze, Topologie, Protokolle und Protokollschichten, Synchronisation nebenläufiger Prozesse, Nichtdeterminismus

ten Modellierungsfertigkeiten spielen bei der Beschreibung der Konzepte nun eine entscheidende Rolle. Der Lehrplan im Überblick ist aus der Tabelle oben ersichtlich.

5. Ausblick

Die Einführung des neuen Pflichtfaches stellte eine sehr mutige Entscheidung der bayerischen Staatsregierung dar. Wie gewagt dieses Unterfangen wirklich ist, zeigte sich bereits in den ersten Jahren der Vorbereitung: Unterrichtsmodelle müssen entwickelt, Schulversuche ausgewertet, Lehrerinnen und Lehrer aus- und weitergebildet werden. Nicht zuletzt muss für eine geeignete technische Infrastruktur an den Schulen gesorgt werden.

Diese Aufgaben belasten die damit betrauten Institutionen bis an ihre Leistungsgrenze, erschwert durch die leeren Kassen der öffentlichen Hände, die eine Umsetzung ohne jegliche zusätzliche Personalausstattung erzwingen.

Inzwischen zeichnet sich jedoch ab, dass dieses neue Fach zwar in den ersten Jahren etwas rau laufen wird, langfristig jedoch für Bayern einen enormen Standortvorteil darstellt und den Freistaat damit auf seinem Weg zu einem international führenden Zentrum für neue Technologien ein gutes Stück weiterbringt.

Literatur

Hubwieser P.: Didaktik der Informatik. Grundlagen, Konzepte und Beispiele. Springer Verlag, Berlin, Heidelberg, 1. Korrigierter Nachdruck April 2001. ISBN 3-540-65564-6

Hubwieser P.: Object Models of IT-Systems Supporting Cognitive Structures in Novice Courses of Informatics. In: van Weert T., Munro R. (Eds.): Informatics and The Digital Society: Social, Ethical and Cognitive Issues, IFIP TC3/WG3.1&3.2 Open Conference on Social, Ethical and Cognitive Issues on Informatics and ICT, July 22-26, 2002, Dortmund, Germany. IFIP Conference Proceedings 244 Kluwer 2003, ISBN 1-4020-7363-1, pp 129-140

Autor

Univ. Prof. Dr. Peter Hubwieser
Leiter des Fachgebietes "Didaktik der Informatik" an der Fakultät für Informatik der TU München, Vorstand des Zentralinstituts für Lehreraus- und Lehrerfortbildung der TU München, Präsidiumsmitglied der Gesellschaft für Informatik (GI)
E-Mail: Peter.Hubwieser@in.tum.de
Homepage: http://ddi.in.tum.de

Hubwieser ist einer der führenden Informatik-Fachdidaktiker in Deutschland und hat mit seinem Konzept, in Bayerns Gymnasien Informatik als Pflichtfach zu verankern, für Aufsehen gesorgt.

Impressum:

Verleger: CDA Verlags- und Handelsges.m.b.H, A-4320 Perg, Tobra 9, ***Herausgeber:*** Univ.Prof. Mag. Peter Micheuz, ***Redaktionsanschrift:*** A-4320 Perg, Tobra 9, Tel.: (+43) 07262/57557, Fax: (+43) 07262/57557-44, *e-mail:* redaktion@cda-verlag.com ***Internet:*** *http://www.cda-verlag.com, http://www.cd-austria.at/, http://www.vollversionen.com,* ***Richtung:*** Das Multimedia-Magazin für LehrerInnen und ErzieherInnen.

Manuskripte und Programme: Es wird keine Haftung für unverlangt eingesandte Manuskripte und Programme übernommen. Die Einsendung von Manuskripten jeder Art gilt als Zustimmung des Verfassers zum Abdruck in den vom Verlag herausgegebenen Publikationen. Der Verlag behält sich das Recht vor, eingesandte Manuskripte nicht zu veröffentlichen. Eine Gewähr für die Richtigkeit der Veröffentlichung kann nicht übernommen werden. Für den Inhalt der Anzeigen haftet ausschließlich der Inserent, eine Prüfung seitens des Verlags erfolgt nicht!

Urheberrecht: Alle in den Publikationen des Verlages veröffentlichten Beiträge sind urheberrechtlich geschützt. Jegliche Reproduktion oder Nutzung bedarf der vorherigen, schriftlichen Genehmigung des Verlages. Der Verlag übernimmt keinerlei Haftung für eventuell auftretende Kosten oder Schäden, welcher Art auch immer. Für den Inhalt der Programme sind die Autoren verantwortlich.

Was unterscheidet ein Auto von einem Computer?

Braucht es in der Schule ein Fach Informatik? Reicht es nicht aus, wenn im Unterricht in den verschiedenen Fächern der Computer als Werkzeug genutzt wird und die Schülerinnen und Schüler nebenbei die für den späteren Berufsalltag notwendigen Fertigkeiten im Umgang mit den Informations- und Kommunikationstechnologien (ICT) erwerben? Autofahren lernen wir ja auch nicht in der Schule! Auf den ersten Blick mag dieser Gedanke überzeugen. Schnell stellt man aber fest, dass es zwischen der Nutzung eines Autos und dem Stellenwert des Computers in der Informationsgesellschaft gewichtige Unterschiede gibt. Will die Schule auf das Leben vorbereiten, dann braucht es zwingend ein Schulfach „Information und Kommunikation".

Werner Hartmann
ETH Zürich

Informatik als Teil der Allgemeinbildung

Die Informatik hat unsere Gesellschaft im Laufe weniger Jahrzehnte verändert wie kaum eine andere Wissenschaft zuvor. Fast jedermann nutzt heute Anwendungen wie Textverarbeitung, Tabellenkalkulation oder Web-Browser. Um diese Computer-Werkzeuge bedienen zu können, benötigt man im Gegensatz zu früher keine Programmierkenntnisse mehr. Die effiziente und effektive Nutzung dieser Werkzeuge ist aber keineswegs einfach. Es genügt nicht, im richtigen Moment die richtige Taste zu drücken. Die Ausbildung orientiert sich aber trotzdem oft genau an einer solchen Shift-Ctrl-F7 Philosophie. Und Zertifikate bescheinigen den Erwerb entsprechender Fertigkeiten. Dabei wissen wir alle, dass es nicht genügt, in der Schule nur das Einmaleins zu üben oder affenartig Prozentsätze von irgendwelchen Zahlen ausrechnen zu können. Mathematische Allgemeinbildung bedingt auch ein Verständnis für die einem Thema zugrunde liegenden Konzepte. Wer nur Prozentsätze berechnen kann, aber nie ein tieferes Verständnis für die Prozentrechnung erworben hat, wird sich als Tourist nichts ahnend über's Ohr hauen lassen, wenn der Kellner zuerst 5% für das Gedeck, auf dem resultierenden Total 15% für den Service und auf dem neuen Total noch 5% für die MwSt draufschlägt. Auch für die effiziente, tägliche Nutzung der Informatik-Technologien braucht es ein Verständnis der grundlegenden Konzepte. Dieses Verständnis kann nicht später in der Hektik des Berufsalltages erworben werden, sondern muss als Teil der Allgemeinbildung in der Schule vermittelt werden. Fächer wie Mathematik, Physik oder Geschichte gehören zu den Grundlagen unserer Bildung. Heute ist der Umgang mit dem Computer neben Lesen, Schreiben und Rechnen eine weitere Kulturtechnik und muss ebenfalls Eingang in der Bildung finden.

Konzeptwissen versus Produktwissen

Die Informatikausbildung hat sich in den vergangenen Jahrzehnten immer stark an der aktuellen Technik orientiert. In eigentlichen Hauruck-Übungen werden immer wieder neue Ausbildungsgänge aus dem Boden gestampft und Zertifikate geschaffen. So ließen sich in den letzten Jahren Tausende von Interessierten zu Web-Designern und Web-Mastern ausbilden und müssen heute mitten in der Dotcom-Krise feststellen, dass ihr spezifisch auf einen Teilbereich der Informatik ausgerichtetes Wissen nicht mehr groß gefragt ist. Die Frage liegt auf der Hand: Wie kann die Informatikausbildung aus diesem Teufelskreis – angetrieben durch kurzfristige Entwicklungen – ausbrechen und langfristig nutzbares Wissen und Können vermitteln? Der Schlüssel zur Lösung liegt in einer stärkeren Gewichtung des Konzeptwissens. Etablierte Schulfächer wie Mathematik oder Physik orientieren sich nicht an kurzfristigem Zweckdenken und trotzdem ist ihre Bedeutung für unsere Kultur und unser Leben unbestritten.

Der Mathematikunterricht beispielsweise verfolgt verschiedene Zielsetzungen. In den niedrigeren Schulstufen werden „Skills" vermittelt. Das Einmaleins, Bruchrechnen, einfache Prozentrechnung, der Vergleich von Größen und weitere Themen gehören zu den Fertigkeiten, die heute von jedem Schulabgänger erwartet werden. Hierzu gibt es eine Parallele in der Informatik. Von heutigen Schulabgängern wird erwartet, dass sie mit einer Textverarbeitung umgehen, per E-Mail kommunizieren und über das Internet Informationen beschaffen können. Die Versuchung ist groß, diese Fertigkeiten einfach im Stil „drill and practice" zu vermitteln. Wir wissen aber inzwischen aus den großen internationalen Vergleichsstudien im Bildungswesen zur Genüge, dass ein rein auf den Erwerb dieser Fertigkeiten ausgerichteter Unterricht keine nachhaltige Wirkung zeigt. Konfrontiert mit einer neuen, nicht trainierten Situation und schon liegt man mit seiner Prozentrechnung daneben. Ein wirklich nachhaltiger Erwerb von Fertigkeiten setzt auch ein Verständnis für die grundlegenden Prinzipien und Zusammenhänge der Mathematik voraus. In der Informatik ist das nicht anders.

Die verschiedenen Stockwerke des Informatik-Turms

Die Struktur einer Wissenschaft umfasst verschiedene Schichten. Jede Schicht baut auf den Errungenschaften der unteren Schichten auf. Bildlich kann man eine Wissenschaft als Turm mit verschiedenen Stockwerken darstellen. Im untersten Teil befindet sich die Theorie, abstrakt und ohne vertieftes Studium unverständlich. Die Versuchung ist groß, die unteren Stockwerke des Turms

zu vernachlässigen und oben einzusteigen. Ganz oben auf der Aussichtsplattform des Turmes hat man eine perfekte Rundsicht. In der Ferne steigt der Dampf des Kühlturms eines AKW empor und am Himmel fliegt ein neues Großraumflugzeug vorbei. Und alles fängt man mit wenigen Knopfdrücken mittels der digitalen Videokamera ein. Hält man aber inne, wundert man sich über die spezielle Form des Kühlturms. Man fragt sich, wie sich ein so schweres Objekt in der Luft fortbewegen kann und wie die vielen Bilder auf kleinstem Raum gespeichert werden. Zur Beantwortung dieser Fragen muss man sich in die unteren Stockwerke des Turmes begeben. Ob man bei der Besichtigung des Turms zuoberst, ganz unten oder irgendwo dazwischen startet, ist unwesentlich. Entscheidend ist einzig, dass man alle Stockwerke besucht. Die Bedeutung der Theorie als Zugang zur Praxis gilt auch für die Informatik, ist aber noch wenig ins Bewusstsein der Gesellschaft eingedrungen. 1995 wurde im Informatik Spektrum der Informatikturm vorgestellt. [Nievergelt, J.: Welchen Wert haben theoretische Grundlagen für die Berufspraxis? Gedanken zum Fundament des Informatik-Turms. Informatik Spektrum, 18(6), 342-344 (1995)]

Ein riesiger Oberbau ist das Einzige, das die große Mehrheit der Informatikanwender aus der Ferne überhaupt sieht. Hier werden Informatiksysteme zur Lösung konkreter Aufgaben eingesetzt, hier weht die Werbung, hier fließt das Geld.

Im zweitobersten Stockwerk ist eine moderne Fabrik untergebracht, mit ihren Managern, Projektleitern, Software- und Hardware-Ingenieuren. Große Hardware- und Software-Komponenten, halb- oder ganzfertige Bausteine, werden eingeliefert und gemäß voluminösen Spezifikationen zu Standardsystemen zusammengesetzt.

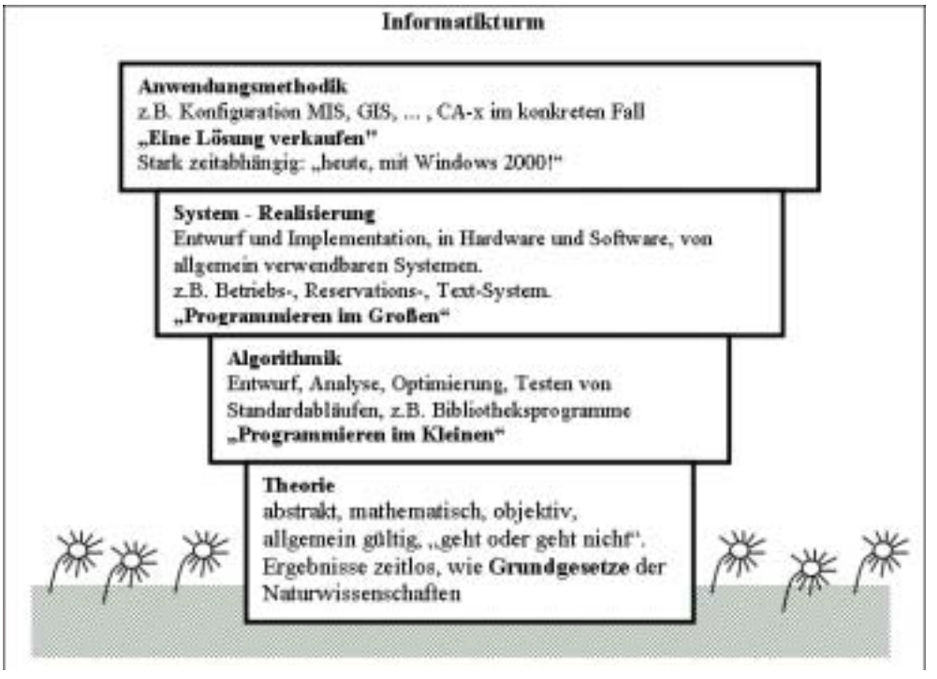
Gleich darunter ist der unscheinbare erste Stock sichtbar. Das Schild an der Haustür verkündet „al Khowarizmi". Sein Name und Werk inspirierte die Wortschöpfungen Algorithmus und Algebra. Hier arbeiten Einzelkämpfer oder kleine Teams.

Im Erdgeschoss oder schon im Keller stößt man auf ein mit Blumen gut verdecktes Refugium, das man kaum sieht. Nur selten hört man etwas aus dieser untersten Etage, und es ist schon lange her, dass einige formal unentscheidbare Sätze nach außen drangen.

In der Ausbildung, bei der Festlegung des Curriculums eines Informatikunterrichts, ist man unweigerlich mit den verschiedenen Stockwerken des Informatik-Turms konfrontiert. Die Ansprüche an eine zeitgemässe Informatikausbildung sind vielfältig. Zum einen erwartet man von Schulabgängern Fertigkeiten im Umgang mit Standardanwendungen. Hier geht es um Produktwissen, das bedingt durch die rasch aufeinander folgenden Rochaden in den oberen Stockwerken nur eine geringe Halbwertszeit aufweist. Zum anderen erwartet man von Schulabgängern aber auch ein Verständnis für die grundlegenden Prinzipien, auf denen unsere Informationsgesellschaft aufgebaut ist. Ohne Konzeptwissen kann man den Einsatz und Nutzen von Informatiklösungen nicht beurteilen. Anders ausgedrückt: Eine fundierte, allgemein bildende und trotzdem praxisorientierte Informatikausbildung setzt voraus, dass man den Informatikturm in seiner ganzen Höhe zumindest besichtigt.

Stetiger Trend nach oben im Informatik-Turm

Wirft man nun einen Blick zurück auf die letzten Jahrzehnte der Informatikausbildung, so stellt man fest, dass die unteren Stockwerke des Informatik-Turms im Laufe der Zeit immer weniger besucht wurden. Man landet direkt auf dem Dach und bestaunt die immer farbigeren Tools, welche hier marktschreierisch angeboten werden. Die Unterrichtsinhalte sind Windows, Office, Multimedia, Internet usw. Nur an wenigen Schulen wird ein Abstecher in das zweitoberste Stockwerk unternommen, und wenn, dann gibt man sich bewusst modern. Kleine Programme schreiben ist uninteressant, Umgang mit großen Systemkomponenten ist „in". Und vieler-



orts möchte man den Computer als Unterrichtsgegenstand gleich ganz aus den Schulen verbannen. Es wird die Fächerintegration des Computers gepriesen. Es wird argumentiert, Autofahren lerne man auch nicht in der Schule, genau so wenig müsse der Computer im Unterricht thematisiert werden. Dabei unterscheiden sich die Computernutzung und das Autofahren grundlegend. Das Auto hat nur einen Zweck, die Fortbewegung vom Punkt A zum Punkt B. Der Computer ist ein Werkzeug mit einem sehr breiten Anwendungsspektrum. Beim Autofahren können nur wenige Parameter beeinflusst werden, hauptsächlich Gas, Bremse und Steuerrad. Im Gegensatz dazu ist das „Armaturenbrett" einer modernen Textverarbeitung wie Word ein Vielfaches komplexer. Wer einen Autotyp fahren kann, vermag recht einfach auf einen anderen Autotyp umzusteigen. Der Umstieg vom Release x.y auf Release x.y+1 einer Textverarbeitung bedingt meistens einige Stunden Einarbeitung. Und auch die Bedeutung von Autofahren und Umgang mit Informatikmitteln ist verschieden: Man kann sich problemlos ohne Führerschein durch das Leben schlagen. Ohne ein Grundverständnis für ICT ist man schon beim nächst besten Billett- oder Geldautomaten überfordert.

Es braucht ein Schulfach Informatik

Autofahren lernt man außerhalb der Schule, also könnten doch die Schülerinnen auch das „Computern" außerhalb lernen. Folgende drei Gründe sprechen für ein eigenes Fach „Information und Kommunikation":

1. Der Faktor Zeit: Die praktischen Fertigkeiten des Autofahrens lernt man in der Fahrschule in 20-30 Lektionen. Für die Vermittlung von Informatikkonzepten ist ein Fach mit mindestens 6 Jahresstunden nötig. Mit rund 250 Lektionen eine ganz andere Größenordnung!

2. Der Inhalt: Schulen haben den Auftrag, allgemein bildende, langlebige Inhalte zu vermitteln. Beim Autofahren geht es primär um reine Fertigkeiten. Der sinnvolle Einsatz von Informatikmitteln bedingt das Verständnis von zahlreichen, grundlegenden Konzepten. Später „on the job" bleibt dafür keine Zeit mehr.

3. Die Bedeutung: Die Beherrschung von Informatikmitteln ist neben Lesen, Schreiben und Rechnen inzwischen eine weitere Kulturtechnik, die alle Schulabgängerinnen beherrschen sollten. Man kann heute problemlos ohne Führerschein eine erfolgreiche berufliche Tätigkeit ausüben, nicht aber ohne gute Informatik-Kenntnisse.

Wie präsentiert sich die Situation rund um die Nutzung von ICT in den Schulen? Selbstverständlich versuchen die Schulen im Rahmen ihrer begrenzten finanziellen Möglichkeiten den Anforderungen der Informationsgesellschaft gerecht zu werden und investieren einiges in den Aufbau einer geeigneten Informatik-Infrastruktur. In der Terminologie des Autofahren ausgedrückt, werden heute in den meisten Schulen unzählige Tankstellen (Internetzugänge in den Schulzimmern) gebaut. Dazu natürlich auch ein Straßennetz (Vernetzung der Schulhäuser) und Autobahnen (breitbandige Anschlüsse ans weltweite Netz). Und einiges Geld wird in die Beschaffung neuer Autos (Personal Computer) gesteckt. Kaum investiert wird hingegen in den Straßenunterhalt (Wartung) und in Autogaragen (Informatik-Support). Und vielerorts fehlt es – wiederum in der Terminologie des Autofahrens – an Fahrlehrerinnen. Kurz: Der Informatikunterricht an den Schulen ist immer noch eine große Baustelle. Deshalb bleibt das Wichtigste oft ganz auf der Strecke, geht in all den Bauarbeiten unter: Das Straßennetz ist erbaut, die Autos sind angeschafft, aber eigentlich weiß niemand, wohin die Fahrt gehen soll.

Soll der bisherige Irrweg in der Schulinformatik beendet werden, dürfen kurzfristige und zufällige Entwicklungen nicht länger dazu verleiten, dauernd neue Wege einzuschlagen. Wie in anderen Fächern muss auch in der Informatik das Schwergewicht auf die Bildung, also langlebige Inhalte gelegt werden. Dazu braucht es ein eigenständiges Fach „Information und Kommunikation" mit einer Stundendotation wie andere klassische Schulfächer. In diesem Fach erwerben Schülerinnen und Schüler das grundlegende Konzeptwissen für die Nutzung von Informatikmitteln. Jede Schulabgängerin muss heute neben Lesen, Schreiben und Rechnen auch mit der Kulturtechnik „Computer" vertraut sein. Autofahren kann man hingegen weiterhin nach der Schule lernen.

Autor

Dr. Werner Hartmann, Information Technology & Education
Departement Informatik, ETH Zürich
E-Mail: hartmann@inf.ethz.ch
Homepage: http://www.educeth.ch
Dr. W. Hartmann leitet die Didaktikausbildung am Departement Informatik der ETH Zürich. Mit EducETH betreut er einen exzellenten Bildungsserver (siehe S. 30)

Informatik und PISA

Anmerkungen zur Debatte um informatische Bildung und deren Notwendigkeit in der Schule

1. Informatikunterricht – einige Aspekte aktueller Debatten

Mit der Einführung der informationstechnischen Grundbildung auf der Basis von Festlegungen der Bund-Länder-Kommission Mitte der 80-iger Jahre in Deutschland wurde sicher ein wichtiger Schritt zur Nutzung des Computers in den verschiedenen Bereichen der Allgemeinbildung gegangen. Es besteht inzwischen vielerorts Konsens darüber, dass es notwendig war und ist, für ein Grundverständnis der Computerbenutzung zu sorgen. Natürlich sollte man auch zur Kenntnis nehmen, dass die jeweiligen Anwendungen schnell veralten und damit eine Bedienung der im Moment aktuellen Anwendung, die nach dem Ende der Schulzeit in der Regel so gar nicht mehr existiert, eben gerade kein ausreichender Inhalt einer informatischen Bildung sein kann. Das ist insbesondere dann der Fall, wenn unter der Überschrift Computer-Führerschein die Bedienung eines aktuellen Softwareprodukts im Vordergrund steht.

Steffen Friedrich
TU Dresden

So sind integrative Ansätze vielleicht geeignet, die Universalität des Rechners deutlich zu machen, verdecken aber sowohl die für den richtigen Einsatz eines solch mächtigen Werkzeugs erforderlichen Grundlagen als auch die dafür in den Schulen oft nicht vorhandenen Rahmenbedingungen. Es gibt bisher auch kaum Vorschläge, wesentliche Teile der Mathematik innerhalb der Naturwissenschaften zu unterrichten und es ist für die Behandlung physikalischer Gegenstände doch nicht allein wichtig, welche Resonanz sie gegenwärtig in der öffentlichen Meinung besitzen, sondern welche physikalischen Grundprinzipien damit vermittelt werden. Ähnliche Überlegungen ließen sich für die anderen Naturwissenschaften und sogar auch für die geisteswissenschaftlichen Fächer anstellen.

Entsprechende Kompetenzen sind allerdings noch bei den Lehrenden und bei den Lernenden zu erwerben. Das macht eine informatische Bildung und darin enthaltenden Informatikunterricht als Grundbestandteil der Allgemeinbildung erst recht notwendig. Es wird Zeit, die Informatik in der Schule nicht mehr nur auf das Erlernen einer imperativen Sprache einzugrenzen oder die Notwendigkeit des Faches an aktuell noch gültigen Plänen zu beurteilen. So erwecken inzwischen multimediale Oberflächen und vorhandene Internetzugänge den Eindruck, dass die Nutzung neuer technischer Möglichkeiten in der Bildung lediglich eine Frage der modernen Ausstattung darstellt. Eine Verlagerung dieser grundlegenden Ausbildung in Fächer und Projekte verlangt erst recht nach hoher technischer Aktualität und größerem personellen Aufwand. Mit dem Blick auf die momentan vorhandenen finanziellen Möglichkeiten bei den öffentlichen Haushalten ist dies auf Dauer nicht zu leisten.

Mitunter wird der Standpunkt vertreten, dass es grundlegender Voraussetzungen für die Benutzung moderner Standard-Anwendungen eigentlich nicht bedarf. Im Gegenteil erfordern die vielfältigen Systeme gerade gegenwärtig eine grundlegende und systematisierende Bildung in der Informatik. Erst auf der Basis der Kenntnis von Wirkprinzipien von Informatiksystemen ist es sinnvoll, über Möglichkeiten und Grenzen von Computern zu diskutieren und deren gesellschaftliche Wirkungen im Informationszeitalter zu verstehen.

Schließlich ist auch die Diskussion um Inhalte von Schulfächern und deren quantitative und qualitative Bestimmung so alt wie die Schule selbst. Dabei hatten es neue Inhalte gewöhnlich schwer, sich durchzusetzen oder haben sich bis heute nicht etablieren können.

Ein Anspruch an und der Nachweis von Bildung ist auf diesem Gebiet auch keinen modernen Strömungen unterworfen, sondern kann die Substanz und auch die Kompetenzen vermitteln, die für ein Leben und Arbeiten in der künftigen Informations- und Wissensgesellschaft benötigt werden. Eine in diesem Sinne allgemeine Bildung wird sich somit nicht an Oberflächen eines momentanen Entwicklungsstandes orientieren, sondern wird die Dinge genauer betrachten und Effekte hinterfragen, um so grundlegende Wirkmechanismen zu vermitteln.

Eine solche Auffassung umfasst eine Benutzung von Anwendungen ebenso, wie die entsprechenden theoretischen und praktischen Grundlagen zur Methodik, Analyse und Konstruktion von Informatiksystemen sowie die Auswirkungen ihres Einsatzes. Schließlich ist es Anliegen der Fachdisziplin Informatik, sich mit Hard- und Software, mit Organisation und Struktur von Informatiksystemen und deren Daten sowie mit Aspekten bei Nutzern und Betroffenen dieser Systeme zu beschäftigen. Sie ist in diesem Sinne weder auf das Erlernen der Bedienung von Anwendungen reduzierbar, noch ist sie bloße Informationstechnik.

2. Informatik - wirklich ein Schulfach?

Diese Gesamtsicht auf eine informatische Bildung macht die Komplexität in diesem Bereich deutlich und charakterisiert unverzichtbare Bestandteile. Hinterfragt man in diesem Zusammenhang den Begriff der Bildung, kann man sich durchaus folgender Darstellung anschließen:

„Erteilt man ihm (dem Menschen) jedoch eine grundlegende Bildung im Sinne einer ‚Allgemeinbildung', wird er in den Stand gesetzt, auf solche Veränderungen, die vorher niemand voraussehen kann, durch Weiterlernen flexibel zu reagieren."[1]

Dies impliziert Lernende natürlich auch hinsichtlich der Informatik in eine solche Position zu bringen, damit sie künftigen Anforderungen gewachsen sind, weiterlernen und flexibel reagieren können. Es liegt auf der Hand, dass eine lediglich an aktuellen Applikationen orientierte Benutzerschulung diesem Anspruch nicht gerecht wird. Seitens der Informatik und deren Anwendung im Fach gilt es allerdings, solche Themen deutlich herauszuarbeiten, die den jeweiligen Bildungsanspruch auch erfüllen können.

Das soll an einem Beispiel zum Lernbereich „Präsentieren von Informationen"[2] kurz gezeigt werden:

Der Unterricht soll vor allem dazu beitragen, den Schülern einen Einblick in die Unterstützung von Präsentationen durch Informatiksysteme zu vermitteln. Anhand der vielfältigen Erfahrungen aus dem Alltag, aus dem bisherigen Unterricht im Fach Informatik sowie dem Unterricht in anderen Fächern dient dieser Lernbereich der Systematisierung des Vorwissens, der Ausprägung der informatikbezogenen begrifflichen Grundlagen und Erarbeitung von Herangehensweisen zur Auswahl bzw. zum effizienten Einsatz geeigneter Anwendungen. Eine besondere Rolle spielen dabei die Möglichkeiten und Grenzen der Abbildung von Informationen auf Daten. Unter dem Aspekt der Präsentation lernen die Schüler Grundlagen zur Strukturierung von Informationen und verschiedene Verfahren ihrer Darstellung (in Abhängigkeit von den verwendeten Medien) kennen.

Bei Beachtung des zeitlichen Umfangs (am Beispiel des gültigen Lehrplanes in Sachsen) und der zu vermittelnden Grundlagen könnte sich der Unterricht auf folgende Schwerpunkte konzentrieren:

- ❑ Grundlagen der computergestützten Präsentation
- ❑ Darstellen von Informationen in Abhängigkeit der Ausgabemedien
- ❑ Planen, Erstellen und Verteidigen einer projektartigen Aufgabe in selbständiger Schülerarbeit

Auf folgende Inhalte und Begriffe aus der Informatik sollte in diesem Lernbereich nicht verzichtet werden:

- ❑ Strukturierung von Informationen (Daten), Verknüpfung von Objekten (aus unterschiedlichen Anwendungen)
- ❑ Dateiformate und zugehörige Anwendungen
- ❑ Austauschbarkeit und Konvertierung von Daten (Filter)
- ❑ Realisierung von Layoutgestaltungen; Schutz von Informationen (Urheberrechte; Datensicherheit)

Die Fokussierung auf die Präsentation von Informationen macht es allerdings auch notwendig, dass zwischen den Gegenständen des Faches Informatik und den notwendigen Voraussetzungen aus anderen Fächern (z.B. Deutsch, Kunst) differenziert wird. Vielleicht ist es auch hilfreich, fachübergreifende Abstimmungen so zu treffen, dass dieses Thema möglichst zeitnah in verschiedenen Fächern behandelt wird. Das vereinfacht einerseits den Unterricht, weil Beispiele aus gerade behandelten Themen, sprachliche und gestalterische Entwürfe aus den Fächern und die Realisierung von entsprechenden Präsentationen verteilt erfolgen könnten. Andererseits besteht natürlich immer die Gefahr, dass die eigentlichen fachlichen Grundlagen aus der Informatik in den Hintergrund geraten. Hier sind eine exakte Planung und die konkrete Beschreibung der Anforderungsstruktur notwendig." [3]

Es bleibt für die Informatik die Frage, wie und warum sie als Schulfach in diese Strukturierung vorhandener Fächer neu einzuordnen ist. Im Sinne des genannten Anspruchs an Bildung in einem Schulfach Informatik werden gerade solche Inhalte und Methoden vermittelt, die sich deutlich von denen anderer Fächer unterscheiden und zur Flexibilität bei der Bewältigung künftiger Anforderungen beitragen. Also steht mehr die Frage, die Grundprinzipien der Informatik als Bildungsgut zu fordern, die unabhängig von aktuellen Entwicklungen im Bereich der Hard- und Software Allgemeingut für den Absolventen der jeweiligen Schulart sein müssten. Die aktuellen Bezugspunkte spielen dabei eher eine mittelbare Rolle, zumal sie sich in der gegenwärtigen Zeit gerade auf diesem Gebiet sehr schnell ändern.

3. Informatische Bildung - Entwicklungsstufen als neuer Anfang!

Ausgehend von Leitlinien zur informatischen Bildung[4] wurden die im Rahmen der PISA-Studie in den Bereichen Mathematik und Naturwissenschaften ausgearbeiteten Kompetenzstufen dahingehend untersucht, ob diese auch einer ähnlichen stufenweise Darstellung informatischer Bildung dienen könnten. Betrachtet man die dabei gefundenen Stufen zur informatischen Kompetenz[5] genauer, wird deutlich, dass diese ohne einen systematisierenden Fachunterricht nicht erreichbar sind, sich aber auch nicht darauf beschränken.

In der Tabelle auf Seite 13 werden die Stufen zusammenfassend über alle Leitlinien benannt, um in weiteren Tabellen differenziert dargestellt zu werden.

---

[1] *Giesecke. Vom Sinn der Bildung. 1999 (http://home-t-online.de/home/hermann.giesecke)*
[2] *Orientierungsrahmen Informatik (Gymnasium) Sachsen - Klasse 8 (http://lehrer.sn.schule.de/~infogy/didaktik/lplan/or8_10.html)*
[3] *Handreichung „Gymnasium - Informatik - Klasse 8 bis 10" (http://lehrer.sn.schule.de/~infogy/didaktik/lplan/or8_10.html)*
[4] *Gesamtkonzept informatischer Bildung der GI. Bonn, 2000*
[5] *Friedrich. Informatik und PISA – zum Wohl und Wehe der Schulinformatik. INFOS'03 (unveröff.)*

Informatische Kompetenz	Interaktion / Problemlösung	Wirkprinzipien / Konzepte	Modellierung / Abstraktion	Wechselwirkung / Allgemeinbildung
Stufe I	Bedienung von Informatikanwendungen			
Stufe II	Benutzung von Informatiksystemen			
Stufe III	Kenntnis fachsystematischer Grundlagen			
Stufe IV	Verständnis von Konzepten der Informatik			
Stufe V	Entwicklung und Bewertung von Informatiksystemen			

Mit Blick auf die PISA-Studie wurde darauf verzichtet, diesen Stufen eine äußere Schulstruktur zuzuordnen, da die Beschreibung des erzielten Abschlussniveaus den fachlichen Anspruch ausreichend charakterisiert. Es zeigte sich auch dass die genannten Leitlinien ein geeignetes Herangehen darstellen, um grundlegende Aspekte informatischer Bildung, wie Abstraktion, Problemlösung, Konzeptanalyse und Stellung in der Allgemeinbildung zu strukturieren. Sie stehen somit für die Ganzheitlichkeit des Bildungsanspruchs, der durch die Betonung der Problemlösung („Algorithmenkonzept") ebenso einseitig interpretiert wurde, wie es möglicherweise durch die Betonung der Abstraktion („Modellierungskonzept") geschehen könnte.

Hinsichtlich der Stufung einzelner Leitlinien lassen sich in gleicher Weise einzelne Aspekte beschreiben und vergleichend darstellen. Anhand ausgewählter Punkte konnte gezeigt werden, dass eine entsprechende Differenzierung für die Leitlinien möglich ist und das didaktische Verständnis für die Unterrichtsinhalte bei Lehrern unterstützt.

Eine Ausarbeitung und Strukturierung informatischer Kompetenzen kann sich als Basis einer Bildungsnotwendigkeit auf diesem Gebiet erweisen, weil sie die scheinbaren Gegensätze zwischen Anwendungsbedienung und Grundlagenwissen aufbricht. Bereits vor 20 Jahren waren diese Aspekte durch Entgegensetzung einer informationstechnischen Grundbildung und des Informatikunterrichts Gegenstand der didaktischen und schulpolitischen Debatte um das Schulfach Informatik, die der Positionierung dieses Faches in der Allgemeinbildung nicht gerade nützlich waren.

„Mit anderen Worten: Von ihren geistigen Qualitäten her könnten die Naturwissenschaften genauso wie die Musik oder die Literatur zur Bildung gehören ... doch scheint dieser intellektuell-lustvolle Aspekt der Wissenschaften kaum wahrgenommen zu werden. Der Grund für diesen Mangel liegt unter anderem darin, dass es kam Institutionen gibt, die Menschen so ausbilden, dass sie selbst jenes Vergnügen an den Naturwissenschaften erfahren, das deren Schöpfern selbstverständlich ist. ..." [6]

Buchvorstellung

Prof. Dr. Steffen Friedrich ist zusammen mit Dr. Norbert Breier Herausgeber eines interessanten Lehrbuches für den schulinformatischen Anfangsunterricht "Informatische Grundbildung", erschienen im Paetec-Verlag.

Autoren dieses lesenswerten, unterrichtspraktischen Werkes sind die beiden Herausgeber des "LOGIN", der fachdidaktischen Fachzeitschrift für die Schulinformatik im deutschsprachigen Raum schlechthin, Bernhard Koerber und Ingo-Rüdiger Peters.

Link: http://www.log-in-verlag.de

In der aktuellen Ausgabe der Zeitschrift LOGIN (Nr. 122/123) sind auch drei österreichische Beiträge (Mittermeir, Caba, Micheuz) zu finden. Alle drei Autoren haben auch Beiträge für diese Sonderausgabe beigesteuert.

Autor

Univ. Prof. Dr. Steffen Friedrich
TU Dresden, Institut für Informatik
Vorsitzender der Gemeinsamen Kommission Lehrerbildung (GKL) an der TU Dresden, verantwortlich für Lehrerbildung an der Fakultät Informatik
GI-Fachausschuß 7.3 (Mitglied)
E-Mail: sf2@inf.tu-dresden.de
Homepage: http://dil.inf.tu-dresden.de
Friedrich ist Initiator der Königsteiner Fachgespräche, einem informellen Treffen führender Informatik-Fachdidaktiker im deutschsprachigen Raum, das heuer zum 10ten Mal stattgefunden hat. http://koenigstein.inf.tu-dresden.de/

[6] *Giesecke. Vom Sinn der Bildung. 1999 (http://home-t-online.de/home/hermann.giesecke)*

Modell – Experiment – Validierung

Eine Herausforderung für den Informatik Unterricht

Vermittlung von Modellierungsfähigkeit ist ein Hauptziel anspruchsvoller Mathematik-Didaktik wie auch moderner Informatik-Didaktik. Dennoch unterscheiden sich die beiden Fächer in den Möglichkeiten, Jugendlichen diese Fähigkeit zu vermitteln. Ausgangspunkt sind wohl in beiden Fächern Textaufgaben. Dabei ist freilich zu beachten, dass nicht jede Textaufgabe geeignet ist, dieses Ziel zu erreichen. Ebenso reicht es nicht, von Schülern zu verlangen, ein bestimmtes Szenario in UML [1, 2] zu beschreiben. Kern der Modellierung ist doch letztlich die Überführung eines (möglichst) realen vielschichtigen nichtformalen Problems in eine formale Beschreibung der als wesentlichst erachteten Aspekte, sodass durch Bearbeitung des formalen Modells Rückschlüsse auf die Realität gezogen werden können bzw. eine Wirkung in der Realität erzielt werden kann.

Roland Mittermeir
Universität Klagenfurt

Hierbei unterscheiden sich freilich Mathematik- und Informatik-Unterricht. Der Einsatz eines Modells im Mathematik-Unterricht muss sich letztlich auf eine Berechnung beschränken. Die Transformation der Realität über die Zwischenstufen einer Textaufgabe und einer zugehörigen (algebraischen) Formalisierung endet in der Berechnung. Niemand kann verlangen, dass die Ergebnisse der Berechnung durch Bau einer Brücke, Spannen eines Seils, Verladen von Holz, etc. überprüft wird. Anders im Informatik-Unterricht. Hier führt der Weg vom formalen oder halbformalen Modell zur Implementierung in einer entsprechenden Programmiersprache oder in einem entsprechenden Entwicklungssystem (Tabellenkalkulation, Datenbanksystem, ...). Diese Implementierung sollte nicht in der Schublade verschwinden. Sie kann genutzt werden, um ausgeführt zu werden, und zwar in einer Form, die über billiges „es-funktioniert-(manchmal)"-Testen hinausgeht. Vielmehr ist es möglich, mit dem fertigen Programm/System zu experimentieren und so Implementierung und Modell zu evaluieren. Die Kette Realität – Modell – Implementierung – Wirkung auf die Realität bzw. in der Realität kann somit zu einem Kreis geschlossen werden.

Durch diese Art des „round trip engineering" kann der Unterricht nicht nur lebendiger und anschaulicher werden. Die Rundreise ermöglicht es auch Schülerinnen und Schülern, den Wert der Modellierung anschaulich zu demonstrieren. Man muss ja nicht die unterschiedlichen Modelle vor Implementierung zu einer gemeinsamen Standardlösung korrigieren. Anfängern ist es ohnehin schwer klar zu machen, worin der Unterschied zwischen einem guten und einem schlechten Modell liegt. Gegen: „Mein Modell kann doch auch ..." ist nicht immer leicht zu argumentieren. Lässt man aber unterschiedliche Modelle implementieren, ist es ein Leichtes, experimentell zeigen zu lassen, warum so mancher Entwurf leistungsfähiger war als ein anderer.

Freilich ist zu berücksichtigen, dass es Zeit benötigt, die Rundreise auch zu Ende zu gehen und darüber hinaus die Ergebnisse der Modell- und Implementierungsevaluation noch mit der Klasse zu diskutieren sind. Doch diese Zeit ist nicht verloren! Ist es nicht ein übergeordnetes Ziel jeden Unterrichts, die Jugendlichen zu urteils- und kritikfähigen Staatsbürgern zu erziehen. Dies wird nur schwer gelingen, wenn sie stets nur Subjekt der Beurteilung sind. In der Evaluation des Experiments werden sie selbst zu Urteilenden und können so auch leicht erkennen, dass es neben jenem Kriterium, das im Experiment geprüft wurde, wohl auch noch andere Kriterien geben wird, anhand derer die Qualität unterschiedlicher Lösungen beurteilt werden kann.

Moderne Informatik-Didaktik ruft nach Objektorientierung und daher beim Schlagwort Modellierung wohl nach UML. Ich widerspreche nicht, wenn der Unterricht bereits jenes Niveau erreicht hat, auf dem diese Konzepte lehrstoff- und schüleradäquat eingesetzt werden können. Doch Modellierung ist weiter zu sehen und nicht immer muss man zu solch ausgereiften Mechanismen greifen, um den Wert von Überlegungen auf Modellebene zu demonstrieren. Manchmal genügt es auch, zur Motivation effizienter Algorithmen analogistische Modelle zu präsentieren. Dies kostet kaum Zeit, erlaubt aber dennoch alternative Implementierungsformen, die bewertet und diskutiert werden können.

Die Umsetzung dieses didaktischen Konzepts möchte ich vorerst im Kontext von Suchalgorithmen beschreiben. Selbstverständlich kann man argumentieren, dass Binärsuche im allgemeinen Fall effizient ist. Man kann es sogar berechen. Anschaulich ist das aber nicht. Wie wäre es, die Suche im Telefonbuch als Modellfall zu nehmen? Das Anwendungsgebiet ist selbst Jüngsten vertraut. Als Modellalternativen bieten sich an:

a) Ziehen der gewünschten Telefonnummer aus einer Urne (Zylinderhut) in dem sich alle Telefonbucheinträge in zu Kügelchen gerollter Form befinden (man könnte dies ja zu einer Silvesteraufgabe machen).
b) Lesen des Telefonbuchs von der ersten Seite weg, bis der gewünschte Anschluss gefunden ist.
c) Beschreibung des Verfahrens, mit dem Sie nach einem Eintrag suchen.
   Lösung a) und b) lassen sich problemlos implementieren. Lösung c) wäre aufwändiger. Sie ist eigentlich nicht „computergerecht". (Warum?). Also versuchen wir Lösung c) so zu transformieren, dass sie ihre wesentlichen Eigenschaften behält, aber algorithmisch einfacher wird.

Daraus entsteht wohl problemlos
d) die Binärsuche.
Als Querbeziehung zum Mathematikunterricht kann man noch die algorithmische Komplexität der Verfahren a) b) und d) berechnen und letztlich können unterschiedliche Gruppen die einzelnen Verfahren implementieren und anschließend instrumentieren, um so festzustellen, wie weit sich das gemessene Verhalten der jeweiligen Programme mit dem berechneten deckt.

Klassisches Feld für Modellierungsaufgaben findet man selbstverständlich im Datenbank-Bereich. Hier bieten sich auch Querbeziehungen zu nahezu jedem beliebigen anderen Unterrichtsfach an. So ließen wir etwa einmal eine Datenbank historischer Persönlichkeiten [3] erstellen. Dass historische Persönlichkeiten die Attribute Familienname, Vorname, Geschlecht, Funktion oder Titel (Zuordnung zu einem Wissensgebiet), Beruf, Geburtsjahr, Sterbejahr, Geburtsort hatten, war soweit klar. Ob man eine Identifikation (Personen-Nummer) benötigt, war schon nicht so klar. Jedenfalls reichte dies um eine einfache Tabelle füllen zu lassen. Dabei traten erste Probleme auf: Welchen Familiennamen soll man für Maria Theresia eingeben? Welchen bei Euklid? Bei Daniel Bernoulli wurde Mathematiker und Arzt gefunden. Was machen wir mit den laufenden Nummern (Franz I, Franz II) und was, wenn neben Franz I von Österreich auch noch Franz I von Frankreich aufgenommen werden soll. – Ad hoc Lösungen findet man immer, doch halten sie? Allein das Befüllen der Datenbank zeigte, wie bedeutsam es gewesen wäre, all diese Fragen auf Modellebene zu klären, und nicht durch Veränderung der Datenbank zur Abdeckung neu aufzunehmender Datensätze ein schon befülltes Datenbank-Schema mühsam nachführen zu müssen. Freilich ergeben sich Modelländerungen nicht nur aus dem unveränderten Betrieb der Datenbank. Es ist doch fair, nachdem alles funktioniert, auch noch zu fordern, dass die Datenbank sagt, wer mit wem verwandt ist. Spätestens an dieser Stelle stolpern wir über das, was in der Datenbank-Theorie Normalisierung genannt wird und spätestens an dieser Stelle erkennen wir auch den Wert eines halbformalen Modells. Entity-Relationship-Modellierung reicht für diese Fragestellungen aus. Ob man UML verwendet und sich dabei auf jene Teile der UML-Statik beschränkt, die auch in ER ausdrückbar wären, hängt vom weiteren Programm ab, das man mit dieser Klasse vor hat.

Etwas anspruchsvoller als bei den historischen Persönlichkeiten gestaltete sich die Modellierungsdiskussion bei der Realisierung einer Literaturdatenbank. Gewünscht war die Beantwortung von Fragen nach einem Autor (oder Quelle) zu gegebenem Werk bzw. auch die Ausgabe aller Werke einer bestimmten Autorin. Doch ein „Test" des Modells zeigte rasch, dass dies so einfach nicht sein wird. Was ist mit Autorengemeinschaften, was mit Sammelbänden, mit Aufsätzen in Fachzeitschriften, mit Mehrfachauflagen, etc.

Ich spare an dieser Stelle klassische Modellierungsbeispiele von Anwendungssystemen aus. Diese liegen ohnehin auf der Hand. Wesentlich war mir aber zu zeigen, dass Modellierung mehr sein kann als die Vorstufe zur Implementierung, die aufgrund der Einfachheit von schulischen Aufgaben, viele Schülerinnen und vor allem Schüler ohnehin schon vorab im Kopf haben. (Hier habe ich bewusst nicht geschlechtsneutral formuliert. Burschen scheinen eher zur Bastellösung zu neigen als die in technischen Belangen etwas prinzipengeleiteteren Mädchen). Modelle bieten eine Basis der Bewertung nach vielfältigen Kriterien und sie bieten vor allem eine Basis für effiziente Kommunikation innerhalb der Klasse über unterschiedliche Lösungen. Dies sollte didaktisch genutzt werden. Wesentlich ist freilich, dass nicht nur ein einziges Modell erarbeitet wird, das dann implementiert wird. Die Klasse muss Modellvarianten erstellen, die diskutiert und bewertet werden sollten. Aus der Implementierung und aus einem entsprechend gestalteten Betrieb lässt sich dann erkennen, das die Bewertung auf Modellebene keine Marotte der Lehrperson ist, sondern ein entscheidender Faktor für den Projekterfolg ist.

Abschließend sei darauf hingewiesen, dass durch die Modell-Diskussion, Modellbewertung und das anschließende Experiment auch übergeordnete Bildungsziele erreicht werden. Schließlich wollen wir den Jugendlichen doch nicht vermitteln, dass Informatik die Kommunikation mit einer Maschine ist. Informatik ist die Kommunikation über informationstechnische Lösungen zwischen Menschen und die Umsetzung dieser Lösungen auf Maschinen.

Autor

Univ. Prof. DI Dr. Roland Mittermeir
Institut für Informatiksysteme, Universität Klagenfurt
E-Mail: roland@isys.uni-klu.ac.at
Homepage: http://www.ifi.uni-klu.ac.at/ISYS/RM/Staff/Roland.Mittermeir

Mittermeir war maßgeblich an der Einführung des Lehramtstudiums Informatik und Informatikmanagement an der Universität Klagenfurt beteiligt und hat neben seinen vielen Funktionen im Universitätsbetrieb auch einige Beiträge zur Schulinformatik publiziert.

Literatur

[1] Rumbaugh J. Jacobson I., Booch G.: The Unified Modeling Language – Referance Manual; Addison Wesley, 1999.
[2] Hitz M., Kappel G.: UML @ Work: Von der Analyse zur Realisierung; 2. Aufl., dpunkt.verlag, 2003.
[3] Mittermeir R., Kofler E.: Informatik-Einführungsunterricht für Lehramtskandidaten mit besonderer Berücksichtigung geisteswissenschaftlicher Fächer; in: C. Hüffel, A.. Reiter (Hrsg.): Praxis der EDV/Informatik, Reihe: Schule und Erziehung, Jugend & Volk, 1996, pp. 327 - 339.

Excel und VBA

Ein universeller Werkzeugkasten für didaktische Zwecke

Gehen Sie zu einem beliebigen Rechner in einer Schule oder einem Büro und Sie finden fast mit Sicherheit zwei Programme installiert: Microsoft Word und Microsoft Excel. Dass Textverarbeitung ein Hilfsmittel ist, das in nahezu allen Gegenständen sinnvoll eingesetzt werden kann, ist mittlerweile keine Frage mehr. Dass Ähnliches auch für Tabellenkalkulation (also insbesondere für Excel) gilt, ist aber in der Fachdidaktik - und zwar sowohl in Theorie als auch in der Praxis - noch nicht allgemein anerkannt.

Wir wollen uns in diesem Artikel mit drei verschiedenen

Erich Neuwirth
Universität Wien

Themenkreisen beschäftigen, in denen Excel didaktisch ergiebig eingesetzt werden kann:

- ❑ Berechnungen
- ❑ Datenanalyse
- ❑ Informatische Lerninhalte

Die Bedeutung der Tabellenkalkulation nimmt in der Mathematik-Didaktik noch nicht den Stellenwert ein, der ihr (nach Meinung des Autors) zusteht.

Ein Beispiel

Stellen sie sich folgende 10x10-Tabelle vor. Die erste Spalte enthält lauter Einser. Der Rest der ersten Zeile enthält lauer Nullen. An allen anderen Stellen steht die Summe der Zahlen aus der Zelle unmittelbar darüber und aus der Zelle links darüber.

Diese Beschreibung ist (hoffentlich) einfach und verständlich, und sie lässt sich mit Excel unmittelbar umsetzen.

Wenn man das in klassischer mathematischer Schreibweise ausdrückt, sieht es so aus:

Es ist ziemlich einleuchtend, dass die verbale Beschreibung wesentlich einfacher verständlich ist als die (rekursiven) Formeln. Beide Beschreibungen drücken aber genau dieselben Beziehungen aus.

$$F(n,0)=1$$
$$F(0,k)=0 \quad \text{für } k>0$$
$$F(n,k)=F(n-1,k-1)+F(n-1,k) \quad \text{für } n>0 \text{ und } k>0$$

Was wir da gerade beschrieben haben sind die Binomialkoeffizienten, die eigentlich der höheren Mathematik zugerechnet werden. Die verbale Beschreibung ist deswegen so einfach, weil so selbstverständlich Begriffe wie „die Zelle darüber" und „die Zelle links darüber" verwendet werden. Diese Begriffe waren bisher in der mathematischen Schreibweise nicht „zugelassen". Im Rahmen der Tabellenkalkulation sind das aber nicht nur informelle Beschreibungen sondern tatsächlich die Formeln, die man dann in Excel auch tatsächlich eingibt.

In absehbarer Zeit werden für Excel auch Computeralgebra-Systeme verfügbar sein, z.B. YACAS (Yet Another Computer Algebra System).

Bereits jetzt lassen Formularelemente wie Buttons, Eingabefelder und Schieberegler interaktive Animation zu. Das ist fast nirgends so einfach wie in Excel, und schon allein deswegen sollte man Excel (oder ein anderes gleich leistungsfähiges Tabellenkalkulationsprogramm) auf jeden Fall im Unterricht verwenden.

Abgesehen von vielen Funktionen ist Excel aber auch ein sehr leistungsfähiges Datenbank- und Auswertungsprogramm. Die Verwaltung von Datenlisten ist einfach und die Vermittlung vieler Grundbegriffe aus dem Datenbankbereich (Datensatz, Feld, Sortieren, Filtern, ...) ist anschaulich vermittelbar. Damit lassen sich in vielen Unterrichtsfächern konkrete Projekte durchführen. Z.B. Taufbücher der eigenen Gemeinde zu erfassen und dann nach Berufen und Wohnorten der Eltern auszuwerten, wäre eine konkrete (interdisziplinäre) Anwendung. Im Internet gibt es reichlich Daten über wirtschaftliche und soziale Bedingungen in verschiedenen Ländern. Damit kann man in Geografie selbst Ländervergleiche durchführen. Eine sehr gute Website zu diesem Thema biete das US Bureau of Census, http://www.census.gov. Daten über Österreich in computerverarbeitbarer Form gibt es bei Statistik Austria, http://www.statistik.at.

Techniken wie die die Anwendung von Pivot-Tabellen (Kreuztabellen) machen Excel zu einem leistungsfähigen Analyseinstrument, mit dem man Daten nach mehreren Merkmalen gleichzeitig gliedern und statistische Maßzahlen berechnen kann.

Die ganz wichtige didaktische Botschaft bei diesem Einsatz von Excel ist, dass man bei statistischen Analysen nicht darauf angewiesen ist, dass jemand anderer (ein Experte) die interessanten Auswertungen macht, sondern diese Analysen selbst durchführen kann.

Mit Hilfe der Datenbankfunktionen und der Pivot-Tabellen kann man in Excel alle Aufgaben, bei denen es um systematisches Erfassen und Auswerten geht, sehr einfach durchführen. Anwendungen sind in vielen verschiedenen Fächern denkbar. Im Sprachunterricht könnte man Vokabellisten nach Lektionen des Lehrbuchs und Wortarten gliedern und dann gezielt Wortlisten als Lernhilfen für einzelne Wortarten erstellen.

Die Datenbank-Funktionen von Excel sind sehr umfangreich. Es besteht fast kein Grund, im didaktischen Einsatz ein spezialisiertes Datenbankprogramm einzusetzen, viele normalerweise auftretenden Fragen und Probleme lassen sich mit Excel beantworten.

Nun aber zu dem Thema, das vielleicht am überraschendsten ist. Mit Excel lassen sich auch praktisch alle Fragen des Informatik-Unterrichts behandeln, und zwar in sehr verschiedenen Darstellungsformen.

Schon das Tabellen-Modell mit automatischer Neuberechnung ist sehr ergiebig. Funktionsiteration und vor allem Rekursion gelten in der Regel als eher anspruchsvolle Konzepte. Wenn man in Excel in einer Zelle eine Formel erstellt, die aus der Zahl in der Zelle darüber einen neuen Wert berechnet und diese Formel dann nach unten kopiert, dann hat man mit diesen Begriffen ganz selbstverständlich hantiert. Eine Vielzahl von Aufgaben, die Thema des Informatikunterrichts sind, können sehr effizient und elegant im Tabellenkalkulationsmodell behandelt und bewältigt werden.

Excel bietet aber noch mehr. Haben sie schon einmal die Tastenkombination Alt-F11 gedrückt? Wenn sie das tun, dann starten sie die VBA-IDE (also die Programmierentwicklungsumgebung für Visual Basic for Applications). VBA ist keine Sparversion einer Programmiersprache, es handelt sich um eine augereifte Programmiersprache mit allen üblichen Konstrukten (inklusive Rekursion) und zusätzlich einem sehr umfangreichen Objektsystem. Praktisch alles, was man in anderen Sprachen und Implementationen wie Pascal (Delphi) oder auch Python machen kann, ist auch in VBA möglich. Alle üblichen Einführungskurse ins Programmieren kann man daher auch in Excel abwickeln. Es gibt noch dazu einige weitere Vorteile. Die Ausgaben von Programmen (immer eine sehr mühselige Angelegenheit) kann man in eine Tabelle schreiben, dadurch wird Formatieren viel einfacher. Es gibt viele User-Interface-Elemente, man kann auf Mouse-Events reagieren usw. Besonders ergiebig sind Projekte, die Tabellenkalkulationsmodelle mit prozeduraler VBA-Programmierung verbinden. Ein schönes Beispiel ist das „Game of Life". Zur Erinnerung. Das Spiel findet auf einem karierten Blatt (also einer Tabelle) statt. Manche Zellen leben (sind markiert), alle anderen Zellen leben nicht. Wenn ein Zustand (eine Generation) gegeben ist, wird daraus die nächste Generation errechnet. Jede Zelle mit genau 3 lebenden Nachbarn lebt in der nächsten Generation. Eine lebende Zelle mit genau 2 lebenden Nachbarn lebt auch in der nächsten Generation. Alle anderen Zellen sterben. Die entsprechenden Formeln sind sehr leicht in Excel zu implementieren. Zusätzlich ist es sehr angenehm, wenn man die Ausgangskonfiguration mit der Maus erzeugen kann. Das ist in VBA mit ca. 10 Zeilen Code möglich.

Codeentwicklung mit VBA ist auch aus einem weiteren Grund sehr lehrreich. Es gibt einen Lernmodus (also einen Makro-Recorder). Man „führt Excel vor", was man machen möchte, und Excel erzeugt den VBA-Code, der genau das macht. Diesen Code sieht man sich an, und meistens sieht man dann, was man noch ändern möchte. Man verfeinert also den Code (Stepwise refinement ist eine wichtige Technik der Programmierung). Wenn man das macht, erkennt man auch, welche Teile des Codes man gerne parametrisieren möchte, man lernt also aus einer konkreten Handlung das allgemeine Handlungsmuster herauszudestillieren, ebenfalls eine sehr wichtige informatische Fähigkeit.

Solche Projekte haben neben der Vermittlung „kerninformatischer" Inhalte noch einen weiteren ganz wichtigen Effekt: wir vermitteln die Botschaft, dass Standardsoftware den eigenen Bedürfnissen angepasst werden kann. VBA nicht nur Skriptsprache von MS-Office, sondern z.T. auch bei CAD-Software im Einsatz. Die Verwendung dieses Werkzeugs im Informatikunterricht vermittelt daher Fertigkeiten, die unmittelbar nutzbringend eingesetzt werden können.

Die meisten Firmen, die heutzutage mathematische Berechnungen durchführen, machen das mit Excel. Excel ist mittlerweile d a s Programm für Anwendungen der numerischen Mathematik geworden. Das spiegelt die Alltagspraxis der Schule aber noch nicht wieder.

Excel-Kenntnisse sind also auf dem Arbeitsmarkt sehr gefragt. Es ist natürlich nicht primäre Aufgabe der Schule, ausschließlich auf dem Arbeitsmarkt verwertbare Fertigkeiten zu vermitteln. Wenn man aber einen Weg findet, zentrale Informatik-Konzepte an Hand eines Werkzeugs zu vermitteln, dessen Kenntnis auf dem Arbeitsmarkt sehr gefragt ist, dann ist das eine Situation, die man sich öfter wünschen würde. Dann gelingt es uns nämlich, die beiden zentralen Aufgabe der Schule, Bildung und Ausbildung, miteinander zu verbinden.

Die Verwendung von Excel vermittelt noch eine weitere wichtige Botschaft. Auch in so abstrakten Fächern wie theoretischer Informatik und Mathematik kann man mit jener Software arbeiten, die auf (fast) jedem Arbeitsplatzrechner installiert ist. Es sind keine Spezialwerkzeuge notwendig, die außerhalb der Fächer und der Schule wenig Verbreitung haben. Wir verbinden also das Alltagsleben und die Alltagserfahrungen mit dem, was wir in der Schule lernen.

Verschiedene Projekte zum Einsatz von Excel finden Sie auf http://sunsite.univie.ac.at/Spreadsite/.

Autor

Ao. Univ. Prof. Dr. Erich Neuwirth
E-Mail:erich.neuwirth@univie.ac.at
http://mailbox.univie.ac.at/erich.neuwirth
http://sunsite.univie.ac.at
Neuwirth ist der Öffentlichkeit vor allem als Statistikprofessor bekannt, Insidern als "Mr. Excel" und ist an der Universität Wien auch im Bereich des Lehramtsstudiums Informatik und Informatikmanagement tätig.

Schulinformatik, quo vadis?

Über die Notwendigkeit zentraler Leitideen

Sicherlich überrascht die Frage ‚Schulinformatik quo vadis?' zunächst, aber dies ist beabsichtigt und ich möchte auch eine Rechtfertigung und Motivation für meine provokante Frage geben.

Bestandsaufnahme

An welchen Inhalten, welchen didaktischen Konzepten orientiert sich die Schulinformatik? Natürlich am Lehrplan wird die erste Antwort lauten, er gibt doch verbindlich Bildungsziele, didaktische Grundsätze und Lehrstoff vor. Nun weiß ich aber aus meiner fast zwanzigjährigen Tätigkeit als Informatiklehrer, dass der aktuell gültige Lehrplan aus Informatik dem Lehrer einen großen Interpretationsspielraum lässt. Engagierte Kollegen füllen diesen auch im besten Sinn für ihre Schüler. Daneben gibt es aber Kurse, in denen ein Minimalprogramm angeboten wird.

Karl Fuchs

Diese Angebotsbreite hat eine Ursache sicherlich in der spezifischen Situation der Lehrerausbildung in Österreich, gibt es doch erst seit ungefähr drei Jahren das Lehramtsstudium aus Informatik und Informatik - Management an österreichischen Universitäten. Seinerzeitige Ausbildungs- und Fortbildungsangebote aus Informatik an Universitäten und Pädagogischen Instituten wurden von den Lehrern in höchst unterschiedlicher Weise genützt.

Ausgelöst durch Profilbildungen und Schwerpunktsetzungen an einzelnen Schulen wurden zusätzlich Speziallehrgänge an den Pädagogischen Instituten und Erwachsenenbildungseinrichtungen eingerichtet.

Bei aller Berechtigung, ja Notwendigkeit von Spezialangeboten für unsere Schüler, so trugen doch auch diese Kurse wesentlich dazu bei, dass die ‚bunte Wiese' im allgemeinbildenden Unterrichtsfach Informatik, durch diese Schwerpunkte beeinflusst, noch bunter wurde und wird.

Will man also die Stellung des Unterrichtsgegenstandes als selbstständiges Fach sichern und argumentieren, so wird man die Frage stellen müssen: Gibt es in dieser Vielfalt so etwas wie Grundprinzipien oder integrierende Ideen? Zeitlose Ordnungsprinzipien herauszufinden und zu benennen gehört wohl zu den schwierigsten Aufgaben des Didaktikers, denn bei diesen Ideen handelt es sich nicht nur um einzelne Kapitelüberschriften, sondern um Handlungen und Strategien, die in unterschiedlichen Themen und wechselnder Komplexität zum Einsatz kommen. Bereits Anfang der 90er Jahre wurde in didaktischen Arbeiten auf die strukturierende Kraft fundamentaler Ideen für die Informatik hingewiesen [FUCHS, 1994; KNÖß 1989]. Zusätzlich wurden Fundamentale Ideen in Lehrwerke zur Didaktik der Informatik [*Modellierung* in HUB-WIESER, 2000, S. 85ff; *Formalisierung, Automatisierung und Vernetzung* in BAUMANN, 1996, S. 51ff] aufgenommen.

Diese Bemühungen wurden von den Didaktikern an den Hochschulen und Universitäten mit großem Interesse wahrgenommen. Ich bezweifle aber, ob diese Diskussion und ihre Ergebnisse einen großen Adressatenkreis in der Praxis gefunden hat. Dies ist bedauerlich, denn in der Lehre an der Universität sowohl im Lehramt Informatik in Salzburg als auch im Universitätslehrgang ‚Informatik für Lehramtsstudierende" an der Universität Innsbruck konnte ich das Bemühen der Studierenden nach Präzisierung und Detaillierung in der Informatik spüren. Vor allem waren sie für die Hinweise, welche Orientierungshilfe die Fundamentale Ideen bei der großen Stofffülle und dem raschen Anwachsen aktueller Themen geben können, dankbar.

Wenn wir uns also nun darauf einigen können, dass es für den allgemein-bildenden Unterrichtsgegenstand Informatik unbedingt erforderlich ist, einige wenige ‚inhaltliche Eckpfeiler' zu formulieren, dann kann ich es auch wagen, diese fundamentalen Ideen, die mir in den letzten Jahren geholfen haben, die Fülle an Lehrstoff in Informatik über die einzelnen Schulstufen hinweg vertikal zu gliedern, zu benennen. Zusätzlich möchte ich darauf hinweisen, dass die Diskussion vor allem dem selbstständigen Unterrichtsfach Informatik und nicht so sehr den in Österreich zahlreichen Bemühungen um Integration des Computers in viele Unterrichtsfächer gilt.

Fundamentale Ideen der Informatik

Bereits 1992 habe ich *Daten- und Beziehungsstrukturen* und *Objekt - Operation* als Fundamentale Ideen diskutiert und ihre strukturierende Kraft anhand zahlreicher Beispiele illustriert [FUCHS; 1992, S. 43ff]. Obwohl jegliche Bewertung fundamentaler Ideen schwierig ist, gehören die beiden Ideen wohl bis heute zu den anerkannt zentralen Leitideen der Informatik. Beide Ideen wurden auch in einer Liste von Grundprinzipien von Studierenden im Lehramt im fachdidaktischen Seminar bei Kollegen Helmut Caba genannt [vgl. LA *Studium in Salzburg*, CABA, 2003]. Eine sehr umfangreiche und gründliche didaktische Diskussion über die Bedeutung von Daten- und Beziehungsstrukturen für die (angewandte) Informatik findet sich in der von mir betreuten Dissertation von Frau Hilde Kletzl [KLETZL, 2002]

Neben den beiden genannten Ideen habe ich mittlerweile Modellbildung /Modellierung und Algorithmus als Ordnungsprinzipien erkannt und in meinen Katalog fundamentaler Ideen der Informatik aufgenommen [FUCHS, 2002]. Wie bereits erwähnt, nimmt der Modellbegriff eine sehr zentrale Rolle im Lehrbuch für Didaktik der Informatik ein.

In diesem Buch finden sich auch sehr schöne Beispiele den Modellierungsbegriff auf unterschiedlichste Weise im Unterricht anzusprechen [HUBWIESER, 2000, S. 135ff].

Dass es eigentlich unmöglich ist, einzelne Ideen scharf zu trennen, soll die nachfolgende Charakterisierung eines Algorithmus von Gerald Futschek zeigen: „... Zur Lösung eines Anwenderproblems entwirft der Informatiker zunächst ein Modell der Anwendung, (...) Das erste Modell wird in eine Reihe neuer Modelle umgeformt, die immer genauer und formaler werden, bis ein ablauffähiges Modell in einer bestimmten Programmiersprache erreicht ist..." [FUTSCHEK, S. 2/3, in: REITER, RIEDER, 1990]

Die Orientierung seines Algorithmusbegriffs am Modellierungsbegriff ist evident und auch vernünftig, denn zwangsläufig wird man, wenn man sich mit der Rolle des Programmierens im Informatikunterricht kritisch auseinandersetzt, über das ausschließliche Erlernen der Syntax einer Programmiersprache hinaus zu einem logischen Herangehen an ein Problem und einem modulartigem Lösen von Problemen gelangen müssen. Dem gemäß findet sich auch in manchen Katalogen berechtigterweise das Modulprinzip/Modularisierung [HUBWIESER, 2000, S. 103; BAUMANN, 1996, S. 261ff] als fundamentale Idee. Somit gehört zum Algorithmisieren die Analyse der Beziehungen der einzelnen Schritte zur Problemlösung und deren grafische Repräsentation ebenso wie die Implementation zum Testen des (einfachen) Moduls/zur Simulation des abstrakten Modells und es ist „... jedenfalls das Verfahren und nicht das Programm selbst, welche untersucht werden muss, um zu erfahren, wie an ein Problem herangegangen wird..." [SEDGEWICK, 1994].

Nicht zuletzt wird damit also deutlich, welchen Beitrag fundamentale Ideen zur Wissensorganisation und Wissensrepräsentation (also insgesamt zum Wissensmanagement) in einem allgemeinbildenden Informatikunterricht leisten.

Ich bin mir bewusst, dass jede Liste - so auch diese - fundamentaler Ideen unvollständig ist. Vielmehr möchte ich mit diesem Beitrag meinen Kollegen bewusst machen, wie wichtig es für das Fach Informatik ist, zunächst für sich selbst, dann aber für die Schüler, grundlegende Strukturen sichtbar und fassbar zu machen.

Literatur

[1] BAUMANN, Rüdeger (1996): Didaktik der Informatik. Klett Verlag, Stuttgart, München

[2] FUCHS, Karl Josef (1994): Didaktik der Informatik: Die Logik fundamentaler Ideen. In: Schulpraxis Heft 4+5, S. 42-45

[3] FUCHS, Karl Josef (2002): Methodik und Didaktik des Informatikunterrichts. Skriptum zu Vorlesung und Übung, Universität Innsbruck, erstellt von Cornelia Lederle

[4] FUTSCHEK, Gerald (1990): Informatik als Wissenschaft. In: REITER, Anton/RIEDER, Albert: Didaktik der Informatik, Verlag Jugend und Volk, Wien

[5] HUBWIESER, Peter (2000): Didaktik der Informatik. Springer Verlag, Berlin, Heidelberg

[6] KLETZL, Hilde (2002): Daten- und Beziehungsstrukturen - Eine didaktische Analyse im Spannungsfeld von angewandter Informatik und angewandter Mathematik. Dissertation Universität Salzburg

[7] KNÖß, Petra (1989): Fundamentale Ideen der Informatik im Mathematikunterricht: Grundsätzliche Überlegungen und Beispiele für die Primarstufe, E. Chr. Wittmann, Ed. Wiesbaden, DUV

[8] SEDGEWICK, Robert (1994): Algorithmen. Verlag Addison - Wesley, Bonn, München

Autor

Univ. Doz. Prof. Mag. Dr. Karl Fuchs
Universität Salzburg, Institut für Didaktik der Naturwissenschaften
E-Mail: Karl.Fuchs@sbg.ac.at
Homepage: http://www.sbg.ac.at/did/mathdid/personal/mathdid_personal_fuchs.htm
Fuchs unterrichtet am BG/BRG Hallein Mathematik, Informatik
Mitglied in der Didaktikkommission der Österreichischen Mathematischen Gesellschaft;
Betreuung und 10 Jahre Delegationsleiter der Schülergruppen des BMUK zur Internationalen Olympiade in Informatik; Professor für Didaktik der Mathematik und Informatik an der Universität Salzburg und Innsbruck (http://www.sbg.ac.at/did/home.htm);
Positionierung der Fachdidaktik

Fundamentale Ideen im Informatikunterricht am Beispiel der objektorientierten Softwareentwicklung

Die Informatik war als Wissenschaft schon immer von einer gewaltigen Dynamik geprägt. Eine Neuerung jagt die andere. Neue Paradigmen und Technologien, die ganze Teilbereiche der Wissenschaft grundlegend verändern oder sogar neu schaffen, kündigen sich in relativ kurzen Zeitabständen an. Um dieser Dynamik in der Schule Rechnung tragen zu können, ist eine Konzentration auf grundlegende Inhalte notwendig.

Helmuth Caba
Claudio Landerer
Universität Salzburg

Konzept der fundamentalen Ideen

Andreas Schwill schreibt dazu in [SCH]:
„Daher müssen sich die Inhalte im Informatikunterricht bis auf weiteres an den langlebigen Grundlagen der Wissenschaft orientieren. Es ist unverzichtbar, dass den Schülern ein Bild von den grundlegenden Prinzipien, Denkweisen und Methoden (den fundamentalen Ideen) der Informatik vermittelt wird."

Nun ist dieses das Konzept der fundamentalen Ideen natürlich nichts Neues ([BRU]). Dieser Artikel hier beschäftigt sich auch nicht mit dem Konzept an sich, sondern mit den Konsequenzen, die sich aus dessen Anwendung für konkrete Unterrichtsinhalte ergeben. Definitionsgemäß sind fundamentale Ideen durch vier Kriterien charakterisierbar:

(1) Sie sind in mehreren Teilgebieten des Fachbereichs zu identifizieren (**Horizontalkriterium**),
(2) über mehrere intellektuelle Niveaus hinweg vermittelbar (**Vertikalkriterium**),
(3) müssen einen Bezug zur Alltagswelt feststellen lassen (**Sinnkriterium**)
(4) und in einer Wissenschaft über längere Zeit von Relevanz sein (**Zeitkriterium**).

Wie sich unschwer erkennen lässt, sind fundamentale Ideen insbesondere wegen der Erfüllung dieser Kriterien für den Unterricht interessant. Der wichtigste Vorteil, der durch eine Vermittlung von fundamentalen Ideen jedoch entsteht, ist die Befähigung zum so genannten nicht-spezifischen Transfer. Die Schüler sollen dabei lernen, wie in der Vergangenheit erworbene Kenntnisse, durch Anpassung oder Erweiterung auf neue Problemsituationen angewendet werden können.

Umsetzung von fundamentalen Ideen im Unterricht

Für die konkrete Umsetzung von fundamentalen Ideen im Unterricht hat Jerome Bruner das **Konzept der Curriculumspirale** eingeführt (hier aus [SCH]):
„Der Anfangsunterricht ... sollte so angelegt sein, dass diese Fächer mit unbedingter intellektueller Redlichkeit gelehrt werden, aber mit Nachdruck auf dem intuitiven Erfassen und Gebrauchen dieser grundlegenden Ideen. Das Curriculum sollte bei seinem Verlauf wiederholt auf diese Grundbegriffe zurückkommen und auf ihnen aufbauen, bis der Schüler den ganzen formalen Apparat, der mit ihnen einhergeht, begriffen hat."

Eine Umsetzung von fundamentalen Ideen im Unterricht wird bedeutend erleichtert bzw. überhaupt erst ermöglicht, **wenn die grundsätzliche Vorgangsweise am Spiralprinzip von Jerome Bruner ausgerichtet ist und die Vorteile, die sich aus der Erfüllung der genannten Kriterien ergeben, ausgenutzt werden**. Diese einfachen Feststellungen haben weitreichende Konsequenzen.

(1) Inhalte, mit denen fundamentale Ideen im Unterricht vermittelt werden sollen, sollen demnach so früh wie möglich eingeführt und in diesen frühen Phasen möglichst praxis- und anwendungsorientiert umgesetzt werden. Die entsprechenden Grundlagenkonzepte müssen sich wie ein roter Faden durch die Schulkarriere der Schüler ziehen. Immer wieder sollte der Lehrer auf einer neuen intellektuellen Ebene bzw. auf höheren Abstraktionsniveaus auf zu diesen roten Faden zurückkommen, die vorherig eingeführten Konzepte aufgreifen, vertiefen und mit neuen Konzepten erweitern (Vertikalkriterium). Fundamentale Ideen sind dazu gut geeignet, weil sie dem Vertikalkriterium genügen.

(2) Ebenso wird der Lehrer in verschiedenen Teilgebieten des Gegenstandes immer wieder auf die grundlegenden Ideen zurückkommen und die Schüler auf Parallelen zwischen den Bereichen hinweisen. Erst dadurch können Schüler den fundamentalen Charakter einer Idee wirklich verstehen: (Horizontalkriterium).

(3) Zusätzlich sollten sich die für den Transport von fundamentalen Ideen ausgewählten Inhalte & Problemstellungen sowohl am Sinn- als auch am Zeitkriterium orientieren.

Fallbeispiel: Programmieren Softwareentwicklung im Informatikunterricht

Was diese drei Feststellungen konkret bedeuten könnten, soll nun am Beispiel der Programmierung Softwareentwicklung im Unterricht verdeutlicht werden.

1) Identifikation der fundamentalen Ideen

Führen wir uns zunächst vor Augen, welche fundamentalen Ideen in diesem Bereich der Informatik anzusprechen sind. Dazu hat Andreas Schwill in einem seiner Artikel ([SCH]) aufgezeigt, wie sich fundamentale Ideen identifizieren lassen und hat für den konkreten Inhalt der Softwareentwicklung folgende Metaideen Ideen angeführt:

(1) Algorithmisierung mit den zugehörigen Entwurfsprinzipien (Divide & Conquer, Branch and Bound usw.), den Programmierkonzepten (Iteration, Alternative usw.), dem Ablauf (Prozess, Nebenläufigkeit) und der Evaluation.

(2) Strukturierte Zerlegung mit den Teilgebieten Horizontalisierung, Vertikalisierung und Orthogonalisierung.
(Ein System kann immer weiter zerlegt werden, bis ich schließlich an atomaren Bausteinen angelangt bin, die sich nicht mehr zerlegen lassen. Zerlegung eines Systems in seine atomaren Bestandteile aus welchen das gesamte System rekonstruiert werden kann.)

(3) Konzept der Sprache mit Syntax und Semantik.

2) Mögliche Fehler in der Umsetzung

Der Stellenwert der Programmierung wurde seit ihrer Einführung im Informatikunterricht immer wieder kontroversiell diskutiert. Einige Argumente für diesen Unterrichtsinhalt waren z.B., dass Programmieren Spaß macht, einen handlungs- und problemorientierten Unterricht ermöglicht (wodurch die Motivation der Schüler steigt) und dass durch die Programmierung im Unterricht eine Reihe von fundamentalen Ideen transportiert werden kann. Aber es existierten auch viele Gegenargumente. So wurden z.B. kritische Stimmen laut, die die geringe Brauchbarkeit und Attraktivität der Lösungen bei "unverhältnismäßig großem" Zeitaufwand oder aber die Kurzlebigkeit der erworbenen Kenntnisse und vor allem die Begebenheit, dass Schüler den fundamentalen Charakter der Programmierung eigentlich gar nicht begreifen, als Nachteile nannten. Was dabei jedoch nicht gesehen wurde, war die Tatsache, dass nicht der Inhalt der Programmierung an sich sondern dessen konkrete Umsetzung diese Probleme verursachte. Bloße (oft von mathematischen Problemstellungen geprägte) Algorithmierung ohne passende Modellierung, Konzentration auf die Syntax der Sprache, schlechte Beispiele im Sinne der allgemeinen Bedeutung, die unzureichende Integration in andere Teilbereiche der Informatik und vor allem das Kurssystem, nach dem Programmieren in der Schule unterrichtet wurde, waren einige der Fehler in der Vorgangsweise zur Umsetzung. Aufgrund dieser Feststellungen kann man nun in Anlehnung an das Spiralprinzip und an die genannten Kriterien einige Forderungen bezüglich dieser Umsetzung stellen.

3) Forderungen

(1) Zu Beginn ist festzustellen, dass im Rahmen einer Programmierung im Unterricht nicht die Sprache im Mittelpunkt der Betrachtung stehen kann, sondern die Problemstellung bzw. deren Modellierung (siehe [HUB] S. 85-90). Demnach gibt es im Informatikunterricht keine Programmierung im engeren Sinne mehr. Wir sollten fortan von Softwareentwicklung sprechen. Dazu werden die Schüler zuerst immer das Problem aus der realen Welt analysieren, die Anforderungen an ein Informatiksystem definieren, anschließend ein entsprechendes System modellieren und dann implementieren bzw. das Ergebnis der Implementierung interpretieren. Den Mittelpunkt bilden dabei die verschiedenen Modellierungstechniken. Im Anschluss an die Modellierung wird das Modell zur Veranschaulichung bzw. Überprüfung implementiert (Simulation). Im Zuge dieser Implementierung liegt der Fokus jedoch wiederum nicht auf der Syntax der Sprache, sondern, wie auch bei der Modellierung selbst, auf der Vermittlung der fundamentalen Ideen, die damit in Kontakt stehen. Die so vermittelten Kenntnisse sind nachhaltig und von allgemeiner Bedeutung für den Gegenstand. Auch hinsichtlich des Zeitkriteriums scheinen diese Forderungen relevant. Würden wir uns nämlich bei der konkreten Umsetzung auf zu spezielle Inhalte konzentrieren (z.B. auf die Syntax einer Sprache), dann würden wir den Vorteil vernichten, der sich für fundamentale Ideen im Unterricht aus der Erfüllung eben dieses Zeitkriteriums ergibt.

(2) Weiters muss mit der Einführung von Modellierungstechniken bzw. Programmiersprachen bereits möglichst früh begonnen werden. Die theoretische Untergrenze liegt bei etwa 12 Jahren. Von da an sind Kinder in entwicklungspsychologischer Hinsicht für einen Einstieg bereit. Über die Jahre hinweg werden dann entsprechend dem Prinzip der Curriculumspirale immer wieder neue Sprachkonzepte Ideen eingeführt bzw. alte Konzepte ausgebaut und vertieft.

(3) Hinzu kommt die Notwendigkeit der Identifikation der grundlegenden Ideen auch in anderen Teilbereichen des Gegenstandes. Die Lehrer sollten also Parallelen erkennen und die Schüler nicht nur darauf hinweisen, sondern diese Chance ausnutzen und

die Themenbereiche integriert behandeln. Dieser Forderung nach einer Einflechtung von Inhalten wurde mittlerweile im neuen Informatiklehrplan für die AHS-Oberstufe in Österreich ebenfalls Nachdruck verliehen. Insbesondere die Punkte 2 und 3 verdeutlichen, dass das traditionelle Kurssystem, nach dem bis in die mittleren bzw. höheren Oberstufenklassen Konzepte von Programmiersprachen oder auch Modellierungstechniken nie zur Debatte standen, dann jedoch in mehrmonatigen abgeschlossenen Blöcken auf einen Schlag eine Programmiersprache "gelernt" wurde (eben ein Kurs), abzulehnen ist. Durch ein Kurssystem wird weder das Konzept der Curriculumspirale berücksichtigt, noch werden die Chancen genutzt, die sich aus der Erfüllung des Horizontal- bzw. des Vertikalkriteriums ergeben. Die fundamentalen Konzepte können nicht einmal in Ansätzen transportiert werden, wodurch ein großer Vorteil und damit eine wichtige Legitimationsbasis des Inhaltes unweigerlich verloren geht.

(4) Schließlich sollte bezüglich der gewählten Problemstellungen noch Folgendes bemerkt werden. wird eine Forderung hinsichtlich der gewählten Problemstellungen gestellt. Wird das Sinnkriterium, das für fundamentale Ideen selbst gefordert wird, auch auf die Inhalte und Problemstellungen übertragen, mit welchen diese fundamentalen Ideen transportiert werden sollen, dann orientiert sich die Auswahl des konkreten Inhaltes in erster Linie an der realen Lebenswelt der Schüler. Hier auf komplizierte Problemstellungen zu setzen, die keinerlei Verbindung zum Alltag der Schüler zulassen, ist eindeutig der falsche Weg.

4) Skizze einer Beispielumsetzung

Es soll nun kurz eine mögliche Umsetzung skizziert werden, die sich in ihrer Ausführung an das Spiralprinzip und an die Berücksichtigung der genannten Kriterien und Forderungen hält. Als aktuelles Beispiel dient uns dazu die objektorientierte Softwareentwicklung in der Schule. Ob die Objektorientierung dabei als fundamentale Idee der Informatik bezeichnet werden kann, ist zu untersuchen. Für die Schulinformatik besitzt sie in jedem Fall fundamentalen Charakter. Die erste Frage, die sich bezüglich unseres Fallbeispiels stellt, ist jene nach dem Zeitpunkt bzw. der Art der Einführung in das Konzept der Objektorientierung. Sofort mit einer objektorientierten Sprache in die Programmierung Softwareentwicklung einzusteigen (OO-First) ist sowohl hinsichtlich der Programmiersprachenkonzepte als auch hinsichtlich der benötigten Modellierungstechniken relativ sinnlos, wenn vorher nicht entsprechende Vorarbeiten geleistet wurden. Eine derartige Vorgangsweise würde zu einem „prozeduralen Missbrauch" führen und kann keinen Mehrertrag im Sinne einer Objektorientierung bringen, sondern lediglich zur Verwirrung beitragen. Trotzdem soll das Konzept an sich anhand von praktischen Anwendungsbeispielen so früh wie möglich eingeführt werden, um eine korrekte Terminologie und ein passendes mentales Modell der Idee zu vermitteln. Hinsichtlich der Einführung der objektorientierten Softwareentwicklung entsteht somit die Notwendigkeit der Verfolgung eines hybriden Ansatzes, der sich angelehnt an das Konzept von Peter Hubwieser ([HUB]) wie folgt skizzieren lässt.

Spirale 1:
Terminologie & mentales Modell der Objektorientierung

Der Lehrer sollte bereits von den ersten Stunden an versuchen, die Ideen der Objektorientierung und auch die Ideen der Softwareentwicklung selbst in anderen (und meist noch relativ konkreten) Unterrichtsinhalten zu identifizieren und über deren konkrete Anwendung zu transportieren (siehe [HUB] S. 111–119). Der Unterstufenbereich bzw. der untere Oberstufenbereich eignet sie dazu in ganz besonderer Weise, weil die Schüler hier für gewöhnlich in verschiedene Arten von Anwendersoftware eingeführt werden. Dabei steht jedoch nicht das Anwenderprogramm an sich, sondern eine analysierende Betrachtung hinsichtlich der Datenstrukturen der jeweiligen Anwendung im Vordergrund. Denken wir bezüglich der Objektorientierung dabei z.B. an die Kontextmenüs oder die Objektinspektoren, mit denen Eigenschaften von Elementen aus Anwenderprogrammen abgerufen und geändert werden können. Dabei steht jedoch nicht das Anwenderprogramm an sich, sondern eine analysierende Betrachtung hinsichtlich der Datenstrukturen der jeweiligen Anwendung im Vordergrund. Hubwieser zeigt in seinem Konzept, wie eine Einführung in die Basiskonzepte Grundideen der Objektorientierung über eine analysierende Betrachtung von Graphikprogrammen, oder auch Textverarbeitungssystemen usw erfolgen aussehen kann. Eine Vertiefung erfolgt dann z.B. im Zuge der datenorientierten Modellierung (ER-Diagramme). Dort werden die Konzepte weiter abstrahiert und hin in Richtung „abstrakter" Objektorientierung ausgebaut (Generalisierung —> Vererbung, Entitäten —> Objekte, Kardinalitäten —> Objektbeziehungen usw). Dieser hier an zwei Beispielen vorgeführte Brückenschlag von einem an sich abstrakten Konzept hin zur konkreten Anwendung gilt als zentraler Aspekt der Currriculumspirale und ist somit für eine Einführung von fundamentalen Ideen von Bedeutung.

Spirale 2:
Modellierung & Programmiersprachenkonzepte

In dieser zweiten Spirale geht es darum, den Schülern zu zeigen, wie man modelliert und programmiert. Es geht nicht um die Vermittlung einer Programmiersprache. In jeder Stufe

werden dabei alte Konzepte aufgegriffen, ausgebaut bzw. vertieft und neue Konzepte eingeführt. Besondere Berücksichtigung verdient die Reihenfolge, in der die Konzepte eingeführt werden sollten.

(1) Am Beginn stehen in jedem Fall die Begriffe Algorithmus, Anweisung und Sequenz, umgesetzt über eine umgangssprachliche Definition von Algorithmen bzw. deren Ausführung über Roboterspiele (Schüler als Roboter).

(2) Danach werden die eingeführten Konzepte um die zentralen Begriffe Ideen der Variablen (Typen, Zuweisung, Operatoren) bzw. deren Zuweisung, der Operatoren und der Ablaufsteuerung (Fallunterscheidung, Wiederholung) erweitert. Dies könnte z.B. über eine Anwendung von Mini-Languages wie LOGO, Roboter Karel ([KAR]) oder auch dem Java-Hamster ([HAM]) erfolgen. Sowohl der Java-Hamster als auch Roboter Karel (in Varianten) unterstützt Objektorientierung. Entscheidet man sich für die objektorientierten Varianten, dann liegt der Schwerpunkt vorerst jedoch nicht auf dem Aufbau von Objekten (Klassendefinitionen) sondern auf der Erstellung und Verwendung von Objekten (new-Operator bzw. Dot-Notation) im Zuge der oben genannten Schwerpunkte. Modellierung bezieht sich in dieser Phase noch auf die Darstellung von Verarbeitungsvorschriften z.B. über einfache Funktionsmodelle (z.B. Programmablaufpläne oder Struktogramme).

(3) Im Anschluss daran werden dann die Prozeduren bzw. Funktionen eingeführt (Modularisierung) und die Konzepte der Ablaufsteuerung vertieft. Insbesondere könnte in dieser Stufe auch eine Einführung von wichtigen Datenstrukturen wie z.B. dem Array oder dem Record als Vorstufe zum Objekt erfolgen. Modellierung wird jetzt als Mittel zur Systembeschreibung gesehen. Nun werden Verhaltensmodelle (z.B. Zustandsdiagramme) eingeführt und die bereits bekannten Funktionsmodelle vertieft (z.B. Prozeduraufrufgraphen oder Datenflussdiagramme) funktionale Modellierungstechniken spielen hier eine wichtige Rolle. Als Werkzeuge zur Implementierung eignen sich z.B. traditionelle Sprachen wie Pascal oder Makrosprachen in Office Produkten. Auch eine Verwendung von LEGO-Mindstorms ([LEG]) oder z.B. von Construction-Sets (siehe [TES]) ist aufgrund ihres motivationalen Charakters zu überlegen. Weiters ist eine fortgeschrittenere Nutzung von Mini-Languages möglich, wenn es die Schülerinteressen zulassen.

Zusammenführung:
Bis spätestens zur 11. Schulstufe müssen beide Spiralen so weit entwickelt sein, dass ein Einstieg in die objektorientierte Softwareentwicklung möglich wird. D.h. alle notwendigen programmiertechnischen Konzepte wurden vermittelt und das Konzeptdie Idee der Objektorientierung wurde eingeführt. Außerdem hat der Schüler bereits einige wichtige Modellierungstechniken kennen gelernt, die im Zuge der objektorientierten Modellierung weiterverwendet bzw. ausgebaut werden können.

(1) Zum Einstieg in die objektorientierte Softwareentwicklung eignen sich nun Werkzeuge wie z.B. BlueJ ([BLU]). oder auch andere Case-Tools. Sie können verwendet werden, um den Übergang zwischen einer praktischen Anwendung von Objekten und deren Verwendung in der Softwareentwicklung zu erleichtern.

(2) Insbesondere in der objektorientierten Softwareentwicklung ist eine gewissenhafte Modellierung des zu erstellenden Systems von Bedeutung. Neben der hier wichtigen umgangssprachlichen Beschreibung. Es sollten sowohl statische Klassen- und Objektmodelle (z.B. Klassendiagramme, Klassendefinitionen) als auch dynamische Interaktionsmodelle (z.B. Interaktionsdiagramme, Sequenz- und Aktivitätsdiagramme, strukturierter Text usw.) objektorientierte Modellierungstechniken eingeführt werden (Tools dazu siehe [UML]). Außerdem können auch bereits bekannte Modellierungstechniken weiterverwendet werden. So eignen

sich z.B. State-Charts zur Verhaltensmodellierung von Klassen oder spezielle Formen des Datenflussdiagramms (Kontextdiagramme, Anwendungsfalldiagramme) zur Interaktionsmodellierung.

(3) Im Bezug auf die Implementierung stehen steht zunächst der Unterschied zwischen Objekt und Klasse bzw., der Aufbau und die Verwendung von Objekten eines Objektes in einer objektorientierten Sprache und die Verwendung von Objekten in einer objektorientierten Sprache im Vordergrund. Als Sprachen werden jetzt z.B. Java oder C# vorgeschlagen. Integrierte Entwicklungsumgebungen ([IDE]) sollten verwendet werden, um die konkrete Codierung zu erleichtern. An dieser Stelle sei auch darauf hingewiesen, dass LEGO-Mindstorms vollkommen objektorientiert programmiert werden können (siehe [LEG]).

(4) Danach werden die wichtigsten Konzepte der Objektorientierung wie z.B. Kapselung, Vererbung, Polymorphismus usw. eingeführt.

(5) Die Realisierung der Kommunikation zwischen Objekten sollte ebenfalls schwerpunktmäßig behandelt werden (Objektvariablen).

D) Vertiefung - Abschluss

(1) In der letzten Schulstufe erfolgt dann eine Einführung bzw. Vertiefung von weiteren wichtigen Datenstrukturen (Keller, Listen, Bäume, Graphen, mehrdimensionale Arrays usw.) und den entsprechenden Algorithmen darauf.

(2) Einen Abschluss findet die objektorientierte Softwareentwicklung in der Schule mit der Einführung einfacher objektorientierter Entwurfsmuster, anhand derer die Vorteile, die sich aus einer objektorientierten Vorgangsweise in der Softwareentwicklung ergeben, verdeutlicht bzw. die bisher erlernten Konzepte vertieft werden sollen ([GAM] und [PRE]).

E) Einflechtung

Insbesondere die Softwareentwicklung ist als Ausgangspunkt für die geforderte Einflechtung in andere Teilgebiete des Gegenstandes gut geeignet. So ergeben sich nicht nur Verbindungen über den zentralen Begriff des Algorithmus, sondern insbesondere auch über die beiden anderen von Andreas Schwill identifizierten Metaideen (siehe [SCH]).

Eine Frage der Umsetzung

Es wurde nun beispielhaft skizziert, welche Auswirkungen es haben kann, wenn fundamentale Inhalte in der Schule transportiert werden sollen. Die Vermittlung von fundamentalen Konzepten bringt einen Mehrertrag für Schüler (nicht-spezifischer Transfer, umfassendes Verständnis, Nachhaltigkeit der Inhalte), für den Lehrer (Grundlagenthemen, Repertoire) und für den Gegenstand an sich (klare Positionierung und Abgrenzung). Im Rahmen der konkreten Umsetzung stehen und fallen diese Vorteile jedoch mit der Beachtung des Spiralprinzips und der Berücksichtigung der Möglichkeiten, die sich für fundamentale Ideen im Unterricht aus der Erfüllung der genannten Kriterien (Horizontal-, Vertikal-, Zeit- und Sinnkriterium) ergeben.

Diese Arbeit ist im Rahmen des Fachdidaktischen Seminars (SS 03) unter der Leitung von Mag. Helmut Caba an der Universität Salzburg entstanden. Sie basiert auf der Annahme, dass der Schüler über mehrere Jahre hinweg Informatik als Freigegenstand oder als Wahlpflichtgegenstand wählen kann.

Literatur

[1] [HUB] HUBWIESER P.: Didaktik der Informatik. Grundlagen, Konzepte, Beispiele, Springer Verlag, Berlin-Heidelberg, 2001;
[2] [SCH] SCHWILL A.: Fundamentale Ideen der Informatik, Zentralblatt für Didaktik der Mathematik, 1993;
[3] [BRU] BRUNER J.: The Process of Education. Harvard University Press, Cambridge Mass. 1960;
[4] [GAM] GAMMA E. [et al]: Design Patterns, Elements of Reusable Object-Oriented Software, Addison Wesley, 2002;
[5] [PRE] PREE W: Design Patterns for Object-Oriented Software Development, Addison Wesley, 2000;

Autoren

Prof. Mag. Helmuth Caba,
Pädagogisches Institut und Universität Salzurg
E-Mail: helmut.caba@salzburg.at
Homepage:
http://www.sbg.ac.at/did

Bundeslehrer im Hochschuldienst, Mitwirkung am Studienplan Lehramt an der Naturwissenschaftlichen Fakultät, insbesonders im Unterrichtsfach Informatik und Informatikmanagement, Betreuung der Internetplattform Salzburger Bildungsservice, Betreuung der Internetplattform Begabtenförderung, Projekt IKT-Initiative Unterstufe AHS

Claudio Landerer,
Student des Lehramtsstudiums Informatik an der Universität Salzburg

Aporien[1] in der Schulinformatik

Peter Micheuz
Universität Klagenfurt

Gibt es in den traditionellen Schulfächern weitestgehend Konsens darüber, wann welche Lehrinhalte wie und wozu – das sind die Kernfragen der Fachdidaktik - unterrichtet werden, so ist das in der Schulinformatik schon wegen der Vielfalt an (schulautonomen) Organisationsformen nur schwer möglich. Die missliche Situation, dass im Informatikunterricht bisweilen irgendwas, irgendwann von irgendwem aus irgendeinem Grund unterrichtet wird, sollte bald der Vergangenheit angehören. Aber: „Es" wird einem ja wirklich nicht leicht gemacht. Zur Illustration möchte ich den neuen Informatik-Lehrplan für die 6. bis 8. Klasse der AHS-Oberstufe (Wahlpflichtfach) anführen. Als Lehrstoff werden folgende Themen vorgeschlagen: Grundprinzipien der Informationsverarbeitung, Konzepte von Betriebssystemen, Aufbau und Funktionsweise von Netzwerken, Datenbanken, Lern- und Arbeitsorganisation, Konzepte von Programmiersprachen, Künstliche Intelligenz, Erweiterung der theoretischen und technischen Grundlagen der Informatik, Grundlegende Algorithmen und Datenstrukturen, Informatik, Gesellschaft und Arbeitswelt, Rechtsfragen. Der Informatiklehrer ist für die Verteilung dieser Bereiche auf drei Jahre und die Tiefe der Vermittlung selbst verantwortlich. Als Bildungs- und Lehraufgabe ist das sattsam bekannte Kategorientripel Sach-, Selbst- und Sozialkompetenz angeführt und die Schulinformatik soll auch zur Schulung abstrakten Denkens beitragen. Als didaktische Grundsätze werden modul- und projektartiges Arbeiten sowie Aktualitätsbezüge in den Vordergrund gestellt. Die Lehrinhalte sollen, wenn möglich, in vernetzter Form behandelt werden. Dieser offene Rahmen-Lehrplan führt uns zur ersten Aporie.

Zentral verordnete Lehrpläne versus IT-Zertifikate und Standards

Die Informatik unterrichtende Lehrperson sieht sich Lehrplänen gegenüber, die zwar den Bildungsauftrag[2] der Schule erfüllen, aber sehr viele Fragen aufwerfen. Wer bestimmt bei sehr offenen Lehrplänen den detaillierten Lehrstoff und die sogenannte Lehrstoffverteilung? Der verantwortungsvolle, über den Dingen stehende, selbstbewusste Lehrer? Es ist ein offenes Geheimnis, dass in vielen Fällen approbierte Lehrbücher diese nicht einfache Aufgabe übernehmen. Das geflügelte Wort „Lehrbücher sind die geheimen Lehrpläne" dürfte in Lehrerkreisen nicht unbekannt sein. Und gute approbierte Informatik-Lehrbücher sind in Österreich (noch) nicht vorhanden.

Eine absolut notwendige Qualitätsentwicklung und –sicherung im schulischen IT-Bereich kann sich nicht auf verordnete, weit gesteckte Rahmen-Lehrpläne (Input-Steuerung) verlassen. Der Gesetz- und Auftraggeber, also das zuständige Ministerium sollte daran interessiert sein, dass die z.T. nicht unerheblichen Ressourcen (Hard- und Software, Lehrergehälter) auch zu einem entsprechenden Output (Kenntnisse, Fertigkeiten, Kompetenzen der SchülerInnen) führen. Im Zuge transnationaler Studien wie z.B. PISA[3] werden die Standards unserer SchülerInnen auf den Prüfstand gestellt. Der ECDL[4], wie immer man zu ihm stehen mag, gibt einen internationalen Quasi-Standard vor, indem er das Erreichen detaillierter Lernziele durch eine Prüfungsorganisation sicherstellt bzw. sicherstellen soll. Sind IT-Zertifikate also der Schulweisheit letzter Schluss? Ich meine, dass ein konzeptuell gut angelegter Informatikunterricht die Basis dafür legen soll, SchülerInnen in die Lage zu versetzen, diese Zertifikate – wenn möglich im Selbststudium - auf freiwilliger Basis erlangen zu können. Es wäre unklug – obwohl in vielen Schulen Realität – IT-Zertifikatskurse als totalen Ersatz für qualifizierten Unterricht zu instrumentalisieren. Ein zu starkes Diktat außerschulischer Institutionen und Firmen käme der Selbstaufgabe des Bildungsauftrages des (öffentlichen) Schulwesens gleich. Die im Zuge der Diskussion um Standards im Informatikunterricht befürchtete Einschränkung der Lehrfreiheit ist meiner Meinung nach unbegründet, wenn ein vernünftiger Kompromiss gefunden wird. Standards dienen nicht nur im Informatikunterricht der Orientierung für LehrerInnen und SchülerInnen und bieten ein gewisses Maß an Qualitätsabsicherung nach unten. Ein adäquater Informatikunterricht sollte sowohl der Vermittlung von Pflichtteilen (Kernbereich, Standard) als auch von „Küranteilen" (Erweiterungsbereich) Rechnung tragen. In Abhängigkeit der Lehrerkompetenz und der Schülerdisposition – nicht selten gibt es im Informatikunterricht das Problem, dass Informatik-Lehrkräfte bisweilen nicht können und SchülerInnen nicht immer wollen – soll und muss es natürlich von Fall zu Fall Freiräume im Sinne von speziellen, auch vertiefenden, der Unterrichtssituation angepassten und die SchülerInnen fordernden Themenbereichen geben.

Konzeptwissen versus Produktwissen

Folgender Dialog spielte sich heuer als Auftakt für eine angeregte Diskussion in einem österreichischen Lehrer-

[1] *Aporie = Widerspruch, der nicht durch eine logisch-rationale Entscheidung eliminierbar ist bzw. Konflikt, in dem beide Parteien Recht haben (vgl. Pietschmann: Eiris&Eirene, Ibera Verlag, 2002).*

[2] *Die österreichische Schule hat die Aufgabe, an der Entwicklung der Anlage der Jugend nach den sittlichen, religiösen und sozialen Werten des Wahren, Guten und Schönen durch einen ihrer Entwicklungsstufe und ihrem Bildungsweg entsprechenden Unterricht mitzuwirken" (Paragraph 2 des Schulorganisationsgesetzes 1962, das ist der Zielparagraph der Schule).*

[3] *Im Jahr 2006 sollen in der Altersstufe der 15jährigen auch IT-Kompetenzen überprüft werden.*

[4] *European Computer Driving License – Europäischer Computerführerschein*

forum ab. Ein AHS-Lehrer: „Ich bin der Meinung, dass wir unseren Schülern Konzepte und Grundlagen vermitteln sollen, nicht allzu spezielle Kenntnisse in einem bestimmten Office-Paket oder MCP."

Die harsche, aus Sicht des Autors nicht nur im Ton verfehlte Antwort eines BHS-Lehrers im Originaltext: „Nur Kleinkarierte, Dummköpfe, Verbohrte, Scheuklappenmenschen und eigentlich alle jene, die NICHT auf unsere Schüler losgelassen werden sollten, können gegen etwas sein, was - in welcher Form auch immer - unseren Schülern Starthilfe in Beruf oder Studium bringt. Ob Cisco, Linux, Microsoft, Mesonic, SAP, Macromedia, Adobe, C++, C# oder sonstiges: Irgendeiner findet immer das andere Paket besser und ein Haar in der Suppe! Aber soll dieser dümmliche „Glaubenskrieg" auf den Schultern unserer Schüler ausgetragen werden? Konzepte und Grundlagen sind gut, vor allem Konzepte über Konzepte der Grundlagen. Dafür hat ein anderer den Arbeits- oder Studienplatz! Gratuliere!"

Die Wahrheit liegt wohl wie so oft in der Mitte. Dieser scheinbare Widerspruch kann durchaus friedlich gelöst werden, indem der Unterricht so gestaltet wird, dass beide antagonistischen Standpunkte befriedigt werden.

Konzeptwissen und Produktwissen müssen keinen unüberwindbaren Gegensatz darstellen.

Als erstes Beispiel möchte ich anführen, dass der im Syllabus des ECDL im Modul Textverarbeitung aufscheinende Serienbrief, wenn er „konzeptlos" trainiert wird, mit Sicherheit zu dem erwartenden Effekt führt, dass der Anwender bei wenig Training oder bei einem Versionswechsel große Probleme hat.

Ein weiteres Beispiel sind assistentengestützte Abfragen einer Datenbank, wo vom didaktischen bzw. methodischen Standpunkt aus die begleitende Vermittlung von Datenbankkonzepten wie dem relationalen Datenmodell und SQL unerlässlich ist. Drittens möchte ich die derzeit so aktuelle Präsentation von Information im Web anführen, wo es methodisch äußerst unklug wäre, sich auf eine reine Produktschulung von bekannten Webeditoren zu verlassen. Das darunterliegende Konzept von Auszeichnungssprachen wie (X)HTML sowie das Verzeichnis- und Client-Server-Konzept muss parallel dazu unterrichtet werden. Sonst passiert es, dass ausschließlich die Software den Anwender steuert und nicht umgekehrt und beim geringsten Problem der Anwender im „Regen" steht.

Fragen Sie einen (Webeditor)Produkt-Geschulten ohne Konzeptwissen, wie ein einfacher Zähler auf einer Internetseite funktioniert!

Gute Lehrbücher tragen diesem Umstand Rechnung. Leider wird der österreichische Schulbuchmarkt unter dem Titel „Informatik" von vielen firmenspezifischen Produktschulungsunterlagen dominiert. Hier besteht sehr großer Handlungsbedarf nach zeitloseren Lehrbüchern, die mehr bieten als Online-Hilfen, die ohnedies Bestandteil jeder Software sind.

Eine entsprechende Lehreraus- und Fortbildung, und vor allem geeignete Lehrmittel und aufbereitete Inhalte sind mehr denn je gefragt. Das an mehreren Standorten[5] eingeführte Informatik-Lehramtsstudium sollte diesem Umstand Rechnung tragen.

Monolithische Kurssysteme versus curriculares Spiralprinzip

„Das Maß ist das Ziel" gilt wohl auch für den Informatikunterricht. Es macht wenig Sinn und ist schlichtweg langweilig, über einen langen durchgehenden Zeitraum im Unterricht z.B. nur Textverarbeitung, eine einzige Programmiersprache oder nur ein Programmierparadigma zu behandeln. Im Fall von Kursen an berufsbildenden Schulen, in denen man intensiv mit spezieller Software arbeitet, mag das sinnvoll sein. Im Informatikunterricht einer allgemeinbildenden Schule ist das nicht akzeptabel. Im Sinne von „Variatio delectat" sollten Lerninhalte abwechslungsreich dargeboten und von den SchülerInnen aktiv erworben werden.

Als Beispiel für variantenreiches, verzahntes Unterrichten soll folgendes scheinbar banale Beispiel der Ein- und Ausgabe einer einfachen Schülerliste dienen. Wer sagt denn, dass nur die professionelle Textverarbeitung einer speziellen Firma dafür geeignet ist? Einfache Editoren tun es am Anfang doch auch! Das Konzept der Formatierung und Strukturierung kann parallel dazu ebenso gut, wenn nicht besser, mit einer Tabellenkalkulation oder mit Präsentationssoftware gelöst werden! In weiterer Folge kann diese einfache Aufgabe (oder ist es ein Problem?) auch mit einem Webeditor gut erfüllt werden. Wenn es gar um die Schülerliste der ganzen Schule geht und damit die einfache Verwaltung eines Datenbestandes notwendig wird, ist der baldige Einsatz einer Datenbank angezeigt. Dieses einfache Beispiel zeigt auch, wie wichtig eine exakte Spezifikation ist. Das Problemlösen durch variantenreiche Anwendung von Software ist dazu geeignet, viele Konzepte und die Philosophie der Benutzeroberflächen transparent zu machen.

Möglichst bald soll den Schülern klar werden, dass die Informatik eine enormes Spektrum an Werkzeugen zur Verfügung stellt. Eine zu starke Produktfixierung ist abzulehnen. „Blindness of awareness" – eine zu enge Bindung einer Aufgabenstellung bzw. Problemstellung an ein Werkzeug bzw. Produkt - sollte bereits im Anfangsunterricht vermieden werden.

Nach dieser anfänglichen Breite können dann über längere Zeiträume hinweg (z.B. in der Oberstufe) Vertiefungen im Sinne des Spiralprinzips erfolgen. Es muss allerdings eingeräumt werden, dass Wiederholen und Üben, als Sportler würde man Training dazu sagen, Zeit kostet, aber für eine Nachhaltigkeit unabdingbar ist. Bereits erworbenes Wissen und Fertigkeiten in bestimmten Zeitintervallen aufzufrischen und zu vertiefen, sind für das Erreichen dieses Zieles sehr förderlich.

---

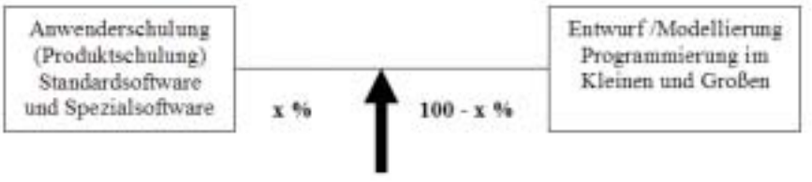
[5] *An den Standorten Wien, Salzburg und Klagenfurt gibt es das Informatik-Lehramtsstudium seit drei Jahren, in Linz seit einem Jahr.*

Beispiele dazu lassen sich viele anführen. So ist z.B. Standardsoftware heutzutage dermaßen komplex, dass sich viele Möglichkeiten eines spiralförmigen Ansatzes finden lassen. Sei es in der Textverarbeitung beginnend mit einem sanften Einstieg mittels einfach formatierter Dokumente, über einfache Automatisierungsoptionen (Vorlagen, Textbausteine, ...) bis hin zur Bearbeitung großer Dokumente und Layoutierung als Vorbereitung für das Verschriftlichen und Ausarbeiten von Referaten und Fachbereichsarbeiten.

In der Tabellenkalkulations- und Datenbanksoftware sind die möglichen, systematisch oder exemplarisch aufbauenden Betätigungsfelder weit größer (vgl. Neuwirth in dieser Sonderausgabe).

Anwendung versus Entwicklung

Wohl kaum ein anderes Schulfach hat sich mit dieser Dialektik so auseinanderzusetzen wie der Gegenstand Informatik. Das entbehrt nicht eines gewissen Reizes, gibt es doch darüber viele Debatten von Informatiklehrern bzw. Fachdidaktikern. An dieser Stelle sind Sie herzlich eingeladen, ihren Unterricht zu reflektieren und zu überlegen, wo sie – über ein Semester, ein Jahr oder gar über die ganze Unter- oder Oberstufe gesehen - folgenden Schieberegler ansetzen würden.



Kein anderes schulinformatisches Thema fordert so zum Widerspruch heraus, wie das Thema „Programmieren". In der noch jungen Geschichte der Schulinformatik seit ca. 1980 hat der Autor fast keine schulinformatisch relevante Programmiersprache ausgelassen. In historischer Reihenfolge waren dies GW-Basic[6], Turbo Pascal, Dbase, Clipper, Prolog, Delphi, Visual Basic, VBA, Javascript, PHP und schließlich Java. Man muss sich vorstellen, dass sich meine erste Interaktion mit dem Computer darauf beschränkte, nach einer angezeigten Zeilennummer 10 einen Befehl einzugeben. GW-Basic ließ grüßen!

Und mit ein bisschen Glück brachte man ein einfaches Programm zum Laufen. Keine Notwendigkeit, sich über ein Betriebssystem Gedanken zu machen, keine Graphik und keine Standardsoftware trübten den Blick auf Programmierung und Algorithmen. Und damit habe ich 3 Jahre Informatik-, pardon: EDV-Unterricht, gemacht und im Jahr 1984 die erste Matura aus EDV abgewikkelt! Das Erstaunliche dabei ist, dass sich ein Teil der Problemstellungen nicht geändert hat, was das Programmieren im Kleinen betrifft. Soviel ich mich noch erinnern kann, war eine Aufgabe das Sortieren durch Einfügen, verpackt in einen simulierten Schilauf, bei dem den Namen der Maturakommissionsmitglieder zufällige Zeiten zugeordnet wurden und auf einer „Anzeigetafel" die richtige Reihung angezeigt wurde. Ein absolut maturables Beispiel auch noch im Jahr 2003! Es muss/sollte aber nicht mehr GW-Basic sein – und ist es auch nicht! Heute wird der alte Wein in neuen Schläuchen transportiert. Zum Beispiel könnte ein relationales Datenmodell zugrundegelegt werden. Auch der Entwicklung einer benutzerfreundlichen Oberfläche in einer ereignisgesteuerten Umgebung steht nichts im Wege. Wer es auf die Spitze treiben will, kann dieses Problem auch noch in einem Client-Server System implementieren. Das wäre vor 20 Jahren definitiv nicht möglich gewesen.

Wenn ich meinen Unterricht reflektiere, bestimmten in der Anfangszeit Algorithmen und Programmierung fast den gesamten EDV-Unterricht, abgesehen von ein wenig Theorie über Zahlensysteme und die Behandlung logischer Grundlagen. Meine erste Textverarbeitung fand mit dem Zeileneditor von GW-Basic statt, ein Ausdruck erfolgte mit dem Befehl PRINT (oder so ähnlich...) in jeder Zeile.

Mit dem Auftauchen der ersten Tabellenkalkulationen und etwas später von Textverarbeitungen und Datenbanken kam der Siegeszug der Anwendersoftware, die den Schieberegler (siehe Grafik oben) weit nach links ausschlagen ließ. In vielen Fällen steht der Zeiger auch heute noch weit links, weil vielfach die nachhaltige Anwenderschulung im Bereich der Standardsoftware fast keine Zeit für Programmentwicklung lässt.

Derzeit herrscht aber der Eindruck vor – verlässliche Studien dafür gibt es allerdings noch nicht – dass im Informatikunterricht wieder mehr entworfen, entwickelt und programmiert wird. Drei Gründe könnten dafür verantwortlich sein:

- Die Omnipräsenz des Internet und die damit einhergehende Gestaltung interaktiver Webseiten, die nur mit „Programmierung" möglich ist.
- Die Möglichkeit der Automatisierung von Abläufen in der Standardsoftware (z.B. VBA in Excel und Access), die nur durch Programmierkenntnisse wahrgenommen werden kann.
- Die Attraktivität der Entwicklungsumgebungen mit schnell sichtbaren Erfolgen.

Objektorientierte Sicht versus objektorientierte Softwareentwicklung

Es ist eine unbestreitbare Tatsache, dass die objektorientierte Softwareentwicklung in der österreichischen Schulinformatik weitgehend ein Stiefkind ist, das (noch) nicht gut behandelt wird. Dafür lassen sich einige Gründe anführen:

[6] *GW-BASIC wurde nach Greg Whitten, einem Microsoft Angestellten benannt*

- Unter vielen InformatiklehrerInnen bestehen über dieses Thema noch viele Unsicherheiten.
- Die Notwendigkeit von objektorientierten Entwürfen beim Programmieren im Kleinen (Algorithmieren) ist nicht erkennbar. Viele schuladäquate Probleme lassen sich auch ohne OOP realisieren.
- Der Paradigmenwechsel von der rein prozeduralen, imperativen Programmierung hin zur objektorientierten Programmierung ist schwierig (der Mensch ist ein „Gewohnheitstier").
- Objektorientierung ist abstrakt und erfordert sehr viel Vorwissen.
- Ein so markanter Paradigmenwechsel, wie ihn nun einmal die objektorientierte Softwareentwicklung darstellt, findet auch im Informatikunterricht nicht von heute auf morgen statt.[7]

Wenn unter Objektorientierung nur der konstruktivistische Ansatz - d.h. Entwurf und Programmierung von wiederverwertbaren Klassenhierarchien - verstanden wird, dann ist zu überlegen, ob das für den durchschnittlichen Schüler – ich spreche nicht von Ausnahmekönnern und Talenten – zu abstrakt und zu schwierig ist[8]. Der Anspruch der Schulinformatik kann nicht sein, alle Schüler zu professionellen Softwareentwicklern auszubilden.

Wohl aber – und damit wäre ein akzeptabler Kompromiss gefunden - kann von der Schulinformatik eingefordert werden, dass von allem Anfang an auf eine saubere Begriffsbildung geachtet wird. Mit den Begriffen Klasse, Objekt, Attribut und Methode können Schüler bereits sehr früh vertraut gemacht werden[9], indem die Benutzerschnittstellen und Standardsoftware bewusst unter dem objektorientierten Aspekt betrachtet werden. Grafische Benutzeroberflächen sind heutzutage extrem objektbasiert.

- Textverarbeitung: Zeichen und Absätze sind Klassen mit den Eigenschaften „Schriftart", „Schriftgröße" bzw. „Ausrichtung", „Einrückung", etc.
- Grafik-Software: Zeichnungswerkzeuge sind Klassen mit den Methoden „Löschen", „Verschieben"

Diese objektorientierte Sicht von Software kann als eine Voraussetzung dafür angesehen werden, dass dem Schüler übertragbare Kenntnisse (Konzeptwissen) vermittelt werden.

Ein sanfter Einstieg in die Objektorientierung (nicht OOA, OOD, OOP[10]) – Puristen werden dem sicher nicht zustimmen - ist aus meiner Sicht das Programmieren mit vorgegebenen Klassen (Objektprototypen) in ereignisgesteuerten Entwicklungsumgebungen. Ich würde der obigen objektorientierten „Dreiheiligkeit" noch das Akronym OOS (Objektorientierte Sicht) hinzufügen.

Mittlerweile hat – in Ansätzen - auch die Objektorientierte Programmiersprache Java in die Schulinformatik (Österreichs) Einzug gehalten. Ich befürchte, dass in vielen Fällen das Konzept der Objektorientierung im „Kampf" mit der Syntax und der Qual der Wahl einer geeigneten Entwicklungsumgebung beim Schüler eher auf Unverständnis stößt.

Sanfte und didaktisch sehr interessante Einstiegsmöglichkeiten, objektorientierte Konzepte zu vermitteln bieten auf (fast) allen Alterstufen z.B. die komplexen Werkzeuge Excel-VBA und Flash.

Ein Aphorismus des bekannten Komponisten A. Bruckner, der im übertragenen Sinn auch Software (Noten für ein Orchester) entwickelt hat, möge im Hinblick auf Softwareentwicklung zum Nachdenken anregen: „Willst du hohe Türme bauen, mußt du lange beim Fundament verweilen". Womit eine weitere Aporie angedeutet ist. Wir reden heute von Breiten- und Spitzensport. Was bedeutet in Analogie dazu die Aporie „Breiteninformatik und Spitzeninformatik" und welche Konsequenzen lassen sich daraus ableiten?

Die geneigte Leserin/der geneigte Leser ist eingeladen, darüber nachzudenken.

Literatur

[1] Schubert Sigrid, 2002, Schulinformatik in der Wissensgesellschaft, S 15-23, Tagungsband ME-2002

[2] Kalfa Winfried, 1996, Dialektik und Curricula

[3] Timmermann Bettina, 2002, Konzepte zur informatischen Bildung

Autor

Prof. Mag. Peter Micheuz, Gymnasium Völkermarkt und Universität Klagenfurt
E-Mail: peter.micheuz@aon.at
Homepage:
http://www.schulinformatik.at

Unterrichtet Informatik am Gymnasium Völkermarkt und ist Bundeslehrer im Hochschuldienst an der Universität Klagenfurt und seit zwei Jahren im Bereich Fachdidaktik Informatik für das Lehramtsstudium Informatik zuständig.
Leitung der ARGE Informatik an AHS in Kärnten, Projekt IKT-Unterstufe AHS
Betreiber des Schulinformatik-Portals
http://www.schulinformatik.at

[7] *Es ist zu bedenken, wie lange es gebraucht hat, bis das Thema Stochastik im Mathematikunterricht Einzug gehalten hat.*
[8] *In diesem Zusammenhang wären auch andere Themen wie z.B. die Schwierigkeit von Rekursion in der Schulinformatik zu beleuchten.*
[9] *Vgl. Hubwieser in dieser Sonderausgabe*
[10] *OOA: Objektorientierte Analyse, OOD: Objektorientiertes Design, OOP: Objektorientierte Programmierung*

Allgemein bildender Informatikunterricht – wie geht das?

Ein in der Informatikdidaktik breit akzeptierter Allgemeinbildungsbegriff, der auch den hier vorgestellten Gedanken zugrunde liegt, stammt von Bussmann und Heymann [1].
Allgemeinbildung soll

- auf zukünftige Lebenssituationen vorbereiten,
- kulturelle Kohärenz stiften (nicht nur rezeptive, sondern erneuernde Aneignung von tradierten Kulturgütern),
- ein zeitgemäßes Weltbild vermitteln,
- zum kritischen Vernunftgebrauch anleiten,
- zur Entfaltung eines verantwortungsvollen Umgangs mit den zu erwerbenden Kompetenzen und
- zur Stärkung des Schüler-Ichs beitragen.

Evelyn Stepancik
Universität Wien

Ein Unterricht, der sich an diesen Zielen orientiert, vermag mehr als nur die Schulung von Bedienerfertigkeit zu leisten. Doch möchte ich die folgenden Ausführungen nicht als Rezept zur Unterrichtsgestaltung anbieten, sondern lediglich Fragestellungen und Anregungen zur individuellen Realisierung eines allgemein bildenden Informatikunterrichts geben.

Eine erste Orientierung bietet der von Hubwieser [2] geprägte Begriff der „informatischen Bildung", der folgende drei Bereiche der schulischen Auseinandersetzung mit Informationssystemen zusammenfasst:

1. Einsatz als Medium oder als Lernhilfe
2. Schulung von Bedienerfertigkeit
3. Beherrschung grundlegender Konzepte

Die Gewichtung dieser drei Bereiche soll zum einen an den Kenntnissen der Schüler und Schülerinnen erfolgen, zum anderen sind jedoch auch die Ziele des gültigen Lehrplans zu beachten. Zu guter Letzt kann und soll der / die Lehrende seine / ihre ganz persönliche Gewichtung basierend auf den vorhergehenden Überlegungen vornehmen. Dabei ist auf die Vermittlung grundlegender Konzepte bereits ab dem ersten Lernjahr zu achten. So können beispielsweise Prinzipien von Rechnernetzen anhand des Versendens elektronischer Nachrichten schon im ersten Lernjahr in reduzierter Form vermittelt werden. Auch die Funktionsweise des World Wide Web kann auf einfachste Art und Weise unterrichtet und von den Schülerinnen und Schülern entdeckt werden. Schon 11-jährige sind für die Frage: „Wie kommt diese Webseite auf meinen Computer?", zu begeistern. Wird zudem noch der Datenschutz thematisiert, dann kann die Unterrichtssequenz wesentliche Beiträge zur Allgemeinbildung leisten.

Die enorme Bedeutung von Informatiksystemen für das tägliche Leben ist hinlänglich erwiesen. Am Arbeitsmarkt sind in vielen Bereichen grundlegende Kenntnisse und Fertigkeiten im Umgang mit Informations- und Kommunikationssystemen nötig. Wie aber können wir Schülerinnen und Schüler darauf vorbereiten? Es ist nicht die Aufgabe eines allgemein bildenden Informatikunterrichts, auf die Ausübung eines bestimmten Berufes vorzubereiten, vielmehr müssen Qualifikationen vermittelt werden, die zur Bewältigung realer Lebenssituationen beitragen, die aber nicht automatisch und nebenher erworben werden können und die durch eine gewisse Universalität, also Anwendbarkeit in sehr verschiedenen Situationen, gekennzeichnet sind.

Die Problemstellungen des Unterrichts haben sich also an realen Lebenssituationen zu orientieren. Dabei sollte nicht die exakte Bedienung einer Software oder die perfekte Beherrschung einer Programmiersprache im Vordergrund stehen. Das Erlernen einer Programmiersprache oder Software kann nur an dafür typischen und geeigneten Beispielen erfolgen, jedoch sind auch die Grenzen der Anwendbarkeit aufzuzeigen. Das Ziel eines derartigen Unterrichts ist es, dass Schüler und Schülerinnen ein der Problemstellung adäquates Werkzeug wählen und ihre Wahl begründen.

Für den Lehrenden / die Lehrende bedeutet dies, Aufgaben zu stellen, Probleme zu finden, die paradigmatisch für den zu erlernenden Inhalt sind. Dabei werden möglicherweise unterschiedliche Lösungswege auftreten, die in einer anschließenden Reflexionsphase zu diskutieren sind. Neben der Vorbereitung auf zukünftige Lebenssituationen wird zum kritischen Vernunftgebrauch angeleitet und zur Entfaltung eines verantwortungsvollen Umgangs mit den zu erwerbenden Kompetenzen beigetragen.

Wesentliche Hinweise zur Konzeption eines allgemein bildenden Informatikunterrichts und zur Bestimmung von Lernzielen formuliert Baumann [3] anhand von drei didaktischen Leitlinien bzw. Leitfragen.

Die erste Leitfrage „Wie können durch Entwicklung, Gestaltung und Anwendung von Informatiksystemen Probleme der Lebenswelt gelöst werden?" inkludiert obige Ausführungen, der praktisch, anwendungsbezogene Aspekt tritt hierbei in den Vordergrund.

Die zweite Leitfrage „Wie sind Informatiksysteme aufgebaut, welches sind die Prinzipien des Zusammenwirkens ihrer Komponenten und wie ordnen sie sich in größere Systemzusammenhänge ein?" zielt auf die Beschäftigung mit Informatiksystemen ab, die aus Teilsystemen aufgebaut sind und sich zu größeren Netzen zusammenschließen können.

Die dritte Leitfrage „Welches sind die Grundlagen und wo liegen die Grenzen formaler bzw. technischer Wissensverarbeitung, und wie kann die kognitive Autonomie menschlicher Subjekte gewahrt werden?" thematisiert einerseits die Grenzen eines verantwortungsvollen Computereinsatzes und andererseits die prinzipiellen Grenzen der Idee des Wissens und seiner technischen Verarbeitung.

Das folgende Beispiel vermag diese Leitlinien zu verdeutlichen. Die Implementierung und Realisierung einer eigenen Webseite ist für Schülerinnen und Schüler zumeist eine interessante Aufgabenstellung. Zudem stellt diese Aufgabe einen typischen Einsatzbereich und eine exemplarische Anwendung der Informatik dar. Wird im Unterricht die Auswahl der Werkzeuge zur Erstellung der Webseite thematisiert und können Schüler und Schülerinnen die getroffene Auswahl begründen, dann trägt auch dies zur ersten Leitfrage bei.

Die Wirkprinzipien von Informatiksystemen (zweite Leitfrage) werden den Schülern und Schülerinnen nahe gebracht, wenn in der Unterrichtssequenz auch die Struktur und Funktion von Netzen – insbesondere des Internets – zur Informationsübertragung analysiert werden.

Die ethischen Grenzen eines verantwortungsvollen Computereinsatzes sollten unbedingt vor der Veröffentlichung der Webseite erörtert werden, womit auch die dritte Leitlinie zumindest in Teilbereichen erfüllt ist.

Zum Abschluss kann ich nur empfehlen, Ansätze eines allgemein bildenden Informatikunterrichts zu erproben. Nicht in allen Klassen, nicht bei allen Themen, sondern vorerst nur bei einer Klasse, bei einem Thema, das den Lehrenden / die Lehrenden immer wieder besonders anzieht.

Literatur

[1] Bussmann H., Heymann H.W.: Computer und Allgemeinbildung, 1987

[2] Hubwieser P.: Didaktik der Informatik, Springer, 2001

[3] Baumann R.: Didaktik der Informatik, Klett, 1996

Autorin

Mag. Evelyn Stepancik, BG/BRG Purkersdorf
E-Mail: estepancik@informatix.at
Homepage: http://www.informatix.at

Unterrichtet am Gymnasium Purkersdorf die Fächer Deutsch, Mathematik und Informatik sowie Informatik-Fachdidaktik an der Universität Wien im Bereich der Lehramtsausbildung Informatik. Engagement auch im Bereich des E-Learning.

Interessante Webseiten

http://www.educeth.ch/informatik

Dieser von Dr. Hartmann und Dr. Reichert betriebene Bildungsserver in der Schweiz ist eine wahre Fundgrube für den Informatikunterricht auf vielen Altersstufen.

http://www.lugsp.at/~alfred

Diese private Seite von Prof. Mag. Alfred Nussbaumer, der im Stiftsgymnasium Melk, Niederösterreich, unterrichtet, bietet sehr übersichtlich gestaltete und selbst erstellte Unterrichtshilfen für den Informatik-Unterricht.

http://www.schulinformatik.at
Neben einer umfangreichen Linksammlung bietet dieses Portal viele Anregungen für den Informatikunterricht,
Aufgabensammlungen (z.B. Maturaaufgaben), Quizzes u.a..
Der verstärkte Ausbau zu einem anspruchsvollen Bildungsserver mit wertvollen Artikeln und Unterrichtsunterlagen ist geplant und im Gange.

http://www.eduhi.at/gegenstand/informatik
Dieser Teil des oberösterreichschen Eduhi-Bildungservers wird vom Fachinspektor für Informatik, Prof. Mag. Günther Schwarz, betrieben und stellt eine sehr gut strukturierte und dokumentierte Linksammlung für fast alle Belange der Schulinformatik dar.

http://www.informatikserver.at
Das steirische Portal http://www.informatikserver.at, die von den Kollegen Dir. Mag. Hans Adam und Realschullehrer Rainer Blaschke stellt aktuelle Hintergrundinformationen aus der IT-Welt für interessierte Informatik-Lehrkräfte zur Verfügung. Interessante Links und Online-Unterlagen bereichern die österreichische Bildungsserverlandschaft.

bm:bwk

CD Austria

Literaturempfehlungen

Ankündigung der Buchneuerscheinung

„Schulinformatik in Österreich – Erfahrungen und Beispiele aus dem Unterricht“ für November 2003 (erscheint bei Ueberreuter und wird zu Sonderkonditionen von Lehrer/innen und Studierenden bei der Amedia bestellt werden können)

Diese aktuelle Bestandsaufnahme der Informatik in Österreich wendet sich an Informatiklehrer/innen, aber auch Studierende an den Pädagogischen und Berufspädagogischen Akademien und den (Fach-)Hochschulen und Universitäten.
Nach einer umfassenden Einführung in den Gegenstandsbereich und einer Standortbestimmung der Informatik folgen 23 praxisbezogene Autorenbeiträge aus der Allgemeinen Pflichtschule (APS), der Allgemeinbildenden Höheren Schule (AHS), der Pädagogischen Akademie (PA) sowie der Berufsbildenden mittleren und höheren Schulen (BMHS) und ein Ausblick auf das Lehramtsstudium „Informatik und Informatikmanagement“.
Ein ausführliches Literaturverzeichnis, eine CD-ROM (mit Daten zu den Beiträgen und dem elektronischen Lexikon „Basiswissen Angewandte Informatik“) und ein DVD-Video mit Statements und Kommentaren der Autoren runden diesen „Medienverbund“ ab.

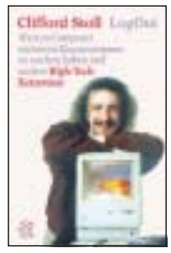
Bücher über Schulinformatik, die wegen ihres invarianten Charakters in keiner Bibliothek von Informatiklehrkräften fehlen sollen.

Einführung in die Informatik
Heiz-Peter Gumm, Manfred Sommer
Oldenbourg

Informatische Grundbildung – Anfangsunterricht
Koerber-Peters
Paetec-Verlag,

Schülerduden – Informatik
Volker Claus, Andreas Schwill
Bibliographisches Institut, Mannheim
ISBN: 3411044845

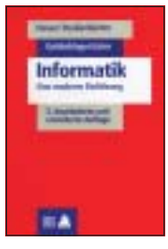
PC-Grundlagen – echt einfach
Martin Schultheiß
Franzis Verlag
ISBN 3-7723-6536-1



Logout
Clifford Stoll
Fischer
ISBN: 3596155126

Was ist Informatik
Peter Rechenberg
Hanser Verlag,
ISBN: 3446213198

Didaktik der Informatik
Peter Hubwieser
Springer-Verlag Berlin Heidelberg
ISBN: 3540655646



Informatik - Eine moderne Einführung
Goldschlager, Lister
Hanser Fachbuch,
ISBN: 3446157662

Didaktik der Informatik
Rüdeger Baumann
Klett Schulbuch
ISBN: 3129850104