# Studie zum pädagogischen Mehrwert eines kontextorientierten Unterrichtskonzepts zur Mensch-Maschine-Kommunikation mit gesprochener Sprache

Dissertation zur Erlangung des Doktorgrades
an der Fakultät für Mathematik, Informatik und Naturwissenschaften
Fachbereich Informatik
der Universität Hamburg
vorgelegt von Sven Alisch

Hamburg, 2017

Datum der Einreichung: 28.03.2017

Referent: Prof. Dr. Norbert Breier

Koreferent: Prof. Dr. Wolfgang Menzel

Vorsitz: Prof. Dr. Leonie Dreschler-Fischer

Tag der mündlichen Prüfung: 07.11.2018

# Abstract

This study is based on experience gained while teaching informatics at a state grammar school of the Free and Hanseatic City of Hamburg. In the past, teaching styles did not meet the needs of students, leading many to lose motivation and give up. Examinations and other performance measures show that teaching styles can reinforce a predominantly negative image of informatics. The starting point of this study is the personal knowledge gained from personal informatics lessons at a state grammar school (german Gymnasium) of the Free and Hanseatic City of Hamburg. The teaching that has been practiced so far on the subject of language processing does not meet the requirements and leads to the fact that the students partly give up demotively. This study uses qualitative interviews to analyse the Hamburg Informatics Curriculum and shows, that these failures are caused by application-oriented informatics didactics, particulary emphasizing algorithms and functional programming techniques in connection with the subject language processing and the mandatory use of a functional programming concept. A revised teaching unit and newly developed digital learning environments are presented, reflecting modern computer science teaching methods as described by Alisch and Breier (cmp. [AB14]). The pedagogical value of the teaching unit is demonstrated by a quantitative study using quasi-experimental methods. The new teaching concept and learning environments, inES and inGE, make it possible to achieve the learning objectives of earlier framework plans in a more efficient and sustainable way. With the new teaching unit, all A-Level requirements, as defined in the so called „Abitur Anforderungsheft“, are fully achieved. Empirical evidence is provided for the first time concerning how students can acquire skills from informatics lessons. It is shown that the concept of teaching informatics in context (IniK) is also suitable for the upper secondary school level, and for teaching other informatics topics at the grammar school level.

# Kurzfassung

Ausgangspunkt dieser Studie ist die persönliche Erkenntnis aus dem eigenen Informatikunterricht an einem staatlichen Gymnasium der Freien und Hansestadt Hamburg, dass der bisherig praktizierte Unterricht zum Themenbereich Sprachverarbeitung nicht den Anforderungen genügt und dazu führt, dass die Schüler teilweise demotiviert aufgeben, in Klausuren und anderen Leistungsnachweisen nicht überzeugen, und der Unterricht im schlimmsten Fall dazu beiträgt, das teilweise noch vorherrschende Negativimage der Informatik zu verstärken. In einem ersten Schritt arbeitet die Studie mithilfe von Experteninterviews heraus, dass die konkreten Ursachen für das Scheitern der Ziele des bisherigen Hamburger Informatik Curriculums über die Sprachverarbeitung vor allem im anwendungsorientierten Informatikunterricht mit starker Betonung der Algorithmik und der Besonderheit zwingend funktional programmieren zu müssen, liegt.

Auf der Grundlage dieser Erkenntnisse wird unter Einbeziehung der von Alisch und Breier postulierten Merkmale eines zeitgemäßen Informatikunterrichts (vgl. [AB14]) und der Analyse aktueller fachdidaktischer Literatur in einem zweiten Schritt eine neue überarbeitete Unterrichtseinheit sowie eine eigens dafür entwickelte Lernumgebung vorgestellt. In einer sich anschließenden quantitativen Untersuchung mit quasi-experimentellem Charakter wird der pädagogische Mehrwert dieser neuen Unterrichtseinheit nachgewiesen. Dieser besteht darin, dass mit dem neuen Unterrichtskonzept in Kombination mit den Lernumgebungen inES und inGE die Lernziele früherer Rahmenpläne besser im Sinne von effizienter und nachhaltiger erreicht werden und im Gegensatz zum alten Ansatz mit der neuen Unterrichtseinheit alle im aktuellen Hamburger Abitur Anforderungsheft angeführten Zielsetzungen nun wirklich erreicht werden.

Des Weiteren wird in dieser Arbeit erstmals der empirisch gesicherte Nachweis erbracht, dass und wie sich Schüler die in den Bildungsstandards für die Sekundarstufe II im Inhaltsbereich „Automaten und Sprache“ formulierten Kompetenzen im Informatikunterricht aneignen können, die deutlich über die im Hamburger Abitur Anforderungsheft für Informatik formulierten Anforderungen hinausgehen. Diese Studie zeigt außerdem, dass das und wie sich das Unterrichtskonzept Informatik im Kontext (IniK) auch in der gymnasialen Oberstufe erfolgreich einsetzen lässt und das bei geeigneter Wahl des Kontextes dieses Vorgehen auch für andere Themenbereiche des Informatikunterrichts der gymnasialen Oberstufe eignet.

# Danksagung

Diese Arbeit wäre ohne die tatkräftige Unterstützung vieler Menschen nicht entstanden.
Ganz besonders danken möchte ich meinem Doktorvater, Herrn Prof. Dr. Norbert Breier, der mich über die vielen Jahre immer wieder motiviert und an mich geglaubt hat. Seine herzlichen Ratschläge, sein großes Vertrauen in meine Arbeit und die vielen Anstrengungen seinerseits haben mir immens geholfen.
Danken möchte ich auch meinem Zweitbetreuer, Herrn Prof. Dr. Wolfgang Menzel, für die vielen fachwissenschaftlichen Ratschläge, die mir ebenfalls sehr geholfen haben.
Des Weiteren gilt mein Dank den an dieser Untersuchung beteiligten Schülerinnen und Schülern, die engagiert und interessiert an der Untersuchung der beiden Unterrichtseinheiten einschließlich der Befragungen und Tests teilgenommen haben. In diesem Zusammenhang bedanke ich mich auch bei allen Lehrkräften und Schulleitungen, die mich bei den Befragungen und Tests unterstützt haben.
Meinem lieben Kollegen Dr. Silvio Jacobs danke ich für seine hilfreichen Anregungen.
Meiner Frau Stefanie und meinen beiden Töchtern Helene und Pauline danke ich für ihr großes Verständnis, ihre große Geduld und Hilfe, die sie mir während der vergangenen Jahre an meiner Dissertation und der sehr anstrengenden Arbeit bei der Entwicklung der Unterrichtssoftware entgegengebracht haben. Mir ist bewußt, dass ihr in dieser Zeit viele Stimmungsschwankungen meinerseits ertragen musstet. Danke dafür. Zuletzt möchte ich noch meinem Vater und meiner Mutter für ihre liebevolle und tatkräftige Unterstützung in allen Jahren danken.

# Inhaltsverzeichnis

# Kapitel 1

# Einleitung

Das Schulfach Informatik befindet sich wie kein anderes Fach im stetigen Wandel. Es gibt heute unter den Fachdidaktikern[1] und Lehrkräften einen breiten Konsens darüber, dass der Informatikunterricht bereits heute Teil der Allgemeinbildung sein muss, damit Schüler am sozialen, gesellschaftlichen, wirtschaftlichen, beruflichen und politischen Leben teilhaben können. Heymann schrieb 1997, dass Lebensvorbereitung, die Stiftung kultureller Kohärenz, die Weltorientierung, eine Anleitung zum kritischen Vernunftgebrauch, die Entfaltung von Verantwortungsbereitschaft, die Einübung in Verständigung und Kooperation sowie die Stärkung des Schüler-Ichs die grundlegenden Aufgaben von Schule sind (vgl. [Hey97], S. 42-45). Jedwedes Informatikcurriculum muss diesen Anforderungen Rechnung tragen. Damit zukünftige Rahmenpläne diesem Anspruch gerecht werden können, hat die GI im Jahre 2008 Grundsätze und Standards für den Informatikunterricht in der Sekundarstufe I empfohlen, „[d]ass [das] übergeordnete Ziel informatischer Bildung in Schulen ist [...], Schülerinnen und Schüler bestmöglich auf ein Leben in einer Informationsgesellschaft vorzubereiten [...]“ (vgl. [Bil08]). Im Kern der Standards und Grundsätze steht der produktunabhängige Zugang zu Denk- und Arbeitsweisen von Informatiksystemen und mit ihm die Darstellung und Repräsentation von Informationen. Dafür wurden aus der Fülle von möglichen Informatikinhalten die wesentlichen Inhaltsbereiche abgeleitet (vgl. Abbildung 1.1) und mit allgemeingültigen an Schulunterricht orientierten Prozessbereichen verknüpft. Mit der Erarbeitung dieser Grundsätze und Standards geht der Wunsch einher, dass sich in jeder Unterrichtseinheit möglichst

[1] Zur besseren Lesbarkeit der Arbeit schließt die darin durchgängig verwendete männliche Form immer auch die weibliche ein.

viele dieser dargestellten Inhaltsbereiche und Prozessbereiche abdecken lassen. Die Bildungsstandards für die Sekundarstufe II wurden 2016 veröffentlicht und formulieren die für die Oberstufe markanten Anforderungsbereiche, das sind Reproduktion (I), Anwendung (II) und Reflexion (III), konkreter.


Abbildung 1.1: Die Prozess- und Inhaltsbereiche der Bildungsstandards

Die Grundlage eines jeden Informatikunterrichts ist der Rahmenplan für das Fach des jeweiligen Bundeslandes. Auch wenn die Ausarbeitung von Rahmenplänen wegen der föderativen Struktur der Bundesrepublik Deutschland anspuchsvoll und zugleich unterschiedlich erfolgt, so sind die Aspekte dieser bisherigen Grundsätze und Standards mittlerweile in vielen Rahmenplänen erfreulicherweise wiederzufinden. Bislang fehlt es jedoch an konkreten Handreichungen und Schulbüchern für Lehrkräfte, die das Fach Informatik unterrichten. Hinzu kommt das Problem, dass Informatik bisher nur in den Ländern Bayern, Mecklenburg-Vorpommern und Sachsen ein Pflichtfach ist. In den anderen Bundesländern, so auch in Hamburg, ist Informatik ein Wahlpflichtfach. Die Folge ist, dass Schüler teilweise das Fach für nur ein Jahr wählen. In Hamburg wird Informatik i. d. R. ab dem siebten (Stadtteilschule) bzw. ab dem achten Schuljahr (Gymnasium) im Rahmen eines Wahlpflichtfaches angeboten. In der Oberstufe wird Informatik seit der letzten Oberstufenreform im Jahre 2011 in Form von Profilen unterrichtet, d. h. Informatik kann Bestandteil eines Fächerbouquets sein. Das Ziel der Profiloberstufe besteht in der Verbesserung des fächerübergreifenden und fächerverbindenden Unterrichtens. Beispielsweise werden neue Freiräume für übergreifende Projekte eröffnet und die Verzahnung von Wirtschaft und Schule gefördert. Diese Tatsachen stellen die Rahmenplanentwickler und besonders die unterrichtenden Lehrkräfte vor enorme Herausforderungen. Seiffert schreibt deshalb

nicht ohne Grund, dass die Curricula der einzelnen Jahrgangsstufen modular aufgebaut sein müssen, d. h. „solange das Fach Informatik ein Wahlfach ist, das Schülerinnen und Schülern auch nur für ein einziges Jahr wählen können, muss zu jedem Schuljahr ein Neueinstieg möglich sein“ (vgl. [Sei03]).

In Hamburg, wo Informatik mit unterschiedlichen Fächern kombiniert werden kann, musste der Rahmenplan für die Oberstufe eben genau auf diese Bedürfnisse ausgerichtet sein, was zur Folge hat, dass keine inhaltlichen Vorgaben gemacht werden, sondern Kompetenzen vorgegeben, die es in der Oberstufeninformatik insgesamt zu erreichen gilt. Mit der Einführung des Zentralabiturs 2014 und der damit einhergehenden zentralen Aufgabenentwicklung wurden spezielle Anforderungshefte entworfen. Darin werden für jedes Hamburger Semesterthema, das sind Objektorientierte Modellierung von Grafiksystemen, Datensicherheit in verteilten Systemen, Simulation dynamischer Systeme und Sprachverarbeitung, die Inhalte und zu erreichenden Kompetenzen definiert. Diese formulierten Kompetenzen sind durchaus zeitgemäß, aber für den Bereich der Sprachverarbeitung zeigen die eigenen Unterrichtserfahrungen eine große Lücke zwischen Wunsch und Wirklichkeit.

Der Wunsch nach Modularität, nach Unabhängigkeit von Vorwissen und der Möglichkeit auch in der Oberstufe Informatik anwählen zu können ohne als Schüler jemals vorher Informatikunterricht gehabt zu haben sowie einen problem- und systemorientierten Informatikunterricht durchzuführen, der den Schwerpunkt auf die Modellierung legt und sich am Ende ein fertiges lauffähiges System erhofft steht gerade bei dem Thema Sprachverarbeitung im eklatanten Widerspruch zur Wirklichkeit. Dieser Informatikunterricht endet häufig in einem Programmiersprachenunterricht, der weder schülerorientiert, noch modular ist und den didaktischen Grundsätzen des Rahmenplanes widerspricht. Er wird auch nicht den Grundsätzen und Standards gerecht, da dieser Unterricht nur den Inhaltsbereich *Algorithmen* und den Prozessbereich *Modellieren und Implementieren* anspricht. Die Ergebnisse zeigen, dass es ohne Vorwissen für die Schüler fast unmöglich ist, sich Konzeptwissen über die Sprachverarbeitung anzueignen. Die Folge ist, dass die Schüler teilweise demotiviert aufgeben und auch die eigenen Ergebnisse am Ende des Semesters in Form von Klausuren und anderer Leistungsnachweise nicht überzeugen und im schlimmsten Fall dazu beitragen, dass das teilweise noch vorherrschende Negativimage der Informatik verstärkt wird.

Diese Ausführungen beruhen zunächst nur auf eigenen Unterrichtserfahrungen,

so dass eine Untersuchung notwendig ist, die mithilfe von Experteninterviews versucht herauszufinden, inwiefern sich diese eigenen Erfahrungen mit denen anderer Lehrkräfte decken. Insbesondere soll diese Arbeit herausfinden, worin die konkreten Ursachen für das Scheitern der Ziele des bisherigen Curriculums über die Sprachverarbeitung liegen. Auf der Grundlage dieser Ergebnisse sowie der Analyse der aktuellen fachdidaktischen Literatur wird diese Arbeit eine neue überarbeitete Unterrichtseinheit entwerfen und diese auf ihre Tauglichkeit im Anschluss daran untersuchen.

Der Titel dieser Arbeit enthält zwei Leitbegriffe, die im Rahmen dieser Studie untersucht werden. Das ist zum einen der Begriff des Unterrichtskonzeptes und zum anderen der Begriff des pädagogischen Mehrwertes. Der Begriff des Unterrichtskonzeptes wird in Jank und Meyer [JM11] wie folgt definiert:

**Definition 1.1**: Unterrichtskonzepte sind Gesamtorientierungen didaktisch-methodischen Handelns, in denen ein begründeter Zusammenhang von Ziel-, Inhalts- und Methodenentscheidungen hergestellt wird. Sie definieren grundlegende Prinzipien der Unterrichtsarbeit, sie formulieren Leitbilder des Rollenverhaltens von Lehrern und Schülern und sie geben Empfehlungen für die organisatorisch-institutionelle Gestaltung des Unterrichts.

Der Begriff *Mehrwert* ist der Wirtschaft entlehnt und bezieht sich dort auf den Teil des Geldbetrages, der über die *Herstellungskosten hinaus erwirtschaftet* wird. Bei der Herstellung eines Produktes, ist es der Gewinn, der abzüglich der Herstellungskosten übrig bleibt. In diesem Zusammenhang wird auch häufig von der *Wertschöpfung* gesprochen. Stratmann hat diesen Begriff auf die Pädagogik übertragen und in seiner Dissertation über den Einsatz von Notebooks an einer Universität wie folgt konkretisiert:

> „[Der *pädagogische Mehrwert* lässt sich] grundsätzlich in zwei Kategorien einordnen [...]: Zum einen *die bessere Erreichbarkeit alter Ziele* durch *neue Methoden und Medien* und zum anderen *die Anstrebung neuer Ziele*, die *durch neue Methoden und Medien erst ermöglicht* werden“ (vgl. [Str07]).

Es „[...] ergibt sich ein Mehrwert, wenn durch [ein Unterrichtskonzept] eine Effizienz-, bzw. Effektivitätssteigerung oder neue Qualitäten durch ein anderes Lernen erreicht werden kann“ (vgl. [Str07]). Ziele werden durch Kriterien präzisiert (vgl. [His12]). Die Praxis wird mit Hilfe von Indikatoren untersucht. Im Kapitel 2 dieser Arbeit werden zunächst die allgemeinen Ziele des Informatikunterrichtes definiert und anschließend die Kriterien eines guten Informatikunterrichtes erarbeitet.
Die Arbeit benutzt ferner sehr häufig die Begriffe Modellierung und Implementierung. Sie werden sowohl in den Rahmenplänen als auch in dieser Arbeit im Sinne der GI Empfehlungen des Jahres 2000 verwendet, d. h. unter Modellierung wird hier „im wesentlichen die Abgrenzung eines für den jeweiligen Zweck relevanten Ausschnittes der Erfahrungswelt, die Herausarbeitung seiner wichtigen Merkmale unter Vernachlässigung der unwichtigen sowie seine Beschreibung und Strukturierung mit Hilfe spezieller Techniken [...]“ verstanden (vgl. [GI200], S. 3). Der Begriff Implementierung meint das konkrete Umsetzen eines Algorithmus bzw. die Erarbeitung eines direkt ausführbaren Modells in Form von Quellcode.

# Kapitel 2

# Aktuelle fachdidaktische Entwicklungen

Der Begriff des *pädagogischen Mehrwertes* für ein Unterrichtskonzept definiert sich, wie in Kapitel 1 beschrieben, über alte Ziele, die besser zu erreichen sein müssen und neue Ziele, deren Erreichung mit neuen Medien und Methoden erst möglich werden. Doch welche Ziele werden im Informatikunterricht insbesondere im Bereich der Sprachverarbeitung angestrebt und welche Voraussetzungen für das Erreichen dieser Zielsetzungen sind nötig? Für die Beantwortung dieser Fragen ist eine Auseinandersetzung mit den aktuellen fachdidaktischen Entwicklungen notwendig. Hierfür haben Alisch und Breier in 2014 einen Gastbeitrag in d64.org unter dem Titel „Zehn Thesen zu einem zeitgemäßen Informatikunterricht “ veröffentlicht (vgl. [AB14]). Verkürzt formuliert ist zeitgemäßer Informatikunterricht demnach

1. allgemeinbildend,
2. informationsorientiert,
3. kompetenzorientiert,
4. kontextorientiert,
5. medienaffin,
6. gendersensibel,
7. sprachfördernd,
8. fächerübergreifend und fachverbindend,
9. projektorientiert und
10. für alle verbindlich.

Während die zehnte These nur durch politische Veränderungen herbeigeführt werden

kann, stellt sich die Frage, wie ein Unterricht aussehen muss, der die Thesen eins bis neun berücksichtigt und die Erreichung der obigen Ziele ermöglicht.

## 2.1 Allgemeinbildender Informatikunterricht

„Bildungsziele und Inhalte sind immer ein Spiegelbild der politischen und ökonomischen Verhältnisse“ ([AB14]). „Die Naturwissenschaften haben ihre volle Bedeutung entfaltet, nachdem im 18. und 19. Jahrhundert die wesentlichen Naturgesetze über Materie und Energie entdeckt wurden“ (vgl. [AU01], S. 401). „Sie waren mit ihren Erfindungen und Erkenntnissen die Geburtshelfer der ersten und zweiten industriellen Revolution und haben die Grundlagen für unsere heutige Technik gelegt. Heute bildet die Informatik den Kern der dritten industriellen Revolution und ist die Grundlage heutiger und künftiger Technik, zukünftiger Berufsbilder und der Informationsflüsse in der Gesellschaft, denn mit der Entdeckung des informatischen Grundgesetzes hat die *Information* als dritte Grundgröße sowohl die Wissenschaft als auch unsere Technik verändert (vgl. [AU01], S. 402)“ ([AB14]). „Die Vernetzung der weltweit angelegten Informationsquellen durch das Internet führt dazu, dass die global verteilte Information prinzipiell für jeden Menschen, zu jeder Zeit und an jedem Ort verfügbar ist und dass jeder sein individuelles Wissen durch Aneignung und Verarbeitung der Information selbst erweitern kann. Dazu werden in zunehmendem Maße Werkzeuge in Form von Informatiksystemen benötigt, ohne die die Fülle an Information schon heute nicht mehr zu bewältigen ist. Der Informatik als Wissenschaft kommt dabei eine Schlüsselrolle zu, da sie systematisch Möglichkeiten der automatischen Informationsverarbeitung und Wissensrepräsentation untersucht und in Informatiksystemen nutzbar macht“ ([GI200], S. 1).

„Ein zeitgemäßer Informatikunterricht stellt sich dem Auftrag der allgemein bildenden Schule und damit der Forderung Heymanns (1997), dass der Informatikunterricht Lebensvorbereitung, die Stiftung kultureller Kohärenz, die Weltorientierung, eine Anleitung zum kritischen Vernunftgebrauch, die Entfaltung von Verantwortungsbereitschaft, die Einübung in Verständigung und Kooperation sowie die Stärkung des Schüler-Ichs umsetzt (vgl. [Hey97], S. 42)“ ([AB14]). „Er eröffnet ihnen unabhängig von ihrem Geschlecht, ihrer Herkunft und ihren sozialen Verhältnissen einen gleichberechtigten Zugang zu informatischen Denk- und Arbeitsweisen und modernen Informations- und Kommunikationstechniken (vgl. [GI200], S. 1)“ ([AB14]).

## 2.2 Informationsorientierter Informatikunterricht

Im Vergleich mit den tradierten Fächern Deutsch und Mathematik, aber auch gegenüber den naturwissenschaftlichen Fächern Physik, Chemie und Biologie ist die Informatik ein noch sehr junges Schulfach und trotzdem gab es in den wenigen Jahren schon sehr viele „Paradigmenwechsel“ bezüglich der fachdidaktischen Ansätze.
Das älteste Paradigma ist der *hardwareorientierte fachdidaktische Ansatz*. Dieser wurde bereits Mitte der 60er Jahre in ersten Unterrichts- und Schulversuchen zur „Datenverarbeitung“ überwiegend von Fachlehrern der Mathematik und Physik durchgeführt. Die Vermittlung von mathematisch-logischen und physikalisch-technischen Grundlagen der Datenverarbeitung standen im Mittelpunkt dieser „Computerkunde“. Hardware ist sehr kurzlebig und hardwareorientierte Inhalte sind schnell veraltet, sodass dieser didaktische Ansatz heute keine Anwendung mehr findet.
Abgelöst wurde dieses fachdidaktische Paradigma vom *algorithmenorientierten fachdidaktischen Ansatz* in den 70er Jahren. Die systematische Suche nach algorithmischen Lösungen von Problemen mit dem Ziel, diese in einem Programm umzusetzen verfolgt der *algorithmenorientierte fachdidaktische Ansatz* . Anschließend wird das „Gelernte durch Anwendung auf praxisorientierte Probleme“ ([Bre03], S. 3) vertieft und die „Auswirkungen der Datenverarbeitung auf die Gesellschaft“ ([Bre03], S. 3) herausgearbeitet. Der Computer bleibt bei diesem Ansatz jedoch auf seinen algorithmischen Aspekt beschränkt mit der Folge, dass der Unterricht häufig in einen Programmierkurs abgleitet. In der Vergangenheit wurden fast ausschließlich numerische Aufgaben gelöst (vgl. [Bre03], S. 3).
Der *anwendungsorientierte fachdidaktische Ansatz* betont den Vorgang der Modellbildung stärker und orientiert sich an den Methoden des professionellen Software-Entwurfs. Im Unterricht sollen deshalb nur solche Themen behandelt werden, die sich in einen Praxisbezug einbetten lassen. In der Schulrealität bestimmt aber weniger die Relevanz der Anwendung die Wahl der Probleme, als vielmehr die Tatsache, dass die Schüler eine algorithmische Problemlösung erarbeiten sollen (vgl. [For90], S. 34). Deshalb läuft der anwendungsorientierte fachdidaktische Ansatz Gefahr, wieder zu einem *algorithmenorientierten fachdidaktischen Ansatz* reduziert zu werden. Trotz dieser Kritik wird der *anwendungsorientierte fachdidaktische Ansatz* in Hamburg seit vielen Jahren praktiziert (vgl. [DKW11], S. 2). Im Hamburger Rahmenplan findet sich zum *anwendungsorientierten fachdidaktischen Ansatz* dieser Absatz: (vgl.

[AJSS09], S. 13):

> „Für jedes Halbjahr der Vorstufe wird jeweils ein Anwendungskontext gewählt. Fachliche Inhalte der Informatik sind auf diesen Anwendungskontext zu beziehen. Folgende Inhalte sind in jedem Halbjahr verbindlich:
>
> - Exploration des gewählten Anwendungskontextes,
> - Analyse von Einsatzmöglichkeiten eines Informatiksystems in dem gewählten Anwendungskontext,
> - Beschreibung von zu unterstützenden Anwendungsfällen im Hinblick auf den Entwurf eines Informatiksystems,
> - Anforderungsbeschreibung für einen eigenen Prototyp eines Informatiksystems aus dem gewählten Anwendungskontext,
> - Implementierung des eigenen Prototypen,
> - Diskussion der Auswirkungen des Einsatzes von Informatiksystemen in dem gewählten Anwendungskontext.“

Insbesondere bei dem vorletzten Punkt, der Implementierung des eigenen Prototypen, deckt sich Fornecks Kritik mit den eigenen Erfahrungen bezüglich dieses fachdidaktischen Ansatzes.
Bei dem *benutzerorientierten fachdidaktischen Ansatz* wird auf die Hervorhebung des Algorithmus und des Programmierens verzichtet und stattdessen auf die Benutzung von Anwendersystemen und die gesellschaftlichen Auswirkungen gesetzt. Programmiersprachen werden insbesondere durch Standardsoftware ersetzt, mit deren Hilfe die Schüler umfassendere und realitätsnähere Probleme im Unterricht bearbeiten können. Hierdurch ist ein engerer Bezug zu praxisrelevanten Anwendungen gegeben, und die sozialen, rechtlichen und wirtschaftlichen Auswirkungen des Computereinsatzes auf den Einzelnen und auf die Gesellschaft werden thematisiert und reflektiert. Die Algorithmik behält aber einen hohen Stellenwert, denn der Umgang mit und das Verstehen von Standardsoftware schließen algorithmische Denk- und Arbeitsweisen ein.
Diese bisherigen Ansätze werden einem allgemeinbildenden Fach nicht mehr gerecht. Breier forderte deshalb 1994:

> „In einem zeitgemäßen Informatikunterricht steht meines Erachtens nicht

> der Algorithmus, sondern die Information als Erscheinungsform der realen Welt im Mittelpunkt“ (vgl. [Hub07], S. 78).

Dabei sei darauf hingewiesen, dass Information hier nicht im Sinne Shannons, der Information lediglich auf die Wahrscheinlichkeit des Informationsgehaltes eines Zeichens reduziert, sondern im Sinne der Formulierungen der Bildungsstandards für die Sekundarstufe II:

> Information nennt man den abstrakten Gehalt („Bedeutungsgehalt“, „Semantik“) einer Aussage, Beschreibung, Anweisung, Nachricht oder Mitteilung. Die äußere Form der Darstellung nennt man Repräsentation (konkrete Form der Nachricht) (vgl. [Bil16], S. 9).

Hubwieser weist darauf hin, dass nicht alles, was mit „Nachrichten und deren Bedeutungsinhalt zu tun hat, automatisch zum Gegenstand der Schulinformatik [werden darf]“, denn damit würden die in der Philosophie, Rechtswissenschaft, Psychologie oder Biologie beheimateten Themengebiete mit erfasst werden (vgl. [Hub07]. S.79). Ein informationsorientierter Informatikunterricht orientiert sich stark an einem erweiterten EVA[1]-Prinzip (vgl. Abbildung 2.1). Im Kern geht es darum, dass ein solcher Unterricht sich mit den Fragen der Darstellung von Informationen (Repräsentationen) in Form von Daten, der Verarbeitung (Übertragung) und der Interpretation dieser Daten beschäftigt. Dabei spielt es keine Rolle, ob die Daten über eine „Leitung“, also einem Netzwerk oder dem Internet, übertragen oder ob sie lokal auf einer Maschine in einer Datei gespeichert werden.

[1] EVA ist die Abkürzung für **E**ingabe-**A**usgabe-**V**erarbeitungs-Prinzip.

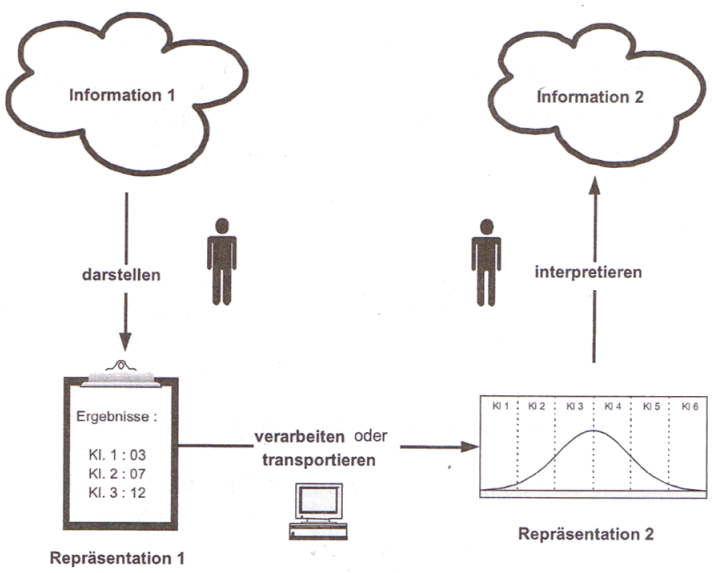



Abbildung 2.1: Informationsorientierter Informatikunterricht aus [Hub07]

Ein Informatikunterricht nach dem *informationsorientierten fachdidaktischen Ansatz* orientiert sich an den Leitlinien (vgl. [HB02], S. 36):

1. Interaktion mit Informatiksystemen,
2. Wirkprinzipien von Informatiksystemen,
3. Informatische Modellierung,
4. Wechselwirkungen zwischen Informatiksystemen, Individuum und Gesellschaft.

Die erste Leitlinie fordert, dass durch die Konfrontation der Schüler mit Informatiksystemen eine Vorstellung über Strategien und Methoden dieser Informatiksysteme geschaffen wird und dieses Wissen auf andere analoge Situationen übertragen werden kann. Die Schüler müssen Informationen sammeln, organisieren, verarbeiten, sichern und wiederherstellen, repräsentieren, präsentieren und evaluieren. Sie sollen so unabhängig von bestimmten Produkten werden und selbstständig geeignete Werkzeuge zur Lösung ihrer Probleme benutzen. Des Weiteren lernen sie hier Aspekte der Softwareergonomie kennen und können ihr erworbenes Wissen auf eigene erstellte Informatiksysteme übertragen.
Die zweite Leitlinie gibt vor, dass Schüler Konzeptwissen über die Konstruktionsweise eines Informatiksystems erlangen. Dazu gehören Erkenntnisse über die Bestandteile eines solchen Systems und das Wissen um die Interaktion zwischen den

einzelnen Komponenten. Am Ende einer Unterrichtseinheit nach dem informationsorientierten Ansatz sollen elementare Ideen und Konzepte wie z. B. die Digitalisierung, Automatentheorie, aber auch die Funktionsweise eines modernen Computers (z. B. Prozessoren, Speicher, Bussystem) vorhanden sein.
Schüler sollen der dritten Leitlinie nach verstehen, dass Informatiksysteme ein Ergebnis des informatischen Modellierens eines realen lebensweltlichen Ausschnitts sind. Dazu gehört der Erwerb einer Modellierungskompetenz. Schüler müssen hierfür die Benutzung sinnvoller Modellierungstechniken (z. B. UML[2]) erlernen.
Der vierte Aspekt stellt einen direkten Bezug zu Coys Forderung her, dass ein zeitgemäßer Informatikunterricht den politischen, rechtlichen, ökonomischen und gesellschaftlichen Kontext, die Frage welche Auswirkungen Informatiksysteme auf den Menschen und die Gesellschaft haben, mit diskutieren muss (vgl. [Coy05], S. 18).

## 2.3 Kompetenz- und projektorientierter Informatikunterricht

Seit 2003 existiert ein Dokument aus den OECD-Veröffentlichungen über Schlüsselkompetenzen (engl. „Key Competencies“ ). Dort formuliert Rychen (vgl. [Ryc03]):

> „A competency is more than just knowledge and skills. It involves the ability to meet complex demands, by drawing on and mobilising psychosocial resources (including skills and attitudes) in a particular context.“

Dabei werden diese Schlüsselkompetenzen in drei übergeordneten Kategorien eingebettet. Das sind 1. „use tools interactivly“ , 2. „interact in heterogenous groups“ und 3. „act autonomously“ . Die Grundlage eines jeden Rahmenplanes und des damit verbundenen Kompetenzbegriffes regelt hierzulande die Kultusministerkonferenz (KMK). Der Kompetenzbegriff wird dort wie folgt definiert:

> „*Kompetenz* bezeichnet den Lernerfolg in Bezug auf den einzelnen Lernenden und seine Befähigung zu eigenverantwortlichem Handeln in beruflichen, gesellschaftlichen und privaten Situationen. Demgegenüber wird unter Qualifikation der Lernerfolg in Bezug auf die Verwertbarkeit,

[2]Unified Modelling Language

d.h. aus der Sicht der Nachfrage in beruflichen, gesellschaftlichen und privaten Situationen, verstanden. (vgl. KMK, 2000, S. 9 in [SS11], S. 36)“ .

Die OECD-Kategorien und die Definition des Kompetenzbegriffes durch die KMK finden sich in den 2008 definierten Bildungsstandards wieder und werden dort für den Informatikunterricht konkret ausformuliert (vgl. [Bil08], S. 11 und [AB14]):

- „Modellieren und Implementieren“ d. h. die Schülerinnen und Schüler sollen eigene Modelle für vorgegebene Sachverhalte entwickeln und diese mit geeigneten Werkzeugen in Programmen umsetzen.
- „Begründen und Bewerten“ d. h. die Schülerinnen und Schüler müssen die Sinnhaftigkeit ihrer Lösungsansätze legitimieren und die Ergebnisse ihrer eigenen Arbeit einordnen.
- „Strukturieren und Vernetzen“ d. h. die Schülerinnen und Schüler sollen Sachverhalte sinnvoll zerlegen und anordnen (strukturieren) und Verbindungen innerhalb der Informatik sowie zu Gebieten außerhalb der Informatik (andere Fächer und Lebensbereiche) erkennen und nutzen.
- „Kommunizieren und Kooperieren“ d. h. die Schülerinnen und Schüler sollen sich fachgerecht über informatische Sachverhalte austauschen und für die Teamarbeit Kommunikationsmedien sachgerecht und sozialverträglich auswählen.
- „Darstellen und Interpretieren“ d. h. die Schülerinnen und Schüler sollen Informationen für die automatisierte Bearbeitung geeignet darstellen und die im Prozess der Verarbeitung gewonnenen Daten korrekt interpretieren.

Ein kompetenzorientierter Informatikunterricht fördert durch geeignete Unterrichtskonzepte diese Kompetenzen, wie sie in den Standards gefordert werden. Doch welche Unterrichtskonzepte sind geeignet? Ein klassischer Frontalunterricht mit langen Lehrermonologen kann nicht in Frage kommen, denn die obigen Kompetenzen stellen den aktiven Schüler in den Mittelpunkt. Diese Forderung ist im Übrigen nicht neu, denn schon Klafki folgerte in seinen fünf Thesen zur kritisch-konstruktiven Didaktik u. a. (vgl. [Kla06], S. 15):

1. Unterricht muss die Entwicklung von **Selbstbestimmungs- und Solidaritätsfähigkeiten**, die Entwicklung der Fähigkeit zur **Reflexion** und der **Be-**

**gründung** sowie die Entwicklung der **Emotionalität** und **Handlungsfähigkeit** unter Einwirkung der gesellschaftliche Wirklichkeit voranbringen.

2. Das Lernen und Lehren versteht sich als Interaktionsprozess. Dabei müssen Schüler die Fähigkeit für das **eigene weitere Lernen** entwickeln können.
3. Lernen muss im Kern des Konzeptes als **sinnhaft**, verstehendes und **entdeckendes** und **nachentdeckendes Lernen** aufgefasst werden können.
4. Der Lern- und Lehrprozess muss das **Selbst-** und **Mitbestimmungsprinzip** als Folge **wachsender Schwierigkeitsgrade** und **wachsendem Anspruch** verwirklichen. Die Folge muss ein **schülerorientierter** Unterricht sein.

In einem schülerorientierten Unterricht eignet sich der aktive Schüler die Informatik handlungsorientiert, problemorientiert, anwendungsorientiert und ganzheitlich an und wird dabei vom Lehrer begleitet (vgl. [SS11], S.31). Zu den damit verbundenen und zugrundeliegenden Unterrichtskonzepten gehören u. a. der Projektunterricht, der handlungsorientierte Unterricht, der offene Unterricht und der problemorientierte Unterricht. Insbesondere findet sowohl der projekt- als auch der handlungsorientierte Unterricht seit mittlerweile zwanzig Jahren immer mehr Unterstützer unter den Fachdidaktikern.
Der handlungsorientierte Unterricht ist ein konstruktivistisches Unterrichtskonzept, d. h. der Schüler oder die Schülergruppe erwirbt das Wissen und Können durch die selbstständige Auseinandersetzung mit einem Unterrichtsgegenstand. Hierbei findet zwischen den Lehrenden und den Lernenden eine Einigung auf ein sogenanntes „Handlungsprodukt“ statt (vgl. [JM11], S. 309). Interessenorientierung, Selbsttätigkeit und Führung, Verknüpfung von Kopf- und Handarbeit, solidarisches Handeln und Produktorientierung charakterisieren diese beiden Unterrichtskonzepte. Das Ziel dieses Konzeptes ist die Förderung der Teamfähigkeit und der Selbstorganisation des Lernens. Die Abbildung 2.2 zeigt einen typischen Unterrichtsverlauf eines handlungsorientierten Unterrichtskonzeptes.

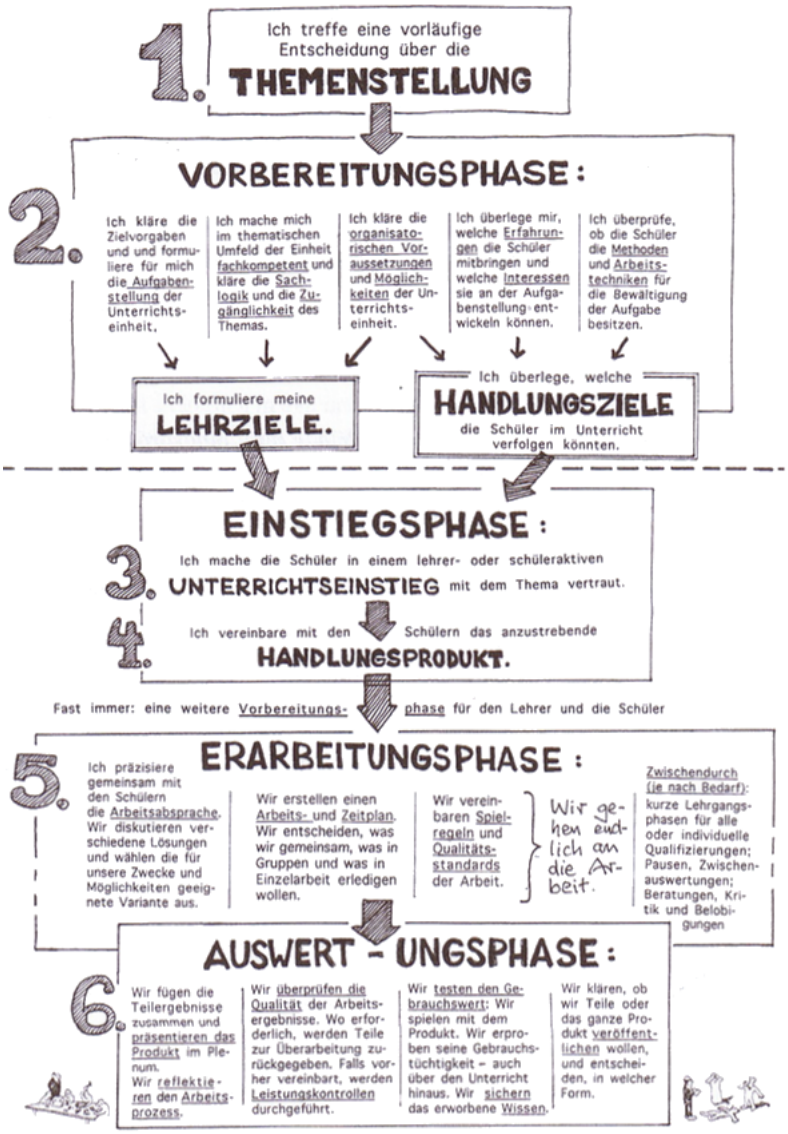



Abbildung 2.2: handlungsorientierter Unterricht aus [JM11], S. 329

Dabei dürfen natürlich nicht die Nachteile, die ein solches Unterrichtskonzept mit sich bringt, außer Acht gelassen werden. Jank und Meyer führen z. B. an, dass handlungsorientierter Unterricht

- im stark verregelten Schulbetrieb Unruhe schafft,
- mehr Vor- und Nachbereitungszeit für die Lehrkraft fordert,
- störungsanfälliger ist, weil er von seinen Zielen her deutlich komplexer ist (vgl. [JM11], S. 334).

Gerade durch die Komplexität, wie sie auch in Abbildung 2.2 zu sehen ist, werden

deshalb in der Praxis immer wieder handlungsorientierte Phasen eingefügt.
Neben handlungsorientierten Phasen leistet auch die Problemorientierung einen Beitrag zur Schüleraktivierung, denn mit ihr werden Schüler vor Anforderungen gestellt, deren Lösung mit dem bisher erworbenen Wissen nur Ansatzweise gelingt. Es macht die Erschließung und Motivierung neuer Lernquellen möglich und es werden dadurch neue soziale Prozesse durch Kooperationen und Kommunikation zwischen Schülern angeregt. Handlungsorientierung und Problemorientierung, schreibt Schwill, „zwingen Lehrer zur Auseinandersetzung mit der informatikspezifischen Vorgehensweise beim Lösen von Aufgaben“ (vgl. [SS11], S. 32). Schüler müssen hier eine geeignete Auswahl an Informatikprinzipien und -methoden treffen und bleiben somit frei in der angemessenen Wahl von Informatikwerkzeugen. Ein solcher Informatikunterricht ist frei von einer reinen Produktschulung, wie sie z. B. im ITG[3]-Unterricht oder in Medienkursen noch häufig praktiziert wird.
Ganzheitliches Lernen bedeutet, wie bereits beim handlungsorientierten Unterricht erwähnt, dass das Lernen mit allen Sinnen (Kopf-, Herz und Handarbeit) stattfindet. Die Anwendungsorientierung verlangt die strengere Verknüpfung zwischen einem Unterrichtsgegenstand und einem lebensweltbezogenen Kontext.

## 2.4 Kontextorientierter Informatikunterricht

In einem zeitgemäßen Informatikunterricht kommen kontextbasierte Unterrichtskonzepte zum Einsatz. Ein neues informatisches und kontextorientiertes Unterrichtskonzept ist „Informatik im Kontext“ (im Folgenden IniK). Das IniK-Konzept wird in der didaktischen Diskussion schon länger verwendet (vgl. [Eng05]), ist jedoch erst ab 2009 im Rahmen einer Initiative heraus weiterentwickelt worden und orientiert sich an anderen ähnlichen Initiativen aus den etablierten Naturwissenschaften. Dazu gehören Biologie im Kontext (BiK), Physik im Kontext (PiCo) und Chemie im Kontext (ChiK). Ein Informatikunterricht der IniK umsetzt basiert auf drei Prinzipien (vgl. [KWSS09], S. 4):

**1. Orientierung an Kontexten**

Informatikunterricht mit IniK wählt Kontexte aus, die sich an der Lebenswelt der

[3]ITG steht für **I**nformations**t**echnische **G**rundlagen

Schüler orientiert. Das ist nicht neu, denn auch der *anwendungsorientierte fachdidaktische Ansatz* verfolgte diese Zielstellung. Anders als der *anwendungsorientierte fachdidaktische Ansatz*, der i. d. R. nur am Anfang einen Kontext zur Motivierung der Schüler nutzt, steht er bei IniK stets im Mittelpunkt der gesamten Unterrichtseinheit.
Für das IniK-Konzept bedeutet Kontext die „Menge von lebensweltlichen Themen bzw. Fragestellungen, die von den Schülern als zusammenhängend geordnet werden und die dadurch sinnstiftend auf deren Handlungen wirken“ (vgl. [KWSS09], S. 5). Durch einen solchen Kontext wird ganz im Sinne Coys ein vieldimensionaler Handlungsrahmen (siehe Kapitel 2.2) betrachtet und der vierten Leitlinie des informationsorientierten Ansatzes gerecht (vgl. [HB02], S. 36, Punkt 4). Es wird über die informatische Dimension hinaus versucht, die gesellschaftliche, politische, rechtliche, ethische und ökonomische Dimension eines Kontextes zu erarbeiten. Auch wenn innerhalb eines Kontextes nicht alle Dimensionen erfasst werden können, erhebt IniK den Anspruch, mehr als nur die informatische Dimension zu betrachten.

**2. Orientierung an den Standards für Informatik in der Schule**

Ein Informatikunterricht nach IniK, der sich an den Standards[4] für Informatik in der Schule orientieren muss, basiert auf dem *informationsorientierten* fachdidaktischen Ansatz, weil dieser Ansatz in den Standards integriert ist. Die Abbildung 1.1 zeigt deutlich, dass der Inhaltsbereich „Information und Daten“ und der Prozessbereich „Darstellen und Interpretieren“ alle anderen Inhalts- und Prozessbereiche einschließt. Noch deutlicher wird das in der Abbildung 2.3, die noch aus der Entwurfsphase für die Standards stammt, weil dort der Inhaltsbereich „Daten und Information“ im Zentrum steht.

[4] [Bil08], S. 11, Abbildung 1.1)

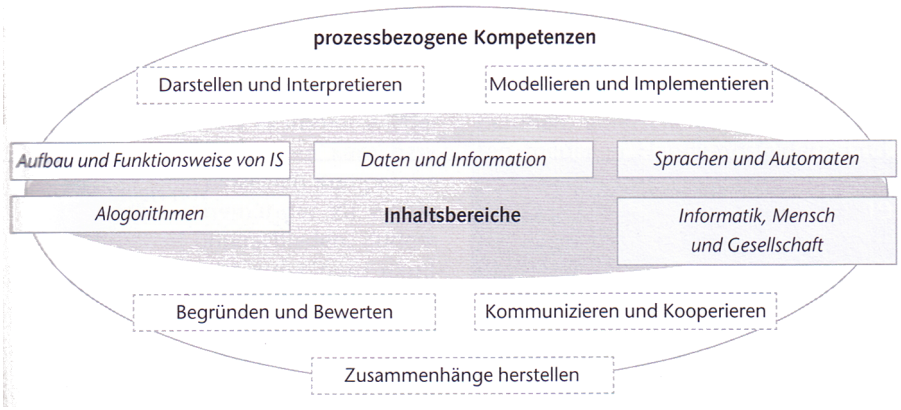



Abbildung 2.3: Prozess- und Inhaltsbereiche der Standards für Informatik in der Schule (vgl. [BBF+06], S. 15)

### 3. Methodenvielfalt

Bereits die Phasierung einer IniK-Unterrichtseinheit lässt auf einen vielfältigen Methodeneinsatz schließen. Die Phasen einer IniK-Unterrichtseinheit orientieren sich wie bei allen modernen kontextorientierten Unterrichtseinheiten an dem Vorläufer ChiK. Die Abbildung 2.4 stellt diese Phasen sowie mögliche Unterrichtsmethoden der Chemie exemplarisch dar.

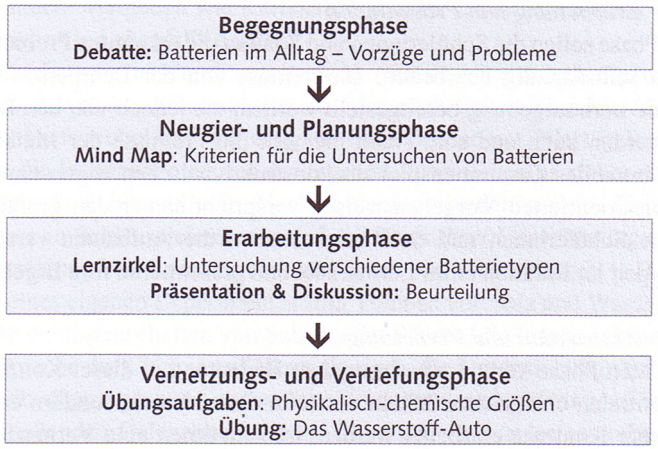



Abbildung 2.4: Phasen einer typischen modernen kontextorientierten Unterrichtseinheit (vgl. [PRF08], S. 27)

Die **Begegnungsphase** dient der Konfrontation der Schüler mit einem lebens-

weltbezogenen Kontext und sorgt dafür, dass das Interesse an Informatik geweckt wird. Mithilfe der Einbeziehung von Fachliteratur und eventuellen Internetrecherchen werden Bezugspunkte zum Thema gesucht und der Kontext aufgebaut. In der **Neugier- und Planungsphase** geht es um die Strukturierung der weiteren Unterrichtsarbeit. Hier werden Fragen der Schüler gesammelt und Arbeitspläne formuliert. Dabei hilft die Formulierung von Leitfragen. Die Erarbeitung von Lösungen findet sich in der **Erarbeitungsphase** wieder. Es müssen die Informationen, die in einem Kontext betrachtet werden, mithilfe von Daten dargestellt, repräsentiert und interpretiert werden. Dabei werden Schüler mit den Modellierungstechniken der Informatik konfrontiert und vertraut. Die dabei notwendigen Hilfsmittel liefern die in der Informatik bereitgestellten Konzepte. Zu nennen sind hier beispielsweise Sprache und Automaten. Diese Konzepte können dann in der Vernetzungs- und Vertiefungsphase grundlegend erarbeitet werden. Konzeptwissen wird hier zu kontextunabhängigem systematischen Fachwissen.

IniK mit einem informationsorientierten fachdidaktischen Ansatz ist darüber hinaus kompatibel mit anderen zeitgemäßen und alternativen fachdidaktischen Konzepten wie z. B. dem Konzept der fundamentalen Ideen nach Schwill. Eine fundamentale Idee ist ein fachinhaltliches „Denk-, Handlungs-, Beschreibungs- oder Erklärungsschema“ , das nach Schwill folgenden Kriterien genügen muss:
„Eine *fundamentale Idee* ist [...]

1. in verschiedenen Gebieten eines Bereichs vielfältig anwendbar oder erkennbar (Horizontalkriterium),
2. auf jedem intellektuellen Niveau aufzeigbar und vermittelbar (Vertikalkriterium),
3. zur Annäherung an eine gewisse idealisierte Zielvorstellung geeignet, die jedoch faktisch möglicherweise unerreichbar ist (Zielkriterium),
4. in der historischen Entwicklung des Bereichs deutlich wahrnehmbar und bleibt längerfristig relevant (Zeitkriterium),
5. der Bezug zu Sprache und Denken des Alltags und besitzt einen Bezug zur Lebenswelt und ist für das Verständnis des Faches notwendig (Sinnkriterium)“ (vgl. [SS11], S. 75-76).

Schwill entwickelte von diesem Konzept ausgehend die Masterideen *Algorithmisierung, Strukturierte Zerlegung und Sprache* für den Informatikunterricht. In einem

Informatikunterricht nach dem informationsorientierten Ansatz lassen sich diese Masterideen inhaltlich zur Deckung bringen, denn für die Darstellung von Informationen mithilfe von Daten müssen sich Schüler mit der strukturierten Zerlegung beschäftigen und Algorithmen für die Verarbeitung oder den Transport dieser Daten entwerfen. Kommen Metasprachen in der obigen IniK-Unterrichtseinheit zum Einsatz, wird die Masteridee Sprache im Unterricht thematisiert.

## 2.5 Medienaffiner Informatikunterricht

In einer digital geprägten Gesellschaft und Kultur ermöglicht die Persönlichkeitsentwicklung, die Medienbildung und das Medienhandeln den gesellschaftlichen Anschluss und die Erwerbsfähigkeit, so dass einer drohenden digitalen Spaltung der Gesellschaft entgegengewirkt werden kann (vgl. [Sch10b], S. 5). Eine breite, aber auch spezialisierte Medienbildung initiiert innovative Impulse für die Arbeitswelt und kann eine Verbesserung der Lebensqualität herbeiführen. „Zeitgemäßer Informatikunterricht beachtet, dass informatische Bildung und Medienbildung zwei sich ergänzende, wechselseitig bedingende Aufgaben schulischer Bildung und Erziehung sind, die sich immer weiter aufeinander zubewegen“ ([AB14]). Das zeigt sich beispielsweise bei der Dekonstruktion von Informatiksystemen unter verschiedenen Sichtweisen:„

- Sicht auf die Hardware bzw. den technisch apparativen Teil des Mediums
- Sicht auf die Benutzungsoberfläche (GUI) und die damit induzierten sozialen Interaktionen bzw. die interaktive Seite des Mediums
- Sicht auf die Vernetzung der Informatiksysteme bzw. die kommunikativen Aspekte der computerbasierten Medien
- Sicht auf die Software und die damit während der Softwareentwicklung realisierten Operationalisierungen von antizipierten sozialen Handlungen im Kontext des Informatiksystems
- Sicht auf die Software im Sinne der dort realisierten informatischen Ideen und Konzepte der Softwaretechnik einschließlich der verwendeten Algorithmen
- Sicht auf die Software im Hinblick auf die Semiotik und die verwendeten symbolischen Repräsentationen
- Sicht auf Software im Hinblick auf von ihr erzeugte virtuelle Realität
- Sicht auf Software im Hinblick auf die Datenbasis, deren Struktur, Inhalt und Repräsentationsformen.

“ ([Mag], S. 7).
Dass ihre Aufgabenfelder nicht überschneidungsfrei sind, liegt in der Natur der neuen, digitalen Medien. Sie sind aus technischer Sicht Informatiksysteme, die aus Hard- und Software bestehen und häufig mit dem Internet als Leitmedium verbunden sind. Der Beitrag des Informatikunterrichts zur Medienbildung liegt heutzutage nicht mehr nur allein in der Bereitstellung grundlegender informatischer Methoden und Sichtweisen, die ein Verständnis digitaler Medien erst ermöglichen (vgl. [fIeV99], S. 7), sondern auch darin, dass digitale Medien selbst Gegenstand des Informatikunterrichts werden und die Schülerinnen und Schüler dort Medienkompetenz erlangen. Das wird möglich, indem bereits bei der Auswahl der Kontexte (vgl. Kapitel 2.4) ein enger Bezug zu digitalen Medien angestrebt wird. Typische Kontexte könnten sein:

- E-Mail nur(?) für Dich
- Mein Computer spricht mit mir!
- Smart und reich durch Apps
- Cybermobbing
- Soziale Netze
- Traue keinem Bild!
- Mobiles Internet - ständig und überall erreichbar sein!?
- Computerspiele - zwischen Spaß und Sucht

Einige dieser Themen (vgl. [Ini14]) haben sich bereits im Unterricht bewährt und gezeigt, dass sie insbesondere Schülerinnen verstärkt ansprechen. Andere Themen sind noch auszuarbeiten“ ([AB14]).

## 2.6 Gendersensibler Informatikunterricht

Ada Lovelace, eine Frau war es, „die das erste komplexe Computerprogramm schrieb - und zwar bereits 1843 in London, fast 100 Jahre vor der Erfindung des ersten elektronischen Rechners Z3, den Konrad Zuse am 12. Mai 1941 in Berlin vorstellte“ (vgl. [FKM16], S. 3). Trotzdem werden die Berufe in der IT-Welt hauptsächlich von männlichen Mitarbeitern dominiert. In der Schule, dass zeigen eigene Erfahrungen, sieht das nicht anders aus. Hier wählen i. d. R. mehr Jungen als Mädchen Informatik als Wahlpflichtfach. Wahlpflichtkurse in denen nur Jungen sitzen und Mädchen fehlen sind keine Seltenheit. Lehrkräfte der Informatik müssen sich dieser

Tatsache bewußt sein und das Fach für beide Geschlechter attraktiv gestalten, d. h. ein zeitgemäßer Informatikunterricht muss deshalb gendersensibler Informatikunterricht sein, für Schülerinnen und Schüler ein gleichermaßen geeignetes Lernklima schaffen und sie ermutigen, die eigenen Begabungen zu entdecken und ihr Selbstbewusstsein zu stärken. Dass Schülerinnen durchschnittlich weniger Erfahrungen im Umgang mit Computern haben als Schüler muss ein zeitgemäßer Informatikunterricht berücksichtigen (vgl. [AB14] und vgl. [JA02], S. 1). Des Weiteren dürfen diese Unterschiede nicht verstärkt werden, sondern müssen abgebaut werden. Hilfreich sind Kontexte, die insbesondere Mädchen ansprechen und ihre Kompetenzen zum Tragen kommen lassen. „Ein Schlüssel liegt [...] auf dem Fokus, mit dem die Beschäftigung mit Inhalten der Informatik vermittelt wird. Erfahrungswerte zeigen, dass ein Großteil der Schülerinnen und Studentinnen nicht an Technik ‚an sich' interessiert sind, sondern an Technik ‚wofür'. Die Anwendungskontexte und die Gestaltung von Technik auf diese hin motivieren sie, sich mit Technik und den Kontexten gleichermaßen zu befassen" (vgl. [Sch13], S. 21). „Ein fachlich breit angelegter, nicht rein Technik-fokussierter Unterricht, der den Interessen und Fähigkeiten beider Geschlechter gerecht wird, wirkt diesem Ungleichgewicht entgegen und schafft damit eine herausfordernde und offene Lernatmosphäre" (vgl. [Kla16], S. 37). Ein einseitiger Informatikunterricht der auf abstrakt mathematische und informationstechnische Fragestellungen reduziert ist, wirkt uninteressant für Mädchen (vgl. [RS12], S. 331-338). Es ist gerade die Freude am Problemlösen, an der Kreativität, an der Kommunikation und Interdisziplinarität sowie soziale und gesellschaftliche Aspekte die insbesondere für Frauen attraktiv sind (vgl. [RS12], S. 331-338). „Insbesondere Programmieraufgaben sollten sich daher nicht auf reine mathematische Problemstellungen oder Computerspiele beschränken. Vielmehr sollten ein direkter Nutzen und ein Bezug zur eigenen Lebenswelt erkennbar sein" (vgl. [Kla16], S. 37). „Gerade die Gestaltung von Technik und die damit verbundene Kreativität bekommt in einem gendersensiblen Informatikunterricht eine besondere Bedeutung, denn verschiedene Studien deuten darauf hin, dass mithilfe von Kreativität das Interesse von Frauen und Mädchen für die Informatik gewonnen werden kann" (vgl. [Rom08], S. 111).
„Des Weiteren steht in einem gendersensiblen Informatikunterricht die Geschlechtergerechtigkeit im Mittelpunkt. Geschlechtergerechtigkeit bedeutet für den Informatikunterricht, stets Auswirkungen auf die Schülerinnen und Schüler bezüglich ihrer überholten und ‚tradierten' Rollenzuschreibung aufzuspüren und aufzubrechen.

Hierfür müssen Lehrerinnen und Lehrer Aufgabenstellungen, bzw. größere Informatikprojekte von beiden Geschlechterperspektiven aus analysieren. Dabei sollten die Projekte immer attraktive und kreative Elemente jeweils zu gleichen Anteilen für beide Geschlechter beinhalten, so dass beide Geschlechter die für die Informatik notwendigen Kompetenzen (Modellieren, Implementieren, usw.) innerhalb der jeweiligen Teilaufgabe, die eine Schülerin oder ein Schüler bearbeitet, erlernen können“ ([AB14]).

## 2.7 Sprachfördernder Informatikunterricht

„Sprachen spielen im Informatikunterricht eine sehr große Rolle und bieten vielfältige Möglichkeiten, sprachliche Kompetenzen bei den Schülerinnen und Schülern aufzubauen. Dabei wird bewusst zwischen den verschiedenen Sprachebenen (Alltags-, Bildungs-, Fachsprache) gewechselt und berücksichtigt, dass Schüler mit Deutsch als Zweitsprache nicht in jedem Fall auf intuitive und automatisierte Sprachkenntnisse zurückgreifen können. Sprachliche Kompetenzen werden im Informatikunterricht systematisch aufgebaut, in dem die Schüler immer wieder und in vielfältiger Form Gelegenheit erhalten, komplexe Zusammenhänge mündlich und schriftlich in unterschiedlichen Textsorten darzustellen (vgl. [AJS11], S. 16). Der Informatikunterricht unterstützt die Entwicklung einer allgemeinen Sprachkompetenz der Schüler“ ([AB14]).

Volker Claus erkannte das bereits 1995 und postulierte den Informatikunterricht damals als Sprachenunterricht und schlug vor, dass „die Analyse und die Verwendung von Sprachen unterschiedlichster Art ausgehend von Sprachen der Informatik den allgemein bildenden Kern für das neu orientierte Schulfach Informatik bilden (sollen). [...] Dabei bevorzugt die Informatik keine spezielle Sprache: Normen, Programmabläufe, Entwürfe, Pflichtenhefte usw. werden in natürlichen Sprachen dargestellt; Algorithmen, Datenstrukturen oder Objekte beschreibt man meist in halbformalen Sprachen, für theoretische Fragestellungen und Modellbildungen benutzt man mathematische Kalküle und Programme müssen letztlich in einer künstlichen Sprache ausformuliert werden“ (vgl. [Cla95], S. 43).

## 2.8 Fachübergreifender und fächerverbindender Informatikunterricht

„Unterricht nach ‚Informatik im Kontext' ist per se fachübergreifender, fächerverbindender Unterricht, denn wenn man von einem lebensweltlichen Kontext ausgeht, gibt es vielfältige Bezüge zu unterschiedlichen Fächern" ([AB14]). „Hier ist nun die Befürchtung, dass man sich im kontextorientierten Informatikunterricht in den verschiedenen Bezugswissenschaften verliert und nicht mehr zur Informatik zurückfindet - also: ‚Lost in Kontext'? [...] Vermutlich ist kein Unterrichtender in der Lage, alle diese Fragen mit der gleichen Fachkompetenz zu behandeln. Das liegt u. a. an der Konzeption der Lehrerbildung. Von der Ausbildung her sind Informatiklehrer in der Kultur einzelner Fächer sozialisiert, dies gilt in besonderem Maß für diejenigen aus der Studienratslaufbahn. Eine ideale, aber leider häufig nicht zu realisierende Lösung für dieses Problem ist der fächerverbindende Unterricht, bei dem sich Kolleginnen und Kollegen aus den betreffenden Fächern zu einem Team zusammenschließen. Beim fachübergreifenden Unterricht, den eine einzelne Lehrkraft stemmen muss, empfiehlt sich der, ‚Mut zur Lücke'!" (vgl. [DKW11], S. 100).

# Kapitel 3

# Zur Verankerung von Themen der theoretischen Informatik im IU

## 3.1 Theoretische Inhalte im IU der Bundesrepublik Deutschland

Die theoretischen Konzepte der Informatik sind in den Bildungsplänen für Informatik in der Bundesrepublik Deutschland überwiegend verankert. Im Schulkontext werden i.d.R. folgende Themen im Rahmen einer Unterrichtseinheit „Formale Sprachen und Formale Grammatiken“ untersucht:

- Alphabete, Wörter und Zeichen (AWZ)
- Syntax und Semantik (SUS)
- Grammatiken (G)
- Ableitungen (A)
- Rekursion (R)
- Notationsformen (NF)
- Endlicher Automat (EA)
- Implementierung (I) eines Endlichen Automaten (EA) oder Parsers (PA)
- Chomsky-Hierarchie (CH)

Die Tabelle 3.1 zeigt, welche der Themen in den einzelnen Bundesländern Unterrichtsgegenstand sind. Es gibt Länder, in denen das Thema Formale Sprachen vollständig weggelassen werden kann, bzw. wo es nicht direkt der theoretischen Informatik zugeordnet wird. Diese theoretischen Konzepte werden im Rahmen eines anderen

Kontextes erarbeitet. Zu diesen Ländern gehören Hamburg, Sachsen und Sachsen-Anhalt. Dagegen findet sich z.B. in Hessen, die Behandlung regulärer, kontextfreier und sogar kontextsensitiver Grammatiken.

| | Geforderte Fachinhalte | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Bundesländer | AWZ | SUS | G | A | R | NF | EA | I | CH |
| B.-Würtemberg (vgl. [BPI05]) | W | W | W | W | W | W | W | W, EA | W |
| Bayern (vgl. [BPI13]) | X | X | X | X | X | X | X | X, EA | W |
| Berlin (vgl. [SfB06]) | X | X | X | X | X | X | X | X, EA | W |
| Brandenburg (vgl. [MfB11]) | X | X | X | X | X | X | X | X, EA | W |
| Bremen (vgl. [Löw09]) | X | X | X | X | X | X | X | X, EA | X |
| Hamburg (vgl. [AJSS09]) | W | W | W | W | W | W | W | W, PA | W |
| Hessen (vgl. [Kul10]) | X | X | X | X | X | X | X | W, PA | X |
| M.-Vorpommern (vgl. [RMe04]) | X | X | X | X | X | X | X | X, EA | W |
| Niedersachsen (vgl. [EPA04]) | X | X | K | X | K | X | K | K | K |
| N.-Westphalen (vgl. [MfSuW99]) | X | X | X | K | K | K | X | X, PA | K |
| Rheinland-Pfalz (vgl. [MfB10]) | X | X | X | K | X | X | X | X, PA | K |
| Saarland (vgl. [Sch10a]) | W | K | K | K | K | K | W | K | K |
| Sachsen (vgl. [fBuS11]) | W | W | W | W | W | W | K | K | W |
| Sachsen-Anhalt (vgl. [EGH+03]) | W | W | W | W | W | W | W | K | K |
| Schleswig-Holstein (vgl. [MfB02]) | K | K | K | K | K | K | X | K | K |
| Thüringen (vgl. [TMfB12]) | X | X | X | X | K | X | X | K | K |

Tabelle 3.1: Formale Sprachen in den Bundesländern

**Abkürzungen:**

- X Fachinhalt wird vorgeschrieben
- W **W**ahlthema, muss nicht zwingend Unterrichtsgegenstand sein
- K **K**eine Angabe
- PA **P**arser wird implementiert
- EA **E**ndlicher Automat wird implementiert

## 3.2 Theoretische Inhalte im Hamburger IU

Der Rahmenplan der Freien und Hansestadt Hamburg (Vgl. [AJSS09]) stellt auf Seite 13 die Forderung „Die Schüler (...) unterscheiden natürliche und formale Sprachen“ . Ferner wird auf Seite 17 die Auseinandersetzung mit „Sprache als Werkzeug der Kommunikation (Aspekte formaler Sprachen, Syntax und Semantik)“ gefordert. Auf Grundlage des Rahmenplans muß in der Studienstufe des Gymnasiums für jedes Semester ein Kontext gewählt werden. Die theoretischen Konzepte der Informatik werden i.d.R. im dritten Semester erarbeitet. Im Rahmen der „Möglichkeiten und Grenzen von Informatiksystemen“ wird der Einsatz von formalen Grammatiken diskutiert. Der Kontext für dieses Semester ist die maschinelle Sprachübersetzung in Form eines Wort-zu-Wort-Übersetzers, bei dem vor der eigentlichen Übersetzung die Syntax des zu übersetzenden Satzes mit Hilfe einer Grammatik überprüft wird. Dabei steht die funktionale Implementierung des Grammatikprüfers in Scheme im Vordergrund. Die kompletten Lerninhalte dieses Semesters können dem Skript von Seiffert und Kück (Vgl. [SK11]) entnommen werden.
Im eigenen Unterricht auf Grundlage des genannten Skriptes zeigten sich wiederholt logische Brüche, die wie folgt zusammengefasst werden können:

1. Der Unterrichtsgang vom Wort-zu-Wort Übersetzer zu formalen Grammatiken
2. Das Abdriften in einen Scheme Programmierkurs
3. Der unreflektierte Einsatz eines Parsers

Zu (1): Die Grammatikprüfung wird im Unterricht als Hilfsmittel für den Sprachübersetzer thematisiert. Dabei wird der Satz vor der Übersetzung auf grammatische Richtigkeit überprüft. Danach erfolgt die Wort-zu-Wort Übersetzung, womit die formale Grammatik und die Überprüfung überflüssig werden. Das Problem besteht bei diesem Unterrichtsgang darin, dass die formale Grammatik für die eigentliche Übersetzung nicht benutzt und damit die eigentliche Notwendigkeit einer formalen Grammatik bei der Sprachübersetzung nicht thematisiert wird. Der Einsatz einer formalen Grammatik vor der Sprachübersetzung wirkt aus Schülersicht künstlich aufgesetzt.
Zu (2): Der Rahmenplan schreibt für einen Kurs auf erhöhtem Niveau die Behandlung einer sogenannten Typ-B-Modellierung vor. Darunter fallen Programmiersprachen, die entweder ein deklaratives oder funktionales Konzept benutzen. In der Vergangenheit hieß das konkret Prolog, Haskell oder Scheme, wobei sich letzteres bei

den hamburgischen Lehrerinnen und Lehrern durchsetzte nicht auch zuletzt wegen des Skriptes von Seiffert und Kück. Scheme ist eine funktionale Programmiersprache aus der LISP-Familie. LISP steht für **Lis**t **P**rocessing. Für die Schüler stellt die Erarbeitung dieser Programmiersprache eine große Hürde dar, denn zum Einen ist der Abstraktionsanspruch, alle Daten als Listen darzustellen, sehr hoch und zum Anderen sorgt die polnische Notation und ihre einhergehenden Verschachtelungen durch Klammern für sehr viel Unübersichtlichkeit. Dieses Problem versucht das Skript zu beheben, in dem es behutsam in die Arbeit mit der funktionalen Programmiersprache einführt. Das gelingt erfahrungsgemäß nur begrenzt. Spätestens bei der Implementierung einer formalen Grammatik in Scheme scheitern die Schüler. Es ergibt sich unmittelbar die Forderung nach mehr Übungsmaterial, um den Umgang mit Listen zu trainieren. Dadurch werden die theoretischen Konzepte der Informatik zu einer Randnotiz und aus einem Informatikkurs wird ein Scheme-Programmierkurs, was dem aktuell gültigen Hamburger Rahmenplan widerspricht.
Zu (3): Der Parser, bzw. Kellerautomat „fällt auf einmal vom Himmel“. Die Notwendigkeit eines solchen Kellerautomaten wird nicht diskutiert. Er wird vorgegebenen, weil er für die Syntaxanalyse wichtig ist. Doch woher kommt dieses theoretische Konzept des Kellerautomaten? Gibt es noch andere Automaten, Maschinen oder Konzepte? Das alles wird nicht im Rahmen dieses bisherigen Unterrichtsganges betrachtet.

## 3.3 Untersuchung zum Hamburger IU

Die bisherigen subjektiven Erkenntnisse und Erfahrungen bezüglich des Semesters über Sprachverarbeitung werden hier denen anderer Lehrer gegenübergestellt. Die Datenerhebung und Auswertung erfolgt auf Grundlage von qualitativen Interviews, die im Mai des Jahres 2013 durchgeführt wurden. Die Ergebnisse, deren Veröffentlichung im Jahr 2013 in Kurzform auf der WiPSCE erfolgte (vgl. [Ali13]), werden hier ausführlich dargestellt.

### 3.3.1 Ziele des Hamburger IU

In Hamburg werden in einem Semester für Sprachverarbeitung die zu entwickelnden Schülerkompetenzen über den Rahmenplan hinaus in einem Anforderungsheft (kurz A-Heft) für das Zentralabitur in Informatik festgelegt. Darin werden folgende fach-

spezifischen Kompetenzen ([Ros14], S. 106) aufgeführt:
Die Schülerinnen und Schüler

- vergleichen natürliche und formale Sprachen,
- beschreiben grundsätzliche Schwierigkeiten maschineller Sprachverarbeitung,
- interpretieren unterschiedliche Darstellungen von Grammatiken,
- analysieren Sätze einer Sprache und entwickeln die zu ihrer Beschreibung notwendigen Grammatikelemente,
- erstellen einen Parsebaum zu einem gegebenen Satz und einer gegebenen Grammatik,
- analysieren und modifizieren Scheme- oder Haskellfunktionen zum Kontext Sprachverarbeitung,
- veranschaulichen rekursive Prozesse, erkennen Endrekursion und erläutern diese,
- entwickeln funktionale Modellierungen für Teilprobleme der Sprachverarbeitung,
- verwenden Sprachelemente von Scheme oder Haskell syntaktisch korrekt, implementieren Wiederholungen durch rekursiven Funktionsaufruf,
- konstruieren und nutzen einfache Listen, Assoziationslisten und Bäume,
- arbeiten sinnvoll mit Parametern unterschiedlicher Typen, auch mit Funktionen als Parameter,
- vergleichen unterschiedliche Vorgehensweisen zur Realisierung eines Parsers (Abbildung der Grammatikproduktionen als Liste von Listen sowie Tiefen- oder Breitensuche, Funktionen zur Auflösung von Nonterminalen).

Für dieses Semester ist das Thema Sprachübersetzung der Anwendungskontext. Im Verlauf dieses Semesters wird eine Sprachübersetzungssoftware in Scheme modelliert und implementiert, die vom Schüler Schritt für Schritt verbessert werden soll.

### 3.3.2 Erhebungsmethode

Der aktuelle Rahmenplan sieht den Unterrichtsgegenstand „Sprachübersetzungssysteme“ i. d. R. nur in einem Kurs auf erhöhtem Niveau vor. Seit der letzten Oberstufenreform im Jahr 2010 gibt es in Hamburg keine Leistungs- und Grundkurse für das Fach Informatik mehr. Der Unterricht findet in sogenannten Profilen, das sind Fächerbouquets, statt. Informatik wird entweder auf erhöhtem oder auf grund-

legendem Niveau unterrichtet. Das Ziel dieser Reform besteht unter anderem in der Förderung des fächerübergreifenden Unterrichtens. Die Reform führte jedoch auch dazu, dass das Fach Informatik insbesondere auf erhöhtem Niveau sehr selten an den Hamburger Schulen angeboten wird. Aufgrund der daraus resultierenden geringen Stichprobengröße fiel die Wahl auf eine qualitative Datenerhebung.
Des Weiteren lässt sich durch eine qualitative Methode ein detailliertes Bild aus Sicht der betroffenen Lehrer, die ebenfalls dieses Curriculum für einen Kurs auf erhöhtem Niveau unterrichten müssen, zeichnen (vgl. [FvKS05], S. 17).
Um qualitative Daten durch die Lehrer zu erhalten, eignen sich Leitfaden-Interviews. Die Wahl fiel auf die Methode der Experteninterviews, spezielle Leitfaden-Interviews, weil die Lehrer hier nicht als Person von Interesse sind, sondern viel mehr als Experten eines bestimmten Handlungsfeldes (vgl. [FvKS05], S. 214). Der Vorteil des Experteninterviews besteht darin, dass der Informationsverlust im Vergleich zu standardisierten Fragebögen geringer ist, denn im Interview können fehlende Informationen durch gezielte Nachfragen ergänzt werden. Des Weiteren können diese Interviews aufgezeichnet und anschließend transkribiert werden. Ein weiterer Vorteil stellt der direkte zwischenmenschliche Kontakt dar, der dazu führen kann, dass der Interviewer auch Antworten auf besonders kritische Fragen erhält. Letzteres kann auch als Nachteil angesehen werden, denn der direkte Kontakt des Interviewers kann z. B. durch falsches Fragen die Antwortmöglichkeiten des Interviewpartners beeinflussen, indem der Interviewer die von ihm gewünschten Antworten erhält.

### 3.3.3 Ziel der Untersuchung

Das Ziel dieser Untersuchung ist es herauszufinden, inwiefern der anwendungsorientierte Informatikunterricht und der Anwendungskontext Sprachübersetzungssysteme für die Erarbeitung theoretischer Informatikinhalte geeignet sind. Dabei sollen folgende Fragen beantwortet werden:

- *Welche Übereinstimmungen bzw. Unterschiede zwischen den eigenen Erfahrungen und denen anderer Lehrer gibt es?*
- *Werden die Unterrichtsziele der theoretischen Inhalte der Informatik bei anderen Lehrern erreicht?*
- *Welche Faktoren begünstigen das Erreichen bzw. das Misslingen von Unterrichtszielen?*

- *Wie könnten Unterrichtsziele besser erreicht werden?*
- *Welche neuen Ziele könnten durch ein überarbeitetes Curriculum erreicht werden?*
- *Welche Methoden kommen im anwendungsorientierten Informatikunterricht bei den Kollegen häufig zum Einsatz?*

### 3.3.4 Die Stichprobe - das Sampling

Die Auswahl der Lehrer erfolgte nach folgenden drei Kriterien. Die Lehrer sollten

- ein grundständiges Studium der Informatik absolviert haben,
- Erfahrungen mit dem Unterrichten nach dem anwendungsorientierten fachdidaktischen Ansatz haben,
- den Anwendungskontext „Sprachübersetzungssysteme“ unterrichtet haben und
- in der Sekundarstufe II (Oberstufe) eingesetzt sein.

Die zu interviewenden Lehrer wurden durch einen Aufruf über eine Hamburger Informatik-Mailingliste gefunden. Es erklärten sich sechs Lehrer für ein Interview bereit.

### 3.3.5 Aufbau des Interviews

Diese vierzehn Fragen bildeten den Leitfaden.

1. Wann unterrichteten Sie diese Unterrichtseinheit zur Sprachverarbeitung das letzte Mal?
2. Bitte beschreiben Sie, wie ihr typischer Unterricht zur Sprachverarbeitung aussah.
3. Was waren aus Ihrer Sicht sowohl positive als auch negative Erlebnisse?
4. Welche unterrichtlichen Methoden kamen in Ihrem Unterricht überwiegend zum Einsatz?
5. Welche Lernsoftware bzw. Lernumgebungen kamen in Ihrem Unterricht zum Einsatz?
6. Können Sie etwas über die Motivation der Schülerinnen und Schüler aussagen?
7. Wie sahen die Rückmeldungen der Schüler aus?
8. Im Rahmenplan und der vorangegangenen Untersuchung werden Kompetenzen genannt. Welche dieser Kompetenzen werden bisher erreicht?

9. Sind die bisher definierten Kompetenzen für den Kontext Sprachverarbeitung sinnvoll?
10. Wie bilden Sie diese Kompetenzen im gegenwärtigen Informatikunterricht aus?
11. Worin lagen aus Ihrer Sicht die Gründe für die Erreichung bzw. die Nichterreichung der Kompetenzen?
12. Wie könnten diese Kompetenzen besser erreicht werden?
13. Welche neuen Kompetenzen könnten oder sollten durch eine überarbeitete Unterrichtseinheit zur Sprachverarbeitung angestrebt werden?
14. Wie könnten aus Ihrer Sicht diese neuen Kompetenzen erreicht werden?

### 3.3.6 Das Code-System

Die Abbildung 3.1 stellt die Kategorien und ihre zugehörigen Unterkategorien dar.

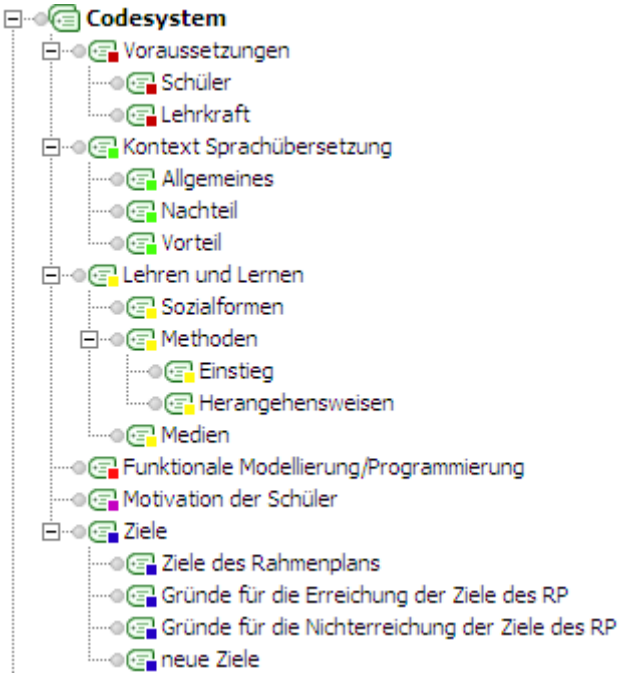



Abbildung 3.1: Coding

Die sechs Hauptkategorien Voraussetzungen, Anwendungskontext Sprachübersetzung, Lehren und Lernen, Funktionale Modellierung/Programmierung, Motivation der Schüler und Ziele ergeben sich aus dem Untersuchungsgegenstand des Anwendungskontextes „Sprachübersetzung“ und dem anwendungsorientierten Informatikunterricht selbst. Die Kategorie Voraussetzungen besitzt die Unterkategorien Schü-

ler und Lehrer. Es ist von Bedeutung, ob ein Lehrer diesen Unterrichtsgegenstand schon mehrfach oder nur ein einziges Mal unterrichtet hat, denn i. d. R. laufen Unterrichtsvorhaben erst nach mehrmaligem Ausprobieren besser. Des Weiteren ist es auch wichtig, wie kompetent sich ein Lehrer mit dem Unterrichtsgegenstand „Sprachübersetzung“ fühlt. Damit können sich Meinungen und Erfahrungen unterschiedlich gewichten lassen. Darüber hinaus spielt auch die Schülerklientel beim Ge- bzw. Misslingen von Unterricht eine Rolle. Es sollte hier der Frage, ob es sich um einen Kurs auf erhöhtem bzw. grundlegendem Niveau handelte, nachgegangen werden. Außerdem könnten die Unterrichtserfahrungen des Lehrers länger zurück liegen und aus einer Zeit vor der Reform stammen, in der es noch Leistungs- bzw. Grundkurse gab. Es ist möglich, dass dieser Unterrichtsgegenstand in der Vergangenheit in einem Grundkurs stattfand. In einem Grundkurs sind bezüglich des Anwendungskontextes und der funktionalen Modellierung deutlich negativere Aussagen zu erwarten.
Mithilfe der Kategorie Sprachübersetzung soll der Aufbau der Unterrichtseinheit erfasst werden.
Die Kategorie Lehren und Lernen steht für die Untersuchung der Sozialformen (z. B. Einzel-, Partner- oder Gruppenarbeit) und der überwiegend eingesetzten Methoden und Medien, denn diese Kategorien können Aufschluss darüber geben, welche vielfältigen Methoden ein fachdidaktischer Ansatz bietet.

### 3.3.7 Auswertung

Die Auswertung der Interviews wird kategoriebasiert durchgeführt.

#### 1. Voraussetzungen der Lehrkräfte und Schüler

Alle befragten Lehrer geben an, Unterrichtserfahrungen bezüglich dieses Unterrichtsgegenstandes sowohl in einem Leistungs- als auch in einem Grundkurs zu haben. Einer der Befragten ist Mitautor des Hamburger Rahmenplans und besitzt mehrjährige Unterrichtserfahrungen mit dem Thema Sprachübersetzung in der gymnasialen Oberstufe. Dieser Befragte hat diesen Unterrichtsgegenstand nach eigenen Angaben im Schuljahr 2008/2009 unterrichtet. Ein weiterer Lehrer ist als Diplominformatiker Seiteneinsteiger in den Lehrerberuf und hat das Thema im Schuljahr 2012/2013 unterrichtet. Lediglich ein Lehrer gab an, dass er sich selbst als nicht sehr „fit“ im Umgang mit Themen der theoretischen Informatik, insbesondere mit

Sprachen im Informatikunterricht, fühlt. Die anderen Befragten haben mehrjährige Unterrichtserfahrungen und unterrichten diesen Unterrichtsgegenstand regelmäßig sowohl in Leistungs- als auch in Grundkursen.

**2. Anwendungsorientierter Unterricht**

Die Unterrichtsverläufe der interviewten Lehrkräfte ähneln sehr und widerspiegeln einen typischen *anwendungsorientierten* Informatikunterricht. Vier der Befragten steigen in den Unterricht problemorientiert ein, indem sie die Schüler mit digitalen Sprachübersetzern bzw. allgemeinen sprachbezogenen Informatiksystemen konfrontieren. Zum Einsatz kommen z. B. der Google-Übersetzer, eine BWInf[1]-Aufgabe, in der eine Wort-zu-Wort-Übersetzung implementiert wird, oder das Computerprogramm Eliza, das die Kommunikation zwischen einem Menschen und einem Computer über natürliche Sprache simuliert. Die Idee dieser Einstiege ist, durch die Benutzung von Wort-zu-Wort Sprachübersetzern typische Problemfelder entdecken zu lassen und in den Anwendungskontext Sprachübersetzung einzusteigen.

> „*L4: Da das ganze ja ein Unterthema ist, unter der großen Überschrift KI, steige ich gerne ein mit einem Beispiel einer Webseite, die ganz offensichtlich von einer Maschine übersetzt worden ist. Einen wunderbaren Text aus der Gegend hier, wo es also um die Undergroundcitizens von Berlin geht, das ist also die U-Bahn-Station Berliner Tor von wo aus man zu Undergroundcitizens zu Stoneford Avenue fahren kann, also U-Bahn Steinfurther Allee. Und dann ist der Text über den Öjendorfer Friedhof und ist irgendwie absolut unverständlich. Man kann es nicht nachvollziehen, worum es da geht, bis man es wörtlich versucht zurückzuübersetzen und dann kann man dazu den deutschen Text erahnen, der auch im Web verfügbar ist. […]*“ (L4,4)

Die Befragten lassen bereits hier erste kleine Wort-zu-Wort-Übersetzer in Scheme implementieren und im Plenum vorstellen.
Im weiteren Verlauf des Unterrichts stehen die Modellierung und Implementierung von Algorithmen mithilfe der funktionalen Programmiersprache Scheme, bzw. Haskell im Mittelpunkt des Unterrichts. Es lassen sich deshalb auch die von Forneck kri-

[1] BWInf: Bundeswettbewerb für Informatik

tisierten Aspekte, d. h. die Reduzierung dieses fachdidaktischen Ansatzes auf einen algorithmenorientierten Ansatz, bei allen sechs Interviewpartnern wiederfinden:

> „*L2: Haben aber das Problem das mit Scheme gescheit umzusetzen*[2]*. Die meisten kommen so weit, das sie eben aus einer doppelt verketteten Liste, wo unsere Vokabelpaare irgendwie drinstehen, diese doppelt geklammerte Liste, das sie da irgendetwas heraus lesen also dieses ganze Grundkonstrukt, indem wir unsere Vokabeln hinterlegt, unsere Bibliothek, das Lexikon, das ist für diese schon so schwierig zu verstehen, das da oft schon Verständnisprobleme des Programms auftreten. Weil wir da schon sehr stark verklammert haben, gleich eine Liste in der Liste benutzen, also Begriffspaare in den Listen haben.*“ (L2,16)

Drei der Befragten geben an, dass sie anders als im Rahmenplan gefordert, zuerst einen Programmierkurs für die funktionale Sprache Scheme durchführen:

> „*L1: Also am Anfang des Semesters habe ich erst einmal, weil ich das ganze mit Scheme mache, was gemacht, was wir gar nicht machen sollen. Ich habe nämlich einen Programmierkurs zu Scheme gemacht. Weil ich mir sage, Scheme ist eine so komplizierte Sprache, dass es für das Verstehen der Schüler, erst einmal eine Einführung in die Sprache zu geben.*“ (L1,4)

> „*L2: Ich habe erst so ein bisschen in das Thema eingeführt. Ich habe die Übungsblätter, die bei der Reihe dabei sind, genommen und überarbeitet. Dass sie diese ganze Scheme-Geschichte da richtig kennenlernen. Das heißt erst einmal die Einführung in Scheme selber, was ist überhaupt diese ganze Listenorientierung. Dann dieses ganze Thema aufgerissen.*“ (L2,4)

> „*L3: Dann habe ich aber auch schon gleich angefangen mit Scheme zu arbeiten, weil ich ahnte, diese Sprache benötigt eine lange Zeit bei den Schülern.*“ (L3,1)

Die obigen Zitate zeigen, dass die Lehrer von den Vorgaben des Rahmenplans abrücken und den Informatikunterricht als Programmierkurs gestalten. Sie gehen nicht

[2]Lehrkraft bezieht sich auf gestellte Aufgaben.

von einem Anwendungskontext aus, sondern lassen die Schüler auf Vorrat lernen. Diese Vorgehensweise kollidiert selbst mit dem Anspruch des *anwendungsorientierten fachdidaktischen Ansatzes.*

**3. Anwendungskontext Sprachübersetzung**

Die Schüler kennen Sprachübersetzungssysteme aus ihrem Alltag und halten deshalb den Anwendungskontext Sprachübersetzung für sinnvoll und motivierend:

> „*L2: Diese Grundidee finde ich, ist gut. Ist ein gutes Beispiel für diese Möglichkeiten und Grenzen, das ist ja das Oberthema. Mit der Sprache da zu arbeiten, da sind die Schüler auch immer dabei und das verstehen sie, warum man das machen will und wissen auch durch google-Übersetzer oder so etwas, dass das schwierig ist, selbst wenn der immer besser wird.*“ (L2,16)

Weitere Vorteile sehen die Befragten in der hinreichenden Komplexität, „*[...] dass es ein relativ beschränkter Bereich ist, das klar ist, was passieren soll.*“ (L1,12).
In Verbindung mit einem *anwendungsorientierten Ansatz* ergeben sich für die Befragten aber Probleme. Es überwiegen die Nachteile, die in der Komplexität bezüglich des Implementierens eines Sprachübersetzers innerhalb dieser Unterrichtseinheit liegen. Fast alle Befragten sehen das als größtes Problem. Fünf der sechs Befragten sehen es als unzureichend, dass am Ende dieser Einheit kein fertiges Produkt entwickelt wird und die Unterrichtseinheit so kein befriedigendes Ende hat. Ein Lehrer äußerst sich dabei wie folgt:

> „*L3: [...] Ich habe mit sehr viel Aufwand versucht einen Sprachübersetzer hinzubekommen. Einmal sehe ich, ich gebe einen deutschen Satz rein und kriege einen englischen Satz raus oder umgekehrt. Weil dann, wenn man einmal richtig was Schönes macht, dann haben die Schüler ein Endergebnis. Und können das auch noch einmal optimieren, verfeinern, erweitern und das machen sie ja nun auch sehr selbstständig dieses Abändern. Dieses Hinführen, so ist mein Unterrichtsstil, da gebe ich ja schon Schritte vor da ist wenig Zeit für forschendes Arbeiten. So aber, weil dieser Weg nicht zum Ende führt, kommen die Schüler niemals in*

*diese Phase, die brechen einfach ab, und es ist frustrierend für die Schüler.*“ (L3,12)

Die Konsequenz daraus ist, dass es „*schwierig [ist], damit durchgängig zu arbeiten, weil man damit einfach nie zu einem Erfolgserlebnis kommt*“ (L4,14). Die Mehrheit der Befragten entwickelt einen Wort-zu-Wort-Übersetzer ohne Einbeziehung des Wissens über Elemente formaler Sprachen und formaler Grammatiken.

**4. Funktionale Modellierung und Programmierung**

Der Rahmenplan und das A-Heft für Informatik schreiben vor, dass Lehrer hier eine funktionale Programmiersprache einsetzen müssen. Über die Jahre hat sich hier vor allem die Programmiersprache Scheme etabliert, welche von fünf der sechs Befragten eingesetzt wird. Einer der befragten Lehrer verwendet die Programmiersprache Haskell. Die Begründung für den Einsatz der funktionalen Programmierung gibt einer der Befragten wie folgt an:

„*L5: Es gibt ja auch andere Bundesländer, die nicht funktional modellieren und auch nicht logisch, also etwa Bayern, die begrenzen sich auf objektorientierter Modellierung, zustandsorientierter Modellierung und Modellierung von Abläufen, sind auch drei Stück auf erhöhtem Niveau. Und setzen daran einen anderen Schwerpunkt, machen mehr theoretische Informatik. Das hielt ich nicht für sinnvoll, weil ich denke, dass es hilfreich ist, wenn Schülerinnen und Schüler erfahren, dass man auch auf eine ganz andere Art und Weise sich eher die virtuelle Maschine vorstellen kann, eben nicht als Befehlsempfänger sondern als Funktionsauswerteapparat, das ist eine ganz anderes Paradigma, eine ganz andere Art von Paradigma. Und ich meine das man auf diese Weise eine gewisse Flexibilität des Denkens befördern kann.*“ (L5,30)

Für die anderen fünf Lehrer überwiegen die Nachteile beim Einsatz dieser funktionalen Programmiersprache:

„*L1: Negativ wird immer wieder von den Schülern bemerkt, dass die Sprache nicht verständlich ist. Scheme.*“ (L1,6).

> „*L2: Die Schüler finden die Programmiersprache Scheme schwierig. Sind bis zuletzt damit am Kämpfen, wo wie viele Klammern gesetzt werden, die ganze Syntax eben. [...] Und da ist ja dann auch die Rekursion ein großes Thema.*“ (L2,16)

Alle Befragten geben an, dass die Benutzung einer für die Schüler neuen und unbekannten funktionalen Programmiersprache neben dem Entwickeln von Algorithmen, eine zusätzliche Hürde darstellt. Das erschwert das Verständnis über formale Sprachen und Grammatiken und führt dazu, dass diese Inhalte gänzlich weggelassen werden und stattdessen reine Programmierkurse durchgeführt werden. Das betrifft z. B. die notwendige Verwendung von eingebetteten Listen in Scheme, die den Schülern sehr schwer fällt. Für das Verständnis werden Erfahrungen im Bereich der funktionalen Modellierung vorausgesetzt, über die Schüler dieser Altersgruppe nicht verfügen. Den Befragten nach sind die Schüler eher gewohnt „*in Prozessen und Befehlsabfolgen*“ (L5,6), also prozedural zu denken.

**5. Methoden und Sozialformen**

In der hier untersuchten Unterrichtseinheit kommen bei den befragten Lehrkräften die Entwicklungsumgebung DrRacket für Scheme, Smultron für Haskell, die Kommunikationsplattform CommSy sowie selbsterstellte Videos zum Einsatz. Sowohl bei DrRacket als auch bei Smultron handelt es sich um Entwicklungsumgebungen, die aus einem Texteditor und einer sogenannten REPL[3] für die Ausführung der Programme bestehen. Diese Entwicklungsumgebungen unterstützen nur die Entwicklung der jeweiligen funktionalen Sprache. CommSy dient als Plattform für den Austausch von Arbeitsmaterialien (i. d. R. Arbeitsblätter). Zu den eingesetzten Methoden, das geben alle Befragten an, findet ein lehrerzentrierter Unterricht, der durch Arbeitsblätter aus der Handreichung von Seiffert (vgl. [SK11]) unterstützt wird, statt. Einer der Befragten setzt das Flipped-Classroom Prinzip ein, bei dem sich die Schüler den neuen Stoff selbstständig auf der Basis von Lernvideos erarbeiten. Als Sozialform überwiegt die Partnerarbeit, bei der sich zwei Schüler einen Rechner teilen, um die erforderlichen Aufgaben zu bearbeiten. Diese Partnerarbeit wird immer wieder durch Lehrervorträge unterbrochen:

[3] Read-Eval-Print-Loop

> „*L1: Und der Unterricht sah meist so aus, dass ich noch einmal zusammengefasst habe, wo wir stehengeblieben waren, dass ich ein neues Problem oder eine Erweiterung gestellt habe, um die Schüler dann weiter machen zu lassen. Also ziemlich viel lehrerzentrierte Unterrichtsgespräche. Aber auch viel Arbeit in kleinen Gruppen am Computer. Wenig innovativen Unterricht.*“ (L1,4)

Ein Lehrer begründet diese Vorgehensweise mit der Komplexität des Themas.

> „*L3: Das liegt an dem Thema. Weil die nie richtig selbstständig in dieses eigene Entwickeln und eigene Modellieren kamen. Auch sonst war das Ganze schon lehrerzentrierter.*“ (L3,16)

Obwohl der *anwendungsorientierte Ansatz* des Rahmenplans die Erarbeitung der informatischen Inhalte innerhalb eines Anwendungskontextes fordert, weichen fünf der sechs Befragten davon ab, indem sie neben dem Kontext Sprachübersetzung weitere Anwendungen betrachten:

> „*L2: [...] Also ein Nummernschild wie man das als Grammatik darstellen kann, dass nachher dann nur gültige Nummernschilder bei herauskommen. [...]*“ (L2,14)

Ein weiterer Lehrer gibt an, dass seine Schüler Chatbots entwickeln. Lediglich zwei der Befragten lassen ihre Schüler Bezüge zur deutschen Grammatik herstellen und Sätze mithilfe einer formalen Grammatik analysieren. Nur einer der Befragten führt das auch innerhalb des Anwendungskontextes durch:

> „*L5:Also durch die Analyse der Schwächen eines Wort-zu-Wort-Übersetzers sind die Schüler auf die Mängel dieser Wort-zu-Wort-Übersetzung gekommen und haben erkannt, dass beispielsweise eine Übersetzung vom Englischen ins Deutsche schon beim Artikel nicht möglich ist, weil man nicht weiß, ob dieser Artikel nun männlich, weiblich oder sächlich übersetzt werden muss. Außerdem ist die Satzstellung eine andere und man muss sich deshalb mit der Grammatik der Sprache befassen, damit man sie vernünftig übersetzen kann.* “ (L5,12)

Die Abweichung von den Vorgaben des Rahmenplans und der Handreichung von Seiffert begründen vier der sechs Interviewten damit, dass die Einschränkung auf

einen einzigen Kontext für sie zu eng ist und sie mehr Zeit für Themen aus der Theorie der formalen Sprachen wünschten.

> „*L1: Wenn man einen theoretischeren Ansatz hätte und nicht jetzt von einem Spezialfall ausgehen würde, sondern wenn man jetzt sagen würde, wir machen jetzt Grammatiken, wir machen jetzt Theoretische Informatik, dann wäre das natürlich möglich, aber nicht im Sinne des Rahmenplans, das ja die Sicht des anwendungsbezogenen hat. Weil das ja immer von einem speziellen Anwendungskontext übergreifend ist. Es wäre sicher einfacher, wenn ich von oben beobachten könnte, was es alles gibt und jetzt suchen wir uns einen speziellen [Kontext] aus.*“ (L1,24)

> „*L6:Na man müsste dann etwas genauer darauf eingehen, was formale Sprachen eigentlich sind, was sie leisten können und was sie nicht leisten können. Unterschiede denn halt auch, dass sie differenzieren können.* “ (L6,42)

> „*L5: [...] Was ich nicht für sinnvoll halten würde wäre, jetzt, sowie im Studium Sprachen, die unabhängig von konkreten Sprachen sind, zum Gegenstand des Unterrichts zu machen. Also nicht die Sprache aa bb cc, sondern es müssten schon, also ich würde schon für sinnvoll halten, bei dem Kontext der natürlichen Sprachen zu bleiben, wegen des Anwendungsbezuges [...].*“ (L5,28)

Das Reduzieren auf die algorithmische Entwicklung und Implementierung mithilfe einer funktionalen Programmiersprache steht jedoch im Vordergrund.
Es stellt sich die Frage, ob es denn wirklich sinnvoll ist, die ganze Theorie nur innerhalb eines einzigen Anwendungskontextes zu erarbeiten. Schüler brauchen Übungen verschiedenster Art, sodass sie Problemstellungen auch auf andere Anwendungskontexte übertragen können.
Am Ende dieser Unterrichtseinheit sollen die Schüler einen Parser programmieren, der nach dem Prinzip eines „*General Problem Solvers*“ arbeitet. Nur zwei der Befragten erreichen mit ihren Schülern dieses Ziel.

> „*L5:Dieser Parser, der arbeitet ja nach dem Prinzip des General Problem Solvers mit entsprechender Breitensuche. [...] Da kommen die Schüler nicht von alleine drauf in diesem Unterrichtsgang. Ich stelle ihnen ein*

*Beispiel vor. Es geht nach dem Prinzip, ein fertiges Stück Software zu analysieren, was aber noch unvollständig ist. Dieses versucht zu verstehen und dann auszubauen. Das heißt also die Idee, die Idee der Breitensuche kann man vorher auch theoretisch erarbeiten, weil die Schülerinnen und Schüler ja einen Parsebaum erstellt haben. Eine Schwierigkeit besteht noch darin, dass der Parsebaum etwas anderes ist als der Suchbaum zur Erstellung des Parsebaumes. Das stellt eine Schwierigkeit in diesem Unterrichtsgang dar. Die Alternative, die ich später kennengelernt habe, die von Theo Stenzel stammt, mit einer sehr sehr funktionalen Vorgehensweise, [...], de[n] ich in einer späteren Ausbaustufe noch ergänzt habe, der arbeitet nach dem Prinzip, dass jedem Nonterminal eine Funktion entspricht. Der erfordert ein höheres Abstraktionsniveau bezogen auf funktionale Denkweise führt aber zu einem sehr viel kürzeren Parser.*“ (L5,14)

„*L6: [...] Also ich führe Grammatiken mit diesen Suchbäumen ganz normal ein und welche Sätze werden erkannt und welche Sätze werden nicht erkannt. Und im Prinzip ist der Parser für mich denn die Umsetzung dieser Grammatik in ein Programm. Und den gebe ich größtenteils vor, weil ich das zu komplex finde, dass sie das selber programmieren sollten, und sie sollen das dann praktisch nachvollziehen und sollen dann bestimmte Funktionen erweitern also zum Beispiel müssen sie dann den Parser verstehen können [...].* “ (L6,20)

Die Schüler sollen sich das theoretische Konzept eines Parsers anhand eines unvollständigen Quelltextes selbst erarbeiten. Es steht, wie die oben zitierten Aussagen von L5 und L6 auch zeigen, nicht das theoretische Konzept eines Parsers im Vordergrund, sondern wieder nur seine funktionale Implementierung. Ob die Schüler das dadurch besser verstehen können, muss im Folgenden dieser Arbeit weiter untersucht werden.
Es bleibt festzuhalten, dass alle Befragten zwar die Handreichung von Seiffert einsetzen, aber dieses Skript mit eigenen Ideen kreativ anreichern.

**6. Motivation der Schüler**

Die Aussagen der Befragten zeigen, dass die Motivation der Schüler sehr stark von der Leistungsfähigkeit und dem generellen Interesse für Themen der Informatik abhängt. Schüler, die gute funktionale Lösungen implementieren können, haben die ganze Zeit über eine hohe Motivation. Das sind jedoch i. d. R. Schüler mit einer ohnehin hohen Motivation für die Informatik.

> „*L4:Unterschiedlich. Diejenigen die das mit der funktionalen Programmierung gut hinbekommen haben, die hatten eigentlich die ganze Zeit über eine hohe Motivation, aber das sind halt auch immer welche, die grundsätzlich eine hohe Informatikmotivation haben und diejenigen die das nicht hinbekommen haben, die vielleicht auch schon mit einer geringen Grundmotivation eingestiegen sind, die waren am Ende nicht mehr so begeistert. Irgendwie kommen sie mit deren Lösung nicht zu Rande, kommen einfach nie vom Fleck. Und das ist ein Problem, das liegt irgendwie in der funktionalen Programmierung, die ganze Zeit meines Lehrerdaseins schon begleitet, dass es immer welche gibt, die irgendwie nichts auf die Reihe bringen.*“ (L4,32)

Die Motivation der Schüler wird von allen Befragten vor allem von der Fähigkeit, „funktional zu Denken“ abhängig gemacht.
Vier der sechs Befragten legen Wert auf ein Endprodukt, mit dem sich Schüler identifizieren können, damit die Motivation der Schüler nicht nachlässt. Einer der Befragten wünscht, dass der Anwendungskontext mehr in den Mittelpunkt gerückt wird und weniger das Programmieren:

> „*L2: Ein Ziel, was ich wichtig fände, dass man den Kontext mehr in den Mittelpunkt stellt, in dem man die ganze komplizierte Syntax drumherum möglichst gering hält.*“ (L2,40)

**7. Ziele des Rahmenplans**

Der zum Zeitpunkt der Befragung gültige Rahmenplan für Informatik stellt die gesamten Ziele in Form von Kompetenzen dar, d. h. sie werden nicht mehr wie in älteren Lehrplänen einem Semester direkt zugeordnet. Deshalb wurden die Befragten

im Rahmen dieser Interviews gebeten, die Kompetenzen, die aus ihrer Sicht explizit für diese Unterrichtseinheit von wesentlicher Bedeutung sind, anzukreuzen. Mit einem „+“ kennzeichnen die Befragten die Kompetenzen, die ihre Schüler im Unterricht erreicht haben. Ein „o“ bedeutet, dass die Kompetenz bei den Schülern nur teilweise ausgebildet wurde und ein „-“, dass die jeweilige Kompetenz nicht erreicht wurde. Kompetenzen, die aus Sicht der Befragten irrelevant sind, werden nicht markiert. Die Häufigkeit der Markierungen, unabhängig davon, ob es sich dabei um ein „+“, ein „o“ oder ein „-“ handelt, sagt insofern etwas über die Bedeutung der Kompetenz für diese Unterrichtseinheit aus. Sie ist aus Sicht der Befragten umso höher, je häufiger sie von allen markiert wurde. Die Ergebnisse dieser Befragung können den Tabellen A.1 bis A.6 im Anhang A entnommen werden. Betrachtet man nur die Kompetenzen, die in den Tabellen vier Mal oder häufiger markiert wurden - das sind die, die aus Sicht der sechs Befragten also mehrheitlich für diese Unterrichtseinheit als relevant angesehen werden - dann ergibt sich folgendes Bild:
Aus dem Bereich „Informatiksysteme analysieren und verstehen“
Die Schülerinnen und Schüler

- untersuchen Algorithmen und vergleichen sie hinsichtlich Effizienz und Qualität der Lösung,
- analysieren Informatiksysteme hinsichtlich der zugrundeliegenden Strukturen.

Aus dem Bereich „Informatiksysteme gestalten“
Die Schülerinnen und Schüler

- wählen bei der Gestaltung von Informatiksystemen passende Algorithmen aus,
- implementieren Modelle sowohl mit Hilfe grafischer Entwicklungsumgebungen als auch mit einer höheren Programmiersprache,
- identifizieren automatisierbare Sachverhalte der realen Welt und modellieren sie mit mindestens drei unterschiedlichen Modellierungsansätzen, und zwar mit objektorientierter Modellierung und jeweils mindestens einem weiteren Modellierungsansatz,
- implementieren einen Interpreter für eine selbst entwickelte einfache formale Sprache.

Aus dem Bereich „Darstellen und Interpretieren“
Die Schülerinnen und Schüler

- beschreiben Modelle und Algorithmen sowohl grafisch als auch verbal,
- unterscheiden natürliche von formalen Sprachen,
- differenzieren formale Sprachen hinsichtlich ihrer Interpretierbarkeit.

Aus dem Bereich „Begründen und Bewerten“
Die Schülerinnen und Schüler

- bewerten die prinzipielle und praktische Realisierbarkeit von Informatiksystemen, ohne dass dabei die Mathematik im Zentrum steht,
- bewerten die Auswirkungen von Informatiksystemen auf die betroffenen Menschen.

Aus dem Bereich „Kommunizieren und Kooperieren“
Die Schülerinnen und Schüler

- nutzen Informatiksysteme zur Kooperation und reflektieren die Kommunikationsprozesse,
- beschreiben Sachverhalte mithilfe von Texten, Bildern und Diagrammen,
- verwenden die informatische Fachsprache angemessen,
- dokumentieren Lernergebnisse, Arbeitsabläufe und Arbeitsergebnisse,
- präsentieren wesentliche Ergebnisse adressatengerecht.

Aus dem Bereich „Möglichkeiten und Grenzen von Informatiksystemen“
Die Schülerinnen und Schüler

- bewerten Verfahren hinsichtlich Effizienz und Bedeutung aufgrund der Einsatzmöglichkeiten,
- kennen prinzipielle und praktische Grenzen der Berechenbarkeit,
- reflektieren über Möglichkeiten und Grenzen von Informatiksystemen, z. B. durch eine fachkundige Diskussion der Frage „Welche Teile der geistigen Tätigkeiten des Menschen können Maschinen übernehmen?“

Es ist nicht ungewöhnlich, dass diese Kompetenzen nicht mehr den konkreten Semestern zugeordnet werden, denn um die eingangs erwähnte Profilbildung zu ermöglichen (siehe Kapitel 1), wurden die Kompetenzen in den Rahmenplänen absichtlich sehr allgemein formuliert. Des Weiteren legt der Rahmenplan im Vergleich zum Lehrplan nur fest, über welche Kompetenzen ein Schüler am Ende der gymnasialen Oberstufe verfügen muss. Zu welchem Zeitpunkt eine Kompetenz ausgebildet

wird, kann ein Lehrer bei seiner Unterrichtsplanung selbst entscheiden. Deswegen ist es auch nicht verwunderlich, dass z. B. gerade die Kompetenzbereiche wie „Begründen und Bewerten“ und „Kommunizieren und Kooperieren“ auch von den Befragten mit einem „+ “ markiert wurden.
Damit aber die Ziele für diese Unterrichtseinheit genauer untersucht werden können, hilft der etwas ältere Lehrplan aus dem Jahr 2004, weil die Unterrichtseinheit und auch die Handreichung von Seiffert seit dieser Zeit existieren und interessanterweise das etwas später erschienene A-Heft diese Kompetenzen mit etwas modifizierten Formulierungen wieder enthielt, weil sich die allgemeinen Formulierungen für ein Zentralabitur Informatik nicht eigneten. Im Lehrplan aus dem Jahr 2004 sind folgende Kompetenzen aufgeführt (vgl. [BFO+04], S.27, S.28):
Die Schülerinnen und Schüler:

K.1. vergleichen natürliche und formale Sprachen

K.2. interpretieren unterschiedliche Darstellungen von Grammatiken

K.3. klassifizieren Automatenmodelle, Sprachen und Grammatiken

K.4. unterscheiden Rekursion und Iteration

K.5. analysieren Sätze einer Sprache, entwickeln die zu ihrer Beschreibung notwendigen Grammatikelemente und stellen diese dar,

K.6. setzen Rekursion fachgerecht ein

Im A-Heft finden sich diese Formulierungen wieder. Zusätzlich werden auf Seite 111 des A-Heftes noch (vgl. [Ros14]) die folgenden Kompetenzen aufgeführt:

A.1. beschreiben grundsätzliche Schwierigkeiten maschineller Sprachverarbeitung,

A.2. erstellen einen Parsebaum zu einem gegebenen Satz und einer gegebenen Grammatik,

A.3. vergleichen unterschiedliche Vorgehensweisen zur Realisierung eines Parsers (Abbildung der Grammatikproduktionen als Liste von Listen sowie Tiefen- oder Breitensuche, Funktionen zur Auflösung von Nonterminalen)

Vergleicht man nun diese zwei Pläne, die zur Zeit der Befragung vorhanden waren und berücksichtigt noch das A-Heft, dann kristallisieren sich drei Kompetenzen

heraus, die zum Zeitpunkt der Befragung definiert waren, dieser Unterrichtseinheit direkt zugeordnet werden können und in dieser Unterrichtseinheit direkt ausgebildet werden müssen. Das sind die Kompetenzen:
Die Schülerinnen und Schüler:

- unterscheiden natürliche von formalen Sprachen,
- differenzieren formale Sprachen hinsichtlich ihrer Interpretierbarkeit,
- implementieren einen Interpreter für eine selbst entwickelte einfache formale Sprache.

Die Hälfte der sechs Befragten geben an, die Kompetenz „[...] unterscheiden natürliche von formalen Sprachen“ nur teilweise ausgebildet zu haben, während die Schüler der anderen Befragten diese Kompetenz erreichten. Die Kompetenz „[...] differenzieren formale Sprachen hinsichtlich ihrer Interpretierbarkeit“ wird dagegen von nur einem der sechs Befragten, das ist L5, als erreicht angegeben. L5 begründet das aber nur theoretisch, d. h. der Befragte benennt hier keine exemplarische Situation, bei der nachvollzogen werden kann, inwiefern ein Schüler zu dieser Kompetenz kommt.

> „*L5: [...] Differenzieren formale Sprachen hinsichtlich ihrer Interpretierbarkeit, dass ist hier genau dieses Problem, also eines der Probleme warum es so schwierig ist einen guten Übersetzer zu machen, weil die natürliche Sprache nicht kontextfrei ist. Das erkennen sie aber nur auf erhöhtem Niveau. Diese Kompetenz, würde ich sagen, erwerben sie nur auf erhöhtem Niveau.*“ (L5,24)

Vier der sechs Befragten bilden diese Kompetenz nicht aus.

> „*L4: Differenzieren formale Sprachen hinsichtlich ihrer Interpretierbarkeit - erfordert ja, dass man mehrere formale Sprachen erst mal anguckt und dann daran überlegt, wie es um deren Interpretierbarkeit steht. Mache ich glaube ich, eher nicht.*“ (L4,24)

L2 gibt an, sich selbst nicht sehr kompetent im Umgang mit formalen Sprachen und formalen Grammatiken zu fühlen und beschränkt sich auf Übungsaufgaben aus einem Lehrbuch.

> „*L2: Oh, da bin ich jetzt selber nicht fit, muss ich gestehen, welche Sprachklassen es gibt. [...] Ich hab mich sehr viel an dem Buch Informatik 5, was die in Bayern verwenden von Klett, orientiert.*“ (L2,14)

Die Erreichung der Kompetenz „[...] implementieren einen Interpreter für eine selbst entwickelte einfache formale Sprache “ wird nur von zwei der sechs Befragten, das sind L1 und L6, bejaht. Sie halten sich dabei streng an die Handreichung von Seiffert, in der ein Parser nur funktional implementiert wird. L2 gibt explizit an, diese Kompetenz nicht ausbilden zu können. L3 schafft das nur teilweise. L4 lässt seine Schüler anstelle des Parsers Chatbots implementieren, weil „das sehr lange dauern würde und weil [er] das für nicht sehr motivierend [hält]“ (L4,10).

> „*L4: Weil man wenn man die ganze Zeit an diesem Übersetzer arbeitet nur bis zum Akzeptor kommt und das ist frustrierend. Weil man feststellt, ok, wir arbeiten hier an einem Übersetzer aber der übersetzt nie. Weiter kommt man halt da nicht. Deshalb hab ich das immer rausgelassen und bin aus dem Skript ausgestiegen, bevor es soweit war, bei Wortübersetzern aufgehört und andere Sachen gemacht und die Theorie als Blog.*“ (L4,10)

L5 ignoriert diese Kompetenz gänzlich.
Es bleibt festzuhalten, dass nur mit Einschränkungen die Kompetenz „[...] unterscheiden natürliche von formalen Sprachen“ bei den Befragten erreicht wird. Die anderen beiden Kompetenzen werden nicht erreicht.
Betrachtet man nun wieder die durch das A-Heft verbindlichen Kompetenzen, so kann man zusätzlich daraus schlussfolgern, dass die Schüler der Befragten auch nicht die Kompetenzen K.2., K.3., K.5. sowie A.2. und A.3. erreichen. D. h. von den neun verbindlichen Kompetenzen werden fünf nicht erreicht. Lediglich K.1. und A.1. sind für die Schüler erreichbar.
Ob die Schüler der Hansestadt in dieser Unterrichtseinheit darüber hinaus auch die Kompetenzen K.4. und K.6. erreichen können, müsste im weiteren Verlauf dieser Arbeit untersucht werden.

## 3.4 Zusammenfassung

Der Informatikunterricht in Hamburg ist *anwendungsorientiert* und die obige Untersuchung zeigt, dass die Kritik von Forneck (vgl. [For90], S. 36) bezüglich der Gefahr

eines Rückfalls in einen *algorithmenorientiert*en Informatikunterricht zutrifft. Im Semester über Sprachverarbeitung kommt zur starken Betonung der Algorithmik noch die Besonderheit hinzu, dass zwingend funktional programmiert werden muss. Diese Überbetonung des funktionalen Programmierens führt dazu, dass andere Dimensionen des Anwendungskontextes Sprachübersetzung nur ansatzweise oder gar nicht berücksichtigt werden können. Das betrifft insbesondere ein tiefes Verständnis für formale Sprachen und Grammatiken im Allgemeinen und für die Arbeitsweise eines Parsers im Besondern. Im Extremfall finden Programmierkurse statt, die unabhängig vom *anwendungsorientierten* Ansatz generell deplatziert sind. Die Programmierung selbst wird mithilfe einer Entwicklungsumgebung durchgeführt, die nichts mit dem Anwendungskontext gemein hat. Es können damit keine Problemstellungen visualisiert werden. Ein direktes Arbeiten an den Problemstellungen, die der Anwendungskontext mit sich bringt, ist unmöglich.
Der Anwendungskontext Sprachübersetzung an sich wird insgesamt positiv bewertet. Die Beschäftigung mit Sprache wird von allen Befragten und deren Schülern als zeitgemäß empfunden. Kritisiert wird hier überwiegend, dass am Ende der Einheit kein Produkt steht und damit einhergehend die anfangs hohe Motivation der Schüler nicht gehalten werden kann.
Die Folge davon ist, dass wichtige im Rahmenplan und im A-Heft geforderte Kompetenzen unzureichend oder gar nicht ausgebildet werden.

# Kapitel 4

# Mensch-Maschine-Kommunikation mit gesprochener Sprache im IU

Bei der Mensch-Maschine-Kommunikation „wirkt der Mensch mit einer Maschine mit dem Ziel zusammen, eine selbstgewählte oder vorgegebene Aufgabe zu lösen. Mit dem Sammelbegriff Maschine werden hier technische Systeme der unterschiedlichsten Art bezeichnet, angefangen von Werk –, Fahr –, und Denk –, bis hin zu Spielzeugen“ (vgl. [Gei90], S. 9). Für die Kommunikation kann der Mensch unterschiedliche Werkzeuge benutzen. Ohne Anspruch auf Vollständigkeit können hierfür Tastaturen, Mäuse, Berühreingabegeräte, Griffel, Graphik-Tablets, Steuerknüppel, Rollkugeln und die gesprochene Sprache zum Einsatz kommen. Letztere kann zwischen der Eingabe von fest vereinbarten Wörtern, sogenannten Sprachschaltern (vgl. [Gei90], S. 44), und natürlich gesprochener Sprache unterschieden werden.
Informatiksysteme, bei denen der Computer die Rolle eines Kommunikationspartners einnimmt und mithilfe gesprochener Sprache bedient werden kann, bezeichnet Herzeg als Dialogsysteme (vgl. [Her94]). Sprachdialogsysteme sind spezielle Informatiksysteme, über die z. B. ein Anrufer mithilfe eines Telefons einen teil- oder vollautomatisierten natürlichsprachlichen Dialog führen kann.
Die folgenden Kapitel stellen die Grundlagen der Spracherkennung, des Sprachverstehens, der Sprachdialogsysteme, der Formalen Sprachen und Grammatiken sowie Elemente der Automatentheorie dar. Diese Arbeit bewegt sich hierbei im Grenzgebiet zwischen der Computerlinguistik und der Theoretischen Informatik. In Folge dessen wird insbesondere der Begriff der Formalen Sprachen zum einen im computerlinguistischen Sinne und zum anderen im Sinne der theoretischen Informatik

verwendet. Formale Sprachen sind eine Menge von Sätzen (computerlinguistische Sichtweise) bzw. eine Menge von Wörtern (Sichtweise der theoretische Informatik), die ihrerseits aus Zeichen bzw. Symbolen (Sichtweise der theoretischen Informatik) bzw. Wörtern im computerlinguistischen Sinne bestehen. Formale Sprachen können mit einer endlichen Menge von Regeln (formale Grammatiken) beschrieben werden. Für die Vermittlung von zugrunde liegenden Konzepten in Form von Definitionen benutzt diese Arbeit die Sichtweise der theoretischen Informatik, während praktische Beispiele, insbesondere bei der Verwendung der in diesem Kapitel vorgestellten und für die Schule konzipierten Software wird überwiegend die computerlinguistische Sichtweise benutzt.

## 4.1 Grundlagen

Sprachdialogsysteme bestehen aus fünf Komponenten (vgl. [Hau00]). Das sind:

- Spracherkennung
- Sprachverstehen
- Dialogsteuerung
- Sprachgenerierung
- Sprachsynthese

Die Abbildung 4.1 zeigt das Zusammenwirken dieser Komponenten.

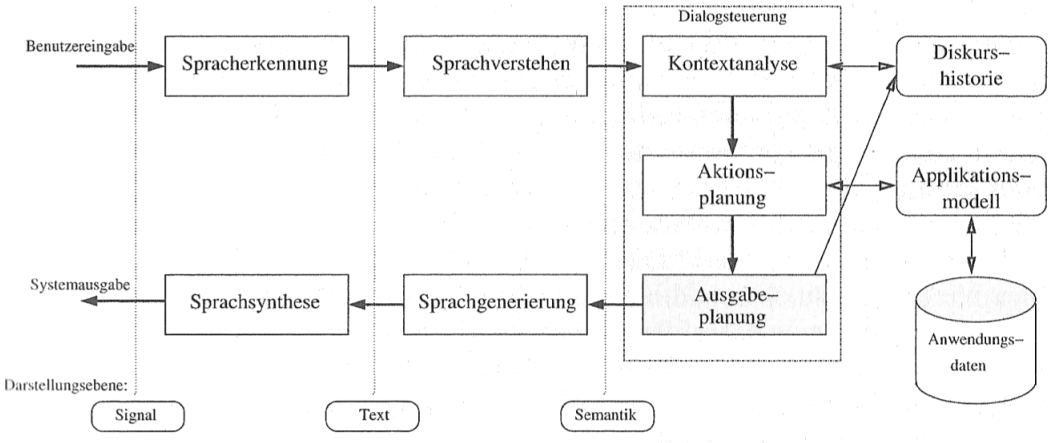



Abbildung 4.1: **Komponenten eines Sprachdialogsystems**[Hau00]

### 4.1.1 Spracherkennung und -verstehen

Bei der Spracherkennung werden akustische Signale in Text umgewandelt. Dabei wandelt das Mikrofon, das zur Aufnahme des gesprochenen Textes eingesetzt wird, die analogen Schallwellen in elektrische Signale um, die mithilfe eines Analog-Digitalwandlers in einen binären Datenstrom übersetzt werden (vgl. [McT04], S. 83). Solche Datenströme lassen sich in einem Audioverarbeitungsprogramm (z. B. Audacity) sichtbar machen (vgl. Abbildung 4.2).

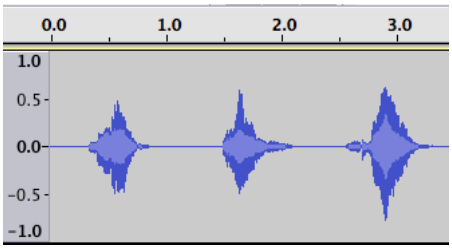



Abbildung 4.2: Aufgezeichnete Sprache in Audacity

Die Abbildung 4.2 zeigt die Aufnahme der drei hintereinander gesprochenen Wörter „null“, „eins“ und „zwei“. Die x-Achse des Diagramms stellt die Zeit und die y-Achse die Amplitude der aufgezeichneten akustischen Schwingungen (Schallwellen) dar. Je größer die Amplitude, desto lauter wurde das Wort gesprochen. Die Aufnahme der in Abbildung 4.3 gesprochenen Wörter „drei“, „vier“ und „fünf“ ähneln sich.

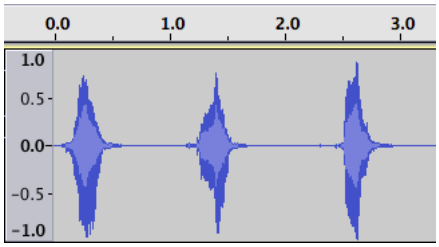



Abbildung 4.3: Aufgezeichnete Sprache in Audacity

Eine Erkennung der Wörter ist bei dieser Darstellung nicht möglich, denn die im Signal enthaltenen Frequenzen lassen sich bei dieser Darstellung nicht erkennen. Für die Analyse des Gesprochenen werden deshalb, wie bei der Basilarmembran des menschlichen Gehörs (vgl. [Sig16]), sogenannte Spektral-Transformationen

(z. B. die Fourier-Transformation) durchgeführt. Diese Spektral-Transformationen überführen ein zeitdiskretes endliches Signal in ein diskretes periodisches Frequenzspektrum. Hiermit lassen sich die Zusammensetzung der verschiedenen Frequenzen des Eingangssignals untersuchen. Die folgenden zwei Abbildungen zeigen das Frequenzspektrum der beiden obigen Beispielaufnahmen.

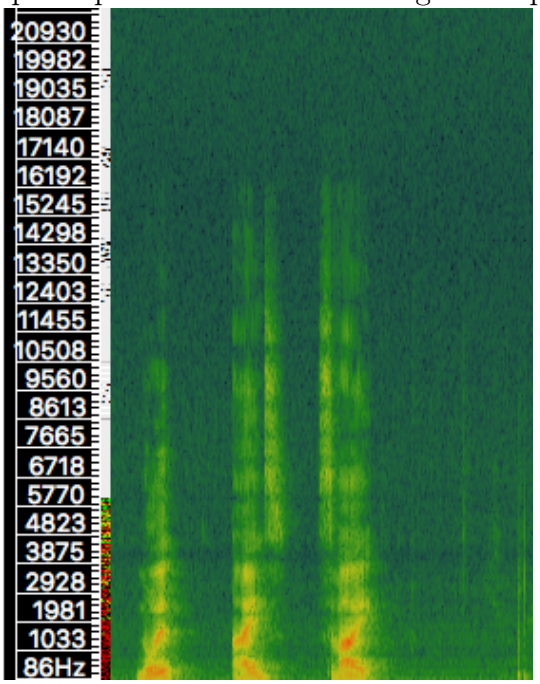



Abbildung 4.4: Frequenzspektrum der ersten Aufnahme

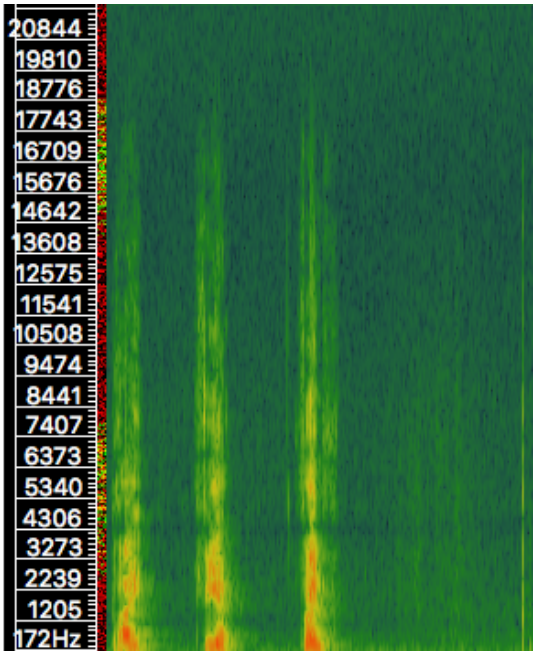



Abbildung 4.5: Frequenzspektrum der zweiten Aufnahme

Die Abbildungen 4.4 und 4.5 zeigen sogenannte Spektrogramme. Spektrogramme stellen die Schallenergie[1] in einem Frequenzbereich dar. Je rötlicher ein Bereich im Frequenzspektrum eingefärbt ist, desto größer ist die Schallenergie in diesem Bereich. Bei der Abbildung 4.4 ist eine große Schallenergie bei 86 Hz und 1033 Hz zu beobachten. Im Vergleich dazu liegt eine große Schallenergie in Abbildung 4.5 bei 172 Hz und zwischen 2239 Hz und 3373 Hz. Die beiden Darstellungen der Frequenzspektren weisen gegenüber dem Zeit-Amplituden-Diagramm deutlichere Unterschiede zwischen beiden Aufnahmen aus.

Bei der Spracherkennung werden diese Spektral-Transformationen für sehr kleine Zeitabschnitte des Signals, üblicherweise alle 25 ms, durchgeführt, damit die Erkennung von Phonemen[2] möglich wird (vgl. [McT04], S. 83). *„Ein Phonem trägt*

[1] Die in einem Schallfeld (einem Medium in dem sich Schallwellen ausbreiten) enthaltene Energie.

[2] die kleinste bedeutungsunterscheidende, nicht selbst bedeutungstragende sprachliche Einheit

*selbst keine Bedeutung, ändert jedoch die Bedeutung eines Wortes, wenn man es durch ein anderes Phonem ersetzt, z. B. ‚**k**ühlen‘ vs. ‚**f**ühlen‘, ‚Ro**b**e‘ (langes o) vs. ‚Ro**bb**e‘ (kurzes o), ‚rasten‘ (kurzes a) vs. ‚rasten‘ (langes a)“* (vgl. [Hil08], S. 15).
Für die Erkennung von ganzen Wörtern reicht diese Analyse nicht aus, denn Laute lassen sich schwer trennen und beeinflussen sich gegenseitig. Beispielsweise kann man bei den Lautfolgen „am“, „um“ und „an“ den vokalen vom nasalen Anteil nicht einfach trennen. Abgesehen von Nebengeräuschen gibt es bei der gesprochenen Sprache anders als in der Orthographie keine Pausen bzw. es können Pausen auch in einem gesprochenen Wort auftreten. Annotierte Spektrogramme und der Einsatz statistischer Verfahren helfen bei der Lösung dieses Problems. Statistische Verfahren, wie z. B. das Hidden-Markov-Modell, kurz HMM, untersuchen die Korrelation zwischen einem Phonem und den damit auftretenden Merkmalen, zu denen die zeitliche Varianz und der Kontext gehören. Die Phoneme werden über die mit ihm am häufigsten auftretenden Merkmalskombinationen erkannt.
Freie Spracherkennungssoftware, z. B. Sphinx für Java, enthalten sogenannte Aussprachelexika, in denen Phonemfolgen und das dazugehörige Wort gespeichert sind (vgl. [V1416]). Das folgende Listing zeigt exemplarisch die Phonemfolge des Wortes „null“.

Listing 4.1: Ausschnitt der Datei voxforge_de_sphinx.dic aus dem VoxForge-Paket

```
1 null    n uu l
```

### 4.1.2 Dialogsteuerung

Die Dialogsteuerung ist die Hauptkomponente in einem Sprachdialogsystem. Sie steuert aktiv die Erkennung gesprochener Sprache, indem sie dem Erkenner die zu erkennenden Wörter bzw. Sätze z. B. mithilfe einer formalen Grammatik vorschreibt. Des Weiteren verarbeitet sie die Texte der Spracherkennungskomponente, interpretiert diese im Kontext des Sprachdialoges, gibt die Ergebnisse über die Sprachausgabekomponente aus und ermöglicht so die Kommunikation zwischen dem Sprachdialogsystem und seinem Benutzer.
Hausser unterscheidet zwei Typen der Dialogsteuerung: den gerichteten Dialog und die sogenannte gemischte Initiative (vgl. [Hau00]). Bei einem gerichteten Dialog liegt die Initiative beim System, d. h. das System stellt gezielte Fragen und erwartet als Eingabe je ein Wort, während die gemischte Initiative (mixed-initiative) dem

Benutzer mehr Freiheiten durch das Sprechen ganzer Sätze erlaubt. Fehlen in den gesprochenen Sätzen des Benutzers dennoch wichtige Daten für den Dialog, können die Systeme mit gemischter Initiative Rückfragen an den Benutzer stellen.

### 4.1.3 Sprachgenerierung und -synthese

Die Sprachgenerierungs- und Spachsynthesekomponente arbeitet im Vergleich zur Sprachverstehens- und Spracherkennungskomponente in umgekehrter Reihenfolge, d. h. Sprache wird über ein künstlich erzeugtes Frequenzspektrum oder durch die Verkettung von Signalabschnitten in ein zeitdiskretes Signal überführt und über die Lautsprecher ausgegeben. Dabei wird zwischen der Vollsynthese und der reproduzierenden Synthese unterschieden.
Bei der Vollsynthese werden die Laute vollständig künstlich und nur vom System selbst erzeugt. Für die Erzeugung der Sprache wird auf segmentübergreifende Informationen, das sind Grundfrequenzen, Rhythmen und Lautstärke, zurückgegriffen. Bei der reproduzierenden Synthese werden hingegen zuvor aufgezeichnete menschliche Wortfragmente aneinandergereiht und ausgegeben. Dieses Verfahren sorgt für ein qualitativ hochwertiges Signal, da es sich letztendlich um Wörter handelt, die zuvor von menschlichen Sprechern aufgezeichnet wurden.

### 4.1.4 Modellieren von Sprachdialogen

Sprachdialoge können verschieden modelliert werden. McTear benennt drei Arten (vgl. [McT04]):

- Zustandsbasierte Dialogsteuerung (Finite State based)
- Formularbasierte Dialogsteuerung (Frame based)
- Agentenbasierte Dialogsteuerung (Agent-based)

#### Zustandsbasierte Dialogsteuerung (Finite State based)

Bei der zustandsbasierten Dialogsteuerung werden Sprachdialoge als deterministische endliche Automaten (kurz DFA) modelliert. Die folgenden drei Definitionen benutzen den Begriff des Wortes dabei im Sinne der theoretischen Informatik (vgl. S. 49).

**Definition 1:**
Ein deterministischer endlicher Automat (DFA) ist ein 5-Tupel $A = (Z, \Sigma, \delta, z_0, Z_{end})$ mit

- der endlichen Menge von Zuständen $Z$,
- dem endlichen Alphabet $\Sigma$ von Eingabesymbolen,
- der Überführungsfunktion $\delta$:$Z \times \Sigma \to Z$,
- der Startzustand $z_0 \in Z$,
- der Menge der Endzustände $Z_{end} \subset Z$.

(vgl. [Sch08], S. 19).

Der Automat liest beispielsweise das Wort $w = a_1 a_2 \cdots a_n$ von links nach rechts und durchläuft nach jedem Lesen eines Buchstabens die Zustände $z_0, z_1, \cdots$ bis $z_n$. Dabei ist $z_0$ der Startzustand und $z_i = \delta(a_i, z_{i-1})$ ein Zustand, in dem sich der Automat nach dem Lesen eines Zeichens befindet. Ein Wort wird vom DFA akzeptiert, wenn nach dem Lesen des Wortes ein Zustand $z_n$ erreicht wird und dies ein Endzustand ist.

Die Arbeitsweise des DFA zeigt die Definition 2.
**Definition 2:** Sei $A = (Z, \Sigma, \delta, z_0, Z_{end})$ ein DFA. Die erweiterte Überführungsfunktion $\hat{\delta} : Z \times \Sigma^* \to Z$ wird für alle $z \in Z$, $x \in \Sigma$ und $w \in \Sigma^*$ rekursiv definiert durch

$$\begin{aligned} \hat{\delta}(z, \lambda) &:= z \\ \hat{\delta}(z, wx) &:= \delta(\hat{\delta}(z, w), x) \end{aligned}$$

**Definition 3:** Es sei $A = (Z, \Sigma, \delta, z_0, Z_{end})$ ein DFA.
Dann ist $L(A) = \{w \in \Sigma^* \mid \hat{\delta}(w, z_0) \in Z_{end}\}$ die Menge aller von $A$ akzeptierten Wörter.

Endliche Automaten können auch grafisch in Form sogenannter Zustandsübergangsdiagramme dargestellt werden. Dafür werden die Zustände durch Kreise bzw. Ellipsen dargestellt, die Überführungen zwischen den Zuständen mit Pfeilen gekennzeichnet und die Endzustände doppelt eingekreist.
Die Abbildung 4.6 zeigt exemplarisch einen Ausschnitt aus einer Darstellung eines Sprachdialoges, der mithilfe eines DFA modelliert wurde.

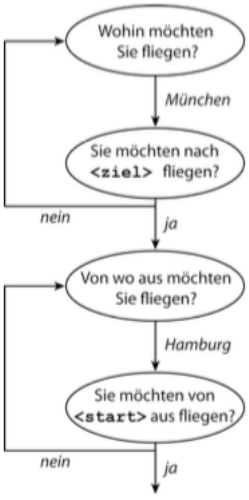



Abbildung 4.6: **Zustandsbasiertes Modell eines Sprachdialoges** (vgl. [Hil08], S. 21)

Nach Hausser entspricht ein solcher Sprachdialog einem gerichteten Dialog, d. h. die Initiative liegt beim System. Die Reihenfolge ist nicht flexibel. Es wird Zustand für Zustand abgearbeitet. Solche Sprachdialoge eignen sich beispielsweise für Finanztransaktionen beziehungsweise Wettervorhersagen (vgl. [Hil08], S. 24).

### Formularbasierte Dialogsteuerung (Frame based)

Formularbasierte Dialoge sind gegenüber zustandsbasierten Dialogen wesentlich flexibler. Solche Dialoge können mit einer offenen Frage beginnen. Beispielsweise könnte die Frage - *Wie sehen ihre Reisepläne aus?* - mit

- *Ich möchte von Hamburg nach München fliegen*,
- *Ich will von Berlin nach Hamburg fliegen* oder
- *Nächste Woche um 18.00 Uhr möchte ich von Hamburg nach Berlin fliegen*

beantwortet werden. Unabhängig davon, wie ausführlich der Benutzer dieses Sprachdialoges antwortet, wartet das System darauf, dass alle definierten Variablen belegt werden. In diesem Beispiel könnten das die Variablen *datum*, *uhrzeit*, *start* und *ziel* sein. Die Abbildung 4.8 zeigt ein solches Formular.

**Formular** `fluganfrage`

| Variable | Wert |
|---|---|
| `start` | Hamburg |
| `ziel` | München |
| `datum` | `undefiniert` |
| `uhrzeit` | `undefiniert` |

Abbildung 4.7: **Formularbasiertes Modell eines Sprachdialoges** (vgl. [Hil08], S. 25)

Werden in einem solchen Sprachdialog bei der ersten Antwort des Benutzers nicht alle Variablen belegt, weil beispielsweise der Benutzer nur **start** und **ziel** angibt, können die restlichen Variablen der Reihe nach abgefragt werden. Damit gehört diese Dialogsteuerung nach Hausser zu den gemischten Initiativen, weil je nach Zustand des Sprachdialoges die Initiative mal beim System und mal beim Benutzer liegt.

#### Agentenbasierte Dialogsteuerung (Agent-based)

Bei agentenbasierter Dialogsteuerung kommen Methoden der Künstlichen Intelligenz (KI) mit dem Ziel, menschliche Dialoge zu imitieren, zum Einsatz. Das System versucht im Kommunikationsverlauf ein Bild von sich selbst sowie vom Benutzer aufzubauen. Für negativ beantwortete Benutzeranfragen muss das System sinnvolle Alternativen vorschlagen können. Der folgende Dialog soll das Verhalten veranschaulichen (vgl. [McT04], S. 117):

```
1 User: I?m looking for a job in the Calais area.
2       Are there any servers?
3 System: No, there aren?t any employment servers for Calais.
4         However, there is an employment server for Pas-de-Calais
5         and an employment server for Lille.
6         Are you interested in one of these?
```

### 4.1.5 VoiceXML

VoiceXML ist eine Beschreibungssprache (Voice Extensible Markup Language, kurz VXML) mit der Sprachdialogsysteme implementiert werden können. Sie ist eine Ausprägung von XML[3].

[3]Extensible Markup Language

Der VoiceXML Standard wird von der W3C[4] Voicebrowser Group betreut. Deren Dokumente beschreiben die:

- Ausgabe von Audiodateien und synthetisierter Sprache
- Erkennung von Wörtern und Sätzen
- Erkennung von DTMF
- Aufnahme der gesprochenen Eingabe
- Kontrolle des Dialogflusses
- Telefoniekontrolle (Weiterleiten und Auflegen)

(vgl. [W3C16a])
Während beim Benutzer installierte Webbrowser zur Interpretation von HTML Dokumenten benutzt werden, kommen bei VoiceXML sogenannte Voicebrowser zum Einsatz, die analog eines Webbrowsers VoiceXML Dokumente interpretieren. Diese Voicebrowser sind jedoch nicht bei einem Anrufenden auf dem Telefon installiert, sondern werden von externen Anbietern zur Verfügung gestellt. Daraus ergibt sich der Nachteil, dass die Spracherkennung, die für die Voicebrowser benötigt werden, nicht auf spezielle Sprecher trainiert ist und somit Fehler in der Erkennung möglich sind. VoiceXML ist derzeit weit verbreitet und liegt aktuell in der Version 2.1 vor.

### Aufbau eines VoiceXML Dokumentes

VoiceXML Dokumente beginnen mit einem <vxml>-Tag. Da VoiceXML auf XML basiert, müssen alle „geöffneten“ Tags, anders als bei HTML, unbedingt mit einem „schließenden“ Tag geschlossen werden (hier </vxml>), da es sonst bei der Interpretation zu Fehlern kommt. Des Weiteren benötigen Voicebrowser analog zu einer index.html - Datei, die für Webbrowser zuerst ausgeliefert wird, ein Root-Dokument. Von diesem Root-Dokument kann auf weitere VoiceXML-Dokumente referenziert werden.

### Dialogelemente

Die Elemente <menu> und <form> bilden die Grundlage jeder VoiceXML Anwendung. Beide Elemente befinden sich stets unterhalb des <vxml>-Tags. Mit dem <menu>- Tag können Menüs implementiert werden. Sie erlauben dem Benutzer die

[4]World Wide Web Consortium

Auswahl aus einer Liste mehrerer Auswahlmöglichkeiten. Nach erfolgreicher Erkennung eines Benutzerwunsches können weitere Dialogelemente angesprungen werden. Bei dem <form>-Tag handelt es sich um ein Formular, mit dessen Hilfe mehrere Daten des Benutzers entgegengenommen werden und innerhalb des Formulars ausgewertet werden können.

### Variablen

Variablen zur Erfassung und Verarbeitung von Daten werden mit dem <var>-Tag implementiert. Das folgende Beispiel zeigt die Deklaration zweier Variablen und ein Rechenbeispiel.

```
1  <var name="a" />
2  <assign name="a" expr="10"/>
3  <var name="b" expr="20"/>
4  <var name="result" />
5  <assign name="result" expr="a+b"/>
```

Gemeinsam mit dem *name*-Attribut wird im Beispiel eine Variable a deklariert. Dieser Variable a wird mithilfe des <assign>-Tags in Verbindung mit dem *name*- und dem *expr*-Attribut der Wert 10 zugewiesen. Alternativ kann bereits bei der Deklaration einer Variable ein Wert zugewiesen werden (siehe Variable b). Die letzte Zeile des Beispiels zeigt, wie mithilfe von Variablen gerechnet werden kann.

### Das Menü

Ein Menü wird mit einem <menu>-Tag begonnen. Damit Menüs auch gezielt angesprungen werden können, benötigen diese eine eindeutige Identifizierung. Die Identifizierung eines Menüs wird durch das Attribut *id* (Kurzform von Identity) möglich. I. d. R. folgt ein <prompt>-Tag, mit dessen Hilfe ein Text über die Sprachausgabe ausgegeben werden kann. Es folgen <choice>-Tags, die eine Liste an Auswahlmöglichkeiten zur Verfügung stellen. Je nachdem für welche Auswahlmöglichkeit sich ein Benutzer entscheidet, sorgt das *next*-Attribut des <choice>-Tags für einen Sprung zu einem nächsten Dialogelement. Das folgende Beispiel zeigt ein Menü einer VoiceXML-Anwendung.

```
1  <menu id="mainMenu">
2    <prompt> Was kann ich für Dich tun? <enumerate/> </prompt>
3    <choice next="#information"> Information </choice>
4    <choice next="#beratung"> Beratung </choice>
5    <choice next="#exit"> Ende </choice>
6  </menu>
```

Der Benutzer wird im Beispiel zuerst gefragt, was für ihn getan werden könne. Das <enumerate>-Tag sorgt dafür, dass die Liste der Auswahlmöglichkeiten vorgelesen wird. Es sollte insbesondere bei sehr vielen Auswahlmöglichkeiten weggelassen werden. Entscheidet sich der Benutzer für *Information*, wird ein Dialogelement *information* angesprungen. Das kann ein weiteres Menü oder ein Formular sein.

### Das Formular und die Formularelemente

Mit dem <form>-Tag werden Formulare für die Erfassung von Benutzerdaten eingeleitet. Es wird zwischen *Steuerungs-* und *Eingabeelementen* unterschieden. Mit *Eingabeelementen* werden DTMF- bzw. Spracheingaben erfasst. Die *Steuerungselemente* bieten Möglichkeiten für Sprachausgaben sowie Variablenzuweisungen.
Zur Gruppe der *Steuerungselemente* gehört das <block>-Tag. Es kann für Dialogabschnitte genutzt werden, bei denen lediglich Informationen ausgegeben werden müssen. Der Dialogabschnitt „Information“ des obigen Beispiels könnte folgende Gestalt haben:

```
1    <form id ="information">
2      <block>
3        <prompt>Hier wird Information ausgegeben.</prompt>
4      </block>
5    </form>
```

Benutzereingaben können mit einem <field>-Tag erfasst werden. Die <field>-Tags sind spezielle Variablen, sogenannte Feldvariablen. Sie erfassen Spracheingaben. Dadurch werden Vergleiche, Rechenoperationen etc. möglich. Das folgende Beispiel zeigt ein Formular, welches Daten eines Benutzers mithilfe von <field>-Tags erfasst.

```
1  <form id="beratung">
2    <field name="fach">
3      <prompt>
4        Zu welchem Fach möchtest Du beraten werden? <enumerate/>
5      </prompt>
6      <option>Englisch</option>
7      <option>Deutsch</option>
8      <option>Informatik</option>
9    </field>
10   <field name="klasse" type="number">
11     <prompt>
12       Zu welcher Klassenstufe möchtest Du Informationen über
13       <value expr="fach"/> haben?
14     </prompt>
15   </field>
```

```
16  </form>
```

Zunächst wird der Benutzer gefragt, zu welchem Fach er Beratung wünsche. Die Auswahl beschränkt sich auf die Fächer Englisch, Deutsch und Informatik. Je nach dem, für welche der Möglichkeiten sich der Benutzer entscheidet, wird der *Feldvariablen* ein entsprechender Wert, das ist entweder *Englisch*, *Deutsch* oder *Informatik*, zugewiesen. Wie Variablen auch, können Feldvariablen mithilfe des <value>-Tags innerhalb eines <prompt>-Tags ausgegeben werden.

### Die Ablaufkontrolle

In VoiceXML implementierte Sprachdialoge werden durch den Formular-Interpretations-Algorithmus, kurz FIA, abgearbeitet. Dieser Algorithmus wird im VoiceXML Standard beschrieben. Der FIA arbeitet die VoiceXML-Tags der VoiceXML Dokumente sequentiell ab. Der Algorithmus endet, wenn nach der Abarbeitung sämtliche Benutzereingaben in den Feldvariablen der VoiceXML-Dokumente erfasst sind, d. h. wenn in den Feldvariablen die zuvor deklarierten Werte zugewiesen wurden. Der Algorithmus kann das erkennen, weil die Feldvariablen zu Beginn des Dialoges den speziellen Wert *undefined* besitzen.
Die Implementierung eines Sprachdialoges kann diese sequentielle Abarbeitung durch Sprünge beeinflussen. Hierfür stellt VoiceXML den <goto>-Tag zur Verfügung. Der <goto>-Tag unterscheidet mithilfe des *nextitem*- und des *next*-Attributes dabei Sprünge innerhalb eines Formulars bzw. zu anderen Formularen des implementierten Sprachdialoges.

### Die Aktion

Einer Feldvariable können nicht nur gesprochene Wörter oder Sätze zugewiesen werden. Es kann auf das Gesprochene auch direkt reagiert werden, wie das folgende Beispiel zeigt:

```
1  <field name="fachbestaetigung" type="boolean">
2    <prompt>
3      Du möchtest über das Fach <value expr="fach"/> informiert werden?
4    </prompt>
5    <filled>
6      <if cond="fachbestaetigung==?false?">
7      <clear namelist="fach"/> </if>
8    </filled>
9  </field>
```

Der Benutzer wird im Beispiel gefragt, ob es richtig sei, dass er sich über das zuvor ausgewählte Fach informieren möchte. Anschließend wird mithilfe des <filled>-Tags eine Möglichkeit der direkten Einflussnahme auf das Gesagte eröffnet. So kann hier das Gesagte auf bestimmte Inhalte überprüft werden. Dafür stellt VoiceXML Bedingungsabfragen, z. B. das <if>-Tag, zur Verfügung.

**Die Ereignisse**

Treten während eines Dialoges Fehler auf, so können diese mithilfe des <catch>-Tags abgefangen werden. Zur Vereinfachung der Fehlerbehandlung existieren die Tags <noinput> (Keine Spracheingabe empfangen), <nomatch> (Keine passende Treffer bzw. nicht erkannte Spracheingabe), <help > (Hilfe) und <error> (allgemeiner Fehler). Für mögliche Zeitüberschreitungen wurde das <prompt>-Tag um das Attribut <timeout> erweitert.

### 4.1.6 Modellierung eines Sprachdialoges in VoiceXML

Durch die Interpretation des FIA und den in den VoiceXML-Dokumenten implementierten Sprüngen können in VoiceXML implementierte Sprachdialoge analog zu endlichen Automaten auch mithilfe von Zustandsübergangsdiagrammen modelliert werden.

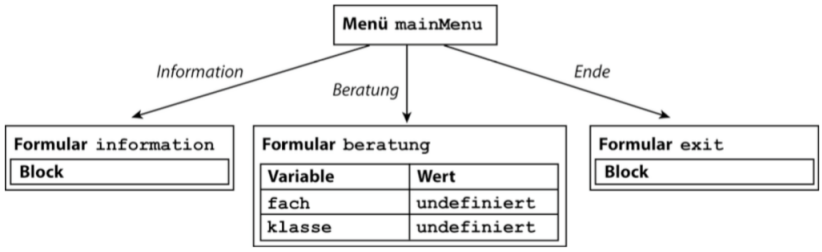



Abbildung 4.8: **Modell eines Sprachdialoges**(vgl. [Hil08], S. 38)

### 4.1.7 Formale Grammatiken

Die Definitionen 4-7 benutzen den Begriff des Wortes im Sinne der theoretischen Informatik.

**Definition 4:**
Eine formale Grammatik ist ein 4-Tupel $G = (V_N, V_T, P, S)$ mit

- einem endlichen Alphabet von Nonterminalen $V_N$.
- einem endlichen Alphabet von Terminalen $V_T$. Es gilt $V_N \cap V_T = \oslash$. $V_N \cup V_T := V$ ist das Gesamtalphabet.
- Eine endliche Menge von Regeln $P \subseteq (V^* \setminus V_T^*) \times V^*$.
- $S \in V_N$, das Startsymbol.

Entwickler von Sprachdialogen können durch die Implementierung formaler Grammatiken im VoiceXML-Quellcode die Erkennungsergebnisse und deren Interpretation zum einen deutlich verbessern und zum anderen können dem Benutzer durch das Sprechen ganzer Sätze mehr Freiräume bei der Kommunikation mit dem System gewährt werden.
**Definition 5:**
Wird ein Wort $v$ aus einem Wort $u$ mittels der Regel einer Grammatik G einschrittig abgeleitet, wird das mit $u \Rightarrow_G v$ notiert. Die Relation $\Rightarrow_G \subseteq V^* \times V^*$ für alle $u$, $v \in V^*$ ist definiert durch:
$u \Rightarrow_G v$ gdw. $\exists u_1, u_2 \in V^* \exists (w_l, w_r) \in P : u = u_1 w_l u_2$ und $v = u_1 w_r u_2$. Wenn der Bezug zur Grammatik bekannt ist, muss das tief gestellte G nicht geschrieben werden. Für mehrschrittige Ableitungen bedient man sich der reflexiven und transitiven Hülle $\Rightarrow_G^*$, die wie folgt definiert ist: $u \Rightarrow_G v$ (in Worten: $u$ geht unter $G$ in endlichen vielen Schritten in $v$ über), wenn $u = v$ oder Wörter $w_0, w_1, \cdots, w_n \in (V \cup \Sigma)^*$ existieren mit $w_0 = u,\ w_n = v$ und $w_i \Rightarrow_G w_{i+1}$ für $i = 0, 1, \cdots n-1$.
**Definition 6:**
Sei $G = (V_N, V_T, P, S)$ eine Grammatik. Die von G erzeugte Sprache ist $L(G) := \{w \in V_T^* | S \Rightarrow_G^* w\}$ (vgl. [Sch08], S. 6).
Für formale Grammatiken gibt es nur zwei Einschränkungen. Dazu gehören:

- Sie darf nur endlich viele Regeln haben.
- Die linke Seite der Regel muss mindestens ein Nonterminal enthalten.

Die Wörter bzw. Sätze können beliebig wachsen oder schrumpfen. Wird die Form der Regeln beschränkt, entstehen verschiedene Grammatiktypen und damit einhergehend verschiedene Sprachtypen unterschiedlicher Schwierigkeitsgrade (vgl. [EP02], S. 56).

**Definition 7 (Chomsky-Hierarchie vom Typ i):**
Sei $G = (V_N, V_T, P, S)$ eine beliebige Grammatik, dann definiert man

| i | Sprachklasse | | definiert als |
|---|---|---|---|
| 3 | regulär, Typ 3 | $\Leftrightarrow$ | für alle $(u, v) \in P$ gilt: $u \in V_N$, $v \in V_T$ oder $v = wA$ mit $w \in V_T$ und $A \in V_N$. |
| 2 | kontextfrei, Typ 2 | $\Leftrightarrow$ | für alle $(u, v) \in P$ gilt: $u \in V_N$, $v \in (V_N \cup V_T)^*$ und $v \neq \lambda$. |
| 1 | kontextsensitiv, Typ 1 | $\Leftrightarrow$ | für alle $(u, v) \in P$ gilt: $u = u_1 A u_2$ und $v = u_1 w u_2$ mit $A \in V_N$, $u_1, u_2, w \in (V_N \cup V_T)^*$ und $w \neq \lambda$. |
| 0 | allgemein, Typ 0 | $\Leftrightarrow$ | für alle $(u, v) \in P$ gilt: $u \in (V^* \setminus V_T^*)$ und $v \in V^*$ |

Tabelle 4.1: Sprachklassen (vgl. [Bre95], S. 30-31)

### 4.1.8 Implementierung kontextfreier Grammatiken in VoiceXML

Für kontextfreie Grammatiken wurden zur vereinfachten Darstellung der Ableitungsregeln eine Reihe von Formalismen (Metasprachen) entwickelt, zu denen ABNF[5] und die SRGS[6] XML gehören und die am gebräuchlichsten sind. Ferner wird hinsichtlich der Implementierung das Wort im computerlinguistischen Sinne benutzt. Bei der ABNF-Darstellung werden die Nonterminale groß geschrieben und mit einem führenden „$-Zeichen“ gekennzeichnet. Das Startsymbol, hier auch Startregel genannt, wird mit dem Schlüsselwort root eingeleitet. Oder-Konstruktionen werden durch einen senkrechten Strich (Symbol) getrennt, während bei Und-Konstruktionen Terminale und Variablen lediglich hintereinander aufgeschrieben werden. Das „leere“ Wort kann mit $NULL gekennzeichnet werden. Das folgende Beispiel zeigt eine solche ABNF-Darstellung.

Listing 4.2: Formale Regeln für die syntaktische Struktur eines deutschen Satzes in ABNF

```
1 root $S;
2 $S= $NP  $VP;
3 $VP= $V  $NP  $PP  $PP;
4 $VP= $V  $NP  $PPs;
5 $NP= $ART  $N | $N;
```

[5] Augmented Backus Naur Form
[6] Speech Recognition Grammar Specification

```
6  $ART= 'der' | 'die' | 'das' | 'eine' | $NULL;
7  $PP= $P  $NP;
8  $PPs = $PP $PP;
9  $P= 'in' ;
10 $N= 'Klausur' | 'Vererbung' | 'Raum105' | 'Schüler';
11 $V= 'schreiben' | $NULL;
12 $P= 'über' | 'im';
```

Beispielsweise kann von der Startregel ausgehend der Satz „die Schüler schreiben eine Klausur über Vererbung im Raum 105“ erzeugt werden:

$$
\begin{aligned}
\$S &=_1 \$NP\ \$VP;\\
&=_5 (\$ART\ \$N)\ \$VP;\\
&=_6 (\text{'die'}\ \$N)\ \$VP;\\
&=_{10} \text{'die'}\ \text{'Schüler'}\ \$VP;\\
&=_3 \text{'die'}\ \text{'Schüler'}\ \$V\ \$NP\ \$PP\ \$PP;\\
&=_{11} \text{'die'}\ \text{'Schüler'}\ \text{'schreiben'}\ \$NP\ \$PP\ \$PP;\\
&=_5 \text{'die'}\ \text{'Schüler'}\ \text{'schreiben'}\ (\$ART\ \$N)\ \$PP\ \$PP;\\
&=_6 \text{'die'}\ \text{'Schüler'}\ \text{'schreiben'}\ (\text{'eine'}\ \$N)\ \$PP\ \$PP;\\
&=_{10} \text{'die'}\ \text{'Schüler'}\ \text{'schreiben'}\ \text{'eine'}\ \text{'Klausur'}\ \$PP\ \$PP;\\
&=_7 \text{'die'}\ \text{'Schüler'}\ \text{'schreiben'}\ \text{'eine'}\ \text{'Klausur'}\ (\$P\ \$NP)\ \$PP;\\
&=_{12} \text{'die'}\ \text{'Schüler'}\ \text{'schreiben'}\ \text{'eine'}\ \text{'Klausur'}\ (\text{'über'}\ \$NP)\ \$PP;\\
&=_5 \text{'die'}\ \text{'Schüler'}\ \text{'schreiben'}\ \text{'eine'}\ \text{'Klausur'}\ (\text{'über'}\ \$N)\ \$PP;\\
&=_5 \text{'die'}\ \text{'Schüler'}\ \text{'schreiben'}\ \text{'eine'}\ \text{'Klausur'}\ \text{'über'}\ \text{'Vererbung'}\ \$PP;\\
&=_7 \text{'die'}\ \text{'Schüler'}\ \text{'schreiben'}\ \text{'eine'}\ \text{'Klausur'}\ \text{'über'}\ \text{'Vererbung'}\ (\$P\ \$NP);\\
&=_{12} \text{'die'}\ \text{'Schüler'}\ \text{'schreiben'}\ \text{'eine'}\ \text{'Klausur'}\ \text{'über'}\ \text{'Vererbung'}\ (\text{'im'}\ \$NP);\\
&=_5 \text{'die'}\ \text{'Schüler'}\ \text{'schreiben'}\ \text{'eine'}\ \text{'Klausur'}\ \text{'über'}\ \text{'Vererbung'}\ (\text{'im'}\ \$N);\\
&=_{10} \text{'die'}\ \text{'Schüler'}\ \text{'schreiben'}\ \text{'eine'}\ \text{'Klausur'}\ \text{'über'}\ \text{'Vererbung'}\ \text{'im'}\ \text{'Raum105'};
\end{aligned}
$$

Die Indizes der Gleichheitszeichen geben an, welche der ABNF-Regeln bei der Erzeugung der Sprache zum Einsatz kamen.
Die gleiche formale Grammatik kann mithilfe von XML wie folgt ausgedrückt werden:

```
1  <grammar langid="407">
2      <rule id="S" scope="public">
3          <ruleref uri="#NP"/>
4          <ruleref uri="#VP"/>
5      </rule>
```

```
6       <rule id="VP">
7           <ruleref uri="#V"/>
8           <ruleref uri="#NP"/>
9           <ruleref uri="#PP"/>
10          <ruleref uri="#PP"/>
11      </rule>
12      <rule id="V">
13          <token>fährt</token>
14      </rule>
15      <rule id="NP">
16          <ruleref uri="#ART"/>
17          <ruleref uri="#N"/>
18      </rule>
19      <rule id="NP">
20          <ruleref uri="#N"/>
21      </rule>
22      <rule id="ART">
23          <one-of>
24              <item>der</item>
25              <item>die</item>
26              <item>das</item>
27              <item>eine</item>
28              <item>
29                  <ruleref special="NULL"/>
30              </item>
31          </one-of>
32      </rule>
33      <rule id="N">
34          <token>Garage</token>
35      </rule>
36      <rule id="V">
37          <token>funktioniert</token>
38      </rule>
39      <rule id="V">
40          <token>schreiben</token>
41      </rule>
42      <rule id="V"/>
43      <rule id="PP">
44          <ruleref uri="#P"/>
45          <ruleref uri="#NP"/>
46      </rule>
47      <rule id="P">
48          <token>in</token>
49      </rule>
50      <rule id="N">
51          <token>Klausur</token>
52      </rule>
53      <rule id="N">
54          <token>Vererbung</token>
55      </rule>
56      <rule id="N">
57          <token>Raum105</token>
```

```
58      </rule>
59      <rule id="N">
60          <token>Schüler</token>
61      </rule>
62      <rule id="P">
63          <token>über</token>
64      </rule>
65      <rule id="P">
66          <token>im</token>
67      </rule>
68  </grammar>
```

In Sprachdialogsystemen werden formale Grammatiken in der sogenannten SRGS[7]-Form dargestellt. Sie werden innerhalb eines VoiceXML-Dokuments mit dem <grammar >-Tag eingebunden. Die SRGS-Grammatiken können sowohl innerhalb desselben VoiceXML-Dokumentes als auch über externe Dateien referenziert werden.
Nach der SRGS-Spezifikation sind Grammatiken sowohl in XML- als auch in der ABNF-Darstellung (vgl. [W3C16b]) möglich.

### 4.1.9 Semantische Tags

Entwickler von Sprachdialogen mit gemischter Initiative setzen sich bei der Implementierung von formalen Grammatiken die Erfassung von Daten aus natürlich gesprochenen Sätzen zum Ziel. Mc Tear bezeichnet solche formalen Grammatiken auch als semantische Grammatiken. Semantische Grammatiken sind kontextfreie und speziell für einen Anwendungsfall angepasste Grammatiken, die Sätze mit nur einer Interpretationsmöglichkeit zulassen (vgl. [McT04], S. 94-95). Das folgende Beispiel zeigt eine solche formale Grammatik in ABNF, die lediglich Sätze für die Tätigkeit einer Flugbuchung erzeugt.

Listing 4.3: Semantische Grammatik für die Buchung eines Fluges in ABNF

```
1  root $FLUG;
2  $FLUG= ( 'Ich' $FLOSKEL $VON $START $NACH $ZIEL $AKTIVITAET );
3  $ZIEL= 'Hamburg' | 'Athen' | 'Mailand';
4  $START= 'Berlin' | 'Muenchen' | 'Stuttgart';
5  $FLOSKEL= $NULL | 'moechte' | 'will';
6  $AKTIVITAET= $NULL | 'fliegen';
7  $VON= $NULL | 'von';
8  $NACH= $NULL | 'nach';
```

[7]Speech Recognition Grammar Specification

Sogenannte semantische Tags können zusätzlich nur die für einen speziellen Anwendungsfall wichtigen Daten des Textes extrahieren. Am Beispiel der Flugbuchung könnte das der Inhalt der Variablen $START und $ZIEL sein. In ABNF werden semantische Tags am Ende einer Regel hinzugefügt und wie folgt deklariert:

Listing 4.4: Die Verwendung semantischer Tags

```
1 $FLUG= ( Ich  $FLOSKEL  $VON  $START  $NACH  $ZIEL  $AKTIVITAET )
2        {out.Start=rules.START;out.Ziel=rules.ZIEL;};
```

Mithilfe der Punktnotation und dem Schlüsselwort *out* werden die aus den erkannten Sätzen extrahierten Daten den dazugehörigen Feldvariablen *Start* und *Ziel* des in VoiceXML modellierten Sprachdialoges zugeordnet.

### 4.1.10 Kellerautomaten

In Kapitel 4.1.4 wurde der deterministische endliche Automat eingeführt, denn die zustandsbasierte Sprachdialogmodellierung basiert auf diesem informatischen Konzept. Deterministische endliche Automaten kommen jedoch nicht nur bei der Modellierung von Sprachdialogen zum Einsatz. Auch für die Spracherkennung ist dieses informatische Konzept bedeutend, denn mithilfe eines deterministischen endlichen Automaten kann überprüft werden, ob ein vom Automaten akzeptiertes Wort mit einem Wort der zuvor modellierten Sprache übereinstimmt.
Jedoch reicht das Konzept des endlichen Automaten nicht für alle kontextfreien Grammatiken bzw. kontextfreien Sprachen aus, d. h. es gibt kontextfreie Sprachen, die nicht vom endlichen Automaten akzeptiert werden. Das sind genuine kontextfreie Sprachen, deren Regeln ihrer zugrunde liegenden kontextfreien Grammatiken eine Mittenrekursion bzw. Zentraleinbettung aufweisen, was sich am Beispiel der folgenden kontextfreien Grammatik in ABNF verdeutlichen lässt.

```
1 root $S;
2 $S           = $NULL;
3 $S           = ( $KLAMMER_AUF  $VARIABLEN  $OPERATION  $S  $KLAMMER_ZU  $S ) ;
4 $S           = ( ( $KLAMMER_AUF  $VARIABLEN )  $OPERATION  $VARIABLEN  $KLAMMER_ZU
  $S ) ;
5 $OPERATION   = +;
6 $KLAMMER_AUF = ( ;
7 $KLAMMER_ZU  = ) ;
8 $VARIABLEN   = x |  y |  a |  b ;
```

Diese Grammatik erzeugt eine Sprache mathematischer Terme, z. B. $(x + y)$ oder $((x + y) + a)$. Die Modellierung eines deterministischen endlichen Automaten ist für

diese kontextfreie Grammatik nicht möglich, denn jeder öffnenden Klammer muss eine schließende Klammer zugeordnet werden können. Dafür müssen jeweils neue Zustände modelliert werden. Die obige Grammatik erzeugt Terme, mit $n$ öffnenden und $n$ schließenden Klammern, wofür $n$ Zustände gebraucht werden. Das widerspricht jedoch dem Konzept endlicher Automaten, bei dem die Anzahl der Zustände endlich sein muss.

**Definition 7:**
Ein nichtdeterministischer Kellerautomat (kurz PDA) ist ein 6-Tupel
$A = (Z, \Sigma, \Gamma, \delta, Z_{start}, \bot)$ mit

- der endlichen Menge von Zuständen $Z$.
- dem endlichen Alphabet $\Sigma$ von Eingabesymbolen.
- dem endlichen Alphabet $\Gamma$ von Kellersymbolen.
- der Überführungsfunktion $\delta : Z \times (\Sigma \cup \{\lambda\}) \times \Gamma \to 2^{Z \times \Gamma *}$.
- dem Startzustand $Z_{start} \subseteq Z$.
- dem Kellerbodensymbol $\bot \in \Gamma$.

**Definition 8:**
Eine Konfiguration eines PDA $A$ ist ein Element $(z, w, v) \in Z \times \Sigma^* \times \Gamma^*$, d. h. dass $A$ im Zustand $z$, das $w$ noch auf dem Eingabeband zu lesen und der aktuelle Kellerinhalt $v$ ist. Die Überführungsrelation $\vdash \subseteq (Z \times \Sigma^* \times \Gamma^*) \times (Z \times \Sigma^* \times \Gamma^*)$ ist definiert durch $(z, xu, yw) \vdash (z', u, y'w)$ gdw. $(z', y') \in \delta(z, x, y)$.
Die von einem PDA akzeptierte Sprache ist die Menge $L_\lambda(A) := \{w \in \Sigma^* | (z_0, w, \bot) \vdash^* (z, \lambda, \lambda), z \in Z\}$. Im Gegensatz zum endlichen Automaten wird beim PDA bezüglich der Endzustände beim Akzeptieren von Wörtern abgewichen. „Tatsächlich kann man zeigen, dass diese Form des *Akzeptierens durch leeren Keller* mit der des *Akzeptierens durch Endzustand* gleichwertig sind“ (vgl. [Sch08], S. 61). Ähnlich wie endliche Automaten können Kellerautomaten ebenfalls grafisch veranschaulicht werden. Das folgende Beispiel zeigt einen möglichen Kellerautomaten, der die obige Sprache der mathematischen Terme akzeptiert.

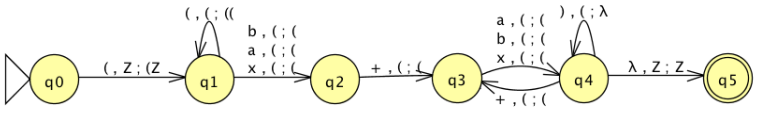



Abbildung 4.9: Kellerautomat einer kontextfreien Grammatik mathem. Terme

## 4.2 inES - integrierte Entwicklungsumgebung für Sprachen

inES ist eine Lernsoftware für die Entwicklung telefonbasierter Sprachdialogsysteme. Sie stellt Schülern dafür eine Sammlung von Werkzeugen für die Modellierung und Ausführung von Sprachdialogen sowie für die Implementierung von Sprachdialogsystemen zur Verfügung. Eine exemplarische Arbeitsweise kann der Handreichung für Lehrer im Anhang D entnommen werden. Ursprünglich wurde inES als Prototyp für das Betriebssystem MacOS X mithilfe der objektorientierten API Cocoa in Objective C entwickelt (vgl. [HB08]). Aufgrund der geringen Verbreitung von Apple-Hardware im schulischen Umfeld wurde inES im Rahmen dieser Arbeit mithilfe der API SWT/JFace in Java neu implementiert. Die Software läuft aktuell unter den Betriebssystemen Linux, MacOS X und Windows. Für die Betriebssysteme MacOS X und Windows wird die vorhandene kostenlos mit ausgelieferte Spracherkennungssoftware nativ eingebunden, während unter Linux die für Java frei verfügbare Spracherkennungsbibliothek Shinx4 benutzt wird.
Die Lernsoftware zeichnet sich durch eine klare Strukturierung und einer leicht bedienbaren Oberfläche aus. Des Weiteren stellt sie den Schülern nur die Funktionen zur Verfügung, die im Unterricht für die Bewältigung der gestellten Aufgaben benötigt werden.
Das Fenster ist vertikal in einen rechten Modell- und einen linken Implementierungsbereich für VoiceXML-Quelltexte geteilt. Der trennende Balken kann mit der gedrückten linken Maustaste verschoben und inES damit an die jeweiligen Situationen angepasst werden. Das Dialogmodell bzw. der VoiceXML - Quelltext können mithilfe des beweglichen Balkens auch ganz ausgeblendet werden.

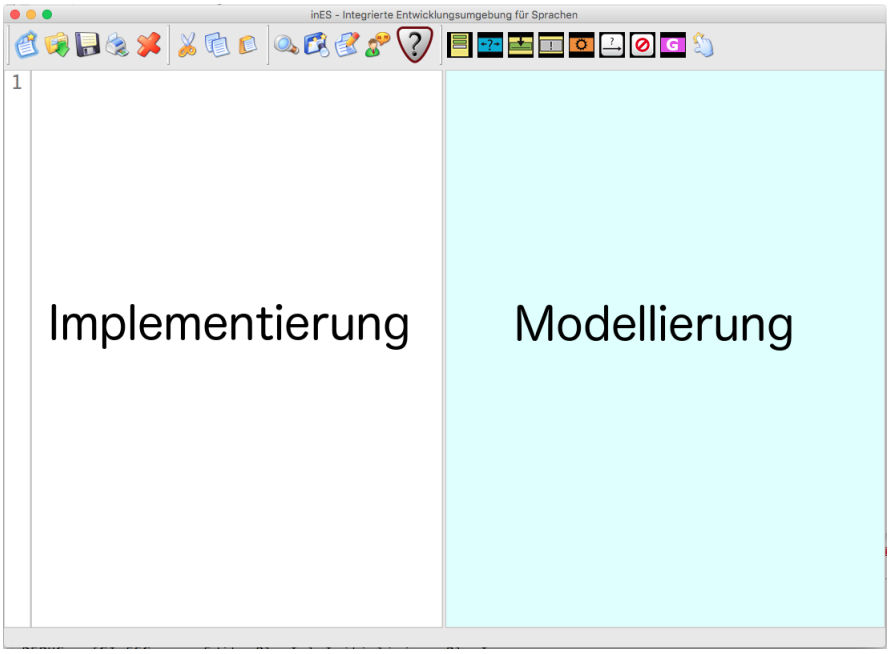



Abbildung 4.10: **inES Hauptfenster mit Beispieldatei**

Für die Modellierung stehen folgende sogenannte Modellobjekte, die sich an der Sprache VoiceXML orientieren, zur Verfügung:

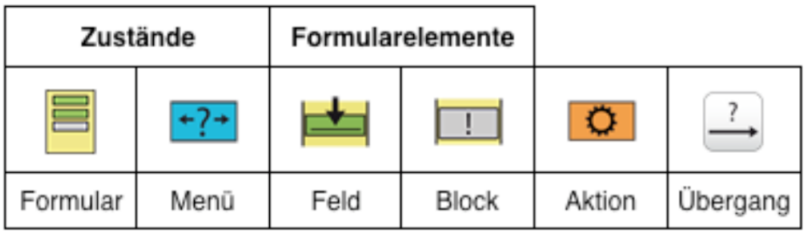



Abbildung 4.11: **inES Modellobjekte**

Damit der Einstieg in die Modellierung von Sprachdialogen möglichst schnell gelingt, wurde die Menge der Modellobjekte (siehe Abbildung 4.13) zwar didaktisch reduziert, dennoch lassen sich hinreichend komplexe Anwendungen entwickeln (vgl. [HB08]).
Mit dem in inES integrierten Editor können die Sprachdialoge in VoiceXML implementiert werden. Der Editor unterstützt die Schüler beim Programmieren durch Syntaxhighlighting, d. h. es werden die verschiedenen VoiceXML-Sprachelemente der Quelltexte je nach Bedeutung unterschiedlich farblich dargestellt und damit die

Lesbarkeit der Quelltexte verbessert.
Modell und VoiceXML-Quelltext sind nicht voneinander abhängig. Weder müssen alle im Zeichenbereich dargestellten Modellobjekte implementiert sein, noch müssen alle implementierten Zustände im Modellbereich vorhanden sein. Diese Trennung soll den Schülern verdeutlichen, dass Modelle keine technische Voraussetzung für eine Implementierung sind, sondern es sich dabei um ein nützliches Hilfsmittel handelt. Anders als im Prototypen kann bei der Neuimplementierung der Lernsoftware inES sowohl der Quellcode als auch das Modell automatisiert abgeglichen werden, d. h. alle programmierten Zustände, die nicht im Modell enthalten sind, können dem Modell automatisch hinzugefügt und umgekehrt können neue im Modell hinzugefügte Zustände automatisch an die passende Stelle im VoiceXML-Quelltext eingefügt werden. Die folgende Abbildung zeigt, welche Teile von Code durch die jeweiligen Modellobjekte erzeugt werden.

| Modellobjekt | Code-Vorlage |
|---|---|
| Menü | `<menu id="menuTitel">`<br>`  <!--Programmiere hier das Menue menuTitel aus.-->`<br>`</menu>` |
| Formular ohne Aktion | `<form id="formTitel">`<br>`  <!--Fuege hier die Formularelemente von formTitel ein.-->`<br>`</form>` |
| Formular mit Aktion | `<form id="formTitel">`<br>`  <!--Fuege hier die Formularelemente von formTitel ein.-->`<br>`  <filled>`<br>`  <!--Programmiere hier die Aktion.-->`<br>`  </filled>`<br>`</form>` |
| Feld ohne Aktion | `<field name="feldTitel">`<br>`  <!--Programmiere hier das Feld feldTitel aus.-->`<br>`</field>` |
| Feld mit Aktion | `<field name="feldTitel">`<br>`  <!--Programmiere hier das Feld feldTitel aus.-->`<br>`  <filled>`<br>`  <!--Programmiere hier die Aktion.-->`<br>`  </filled>`<br>`</field>` |
| Block | `<block>`<br>`  <!--Programmiere hier den anonymen Block aus.-->`<br>`</block>` |

Abbildung 4.12: **Zusammenhang von Modellobjekt und VoiceXML-Quelltext**

Des Weiteren wurde die Anzahl benutzbarer VoiceXML-Tags, d. h. der Befehlsumfang der Programmiersprache VoiceXML, didaktisch reduziert, denn auch hier sollten vor allem die im Unterricht behandelten Problemstellungen lösbar sein. Folgende VoiceXML-Tags werden von inES erkannt und verarbeitet:

| inES - zulässige VoiceXML-Tags | | |
|---|---|---|
| <vxml> | <field> | <option> |
| <form> | <block> | <choice> |
| <menu> | <prompt> | <clear> |
| <value> | <if> | **<initial>** |
| **<grammar>** | | |

Tabelle 4.2: inES - zulässige VoiceXML-Tags

Die hervorgehobenen VoiceXML-Tags sind neu und waren im Prototypen der Lernsoftware noch nicht implementiert. Das VoiceXML-Tag <initial> ermöglicht die Implementierung von Sprachdialogen mit gemischter Initiative. In Verbindung mit dem Tag <grammar> können externe Textdateien mit formalen Grammatiken entweder als SRGS XML- oder als ABNF-Grammatik eingebunden werden. Die in inES implementierten VoiceXML-Sprachdialoge können dadurch beim Ausführen mithilfe der bei inES benutzten Spracherkenner mehrere auf formalen Regeln basierende Wortabfolgen erkennen und verarbeiten. Durch das Benutzen semantischer Tags können des Weiteren mehrere Feldvariablen mit erkannten Inhalten gefüllt werden. Damit eignet sich die aktuelle inES-Version auch für den Einsatz in der gymnasialen Oberstufe, dessen Inhalte in den vorhergehenden Kapiteln dargestellt wurden. Die formalen Grammatiken können entweder mit einem gewöhnlichen Texteditor oder dem Plugin inGE, das im Folgenden vorgestellt wird, entwickelt werden.

Die Syntaxüberprüfung bewerkstelligt inES mit einem validierenden VoiceXML-Parser. Dieser erzeugt auf Grundlage des VoiceXML-Quelltextes einen Automaten, der von dem implementierten VoiceXML-Interpreter ausgeführt wird. Wird ein Fehler gefunden, bricht der Parser ab. Die Zeile, in der der Fehler aufgetreten ist, wird hervorgehoben und eine kurze Fehlerbeschreibung über die Statuszeile der Anwendung ausgegeben. Über die Funktion „Dialog starten“ öffnet sich ein neues Fenster, in dem nach Anklicken des Hörers die Simulation des Telefongespräches gestartet werden kann. inES benutzt unter Windows und Mac OS X die im Betriebssystem vorhandenen Sprachsyntheser, um die vom <prompt> -Befehl umschlossenen Texte über Kopfhörer oder Lautsprecher auszugeben. Unter Linux kann inES die Sprache

über den DFKI MaryTTS Server ausgeben[8]. Der geführte Dialog wird in dem Fenster unterhalb des Tastenfeldes vollständig protokolliert. inES benutzt für die Eingabe unter Windows und Mac OS X die Spracherkennung, die das jeweilige Betriebssystem zur Verfügung stellt. Unter Linux wird hierfür auf die freie Spracherkennungs-API Sphinx4 zurückgegriffen. Die Eingabe kann über ein Mikrofon bzw. einem angeschlossenen Headset erfolgen. Des Weiteren unterstützt inES über die Zifferntasten auch eine DTMF[9]-Eingabe. Das Fenster kann erst nach dem erneuten Anklicken des Hörers geschlossen werden. Da die von den Betriebssystemen kostenlos zur Verfügung gestellten Spracherkenner nicht immer optimale Erkennungsergebnisse liefern, ist die Ausführung des Dialogs auch optional ohne gesprochene Spracheingabe möglich. In diesem Falle erzeugt inES Hyperlinks, die der Benutzer anklicken kann.

## 4.3 inGE - der in inES integrierte Grammatik-Editor

inGE, der in **in**ES integrierte **G**rammatik-**E**ditor, ist eine Erweiterung für die didaktische Lernumgebung inES. Mit inGE können zusätzlich formale Grammatiken für die VoiceXML-Sprachdialoge implementiert werden, sodass die Modellierung von Sprachdialogen mit gemischter Initiative möglich wird und die unter inES benutzten Spracherkenner natürlich gesprochene Wortabfolgen mit mehr als nur einem Wort erkennen können.

inGE kann auf zwei Arten genutzt werden. Entweder als Plugin in inES integriert oder als separate Anwendung. Die Bedienung unterscheidet sich dabei nicht. Mit inGE als Plugin können formale Grammatiken für Sprachdialogsysteme, wie oben bereits beschrieben, implementiert werden. inGE als separate Anwendung gestartet, unterstützt die Erarbeitung theoretischer Inhalte, ohne ganze Sprachdialoge implementieren zu müssen. Wird inGE als Plugin benutzt, dann befindet sich ein entsprechendes Symbol in der Werkzeugleiste von inES (siehe Abbildung 4.15).

[8] Deutsches Forschungszentrum für künstliche Intelligenz (DFKI), http://mary.dfki.de/download

[9] Dial Tone Multiple Frequency

Abbildung 4.13: **Grammatik-Button**

Die Abbildung 4.16 zeigt die Oberfläche von inGE. Sie enthält eine Werkzeugleiste und einen Arbeitsbereich, der durch einen Karteireiter direkt über dem Arbeitsbereich das Implementieren von formalen Grammatiken mithilfe von Syntaxdiagrammen, der XML und der ABNF zulässt. Syntaxdiagramme stellen die Regelmengen formaler Grammatiken grafisch dar. Dabei werden beispielsweise Variablen bzw. Nonterminale mit Rechtecken und Terminale mit Ellipsen repräsentiert. Syntaxdiagramme sind zur XML- bzw. ABNF-Darstellung äquivalent. Durch die grafische Darstellung lassen sie sich von Schülern leichter lesen.
Im Arbeitsbereich lassen sich formale Grammatiken durch Ableitungsbäume oder dem exemplarischen Erzeugen von Sätzen testen. Die Werkzeugleiste besteht aus den Implementierungswerkzeugen für die Syntaxdiagramme und Buttons für das Transformieren einer formalen Grammatik in andere Darstellungsformen (siehe S. 223 im Anhang D2).

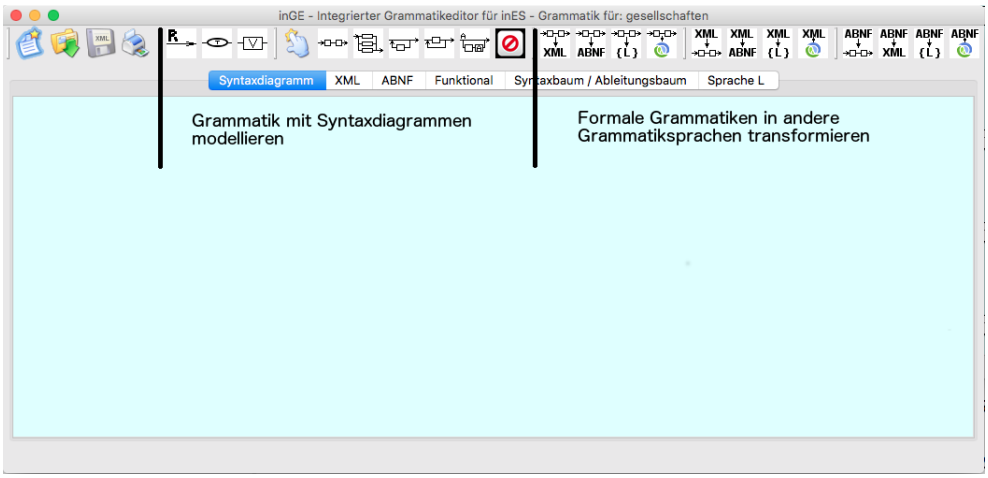



Abbildung 4.14: **Oberfläche von inGE**

Für die Implementierung von formalen Grammatiken in den Darstellungsformen Syntaxdiagramme, SRGS XML und ABNF sowie für das Darstellen von Ableitungsbäumen und das exemplarische Erstellen von Wortabfolgen einer formalen Sprache,

die auf einer zuvor implementierten formalen Grammatik basieren, stellt inGE eine Registernavigation zur Verfügung, mit deren Hilfe das gemeinsam genutzte Programmfenster der jeweiligen Situation angepasst werden kann.

Formale Grammatiken können zunächst als Syntaxdiagramm implementiert werden und anschließend automatisch von inGE in die SRGS XML- bzw. ABNF-Darstellung überführt werden. Umgekehrt kann aus einer formalen Grammatik in SRGS XML- bzw. ABNF-Darstellung ein Syntaxdiagramm erzeugt werden. Das hat didaktische Gründe. Syntaxdiagramme sind grafische Darstellungen formaler Grammatiken und somit klar strukturiert und leicht zu lesen. Soll die formale Grammatik für einen Sprachdialog implementiert werden, muss sie jedoch entweder als SRGS XML- oder ABNF-Grammatik gespeichert werden, denn VoiceXML arbeitet nur mit diesen beiden Darstellungsformen. Stellen die Schüler jedoch fest, dass die Implementierung ihrer formalen Grammatik fehlerhaft ist, kann diese jederzeit wieder als Syntaxdiagramm dargestellt werden. Die Schüler werden dadurch bei der Entwicklung einer formalen Sprache aktiv unterstützt, ohne dass sie sich auch noch um die Syntax der jeweiligen Grammatik-Metasprachen kümmern müssen. Die Schüler können die komplette formale Grammatik als Syntaxdiagramm implementieren und müssen lediglich für die Angabe der semantischen Tags in die SRGS XML- bzw. ABNF-Darstellung durch Drücken des jeweiligen Registers wechseln.

inGE kann auf Basis einer formalen Grammatik und mit Eingabe eines Wortes bzw. einer Wortabfolge sogenannte Ableitungsbäume erzeugen und damit die Arbeitsweise der Ableitungsregeln einer formalen Grammatik grafisch darstellen. Sowohl das Ändern einer formalen Grammatik als auch das Ändern der eingegebenen Wortabfolge aktualisiert den Ableitungsbaum direkt, sodass inGE einen Erkenntnisgewinn für Schüler über das Experimentieren mit formalen Grammatiken und Ableitungsbäumen ermöglicht.

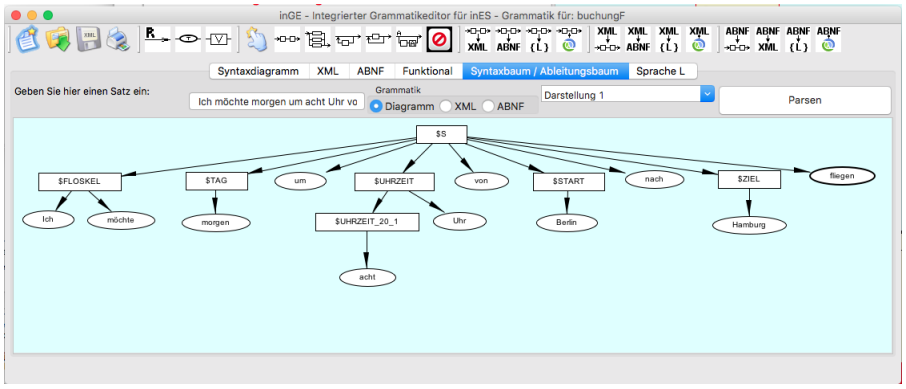



Abbildung 4.15: Ableitungsbäume erzeugen

Schüler können mithilfe dieser Ableitungsbäume auch eine für sie unbekannte formale Grammatik experimentell untersuchen. Des Weiteren werden die Unterschiede zwischen einer formalen und einer natürlichen Sprache klar aufgezeigt. Formale Sprachen entstehen, das zeigt inGE, nach strengen Regeln. Fehlerhafte Zeichen bzw. Ausdrücke führen zu fehlerhaften Ableitungsbäumen.

inGE kann exemplarische Wörter bzw. Wortabfolgen aus einer formalen Grammatik per Knopfdruck erzeugen. Schüler können so erkennen, ob eine von ihnen implementierte formale Grammatik sinnvoll bzw. unsinnig ist und unterstützt dadurch das Analysieren, Beschreiben und Entwickeln von formalen Sprachen.

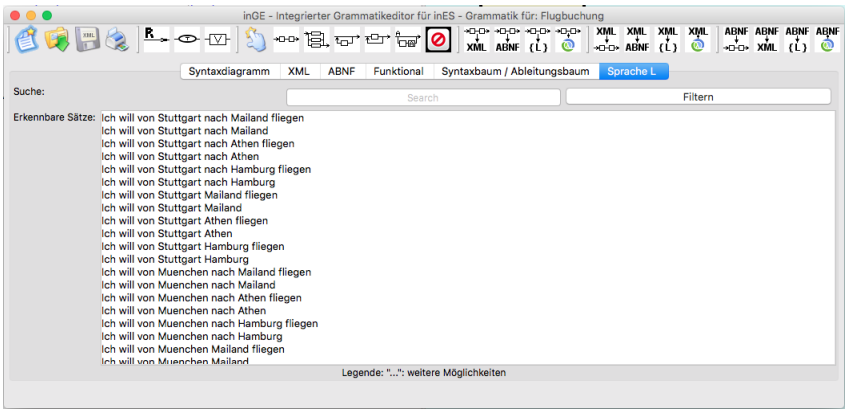



Abbildung 4.16: Sprache erzeugen

inGE unterstützt Schüler einer Oberstufe beim

- Untersuchen von Zusammenhängen zwischen einer Grammatik und ihrer Sprache,
- Vergleichen von natürliche Sprachen mit formalen,
- Ableiten von Wörter einer Sprache,
- Erstellen von Ableitungsbäumen,
- Verwenden von Sprachdefinitionen (z.B. Grammatiken, Syntaxdiagramme) zur Analyse, Beschreibung und Entwicklung formaler Sprachen,
- Überführen von Grammatiken in endliche Automaten und umgekehrt,
- Erläutern von Zusammenhängen zwischen Grammatiken, Sprachen und Automaten (vgl. [Bil16], S.11).

Essentiell ist dabei, dass alle aufgezählten Zielsetzungen des Inhaltsbereiches „Automaten und Sprache“ der Bildungsstandards für die Sekundarstufe II mit inGE und inES in Form von komplexeren Sprachdialogsystemen als Handlungsprodukte und in Einklang mit Kopf-, Herz und Handarbeit umgesetzt werden können.

## 4.4 Planung der Unterrichtseinheit zur Mensch-Maschine-Kommunikation mit gesprochener Sprache

Im Zentrum der Planung und Durchführung der Unterrichtseinheit steht der Erwerb der folgenden Kompetenzen:
Die Schüler

- vergleichen formale mit natürlichen Sprachen,
- untersuchen den Zusammenhang zwischen einer Grammatik und ihrer Sprache,
- leiten Wörter einer Sprache ab und stellen Ableitungsbäume dar,
- verwenden Sprachdefinitionen (z.B. Grammatiken, Syntaxdiagramme) zur Analyse, Beschreibung und Entwicklung formaler Sprachen,
- überführen Grammatiken in endliche Automaten und umgekehrt,
- erläutern den Zusammenhang zwischen Grammatiken, Sprachen und Automaten, analysieren und implementieren Programme zu Problemstellungen auf Kellerautomaten,

(vgl. [Bil16], S.11). Der Unterrichtsverlauf, der in Tabelle 4.3 überblicksartig in Form eines sogenannten Sequenzplanes dargestellt ist, orientiert sich an den vier Phasen

1. Begegnungsphase (L1)
2. Neugier- und Planungsphase (L2)
3. Erarbeitungsphase (L3)
4. Vernetzungs- und Vertiefungsphase (L4)

nach IniK (vgl. Kapitel 2). Im Anschluss daran wird der Unterrichtsverlauf jeder Doppelstunde beschrieben. Nach Durchführung der Unterrichtseinheit und in Kombination mit der Lernsoftware inES werden die Schüler über die Kompetenzen verfügen.

| | Sequenzplan | |
|---|---|---|
| | Thema der Stunde | Ziele |
| L1 | Einstieg in die Welt der Sprachdialoge. | Die Schüler analysieren Spachdialoge.<br>Die Schüler stellen die Dimensionen des Kontextes dar. |
| L2 | Erarbeitung der Grundlagen eines Sprachdialoges. | Die Schüler implementieren ein Sprachdialogsystem in inES. |
| L3.1 | Grenzen bisheriger Sprachdialogsysteme. | Die Schüler erläutern die Grenzen bisheriger Sprachdialogsysteme. |
| L3.2 | Lösungen zur bedingten Überwindung von Grenzen. bisheriger Sprachdialoge (Grammatiken implementieren) | Die Schüler erläutern den Begriff **Formale Grammatik**.<br>Die Schüler implementieren reguläre formale Grammatiken.<br>Die Schüler erläutern Wiederholungsstrukturen in Implementierungen formaler Grammatiken. |
| L3.3 | Grammatiksprachen | Die Schüler implementieren formale Grammatiken in ABNF und XML. |
| L4.1 | Reguläre Ausdrücke | Die Schüler wenden reguläre Ausdrücke bei der Suche nach Daten in Dateien und E-Mails an. |
| L4.2 | Sprachen mithilfe von endlichen Automaten verarbeiten | Die Schüler erläutern den DEA für reguläre Sprachen. |
| L4.3 | Sprachklassen der Chomsky-Hierarchie | Die Schüler vergleichen kontextfreie mit regulären formalen Sprachen.<br>Die Schüler implementieren Kellerautomaten für die Verarbeitung kontextfreier Sprachen. |
| L4.4 | Unsere natürliche Sprache | Die Schüler erläutern Grenzen der Implementierung kontextfreier Grammatiken bezüglich natürlich gesprochener Sprache. |

Tabelle 4.3: Sequenzplan des Unterrichtskonzeptes

Im Folgenden wird der Unterrichtsverlauf jeder Doppelstunde detailliert beschrieben.

### 4.4.1 L1 - 1. Doppelstunde - Einstieg in die Welt der Sprachdialogsysteme

| L1 Verlaufsplanung Stunde 1/2: Einstieg in die Welt der Sprachdialogsysteme | | | | |
|---|---|---|---|---|
| Zeit | Didaktische Funktion | Schüleraktivitäten | Methode / Sozialform | Medien |
| 5 min | Einstieg / Motivation | Video: Zeigen des Videos „Der interaktive Horrorfilm“ | FU | Beamer |
| 20 min | Erarbeitung 1 | Vorspielen des Dialoges mit der Deutschen Bahn.<br>Austeilen des Arbeitsblattes 1.<br>=> Arbeitsauftrag „Analyse des Dialogs“ und<br>Formulierung von Leitfragen. | FU/PA | Arbeitsblatt 1 |
| 10 min | Ergebnissicherung | Fragen auf Flipcharts notieren lassen.<br>Fragen sortieren und Gruppen für deren Recherche initiieren.<br>Die Fragen der Schüler lassen sich i.d.R.<br>nach folgenden Themen clustern:<br>Spracherkennung, Sprachsynthese,<br>Sprachdialogsysteme implementieren,<br>Sprachdialogsysteme in der Gesellschaft,<br>Anwendungsbereiche von Sprachdialogsystemen | sSA/GA<br>sSA/GA | |
| 30 min | Erarbeitung 2 | Schüler recherchieren für die Beantwortung der Fragen.<br>Für jede Frage soll ein kleines Flipchartplakat angefertigt werden.<br>Zusammengefasste Informationen können den beiden<br>zusätzlichen Arbeitsblättern entnommen werden und den<br>Schülern zusätzlich zur Verfügung gestellt werden. | sSA/GA | Flipcharts<br>Infoblatt 1<br>Infoblatt 2 |
| 10 min | Ergebnissicherung | An jedem Plakat bleibt ein Schüler stehen und erläutert<br>den anderen Schülern, die herumgehen,<br>die Ergebnisse der eigenen Gruppenarbeit. | Ausstellung | |
| 5 min | | Ausblick auf die nächste Stunde. | | |

Tabelle 4.4: L1: Einstieg in die Welt der Sprachdialogsysteme

Die Lehrkraft zeigt zum Beginn der Stunde ein Video über einen durch gesprochene Sprache gesteuerten interaktiven Kinofilm, um die Schüler zu motivieren und über das Phänomen „Computer und Sprache“ einen Bezug zu ihrer Lebenswelt herzustellen. Da es sich bei diesem Film um einen sehr zuverlässig funktionierenden Sprachdialog handelt, spielt die Lehrkraft im Anschluss daran die Aufnahme des Radiosenders PSR ab, die einen Sprachdialog zwischen einer Maschine und einem

sächsisch sprechenden Komödianten beinhaltet, um die Schwierigkeiten solcher Systeme für die Schüler sichtbar zu machen. Während die Schüler den Dialog hören, lesen sie den Dialog mithilfe des ersten Arbeitsblattes mit. Nach dieser Phase untersuchen die Schüler mit ihrem jeweiligen Nachbarn den vorgespielten Dialog auf Schwierigkeiten gemeinsam und schreiben ihre Erkenntnisse auf. Darüber hinaus beantworten die Schüler erste Fragestellungen, die ihnen im Zusammenhang mit der Mensch-Maschine-Kommunikation einfallen. In der anschließenden Ergebnissicherung diskutieren die Schüler mit der Lehrkraft über diese Schwierigkeiten. Die Lehrkraft notiert die verschiedenen Fragestellungen der Schüler an einem Whiteboard bzw. Flipchart, denn die Erlebnis- und Erfahrungswelt der Schüler soll innerhalb der Unterrichtsreihe mit berücksichtigt werden und dadurch die Motivation der Schüler für den Kontext steigern. Der Lehrer clustert die Fragestellungen und formuliert mit den Schülern gemeinsam die Leitfragen für diese Unterrichtseinheit. Folgende Leitfragen konnten in vorangegangen Unterrichtsversuchen mit den Schülern gemeinsam erarbeitet werden:

1. Aus welchen Komponenten besteht ein Sprachdialogsystem?
2. Wie kommt die Stimme in den Computer?
3. Wo werden Sprachdialogsysteme bereits eingesetzt? Worin besteht ihr Nutzen?
4. Wo sollten Maschinen nicht als Gesprächspartner eingesetzt werden?
5. Muss man in Zukunft noch fehlerfrei lesen und schreiben können, wenn man mit der natürlichen Sprache kommunizieren kann?
6. Welche Vorteile können Sprachdialogsysteme für behinderte Menschen haben?
7. Wie werden Sprachdialogsysteme programmiert? (vgl. [HB08])

Im Folgenden werden die Schüler in Kleingruppen (max. 4 Personen) Antworten auf diese Fragestellungen unter Zuhilfenahme von Lehrermaterialien bzw. einer Internetrecherche erarbeiten. Die Ergebnisse dieses Prozesses präsentieren die Schüler entweder in Kurzvorträgen oder stellen diese, wie die obige Tabelle zeigt, mit Plakaten dar.

### 4.4.2 L2 - 2. Doppelstunde - Erarbeiten der Grundlagen eines Sprachdialogsystems

| L2 Verlaufsplanung Stunde 3/4: Erarbeiten der Grundlagen eines Sprachdialogsystems | | | | |
|---|---|---|---|---|
| Zeit | Didaktische Funktion | Schüleraktivitäten | Methode / Sozialform | Medien |
| 15 min | Einstieg / Motivation | Die Schüler müssen spielerisch mit Hilfe von Audacity in die Rolle eines Spracherkenners schlüpfen. | FU/PA | Arbeitsblatt 2 |
| 5 min | Ergebniskontrolle | Vergleichen der Ergebnisse. Fragen besprechen. | FU/PA | |
| 10 min | Vorbesprechung | Austeilen der inES-Infoblätter und des Aufgabenblattes 3. Ziel: Aufgabe 1, 2 und 3 sind zu lösen. Zusatzaufgabe für ganz schnelle: Aufgabe 4. | FU/PA | Infoblatt 3 |
| 40 min | Erarbeitung | Schüler arbeiten selbstständig an den Aufgaben. | sSA/GA | PC |
| 15 min | Ergebnissicherung | Schüler das Prinzip der Entwicklungsumgebung erklären lassen. Allgemeine Rückmeldungen. Fragen bezüglich VXML klären. Austeilen des Arbeitsblattes 4 => Fehler im VXML-Code finden lassen. | FU/SV | Beamer Arbeitsblatt 4 |
| 5 min | | Ausblick auf die nächste Stunde. | | |

Tabelle 4.5: L2: Verlaufsplanung Stunde 3/4: Erarbeiten der Grundlagen eines Sprachdialogsystems

Die Schüler einer Oberstufe verfügen i. d. R. nicht über die notwendigen hochschulmathematischen Grundlagen, um die physikalisch-technische Dimension der Spracherkennung im Ganzen zu erfassen. Die Lehrkraft steigt aufgrund dessen in diese Stunde mithilfe der Software Audacity und dem Arbeitsblatt 2 ein. Das Arbeitsblatt 2 stellt drei Mikrofonaufnahmen von gesprochenen Namen deutscher Städte als Zeit-Frequenz-Diagrammen dar. Die Lehrkraft fordert die Schüler auf, sich in die Rolle des Spracherkenners zu versetzen und diese gesprochenen Städtenamen zu erkennen. Dafür müssen die Schüler die auf dem Arbeitsblatt 2 angegebenen Städtenamen zunächst selbst in ein Mikrofon sprechen und anschließend das in Audacity erzeugte eigene Zeit-Frequenz-Diagramm mit dem des Arbeitsblattes vergleichen. Vorangegangene Unterrichtsversuche zeigten, dass sich diese Vorgehensweise gut eignet, denn die Idee der Mustererkennung und der damit verbundenen Schwierigkeiten ist von Schülern aufgrund der Darstellung durch Zeit-Frequenz-Diagramme gut nachvollziehbar. Im Anschluß daran fragt die Lehrkraft nach den möglichen Städtenamen für jedes Zeit-Frequenz-Diagramm. Die Lehrkraft fordert einen Schüler auf, den für

ihn wahrscheinlichsten Städtenamen für das erste Zeit-Frequenz-Diagramm zu nennen. Danach lässt die Lehrkraft über den genannten Städtenamen abstimmen, in dem sie den gesamten Kurs auffordert, sich bei Übereinstimmung zum genannten Ergebnis zu melden. Für das zweite und dritte Zeit-Frequenz-Diagramm wird analog vorgegangen. Mit dieser Methode lassen sich die Schwierigkeiten der digitalen Spracherkennung hinsichtlich verschiedener Sprecher und der Mustererkennung gut diskutieren.

Danach folgt die Einführung in die Lernsoftware inES und damit die Exploration der informatischen Dimension. Die Lehrkraft lässt die Schüler mit dem Arbeitsblatt 3 und der Lernsoftware inES erste kleine Sprachdialogsysteme implementieren. Das Arbeitsblatt 3 ist nach den Grundsätzen des didaktischen Dreischrittes konzipiert, d. h. die Schüler analysieren und testen zuerst einen vorbereiteten Sprachdialog, um die grundlegende Bedienungsweise der Entwicklungsumgebung kennenzulernen. Darüber hinaus müssen sie im vorliegenden dritten Arbeitsblatt bestimmte Ergänzungen im Modell und im Quelltext des Sprachdialogsystems vornehmen, damit sie den Umgang mit der Auszeichnungssprache Voice-XML kennenlernen und sich in die Syntax einarbeiten.

Die Ergebnissicherung erfolgt mithilfe des Arbeitsblattes 4. Die Lehrkraft teilt das Arbeitsblatt aus und lässt den Voice-XML Quelltext auf Fehler untersuchen. Anschließend projiziert die Lehrkraft das Arbeitsblatt 4 z. B. über einen Beamer an ein Whiteboard und fordert die Schüler auf, ihre Ergebnisse mit denen der Musterlösung zu vergleichen. Dabei bespricht die Lehrkraft typische Voice-XML Fehlerquellen mit den Schülern.

### 4.4.3 L3 - 3. Doppelstunde - Grenzen von vielen Sprachdialogsystemen und Lösungen für deren bedingter Überwindung I

L3 Verlaufsplanung Stunde 5/6: Grenzen von vielen Sprachdialogsystemen und Lösungen für deren bedingter Überwindung I

| Zeit | Didaktische Funktion | Schüleraktivitäten | Methode / Sozialform | Medien |
|---|---|---|---|---|
| 5 min | Übung / Motivation | Austeilen des Arbeitsblattes 5.<br>Aufgabe: Vervollständige das Modell mithilfe des Quelltextes! | sSA/PA | Arbeitsblatt 5 |
| 5 min | Ergebniskontrolle | Schüler durch Handmeldungen auffordern die Folien auf dem Overheadprojektor auszufüllen (Fehler korrigieren).<br>Gelenkstelle: Wenn alle Schüler den Fehler gefunden haben => Lob.<br>Alternative: Lösungsfolie auflegen und Problempunkte besprechen. | sSA | OHP,Folie |
| 5 min | Einstieg | Schüler bisherige Grenzen der Sprachdialogsysteme notieren lassen. | EA | Arbeitsblatt 6 |
| | | Grenzen mündlich nennen lassen und an der Tafel notieren. | | Whiteboard |
| 5 min | | Audiodatei von Crealog vorspielen (flugauskunft.mp3 oder reisebuchung.mp3). | FU | |
| 10 min | Erarbeitung | „Das Erkennen natürlicher gesprochener Sprache“<br>Diese Phase gilt als abgeschlossen, wenn alle Schülerinnen und Schüler erfolgreich den vorhandenen Sprachdialog testen konnten. Zur Not auch ohne die „richtige“ Spracherkennung, wenn Hardware/Softwareprobleme auftreten.<br><br>Leitfrage für die nächsten Stunden:<br>„Wie ist es möglich, dass Sätze erkannt und die gesuchten Informationen (z.B. Wochentag, Datum) automatisiert herausgefiltert werden können?“<br>finden lassen. | LV | Beamer<br>Präsentation<br>Videotutorials<br>Arbeitsblatt 7 |
| 10 min | Ergebnissicherung I | Austeilen von unbekannten XML-Grammatiken.<br>Schüler sollen Syntaxdiagramm und Beispiele der Sprache dieser Grammatik per Hand erzeugen. | EA | Arbeitsblatt 8 |
| 5 min | Ergebnissicherung II | Arbeitsblatt 8 vor Unterrichtsbeginn auf eine OH-Folie kopieren. Lehrerlösung aufdecken und gemeinsam die Lösungen besprechen / Fragen klären. | FU | OHP,Folie |
| 5 min | | Ausblick auf die nächste Stunde. | | |

Tabelle 4.6: L3 Verlaufsplanung Stunde 5/6: Grenzen von vielen Sprachdialogsystemen und Lösungen für deren bedingter Überwindung I

Die Lehrkraft teilt für die Reaktivierung und Festigung des vorhandenen Wissens der vorangegangenen Stunden zu Beginn dieser Stunde das Arbeitsblatt 5 mit einer Übung zu Voice-XML und zum Modell eines Sprachdialoges aus. Die Schüler müssen darin ein unvollständiges Modell mithilfe eines Voice-XML Quelltextes vervollständigen. Anschließend vergleichen die Schüler ihre Ergebnisse mit denen der Musterlösung der Lehrkraft. Danach lässt die Lehrkraft die Grenzen der bisherig implementierten Sprachdialogsysteme von den Schülern notieren. Die Lehrkraft fordert die Schüler auf, die aufgeschriebenen Einschränkungen zu nennen. In der Vergangenheit wurden ohne Anspruch auf Vollständigkeit folgende Einschränkungen von den Schülern genannt:

- Bisherige Sprachdialogsysteme erkennen und verarbeiten jeweils nur ein bis max. zwei einzeln gesprochene Worte.
- Die Abarbeitung von Feld zu Feld führt dazu, dass alle Daten einzeln abgefragt werden müssen.
- Floskeln bzw. Dialekte werden nicht berücksichtigt.
- Dialoge werden in Folge dessen sehr lang, was zu hohen Telefonkosten führen kann.

Nun spielt die Lehrkraft einen aufgenommenen Sprachdialog von Crealog, einem sehr progressiven Sprachdialogsystem, ab. Die Schüler erkennen, dass dieses Sprachdialogsystem offensichtlich die notwendigen Daten einem natürlich gesprochenen Satz entnimmt. Das führt zu einer neuen Leitfrage für die kommenden Stunden: „Wie ist es möglich, dass Sätze erkannt und die gesuchten Informationen automatisiert herausgefiltert werden können?“. Ausgehend von diesem Beispiel entwickelt die Lehrkraft in einem Lehrervortrag und mithilfe einer Präsentation eine formale Grammatik. Dafür wird der Begriff der formalen Grammatik zunächst noch nicht formal definiert, sondern er wird induktiv und problemorientiert eingeführt. Nach dem Vortrag fordert die Lehrkraft die Schüler auf, einen in inES mit inGE vorbereiteten Sprachdialog zur Flugbuchung in Verbindung mit einer formalen Grammatik zu untersuchen. Die Aufgabenstellungen entnehmen die Schüler dem Arbeitsblatt 7. Mit diesem Arbeitsblatt erarbeiten sich die Schüler

- die Arbeitsweise des Plugins inGE,
- Grundlagen zu Sprachdialogen mit gemischter Initiative und die Entwicklung von Sprachdialogsystemen/Sprachdialogmodellen zur Verarbeitung natürlich

gesprochener Sätze,

- erste Darstellungsformen (Syntaxdiagramme und SRGS XML) formaler Grammatiken,
- die Möglichkeit gezielte Daten aus einem erkannten Satz zu entnehmen.

Die Schüler erarbeiten sich die Grundlagen formaler Grammatiken zunächst über Syntaxdiagramme und die XML-Darstellung, denn aufgrund des Vorwissens bezüglich Voice-XML sind sie mit der XML-Darstellung vertraut. Für die Ergebnissicherung teilt die Lehrkraft das Arbeitsblatt 8 aus. Die Schüler müssen aus einer gegebenen formalen Grammatik in SRGS XML ein Syntaxdiagramm entwickeln und herausfinden, welche natürlich gesprochenen Sätze damit erkannt werden können. Die Lehrkraft projiziert danach die Musterlösung des achten Arbeitsblattes an ein Whiteboard und bespricht Fragen bzw. Verständnisschwierigkeiten.

### 4.4.4 L3 - 4. Doppelstunde - Grenzen von herkömmlichen Sprachdialogsystemen und Lösungen für deren bedingter Überwindung II

| L3 Verlaufsplanung Stunde 7/8: Grenzen von herkömmlichen Sprachdialogsystemen und Lösungen für deren bedingter Überwindung II | | | | |
|---|---|---|---|---|
| Zeit | Didaktische Funktion | Schüleraktivitäten | Methode / Sozialform | Medien |
| 10 min | Einstieg / Motivation | Letzte Stunde ins Gedächtnis rufen. Schüler erklären prinzipielle Vorgehensweise für das Erstellen eines Sprachdialogsystems mit Hilfe einer Grammatik.<br>Alternative: Schülerinnen und Schüler erstellen eine Art Worflow-Diagramm, wo noch einmal alle grundlegenden Schritte eingetragen werden können. | Lehrer fragt nach und gibt Hinweise.<br>UG | Beamer |
| 40 min | Erarbeitungsphase | Einarbeitung in das Softwaremodul inGE für inES.<br>Lösen der Aufgaben des Arbeitsblattes 7 mit inGE<br>Pflicht: Aufgabe 3) - 5)<br>Aufgabe 6) ist optional. | EA/PA | Rechner/ Software |
| 20 min | Ergebnisvergleich | Schüler stellen Aufgabenlösungen vor. | SV | Beamer |
| 15 min | Ergebnissicherung | Der Begriff der Grammatik. | LV/UG | Beamer Präsentation |
| 5 min | | Ausblick auf die nächste Stunde. | | |

Tabelle 4.7: L3 Verlaufsplanung Stunde 7/8: Grenzen von herkömmlichen Sprachdialogsystemen und Lösungen für deren bedingter Überwindung II

Die Lehrkraft projiziert zu Beginn der Stunde das Arbeitsblatt 8 an ein Whiteboard und lässt exemplarische Wortfolgen auf Grundlage dieser formalen Grammatik nennen. Danach lässt die Lehrkraft die grundlegende Vorgehensweise der Implementierung einer formalen Grammatik in einem Sprachdialog mit gemischter Initiative beschreiben und ruft die ersten grundlegenden Begriffe Syntaxdiagramm, Regel, Terminal und Nonterminal in Erinnerung, um die Schüler zu aktivieren. Des Weiteren dient dieser Einstieg der Festigung dieser ersten Begriffe und der Arbeitsweise mit der Lernsoftware inES und inGE.
Im Anschluss daran leitet die Lehrkraft die Schüler in eine Arbeitsphase über, in dem sie die Schüler auffordert, die Aufgaben 3) bis 5) des siebenten Arbeitsblattes zu bearbeiten. Diese Aufgaben festigen den Umgang mit formalen Grammatiken,

denn die Schüler üben das

- Einfügen von Terminalen,
- Einfügen von Nonterminalen,
- Implementieren von formalen Grammatiken, um bestimmte Wortfolgen zu erkennen und
- Implementieren von semantischen Tags zur Erkennung gesuchter Informationen in Wortfolgen

gehören. Nach der Arbeitsphase von maximal 40 Minuten fordert die Lehrkraft für jede Aufgabe einen Schüler auf, das Ergebnis der Arbeitsphase mithilfe eines Schülervortrages vorzustellen. Danach definiert die Lehrkraft in einem Kurzvortrag den Begriff der formalen Grammatik.

### 4.4.5 L3 - 5. Doppelstunde - Grammatiken in verschiedenen Grammatiksprachen implementieren

| L3 Verlaufsplanung Stunde 9/10: Grammatiken in verschiedenen Grammatiksprachen implementieren | | | | |
|---|---|---|---|---|
| Zeit | Didaktische Funktion | Schüleraktivitäten | Methode / Sozialform | Medien |
| 10 min | Einstieg / | Wiederholung Begriff Grammatik und die Implementierungsarten: Alternative, Sequenz, Terminale, Variablen | UG | Beamer |
| 20 min | Erarbeitungsphase | Schülerinnen und Schüler bearbeiten Aufgabe 6). Lösen der Aufgaben des Arbeitsblattes 7 mit inGE | EA/PA | Rechner/ Software Arbeitsblatt 7 |
| 5 min | Ergebnisvergleich | Schüler stellen Aufgabenlösungen vor. | SV | Beamer |
| 10 min | Ergebnissicherung | Die Lösungen werden diskutiert. Die Begriffe Rekursion und Iteration werden besprochen. Vor- und Nachteile rekursiver Implementierung. | LV/UG | Beamer Präsentation |
| 5 min | Besprechung | Schüler bekommen kleinen „Forschungsauftrag“ : Sie sollen mit Hilfe von inGE die Gemeinsamkeiten und Unterschiede der verschiedenen Grammatiksprachen (Syntaxdiagramm, SRGS-XML Grammar, ABNF) herausfinden. Dafür füllen sie die Tabelle des Arbeitsblattes 9 aus. | PA | Arbeitsblatt 9 |
| 25 min | Erarbeitung | Schüler bearbeiten das Arbeitsblatt 9. | PA | |
| 10 min | Ergebnissicherung | Besprechen des Aufgabenblattes 9. Thematisieren des Begriffes Ableitungsbaum. | LV/UG | |
| 5 min | | Ausblick auf die nächste Stunde. | | |

Tabelle 4.8: L3 Verlaufsplanung Stunde 9/10: Grammatiken in verschiedenen Grammatiksprachen modellieren

Die Lehrkraft projiziert die Implementierung einer unbekannten formalen Grammatik als Syntaxdiagramm an ein Whiteboard und fordert die Schüler zunächst auf, die dargestellten Symbole zu erklären, um die Schüler für diese Stunde zu aktivieren und das vorhandene Wissen zu festigen. Des Weiteren fordert die Lehrkraft von den Schülern exemplarische Wortfolgen, die mit dieser formalen Grammatik erzeugt werden können, zu nennen.
Im Anschluss daran verlangt die Lehrkraft das Lösen der sechsten Aufgabe des siebten Arbeitsblattes. Die Schüler müssen eine formale Grammatik für das Erfassen von Telefonnummern implementieren. In der sich anschließenden Ergebnissicherung werden verschiedene Implementierungen, in diesem Fall iterative und rekursive Im-

plementierungsarten, besprochen. In der Vergangenheit zeigte sich, dass insbesondere rekursive Implementierungen in Programmiersprachen für Schüler nicht trivial sind. Am Beispiel der formalen Grammatiken können sich Schüler durch das Experimentieren mit rekursiven Implementierung formaler Grammatiken, Grundlagen rekursiver Implementierungen sehr anschaulich erarbeiten.
Der zweite Teil der Doppelstunde widmet sich den verschiedenen Darstellungsformen formaler Grammatiken in Sprachdialogsystemen. Zum Einsatz kommt hier neben der SRGS in XML Darstellung auch die Augmented Backus Naur Form (kurz ABNF)[10]. Die Lehrkraft teilt das Arbeitsblatt 9 aus und lässt die Schüler die beiden Aufgaben lösen. In der ersten Aufgabe des Arbeitsblattes füllen die Schüler eine Tabelle aus, in der sie die unterschiedlichen Notationen für Regeln, Terminale, Nonterminale sowie die UND- bzw. ENTWEDER ODER-Implementierungen für die SRGS- und ABNF-Darstellung eintragen. Die Schüler sind zum jetzigen Zeitpunkt mit der SRGS in XML-Darstellung vertraut und erkennen, dass sich die ABNF-Darstellung vor allem in der Notation unterscheidet. Sie fokussiert stärker als die XML-Darstellung auf die Implementierung formaler Grammatiken und ist dadurch übersichtlicher, denn die vielen XML-Tags fallen dort weg. Mithilfe der Tabelle werden die bereits erlernten Begriffe weiter gefestigt und durch die zweite Aufgabe das Implementieren formaler Grammatiken geübt. Nach der Erarbeitungsphase projiziert die Lehrkraft die Lösung an ein Whiteboard und lässt die Schüler ihre eigenen Lösungen mit der Musterlösung vergleichen. Danach fordert die Lehrkraft zwei bis drei Schüler auf, jeweils eine Lösung der zweiten Aufgabe des neunten Arbeitsblattes vorzustellen.

[10] Auf deutsch: Erweiterte Backus Naur Form.

### 4.4.6 L4 - 6. Doppelstunde - Gezielt suchen - Grammatiken helfen dabei, das tägliche Leben zu vereinfachen

| L4 Verlaufsplanung Stunde 11/12: Grammatiken helfen dabei, das tägliche Leben zu vereinfachen | | | | |
|---|---|---|---|---|
| Zeit | Didaktische Funktion | Schüleraktivitäten | Methode / Sozialform | Medien |
| 10 min | Einstieg | Besprechen der Aufgaben des Arbeitsblattes 10 Reguläre Ausdrücke | LV/UG | |
| 30 min | Erarbeitung | Schüler lösen das Aufgabenblatt 10 mithilfe einer Übersicht über die regulären Ausdrücke. | sSA/PA Lehrer gibt Hilfestellungen. | Aufgabenblatt 10 |
| 5 min | Ergebnisvergleich | Schüler stellen Aufgabenlösungen vor. | SV | Beamer |
| 10 min | Ergebnissicherung I | Vergleich der Lösungen mit Lösungsblatt Besprechen der Schwierigkeiten. | LV/UG | Beamer |
| 10 min | Ergebnissicherung II | Zusammenfassender Vortrag über reguläre Ausdrücke und die reguläre Sprache. | LV | Präsentation |
| 5 min | Besprechung | Übertragen (Transfer) des Wissens auf die „Suche nach Dateien (in Dateien) und Ordnern.“ Dabei wird auf die Befehle grep (Linux) und findstr (Windows) zurückgegriffen. Das Arbeitsblatt 11a enthält Übungsaufgaben für grep, 11b für findstr. | LV | Arbeitsblatt 11a Arbeitsblatt 11b |
| 15 min | Übungsphase | Schüler suchen mit Hilfe von regulären Ausdrücken nach Dateien und Ordnern. | sSA | |
| 10 min | Ergebnissicherung III/ Reflexion | Besprechen der Lösungen, besprechen von Problemen bei der Bearbeitung der Aufgaben. | UG | |
| 5 min | | Ausblick auf die nächste Stunde. | | |

Tabelle 4.9: L4 Verlaufsplanung Stunde 11/12: Grammatiken helfen dabei, das tägliche Leben zu vereinfachen

Zu Beginn der Stunde teilt die Lehrkraft das zehnte Arbeitsblatt aus. Die Schüler formulieren in inES und inGE Suchanfragen mithilfe von regulären Ausdrücken, denn reguläre Ausdrücke gehören zu den gebräuchlichen Anwendungen im Bereich formaler Sprachen. Des Weiteren gehört das Suchen nach Informationen auch zur Lebenswelt der Schüler. Die Schüler benutzen für die Formulierung formaler Ausdrücke das vierte Übersichtsblatt als Hilfsmittel. Die Syntax formaler Ausdrücke ist von Informatiksystem zu Informatiksystem unterschiedlich. Es gibt keine standardisierte Syntax formaler Ausdrücke. Deswegen finden die Schüler im vierten Übersichtsblatt die Darstellung der von inES und inGE unterstützten formalen Ausdrücke.

Nach dieser Erarbeitungsphase lässt die Lehrkraft die Schülerlösungen mit der Musterlösung vergleichen und bespricht die verschiedenen Ergebnisse. Die formalen Grundlagen bei der Arbeit mit regulären Ausdrücken fasst die Lehrkraft in einem Kurzvortrag zusammen.
Im zweiten Teil der Doppelstunde teilt die Lehrkraft entweder das Arbeitsblatt 11a, wenn ein Unix-artiges Betriebssystem in der Schule installiert ist bzw. das Arbeitsblatt 11b, wenn ein Windowssystem installiert ist, aus. Das elfte Arbeitsblatt verlangt von den Schülern eine Transferleistung, denn sie formulieren in den beiden Aufgabenstellungen reguläre Ausdrücke mit grep (Unix) oder findstr (Windows), um Daten in Textdateien zu suchen. Im Anschluß an diese Übungsphase zeigt die Lehrkraft die Musterlösung und lässt über zulässige Abweichungen der Schülerlösungen im Plenum diskutieren.

### 4.4.7 L4 - 7. Doppelstunde - Sprache mit Hilfe von endlichen Automaten verarbeiten

| L4 Verlaufsplanung Stunde 13/14: Sprache mit Hilfe von endlichen Automaten verarbeiten | | | | |
|---|---|---|---|---|
| Zeit | Didaktische Funktion | Schüleraktivitäten | Methode / Sozialform | Medien |
| 5 min | Einstieg | Einführung in die endlichen Automaten. | LV | Präsentation eA |
| 20 min | Erarbeitung I | Rückführung auf die Leitfrage: Wie wird Sprache automatisiert verarbeitet?<br>Aufgabe: Entwickle einen endlichen Automaten, der die Sätze „Ich moechte von Berlin nach Hamburg fliegen“ und „Ich will von Berlin nach Hamburg fliegen“, verarbeitet. | sSA | Aufgabenblatt 12 |
| 10 min | Ergebniskontrolle | Vergleich und Besprechen der Schülerlösungen. | LV/SV | Beamer |
| 5 min | Ergebnissicherung II | Sprachdialoge und endliche Automaten. | LV | Präsentation eA am Beispiel inES. |
| 5 min | Besprechung | Lehrer stellt Übungsaufgaben vor. | LV | Arbeitsblatt 13 |
| 25 min | Übung | Schüler müssen, mithilfe von JFLAP, Aufgaben lösen. | sSA/PA | |
| 15 min | Reflexion | Schüler stellen sich gegenseitig ihre Lösungen vor. | UG | |
| 5 min | | Ausblick auf die nächste Stunde. | | |

Tabelle 4.10: L4 Verlaufsplanung Stunde 13/14: Sprache mit Hilfe von endlichen Automaten verarbeiten

Die Funktion dieser Doppelstunde ist die Dekontextualisierung bzw. Vernetzungsphase, in der ausgehend vom Kontext Sprachdialogsysteme auf die dahinterstehenden informatischen Konzepte übergeleitet werden. Dabei steht zunächst die Verarbeitung von formaler Sprache im Mittelpunkt. Im Rahmen dieser Thematik wird das theoretische Konzept eines endlichen Automaten erarbeitet.
Zu Beginn der Stunde hält die Lehrkraft einen Lehrervortrag über die Arbeitsweise eines endlichen Automaten, in dem auch der Zusammenhang zwischen der Modellierung von Sprachdialogen und dem Konzept der endlichen Automaten hergestellt wird. Die Schüler erkennen, dass sie ohne das theoretische Konzept zu kennen, in den vergangenen Unterrichtsstunden bereits endliche Automaten modellierten.
In der sich anschließenden Erarbeitungsphase teilt die Lehrkraft das Arbeitsblatt 12 aus. Dieses Arbeitsblatt führt die Schüler in die Arbeit mit JFLAP (vgl. [Uni13]) ein. Des Weiteren implementieren die Schüler einen ersten endlichen Automaten in JFLAP, der die Wortfolgen

- „Ich möchte von Berlin nach Hamburg fliegen“
- „Ich will von Berlin nach Hamburg fliegen“

verarbeiten kann, denn die formalen Grammatiken für die Erzeugung dieser Wortfolgen kennen sie bereits, sodass die Schüler einen Zusammenhang zwischen formalen Grammatiken und endlichen Automaten erkennen. Nach dieser Phase entwickelt die Lehrkraft die Zusammenhänge zwischen der Modellierung von Sprachdialogen und den endlichen Automaten mithilfe eines Lehrervortrages mit anschließendem Unterrichtsgespräch gemeinsam und führt die wesentlichen Grundbegriffe des Konzeptes eines endlichen Automaten ein.
Danach teilt die Lehrkraft das Arbeitsblatt 13 aus und bespricht die Aufgabenstellungen mit den Schülern gemeinsam. In beiden Aufgaben des Arbeitsblattes erstellen die Schüler endliche Automaten mit der Software JFLAP. Darüber hinaus müssen die Schüler in der ersten Aufgabe des Arbeitsblattes auch eine formale Grammatik für die Erzeugung gleicher Wortfolgen in ABNF implementieren. Die Schüler erkennen, dass sowohl endliche Automaten als auch formale Grammatiken eine Sprache erzeugen können.
Die beiden Aufgabenstellungen des Arbeitsblattes sind umfangreich, deshalb fordert die Lehrkraft am Anfang der Ergebnissicherung die Schüler auf, sich die jeweiligen individuellen Lösungen gegenseitig vorzustellen. Im Anschluss daran wählt die Lehrkraft einige Lösungen für die Reflexion und Besprechung im Plenum aus.

### 4.4.8 L4 - 8. Doppelstunde - Sprachklassen

| L4 Verlaufsplanung Stunde 15/16: Sprachklassen | | | | |
|---|---|---|---|---|
| Zeit | Didaktische Funktion | Schüleraktivitäten | Methode / Sozialform | Medien |
| 5 min | Motivation | Problemstellung für diese Stunde:<br>Zukünftig möchtest Du Deine Mathematikhausaufgaben mithilfe eines Sprachdialogsystems lösen.<br>Dafür müssen Terme erkannt werden.<br>Entwirf eine Grammatik, die mathematische Terme der Gestalt: x y, x ( x y ), y ( ( x y ) x ) erkennt. | sSA/PA | |
| 10 min | Sammlung | Ansehen der Schülerlösungen.<br>Ist eine Musterlösung dabei, dann ist das perfekt, sonst Lehrerlösung mit eingeben und hervorheben, wo das besondere ist.<br>Anknüpfungskontexte: Klammersprache | sSA | Beamer<br>Präsentation<br>Chomsky |
| 5 min | Besprechung I | Aufgabenstellung: Problemreduzierung auf Klammer Auf und Klammer Zu. Schüler sollen einen endlichen Automaten bauen, der die Klammersprache verarbeiten kann. | LV | |
| 10 min | Erarbeitung I | Schüler bearbeiten Aufgabenstellung. | LV | JFlAP |
| 5 min | Besprechung II | Lehrer: Worin liegt das Problem bei dieser Aufgabenstellung? | LV/UG | Beamer |
| 10 min | Erarbeitung II | Lehrervortrag: „Der Kellerautomat“<br>Aufgaben lösen. | LV | Präsentation<br>Kellerautomat |
| 30 min | Übung | Schüler lösen die Aufgaben des Arbeitsblattes 14. | UG | Arbeitsblatt 14 |
| 10 min | Ergebnissicherung | Besprechen und Diskutieren der Ergebnisse. | LV/UG | |
| 5 min | | Ausblick auf die nächste Stunde. | | |

Tabelle 4.11: L4 Verlaufsplanung Stunde 15/16: Sprachklassen

Die Lehrkraft konfrontiert die Schüler zu Beginn dieser Unterrichtsstunde mit einer Aufgabe, in der die Schüler in Partnerarbeit eine formale Grammatik für einen sprachgesteuerten „Mathematikhausaufgabenlöser“ entwickeln sollen. Der Sprachdialog an sich steht nicht im Fokus dieser Aufgabe, sondern nur die Entwicklung einer formalen Grammatik zur Erfassung von Termen. Aufgrund der Vielzahl an Möglichkeiten für die Implementierung einer solchen formalen Grammatik wird der Zeichenvorrat auf runde Klammern sowie den Buchstaben x und y didaktisch reduziert, damit sich die Lösungen in der sich anschließenden Phase übersichtlicher gestalten und vergleichen lassen.
Anschließend fordert die Lehrkraft eine Kleingruppe auf, ihre Lösung zu präsentie-

ren. Nach der Präsentation lässt die Lehrkraft Terme von anderen Schülern nennen und sie von der vortragenden Schülergruppe in inGE mithilfe von Ableitungsbäumen überprüfen. Werden die Ableitungsbäume aufgrund der von den Schülern implementierten formalen Grammatik und unter Eingabe des genannten Termes nicht korrekt erzeugt, wird gemeinsam im Plenum die formale Grammatik überarbeitet. Mit dieser Phase werden die bisher erarbeiteten Grundlagen formaler Grammatiken gefestigt. Des Weiteren wird eine neue Sprachklasse, die der kontextfreien formalen Grammatiken, problemorientiert eingeführt.

Dass es sich bei dieser Klammersprache um eine neue Sprachklasse handelt, wissen die Schüler bis jetzt noch nicht. Deshalb teilt die Lehrkraft einen neuen Arbeitsauftrag aus, der die Schüler auffordert, einen endlichen Automaten für die Verarbeitung dieser Klammersprache in der Anwendung JFLAP zu entwickeln. Die Schüler entdecken selbstständig, dass für die Verarbeitung dieser Klammersprache unendlich viele Zustände benötigt werden und dass dieses ein Widerspruch zur Definition des endlichen Automaten ist. Des Weiteren erkennen die Schüler, dass diese mathematische Klammersprache zwar mit einer kontextfreien Grammatik erzeugt wird, nicht jedoch durch einen endlichen Automaten verarbeitet werden kann. Die Lehrkraft lässt diese entdeckten Erkenntnisse schriftlich fixieren und leitet mit einem Kurzvortrag zum Thema der Stunde „Die Verarbeitung der kontextfreien Sprache mit einem Kellerautomaten“ über. In dem Kurzvortrag führt die Lehrkraft den Kellerautomaten als Erweiterung des endlichen Automaten durch einen Kellerspeicher (engl. Stack) ein. Im Anschluss daran teilt die Lehrkraft das Arbeitsblatt 14 aus. In der ersten Aufgabe des Arbeitsblattes müssen die Schüler noch einmal die Arbeitsweise des Kellerautomaten mit JFLAP und anhand eines Beispiels nachvollziehen. Die zweite Aufgabe zeigt einen Ableitungsbaum, der aus einer formalen Grammatik und der Wortfolge $(x+(a+(a+b)))$ erzeugt wurde. Die Aufgabe fordert die Schüler auf, zuerst eine formale Grammatik für die Erzeugung dieses Termes und danach einen Kellerautomaten für die Verarbeitung dieser Terme zu implementieren. Die Aufgabe vertieft den Umgang mit der Implementierung formaler Grammatiken und verknüpft die bereits erworbenen Kenntnisse mit dem neuen Lerngegenstand des Kellerautomaten. In der letzten Phase dieser Doppelstunde lässt die Lehrkraft wieder eine Schülergruppe die Ergebnisse präsentieren und bespricht noch offene Fragen im Plenum.

### 4.4.9 L4 - 9. Doppelstunde - Unsere natürliche Sprache

| L4 Verlaufsplanung Stunde 17/18: Unsere natürliche Sprache | | | | |
|---|---|---|---|---|
| Zeit | Didaktische Funktion | Schüleraktivitäten | Methode / Sozialform | Medien |
| 10 min | Einstieg | Besprechen des heutigen Stundenthemas (Arbeitsblatt 15).<br>Einteilung der Schüler in 3-4'er Gruppen.<br>Austeilen der Puzzle-Kärtchen. | FU | Arbeitsblatt 15<br>Bastelbogen<br>inGE-Vorlage |
| 30 min | Erarbeitung | Schülergruppen bearbeiten selbstständig Arbeitsblatt 15. | GA | |
| 15 min | Ergebnissicherung | Lehrer bespricht Ergebnisse mit Gruppen gemeinsam.<br>Zusammenfassende Analyse wird mithilfe der Präsentation „Natürliche Sprachen und kontextfreie Grammatiken“ durchgeführt. | LV/UG | Präsentation |
| 25 min | Lerntest | Lerntest | sSA | |

Tabelle 4.12: L4 Verlaufsplanung Stunde 17/18: Unsere natürliche Sprache

Die letzte Doppelstunde dieser Unterrichtseinheit stellt den Bezug zu der natürlichen (deutschen) Sprache her. Die Lehrkraft teilt das Arbeitsblatt 15 und ein Puzzlespiel aus. Die Puzzleteile bestehen aus Terminalen, Non-Terminalen, Überführungspfeilen sowie einer Startregel. Danach teilt sie den Kurs in Gruppen zu je drei bis vier Schülern pro Gruppe auf. Die Aufgabe des ausgeteilten fünfzehnten Arbeitsblattes besteht im Zusammensetzen des Puzzles. Diese Methode soll die Schüler durch das Diskutieren über das Puzzle anregen, ihr erworbenes Wissen unter Beweis zu stellen. Des Weiteren eignet sich diese Methode dafür, sich ohne einen Rechner mit formalen Grammatiken auseinanderzusetzen und aus Diskussionen untereinander herauszufinden, wie das Puzzle zusammengesetzt werden müsste. Die Überprüfung dieser Grammatik erfolgt mit dem Plugin inGE. Die Schüler implementieren nach den Diskussionen innerhalb ihrer jeweiligen Gruppe diese formale Grammatik in inGE. Sie überprüfen, ob sich Ableitungsbäume mit den Wortfolgen des Arbeitsblattes und der von ihnen implementierten formalen Grammatiken bilden lassen. Die Lehrkraft hält einen abschließenden Lehrervortrag zum Thema „Natürliche Sprachen und kontextfreie Grammatiken“. Darüber hinaus thematisiert die Lehrkraft kontextsensitive Grammatiken und schließt mit den Sprachklassen der Chomsky-Hierarchie ab. Die verbleibende Zeit nutzt die Lehrkraft für die Durchführung eines Lerntests.

# Kapitel 5

# Evaluation des Einsatzes der neuen Unterrichtseinheit in der Sekundarstufe II

Die alte Unterrichtsreihe basierend auf dem Konzept von Seiffert wird im Folgenden mit der neuen Unterrichtsreihe nach IniK und unter Verwendung der selbstentwickelten Unterrichtssoftware hinsichtlich des Erreichens der Zielsetzungen des Rahmenplans verglichen. Die Daten basieren auf im Schuljahr 2013/14 durchgeführten Kursen an fünf Hamburger Gymnasien und einer Brandenburger Gesamtschule.

## 5.1 Hypothesen

Nach Entwicklung der Unterrichtssoftware inES und dem darin enthaltenen PlugIn inGE (vgl. Kapitel 4.2, 4.3) sowie der Ausarbeitung der neuen Unterrichtskonzeption zur Mensch-Maschine-Kommunikation (vgl. Kapitel 4.4) wurden folgende Hypothesen für die Untersuchung aufgestellt.

- **H1: Die Mensch-Maschine-Kommunikation mit gesprochener Sprache ist für SuS ein Phänomen aus ihrer Lebenswirklichkeit und motiviert sie, sich mit Informatik zu beschäftigen.**
- **H2: Mit den digitalen Lernumgebungen inES und inGE eignen sich die SuS theoretische Konzepte der Informatik handlungsorientiert an.**

- **H3: Das selbstständige Implementieren und Testen eigener Grammatiken fördert die Kreativität der SuS und ermöglicht verschiedenartige Projekte.**
- **H4: Mit einer zustandsorientierten Modellierung verstehen die SuS das theoretische Konzept eines Parsers.**
- **H5: Durch die Implementierung von formalen Grammatiken mithilfe verschiedener Darstellungsformen verstehen die SuS das theoretische Konzept einer formalen Grammatik.**
- **H6: Die strikte Visualisierung des gesamten Problemlöseprozesses unterstützt einen konstruktiven Umgang mit Fehlern.**
- **H7: Der Kontext Mensch-Maschine-Kommunikation mit gesprochener Sprache spricht SuS gleichermaßen an.**
- **H8: Die kontextorientierte Unterrichtseinheit zur Mensch-Maschine-Kommunikation mit gesprochener Sprache basierend auf IniK und unter Einsatz der Unterrichtssoftware inES hat einen pädagogischen Mehrwert.**

## 5.2 Forschungsdesign

Die aufgestellten Hypothesen implizieren einen Vergleich, weshalb einer quantitativen Forschungsmethode der Vorzug gegeben wurde. Da mit natürlichen Gruppen gearbeitet wurde, hat das Forschungsdesign einen quasi-experimentellen Charakter (vgl. [RSB10], S. 41).
Die Datenerhebung erfolgte durch einen standardisierten Fragebogen in einem Post-Test. Die Methode des Post-Testes wurde ausgewählt, weil die Schüler aufgrund der Bildungsplanvorgaben vor dem Zeitpunkt der Durchführung der Unterrichtseinheit keine Erfahrungen im Umgang mit den theoretischen Grundlagen der Informatik besitzen konnten. Die Elemente der theoretischen Informatik sind in den Bundesländern Hamburg und Brandenburg erst Lerngegenstand der Sekundarstufe II.
Die Untersuchungsobjekte, d. h. die Kursauswahl, erfolgte einmal durch einen Aufruf über eine Hamburger Informatik-Mailingliste sowie einer Anfrage in einem selbst gehaltenen Workshop, der im Rahmen einer Lehrertagung der GI im Februar 2013 in Berlin stattfand. Für die Erprobung der neuen Unterrichtseinheit erklärten sich insgesamt vier Lehrkräfte (drei Hamburger und ein Brandenburger Lehrer) bereit.

Drei Hamburger Lehrkräfte unterrichteten die alte Unterrichtseinheit. Die Auswahl mehrerer Lehrkräfte sollte die Beeinflussung der Ergebnisse durch die Persönlichkeit einer Lehrkraft reduzieren.
Die Durchführung der beiden Unterrichtseinheiten erfolgte zeitversetzt. Die neue Unterrichtseinheit wurde zu Beginn des ersten Schulhalbjahres 2013/2014 erprobt, während die alte Unterrichtseinheit im zweiten Schulhalbjahr durchgeführt wurde. Die ausgewählten Schulen sind räumlich und in einem Abstand von mehreren Kilometern voneinander getrennt, sodass davon ausgegangen werden konnte, dass die Schüler sich keine Ergebnisse des Fragebogens austauschen konnten.

## 5.3 Der Fragebogen

Der eingesetzte Fragebogen war teilstandardisiert, d. h. es wurden sowohl schriftliche Schüleraussagen als auch implementierte Projekte zur Auswertung mit herangezogen (vgl. Anhang C).
Der Fragebogen bestand aus vier Teilen. Im ersten Teil (P01-P06) wurden persönliche Daten abgefragt. Der zweite Teil des Fragebogens enthielt Aufgabenstellungen unterschiedlichen Schwierigkeitsgrades aus der Informatik, damit eine Beurteilung der Vergleichbarkeit beider Gruppen möglich wurde. Für die Untersuchung der Hypothesen, insbesondere der Hypothese H1 bis H7, wurden die Indikatoren des dritten Teils (F01-F32) herangezogen. Die Zugehörigkeit eines jeweiligen Indikators zu einer Variable wird in den sich anschließenden Unterkapiteln dargestellt. Damit die Schüler die zu untersuchenden Forschungsgegenstände nicht erkennen konnte, wurden die Indikatoren vermischt. Die Tabellen 5.1 bis 5.8 ordnen jeder Variablen ihre dazugehörigen Indikatoren (Items) zu. Die unterschiedlichen Ausprägungen eines Indikators wurden durch eine Ordinalskala erfasst, damit der jeweilige Indikator mit der größeren Merkmalsausprägung auch die größere Zahl erhielt und ein Vergleich der Merkmalsausprägungen mit größer/kleiner bzw. besser/schlechter möglich wurde (vgl. [Bor05], S.19). Die Skalen erfassten eine Ausprägung von „trifft nicht zu “ bis „triff vollkommen zu “ mit jeweils sechs Abstufungen. Damit konnte sichergestellt werden, dass die Schüler keine neutrale Position einnehmen konnten.
Im letzten Teil des Fragebogens wurden die in den jeweiligen Unterrichtseinheiten erworbenen Fachkenntnisse in Form von Prüfungsaufgaben abgefragt.

### 5.3.1 H1: Die Mensch-Maschine-Kommunikation mit gesprochener Sprache ist für SuS ein Phänomen aus ihrer Lebenswirklichkeit und motiviert sie, sich mit Informatik zu beschäftigen.

In beiden Unterrichtseinheiten wurden unterschiedliche Anwendungskontexte bearbeitet. Zum einen die Sprachübersetzung (alte Unterrichtseinheit) und zum anderen die Sprachdialogsysteme (neue Unterrichtseinheit). Der erste Teil der Hypothese zielt auf die Nähe des jeweiligen Anwendungskontextes zur Lebenswirklichkeit der Schüler ab. Mit dem zweiten Teil der Hypothese soll herausgefunden werden, inwieweit beide Anwendungskontexte dazu beitragen, dass sich die Motivation der Schüler zur Beschäftigung mit Informatik erhöht. Dementsprechend versuchten die Indikatoren (siehe Tabelle 5.1) des Fragebogens die verschiedenen Ausprägungen für Affekt und Interesse zu messen, denn nach Heckhausen und Heckhausen ist der Affekt eine primäre Bewertungsinstanz, die mit einem Objekt, z. B. einem Anwendungskontext, jeweils positiv oder negativ verknüpft sein kann. Dabei kommt auch dem Interesse eine besondere Bedeutung zu, denn Interesse ist eine besondere Motivationsform, „die durch die Ausrichtung auf einen bestimmten Gegenstand charakterisiert ist“ (vgl. [HH10], S. 367).

Tabelle 5.1: Lebenswirklichkeit und Motivation

| Variablen | Items | Anzahl |
|---|---|---|
| Lebenswirklichkeit | F12SQ002; F13SQ002; F15SQ004 F14SQ004; F16SQ003 | 5 |
| Motivation sich mit Informatik auch außerhalb der Schule zu beschäftigen. | F12SQ004; F13SQ004; F16SQ004 F17SQ003; F32SQ004 | 5 |

### 5.3.2 H2: Mit den digitalen Lernumgebungen inES und inGE eignen sich die SuS theoretische Konzepte der Informatik handlungsorientiert an.

Nach Jank und Meyer bedeutet Handlungsorientierung „[...] ein ganzheitlicher und schüleraktiver Unterricht, in dem die zwischen dem Lehrer und den Schülern verein-

barten Handlungsprodukte die Gestaltung des Unterrichtsprozesses leiten, sodass Kopf- und Handarbeit der Schüler in ein ausgewogenes Verhältnis zueinander gebracht werden können“ (Vgl. [JM11], S. 315). Deshalb sollte der Fragebogen Indikatoren für die Erfassung der Variablen Ganzheitlichkeit, Schüleraktivität, Zielgruppenorientierung und Reflexion ermöglichen.

Tabelle 5.2: Lernumgebung und handlungsorientierter Unterricht

| Variablen | Items | Anzahl |
|---|---|---|
| Ganzheitlichkeit | F17SQ004; F18SQ002; F19SQ002 F20SQ002; F21SQ002 | 5 |
| Aktivität der Lernenden | F18SQ003; F19SQ003; F20SQ003 F21SQ003; F22SQ003 | 5 |
| Reflexion | F18SQ004; F19SQ004; F20SQ004 F21SQ004; F22SQ004; F23SQ003 F29SQ001 | 7 |
| Zielgruppenorientierung | siehe Lebenswirklichkeit | |

### 5.3.3 H3: Das selbstständige Implementieren und Testen eigener Grammatiken fördert die Kreativität der SuS und ermöglicht verschiedenartige Projekte.

Interesse, Faszination, Erfolg und Spaß sind für Romeike wichtige Indikatoren für kreatives Arbeiten in der Informatik (vgl. [Rom08], S. 77). Für den zweiten Teil dieser Hypothese wurden die abgegebenen bzw. hochgeladenen Projekte (siehe Frage P01, des Fragebogens) der Schüler nach dem Kriterium der Vielfältigkeit ausgewertet.

Tabelle 5.3: Kreativität

| Variablen | Items | Anzahl |
|---|---|---|
| Kreativität | F27SQ001; F24SQ003; F22SQ002<br>F26SQ003; F27SQ004; F28SQ001<br>F28SQ004; F23SQ002; F29SQ003<br>F30SQ001; F30SQ003; F30SQ004<br>F31SQ001; F31SQ003; F32SQ002<br>F32SQ003; F24SQ001 | 17 |

### 5.3.4 H4: Mit einer zustandsorientierten Modellierung verstehen die SuS das theoretische Konzept eines Parsers.

Die quantitative Auswertung der Testaufgaben T01-T07 wurde zur Stützung dieser Hypothese mit einbezogen. Mit der Begriffsbeschreibung des Kellerautomaten sollte über die Testfragen T01-T07 hinaus untersucht werden, inwieweit die Schüler nach Durchführung der jeweiligen Unterrichtseinheiten in der Lage waren, den Begriff mit eigenen Worten und unter Verwendung der Fachsprache zu beschreiben. Es wurde keine exakte mathematisch-formale Definition erwartet. Für die Kodierung der eingegebenen Sätze wurde im Folgenden dieses Kategoriesystem zugrunde gelegt.

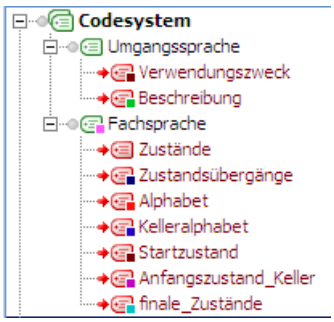



Abbildung 5.1: Codesystem Kellerautomat

### 5.3.5 H5: Durch die Implementierung von formalen Grammatiken mithilfe verschiedener Darstellungsformen verstehen die SuS das theoretische Konzept einer formalen Grammatik.

Analog zu H4 werden für H5 die Antworten der Testaufgabe T08 ausgewertet. Für die Auswertung der Testaufgabe T08 wurde das folgende Kategoriesystem zugrundegelegt.

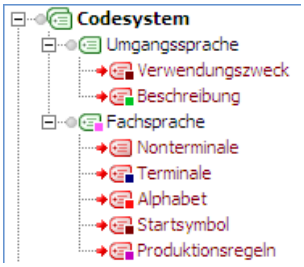



Abbildung 5.2: Codesystem Grammatik

### 5.3.6 H6: Die strikte Visualisierung des gesamten Problemlöseprozesses unterstützt einen konstruktiven Umgang mit Fehlern.

Die in Kapitel 3 dokumentierten und ausgewerteten Experteninterviews zeigten, dass in der bisherigen Unterrichtseinheit ein Nachteil in der fehlenden Visualisierung liegt. Mit den Indikatoren „Eigenständigkeit“ und „Softwareunterstützung“ wurde diese Hypothese dahingehend untersucht, ob die neue Unterrichtseinheit in Kombination mit der eingesetzten Lernsoftware diesen bisherigen Mangel beseitigt hat und damit einen effizienteren und konstruktiveren Umgang mit Fehlern besser als bisher ermöglichte.

Tabelle 5.4: Eigenständigkeit

| Variablen | Items | Anzahl |
|---|---|---|
| Eigenständigkeit | F27SQ002; F28SQ002; F29SQ002<br>F29SQ004; F30SQ002; F31SQ002<br>F31SQ004; F32SQ001 | 8 |
| Softwareunterstützung | F24SQ004; F25SQ004; F26SQ004<br>F27SQ003; F28SQ003; F25SQ001 | 6 |

### 5.3.7 H7: Der Kontext Mensch-Maschine-Kommunikation mit gesprochener Sprache spricht SuS gleichermaßen an.

Gendersensibler Unterricht soll Schülerinnen und Schüler gleichermaßen ansprechen. Deshalb wurden insbesondere die Hypothesen H2, H3 und H6 auf geschlechtsspezifische Unterschiede untersucht. Des Weiteren wurden nur die Gruppen betrachtet, bei denen der Unterricht mit einer vollständig funktionierenden Lernsoftware in-ES (neue Unterrichtseinheit) bzw. DrRacket (alte Unterrichtseinheit) durchgeführt werden konnte.

### 5.3.8 H8: Die kontextorientierte Unterrichtseinheit zur Mensch-Maschine-Kommunikation mit gesprochener Sprache basierend auf IniK und unter Einsatz der Unterrichtssoftware inES hat einen pädagogischen Mehrwert.

Für die Untersuchung dieser Hypothese wurden sowohl die Testaufgaben (T01-T07) als auch Selbsteinschätzungen zu den im Bildungsplan angeführten Kompetenzen herangezogen. Bei den Testaufgaben handelte es sich um Aufgaben aus dem Lehrbuch Informatik 5 (vgl. [HLS+10], S. 16, Aufg. 1; S. 18, Aufg. 9; S. 21, Aufg. 3). Sie untersuchten die Fachkenntnisse, die jeder Schüler nach der Behandlung der Themen beider Unterrichtsreihen haben sollte und orientierten sich an den Zielvorgaben des älteren Lehrplanes (K-Ziele) sowie des A-Heftes (A-Ziele), wie in Kapitel 3 ausführlich dargestellt. Diese Ziele sind (vgl. Kapitel 3):

K.1. vergleichen natürliche und formale Sprachen,

K.2. interpretieren unterschiedliche Darstellungen von Grammatiken,

K.3. klassifizieren Automatenmodelle, Sprachen und Grammatiken,

K.4. unterscheiden Rekursion und Iteration,

K.5. analysieren Sätze einer Sprache, entwickeln die zu ihrer Beschreibung notwendigen Grammatikelemente und stellen diese dar,

K.6. setzen Rekursion fachgerecht ein,

A.1. beschreiben grundsätzliche Schwierigkeiten maschineller Sprachverarbeitung,

A.2. erstellen einen Parsebaum zu einem gegebenen Satz und einer gegebenen Grammatik,

A.3. vergleichen unterschiedliche Vorgehensweisen zur Realisierung eines Parsers (Abbildung der Grammatikproduktionen als Liste von Listen sowie Tiefen- oder Breitensuche, Funktionen zur Auflösung von Nonterminalen).

Die Messung der ausgebildeten Fachkompetenzen des Bildungsplanes wurde mithilfe von Indikatoren (siehe Tabelle 5.5 bis 5.8) durch den Fragebogen (vgl. Anhang B, F01-F32) realisiert.
Sie wurden aus dem aktuell gültigen Bildungsplan unter Berücksichtigung der Operatorenliste des A-Heftes (vgl. [Ros14], S. 103) eigenständig entwickelt.

Tabelle 5.5: Informatiksysteme gestalten

| Kompetenzen | Items | Anzahl |
|---|---|---|
| *Die Schülerinnen und Schüler* | | |
| wählen bei der Gestaltung von Informatiksystemen passende Algorithmen aus. | F01SQ001; F02SQ001; F03SQ001; F04SQ001; F05SQ001 | 5 |
| implementieren Modelle sowohl mit Hilfe grafischer Entwicklungsumgebungen als auch mit einer höheren Programmiersprache. | F06SQ001; F07SQ001; F08SQ001 | 3 |
| vergleichen die Eignung verschiedener Modellierungsansätze für unterschiedliche Problemstellungen und wählen für gegebene Problemstellungen einen geeigneten Ansatz aus. | F09SQ001; F10SQ001; F11SQ001 F12SQ001; F14SQ001; F15SQ001 F13SQ001 | 7 |
| implementieren einen Interpreter für eine selbst entwickelte einfache formale Sprache. | F16SQ001; F17SQ001; F18SQ001 F19SQ001; F20SQ001; F21SQ001 F22SQ001; F23SQ001 | 8 |

Tabelle 5.6: Informatiksysteme analysieren und verstehen

| Kompetenzen | Items | Anzahl |
|---|---|---|
| *Die Schülerinnen und Schüler* | | |
| untersuchen Algorithmen und vergleichen sie hinsichtlich ihrer Effizienz und Qualität der Lösung. | F01SQ002; F02SQ002; F03SQ002 F04SQ002; F05SQ002 | 5 |
| analysieren Informatiksysteme hinsichtlich der zugrundeliegenden Strukturen und Prozesse sowie der Aufgabenteilung zwischen Mensch und Maschine. | F06SQ002; F07SQ002; F08SQ002 F09SQ002; F10SQ002; F11SQ002 F14SQ002; F15SQ002 | 8 |

Tabelle 5.7: Darstellen und Interpretieren

| Kompetenzen | Items | Anzahl |
|---|---|---|
| *Die Schülerinnen und Schüler* | | |
| beschreiben Modelle und Algorithmen sowohl grafisch als auch verbal. | F01SQ003; F02SQ003; F03SQ003 F04SQ003 | 4 |
| unterscheiden natürliche von formalen Sprachen. | F05SQ003; F06SQ003; F07SQ003 | 3 |
| differenzieren formale Sprachen hinsichtlich ihrer Interpretierbarkeit. | F08SQ003; F09SQ003; F10SQ003 F11SQ003 | 4 |

Tabelle 5.8: Begründen und Bewerten

| Kompetenzen | Items | Anzahl |
|---|---|---|
| *Die Schülerinnen und Schüler* | | |
| bewerten, vergleichen und begründen informatische Modellierungen in Bezug auf ihren Anwendungskontext und formale Kriterien. | F01SQ004; F02SQ004; F04SQ004 F03SQ004 | 4 |
| bewerten die prinzipielle und praktische Realisierbarkeit von Informatiksystemen, ohne dass dabei die Mathematik im Zentrum steht. | F05SQ004; F06SQ004; F07SQ004 F08SQ004; F09SQ004; F10SQ004 F12SQ003; F11SQ004 | 8 |
| bewerten die Auswirkungen von Informatiksystemen auf die betroffenen Menschen. | F13SQ003; F14SQ003; F15SQ003 | 3 |

## 5.4 Zusammensetzung der Stichprobe

Den Fragebogen füllten insgesamt 117 Schüler ($n = 117$) vollständig aus. Von einer Gesamtschule des Landes Brandenburg nahmen 31 Schüler an der Studie teil. Die 86 übrigen Schüler verteilten sich auf fünf Hamburger Gymnasien ($n = 86$). Für die Untersuchung der Hypothesen H1 bis H8 wurden folgende Gruppen konstruiert:

- Die Experimentalgruppe $E$, deren Unterricht nach dem neuen informationsorientierten Konzept durchgeführt wurde. Sie bestand aus 35 Schülern.
- Die Kontrollgruppe $K$, deren Unterricht nach dem traditionellen anwendungsorientierten Konzept durchgeführt wurde. Sie bestand aus 36 Schülern.

Bei der Experimentalgruppe $E$ handelte es sich um zwei Kurse eines Gymnasiums, die keine technischen Probleme hatten und die Lernsoftware inES und inGE in vollem Umfang nutzen konnten.
Innerhalb der Untersuchung aller Hypothesen wurden zusätzlich folgende Gruppen betrachtet:

- Die zur Experimentalgruppe gehörenden Kurse $E^1$ ($n_{E^1} = 15$) und $E^2$ ($n_{E^2} = 20$).
- Die zur Kontrollgruppe gehörenden Kurse $K^1$ ($n_{K^1} = 7$), $K^2$ ($n_{K^2} = 16$) und $K^3$ ($n_{K^3} = 13$).
- Die Gruppe $D$ mit ($n_D = 15$), welche die Software nur eingeschränkt nutzen konnte. Diese Gruppe arbeitete mit Apple iMac-Rechnern, die mit einer Windows 7 Installation arbeiteten. Treiberbedingt konnte hier die Spracherkennung und Sprachausgabe nicht genutzt werden.
- Die Gruppe $B$ der Brandenburger Gesamtschule mit 31 Schülern, deren Unterricht ebenfalls nach dem neuen informationsorientierten Konzept durchgeführt wurde. Die Gruppe $B$ konnte die Software wegen der eingeschränkten Benutzerrechte nicht vollständig nutzen. Das PlugIn inGE war nicht einsetzbar und die Software inES stürzte des Öfteren ab.
- Die Gruppe $H$ mit ($n_H = 71$) aller Hamburger Schüler unabhängig davon, welches Unterrichtskonzept zum Einsatz kam.
- $E^w$, die Experimentalgruppe mit $n_{E^w} = 8$ Schülerinnen.
- $E^m$, die Experimentalgruppe mit $n_{E^m} = 26$ Schülern.
- $K^w$, die Kontrollgruppe mit $n_{K^w} = 16$ Schülerinnen.
- $K^m$, die Kontrollgruppe mit $n_{K^m} = 20$ Schülern.

Bei den Informatikkursen $E^1$ und $E^2$ der Experimentalgruppe handelt es sich in beiden Fällen um Kurse auf erhöhtem Anforderungsniveau. Insbesondere sei darauf hingewiesen, das der Kurs $E^1$ vom Verfasser dieser Arbeit selbst unterrichtet wurde. $K^1$, eine Teilgruppe von $K$ war ein Kurs auf grundlegendem Anforderungsniveau. Dieser Kurs fand an einem Gymnasium statt, das selbst kein profilgebendes Fach

Informatik anbietet. Schüler wählen diesen Kurs bewußt im Rahmen ihrer freien Wahlmöglichkeiten und sind vermutlich an Informatik stärker interessiert als andere.

## 5.5 Voruntersuchung

Die Voruntersuchung hatte das Ziel, das Leistungsniveau und die Einstellung der Schüler der Kontrollgruppe $K$ mit denen der Experimentalgruppe $E$ zu vergleichen. Hierfür wurden die Informatik-Vornote, die subjektive Einschätzung des Interesses am Informatikunterricht und die Ergebnisse eines Eingangstestes als Indikatoren herangezogen.

Bei der Interpretation der Ergebnisse wird im Folgenden von

- hoch signifikanten Unterschieden gesprochen, wenn die Nullhypothese, die in der Übereinstimmung der Häufigkeitsverteilungen der Stichproben besteht, mit einer Irrtumswahrscheinlichkeit abgelehnt wird, die kleiner oder gleich $0,1\%$ ist;
- sehr signifikanten Unterschieden gesprochen, wenn die Irrtumswahrscheinlichkeit kleiner oder gleich $1\%$ ist;
- signifikanten Unterschieden gesprochen, wenn die Irrtumswahrscheinlichkeit kleiner oder gleich $5\%$ ist.

Da einige Zellen der dazugehörigen Kontingenztabellen Werte enthalten, die kleiner gleich fünf sind, wurde für die Überprüfung dieser Hypothesen der im Vergleich zum Chi-Quadrat-Test genauere Fisher-Test benutzt.
Dabei kann die Anzahl der Schüler in den einzelnen Gruppen differieren, weil manche Schüler die entsprechenden Fragen des Fragebogens nicht bearbeitet haben. Bei der jeweiligen Untersuchung wurden nur vollständig beantwortete Fragen innerhalb eines Testes berücksichtigt.

### 5.5.1 Die Informatik-Vornote der Schüler

Für eine faire Leistungsbeurteilung der eingeteilten Gruppen wurde zunächst die Informatik-Vornote der Schüler abgefragt. Die vergleichenden Ergebnisse sind in den folgenden Kontingenztabellen und den dazugehörigen Diagrammen dargestellt.

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 12 | 8 | 6 | 6 | 1 | 6 | 1 | 4 |
| 2 | 13 | 13 | 5 | 8 | 0 | 4 | 9 | 2 |
| 3 | 7 | 10 | 4 | 3 | 4 | 4 | 2 | 6 |
| 4 | 1 | 3 | 0 | 1 | 2 | 1 | 0 | 2 |
| 5 | 2 | 2 | 0 | 2 | 0 | 1 | 1 | 1 |
| $n$ | $n_E = 35$ | $n_K = 36$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 13$ | $n_D = 15$ |

Tabelle 5.9: Kontingenztabelle der Informatik-Vornoten

| | $H$ | $B$ |
|---|---|---|
| 1 | 20 | 3 |
| 2 | 26 | 13 |
| 3 | 17 | 13 |
| 4 | 4 | 2 |
| 5 | 4 | 0 |
| $n$ | $n_H = 71$ | $n_B = 31$ |

Tabelle 5.10: Kontingenztabelle der Informatik-Vornoten für Hamburg und Brandenburg

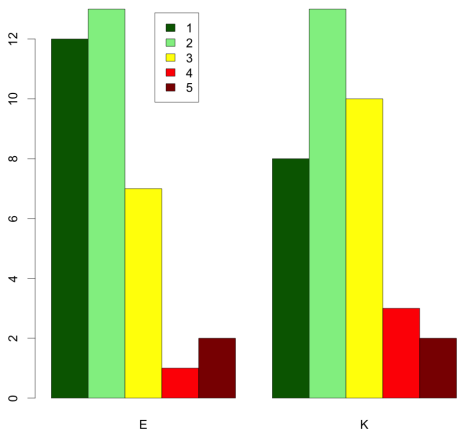



Abbildung 5.3: Vergleich $E$ mit $K$

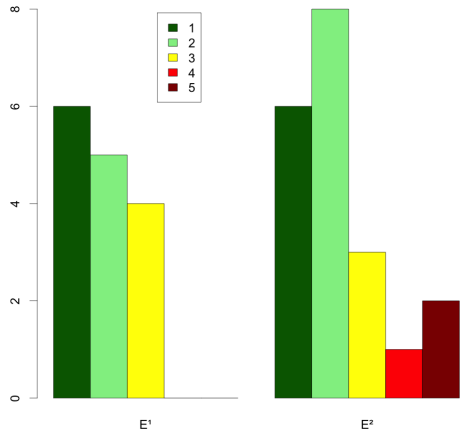



Abbildung 5.4: Vergleich $E^1$ mit $E^2$

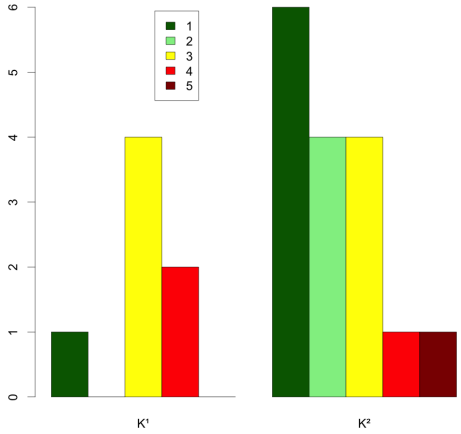



Abbildung 5.5: Vergleich $K^1$ mit $K^2$

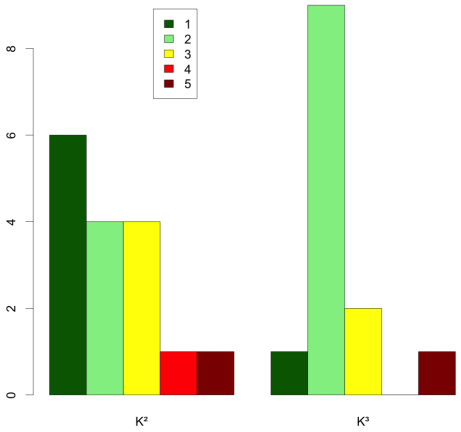



Abbildung 5.6: Vergleich $K^2$ mit $K^3$

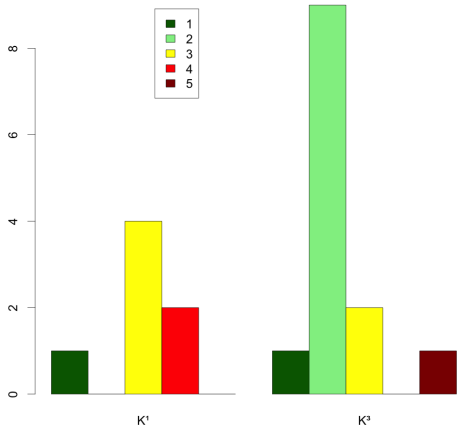



Abbildung 5.7: Vergleich $K^1$ mit $K^3$

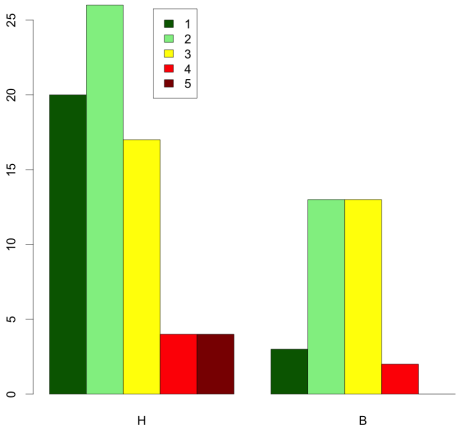



Abbildung 5.8: Vergleich $H$ mit $B$

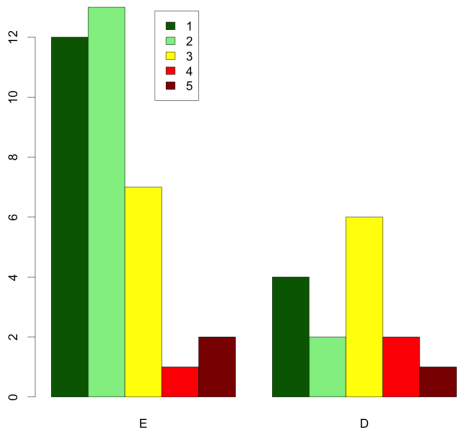



Abbildung 5.9: Vergleich $E$ mit $D$

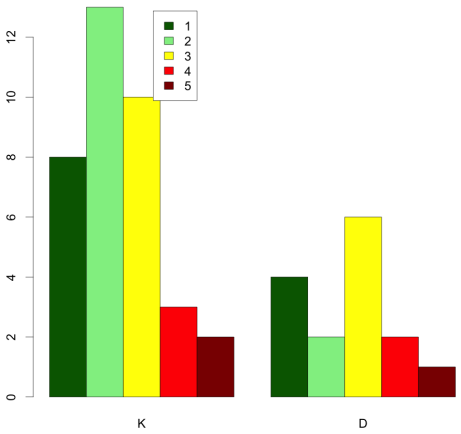



Abbildung 5.10: Vergleich $K$ mit $D$

Für alle Kontingenztabellen lieferte der Fisher-Test einen $p$-Wert von $p$ größer als $0,05$, d. h. die Gruppen unterscheiden sich hinsichtlich der Informatik-Vornote nicht signifikant und waren für die Untersuchung dementsprechend vergleichbar.

### 5.5.2 Interesse am Unterrichtsfach Informatik

Für die Einschätzung des Interesses am Unterrichtsfach Informatik konnten die Schüler aus einer sechsstufigen Skala von 1 (kein Interesse) bis 6 (sehr großes Interesse) auswählen. Die Ergebnisse sind in der folgenden Kontingenztabelle und in den darauffolgenden Diagrammen dargestellt.

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 5 | 0 | 1 | 1 | 3 | 1 | 3 |
| 2 | 4 | 6 | 2 | 2 | 3 | 1 | 2 | 2 |
| 3 | 6 | 2 | 1 | 5 | 2 | 0 | 0 | 2 |
| 4 | 7 | 6 | 4 | 3 | 0 | 3 | 3 | 4 |
| 5 | 6 | 3 | 2 | 4 | 0 | 2 | 1 | 0 |
| 6 | 11 | 14 | 6 | 5 | 1 | 7 | 6 | 4 |
| $n$ | $n_E = 35$ | $n_K = 36$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 13$ | $n_D = 15$ |

Tabelle 5.11: Kontingenztabelle des Informatikinteresses

| | $H$ | $B$ |
|---|---|---|
| 1 | 6 | 0 |
| 2 | 10 | 5 |
| 3 | 8 | 5 |
| 4 | 13 | 8 |
| 5 | 9 | 11 |
| 6 | 25 | 2 |
| $n$ | $n_H = 71$ | $n_B = 31$ |

Tabelle 5.12: Kontingenztabelle des Informatikinteresses für Hamburg und Brandenburg

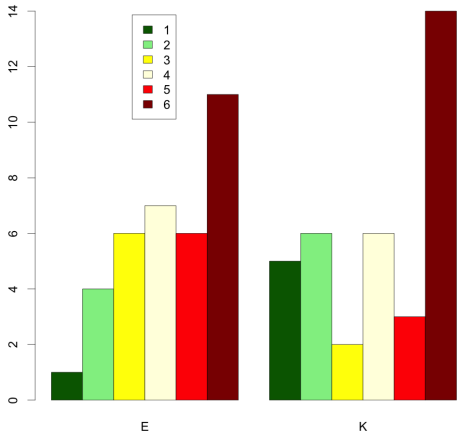



Abbildung 5.11: Vergleich $E$ mit $K$

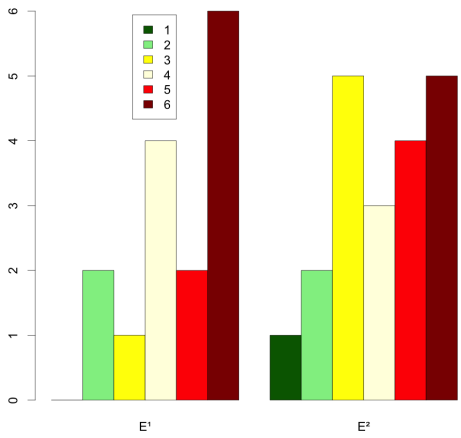



Abbildung 5.12: Vergleich $E^1$ mit $E^2$

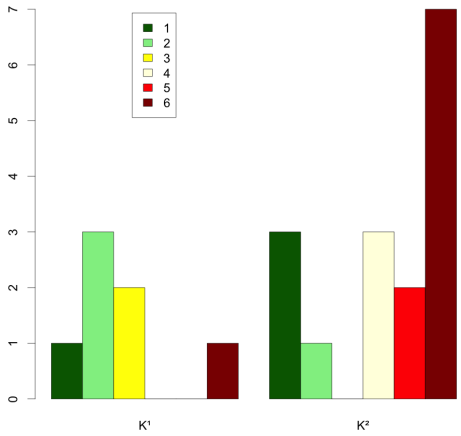



Abbildung 5.13: Vergleich $K^1$ mit $K^2$

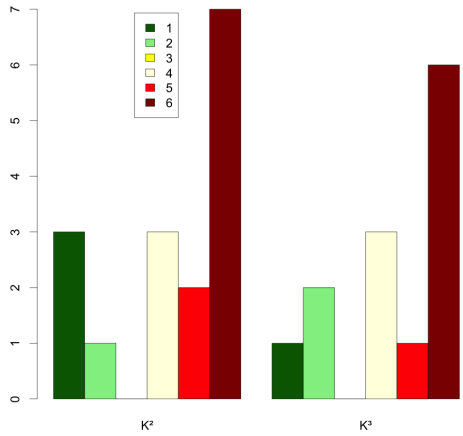



Abbildung 5.14: Vergleich $K^2$ mit $K^3$

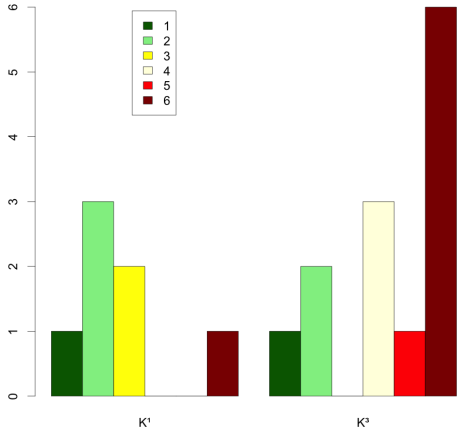



Abbildung 5.15: Vergleich $K^1$ mit $K^3$

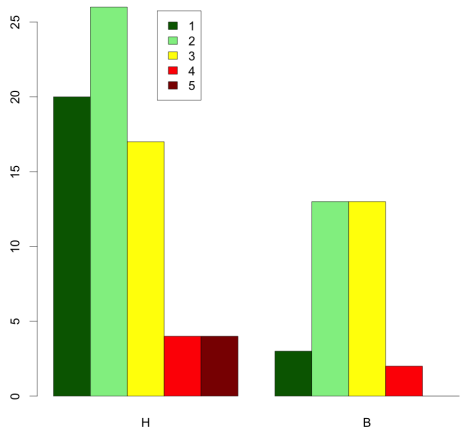



Abbildung 5.16: Vergleich $H$ mit $B$

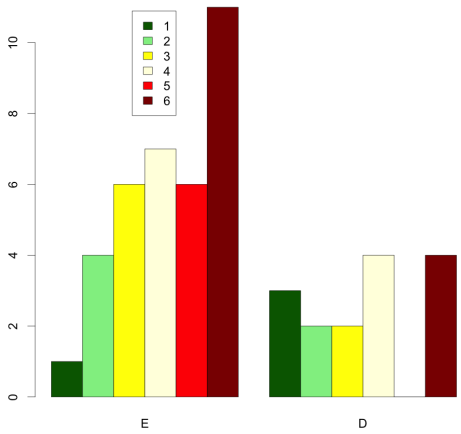



Abbildung 5.17: Vergleich $E$ mit $D$

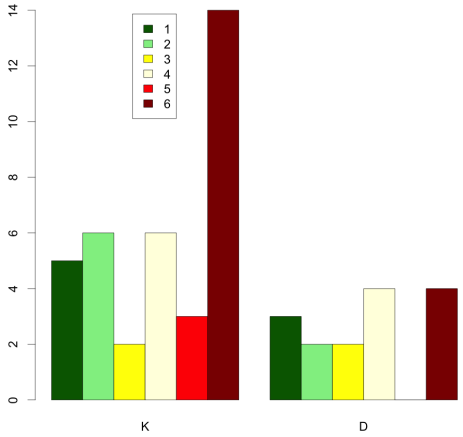



Abbildung 5.18: Vergleich $K$ mit $D$

Bezüglich des Interesses am Unterrichtsfach Informatik gibt es zwischen der Experimentalgruppe $E$ und der Kontrollgruppe $K$ keinen signifikanten Unterschied. Werden jedoch die Unterschiede innerhalb der beiden Gruppen untersucht, dann fällt auf, dass zwischen $K^1$ und $K^2$ bzw. zwischen $K^1$ und $K^3$ ein signifikanter Unterschied besteht, d. h. die Gruppe $K^1$ enthält Schüler, die sich deutlich weniger für das Fach interessieren. Des Weiteren sind die Hamburger Schüler, die Gruppe $H$ dieser Stichprobe, deutlich weniger an Informatik interessiert als die Brandenburger Schüler (Gruppe $B$).

### 5.5.3 Fehlzeiten der Schüler in den Gruppen

| Anzahl Fehlzeiten | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ |
|---|---|---|---|---|---|---|---|---|
| 0-10 | 24 | 34 | 12 | 12 | 7 | 15 | 12 | 13 |
| 10-20 | 4 | 1 | 2 | 2 | 0 | 1 | 0 | 2 |
| 20-30 | 3 | 1 | 1 | 2 | 0 | 0 | 1 | 0 |
| 30-40 | 2 | 0 | 0 | 2 | 0 | 0 | 0 | 0 |
| 40-50 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 50-60 | 2 | 0 | 0 | 2 | 0 | 0 | 0 | 0 |
| 60-70 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 70-80 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 80-90 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 90-100 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $n$ | $n_E = 35$ | $n_K = 36$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 13$ | $n_D = 15$ |

Tabelle 5.13: Kontingenztabelle der Fehlzeiten

| Anzahl Fehlzeiten | $H$ | $B$ |
|---|---|---|
| 0-10 | 65 | 25 |
| 10-20 | 5 | 3 |
| 20-30 | 1 | 2 |
| 30-40 | 0 | 0 |
| 40-50 | 0 | 0 |
| 50-60 | 0 | 0 |
| 60-70 | 0 | 0 |
| 70-80 | 0 | 0 |
| 80-90 | 0 | 0 |
| 90-100 | 0 | 0 |
| $n$ | $n_H = 71$ | $n_B = 30$ |

Tabelle 5.14: Kontingenztabelle der Fehlzeiten für Hamburg und Brandenburg

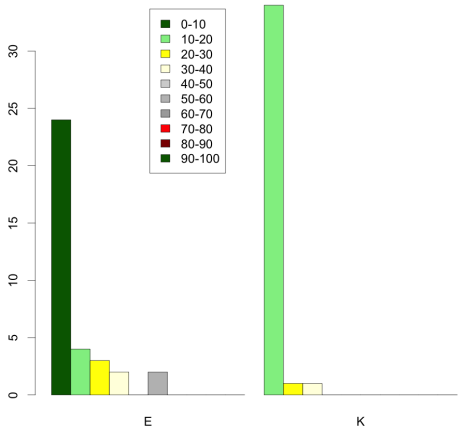



Abbildung 5.19: Vergleich $E$ mit $K$

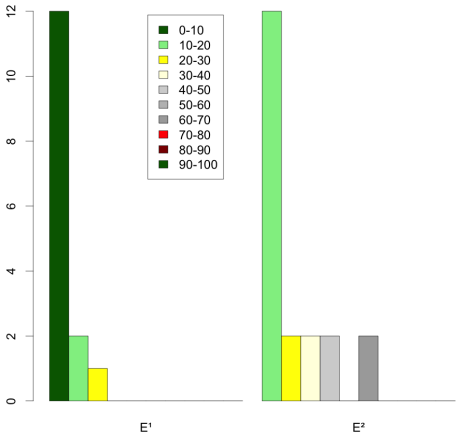



Abbildung 5.20: Vergleich $E^1$ mit $E^2$

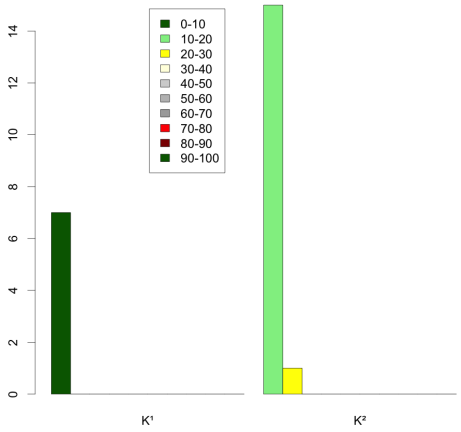



Abbildung 5.21: Vergleich $K^1$ mit $K^2$

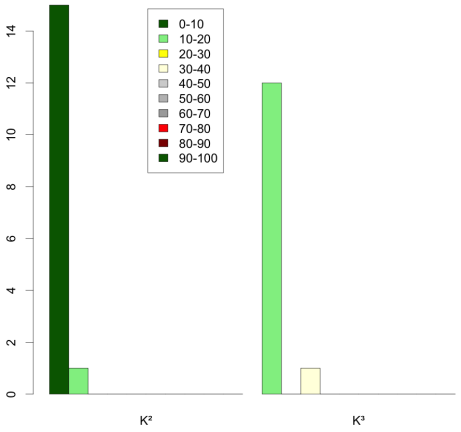



Abbildung 5.22: Vergleich $K^2$ mit $K^3$

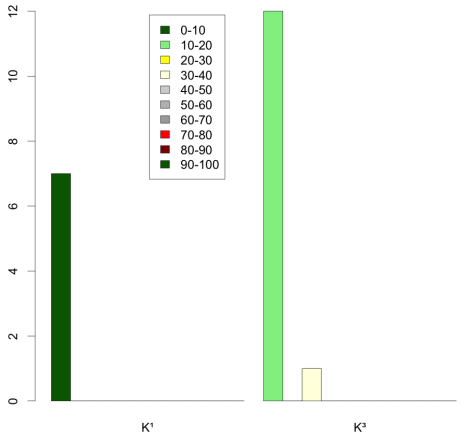



Abbildung 5.23: Vergleich $K^1$ mit $K^3$

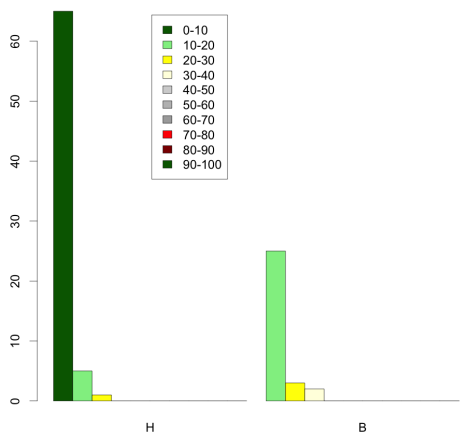



Abbildung 5.24: Vergleich $H$ mit $B$

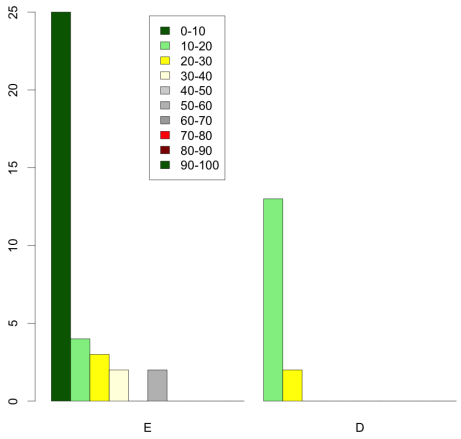



Abbildung 5.25: Vergleich $E$ mit $D$

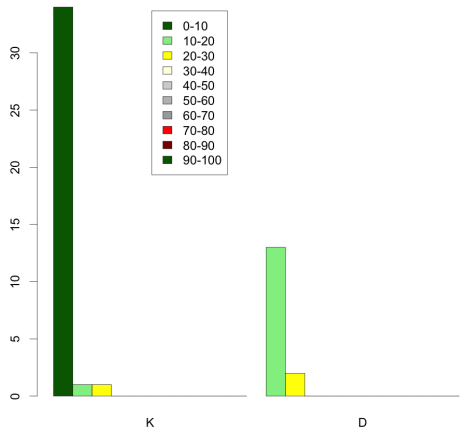



Abbildung 5.26: Vergleich $K$ mit $D$

Die alte Unterrichtseinheit umfasste i. d. R. vierundsechzig Unterrichtsstunden, während die neue Unterrichtseinheit bezüglich der in Kapitel 5.3.8 untersuchten Ziele mit zwanzig Unterrichtsstunden auskam. Dadurch ergibt sich hinsichtlich der Fehlzeiten ein signifikanter Unterschied zwischen den Gruppen $E$ und $K$, d. h. die Schüler der Experimentalgruppe haben signifikant häufiger gefehlt. Innerhalb der Experimentalgruppe $E$ gab es zwischen den Gruppen $E^1$ und $E^2$ keine signifikanten Unterschiede. Das Gleiche gilt für die Teilgruppen $K^1$, $K^2$ und $K^3$ der Kontrollgruppe sowie zwischen den Hamburger (Gruppe $H$) und den Brandenburger (Gruppe $B$) Schülern.

### 5.5.4 Probleme mit der Software inES bzw. inGE

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ | $B$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 4 | 5 | 1 | 3 | 3 | 1 | 1 | 8 | 6 |
| 3 | 7 | 2 | 1 | 6 | 1 | 1 | 0 | 5 | 9 |
| 4 | 21 | 21 | 10 | 11 | 1 | 10 | 10 | 2 | 14 |
| 5 | 2 | 8 | 2 | 0 | 2 | 4 | 2 | 0 | 2 |
| 6 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $n$ | $n_E = 35$ | $n_K = 36$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 13$ | $n_D = 15$ | $n_B = 31$ |

Tabelle 5.15: Kontingenztabelle technischer Probleme

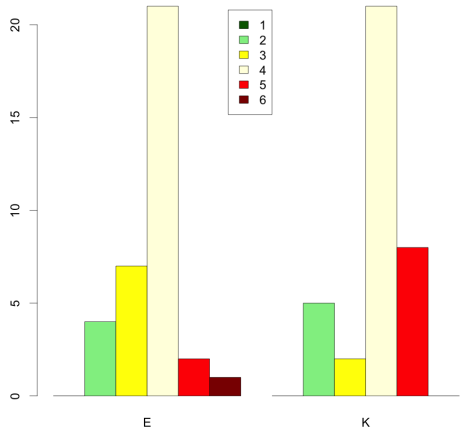



Abbildung 5.27: Vergleich $E$ mit $K$

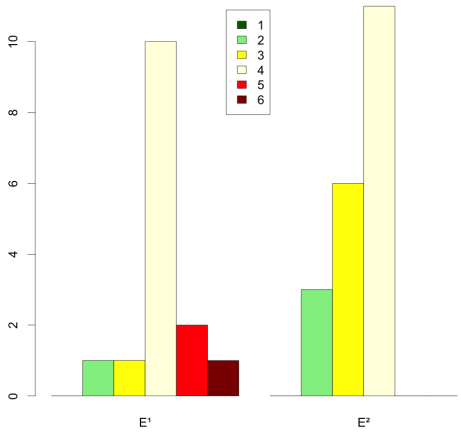



Abbildung 5.28: Vergleich $E^1$ mit $E^2$

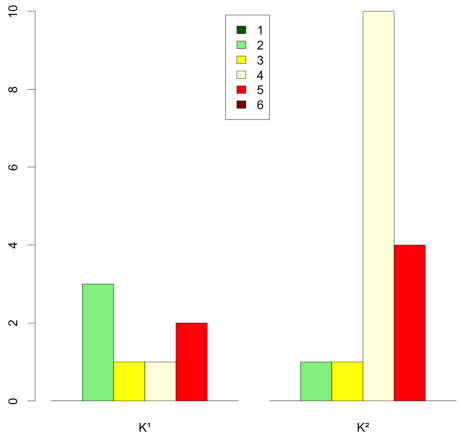



Abbildung 5.29: Vergleich $K^1$ mit $K^2$

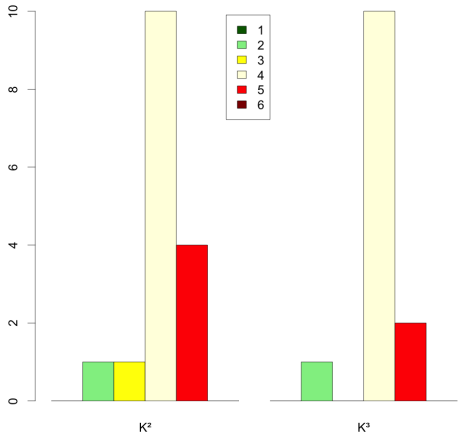



Abbildung 5.30: Vergleich $K^2$ mit $K^3$

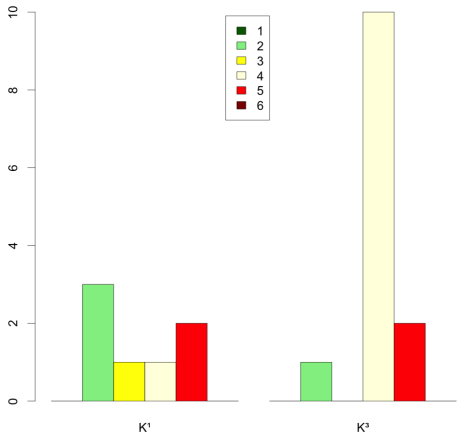



Abbildung 5.31: Vergleich $K^1$ mit $K^3$

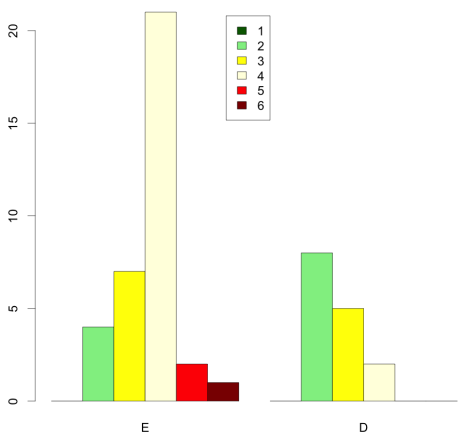



Abbildung 5.32: Vergleich $E$ mit $D$

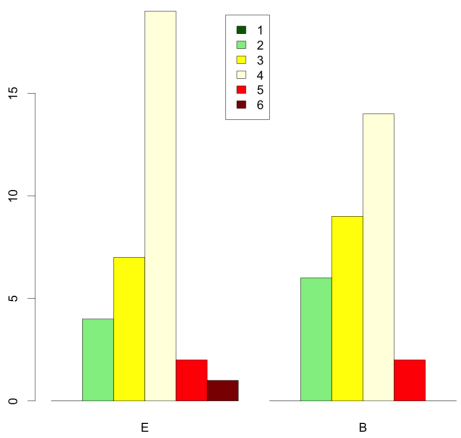



Abbildung 5.33: Vergleich $E$ mit $B$

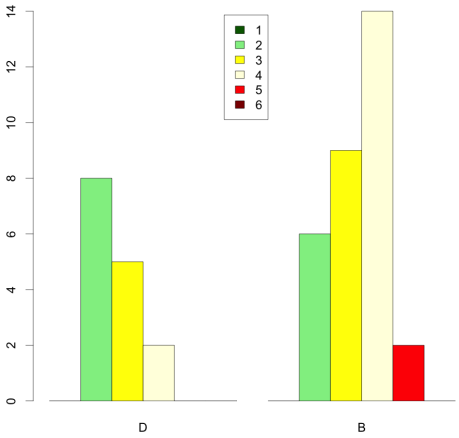



Abbildung 5.34: Vergleich $D$ mit $B$

Zwischen den Gruppen $E$ und $K$, innerhalb der Gruppen $K_1$, $K_2$ und $K_3$ sowie zwischen $E_1$ und $E_2$ gibt es keine signifikanten Unterschiede. Die $p$-Werte des gerechneten Fisher-Tests liegen mit $p$ größer als $0,05$ jeweils über dem Signifikanzniveau. Die Probleme mit der Software inES bzw. inGE in der Gruppe $D$ lassen sich aus der Kontingenztabelle (vgl. Tab. 5.15) und den dazugehörigen Grafiken (vgl. Abb. 5.32 und Abb. 5.34) deutlich erkennen, denn es gibt im Vergleich dieser Gruppe mit allen anderen Gruppen signifikante Unterschiede. Interessant ist, dass, obwohl die Brandenburger Schüler, die Gruppe $B$, die Software ebenfalls nur eingeschränkt nutzen konnten, die Ergebnisse sich hinsichtlich der technischen Dimension von denen der Gruppe $E$ nicht signifikant unterscheiden. Entweder haben die Probanden den Fragebogen generell zu positiv beantwortet, weil sie z. B. mit der Software inES sehr gern arbeiteten, oder die technischen Einschränkungen hatten letzten Endes doch nicht die Störwirkung auf die zu verrichtende Arbeit wie etwa bei der Gruppe $D$. Die technischen Schwierigkeiten betrafen bei der Gruppe $B$ im wesentlichen den Einsatz des Plugins inGE, weswegen die vertiefenden Aspekte der formalen Sprachen und der verschiedenen Automatenklassen überwiegend nur theoretisch besprochen werden konnten.

### 5.5.5 Gesamtergebnisse des Eingangstestes

Für jede Frage (vgl. P201, P301 und P401) konnte ein Schüler einen Punkt bekommen. Die folgenden Kontingenztabellen zeigen die Ergebnisse der Eingangsaufgaben aller Gruppen.

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ |
|---|---|---|---|---|---|---|---|---|
| 1 | 17 | 4 | 10 | 7 | 1 | 2 | 1 | 7 |
| 2 | 12 | 12 | 3 | 9 | 5 | 2 | 5 | 6 |
| 3 | 2 | 15 | 0 | 2 | 0 | 8 | 7 | 1 |
| $n$ | $n_E = 31$ | $n_K = 31$ | $n_{E^1} = 13$ | $n_{E^2} = 18$ | $n_{K^1} = 6$ | $n_{K^2} = 12$ | $n_{K^3} = 13$ | $n_D = 14$ |

Tabelle 5.16: Kontingenztabelle der Eingangsaufgaben der Hamburger Gruppen

| | $H$ | $B$ |
|---|---|---|
| 1 | 21 | 10 |
| 2 | 24 | 7 |
| 3 | 17 | 1 |
| $n$ | $n_H = 71$ | $n_B = 18$ |

Tabelle 5.17: Kontingenztabelle der Eingangsaufgaben für Hamburg und Brandenburg

Die folgende Grafiken stellen die Ergebnisse der jeweiligen Gruppenvergleiche dar.

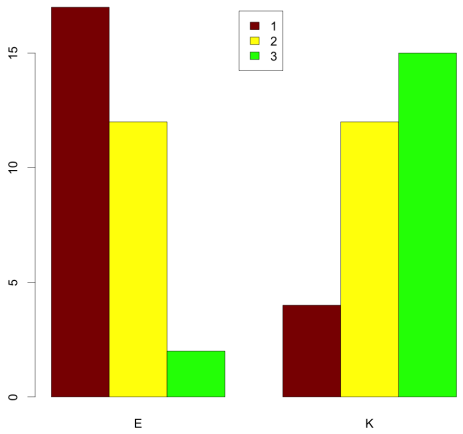



Abbildung 5.35: Vergleich $E$ mit $K$

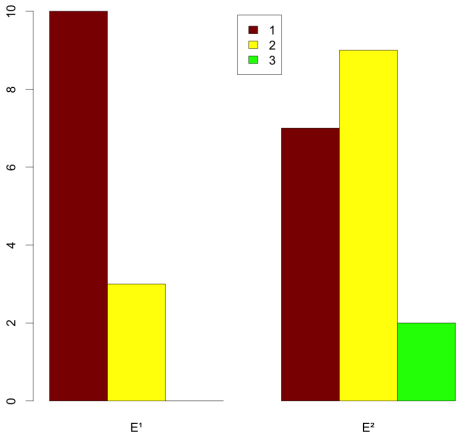



Abbildung 5.36: Vergleich $E^1$ mit $E^2$

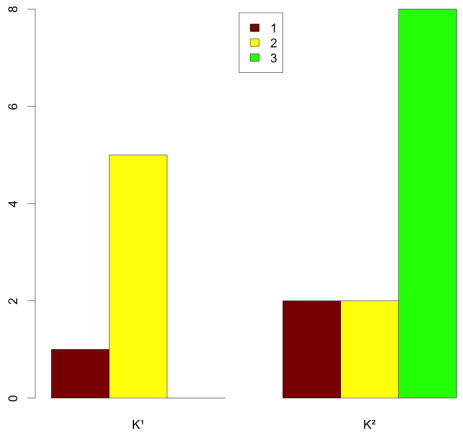



Abbildung 5.37: Vergleich $K^1$ mit $K^2$

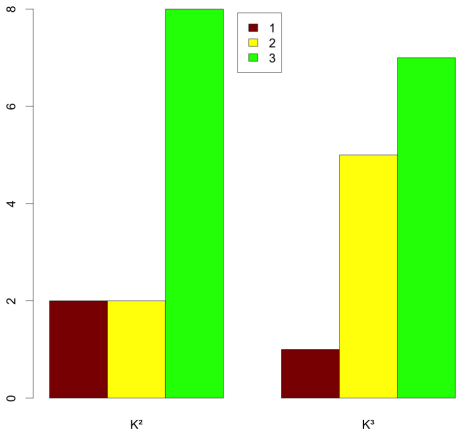



Abbildung 5.38: Vergleich $K^2$ mit $K^3$

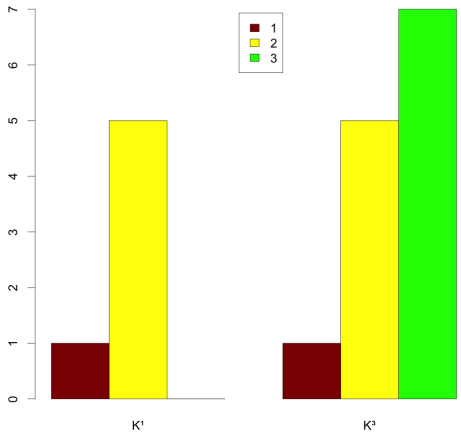



Abbildung 5.39: Vergleich $K^1$ mit $K^3$

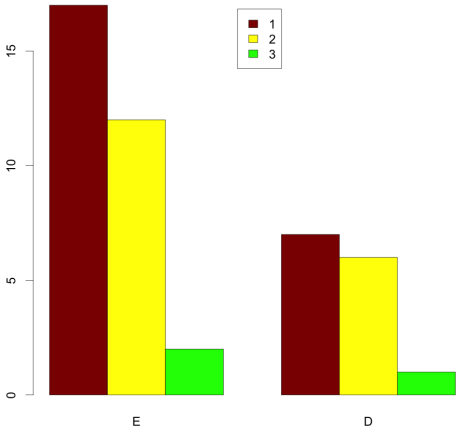



Abbildung 5.40: Vergleich $E$ mit $D$

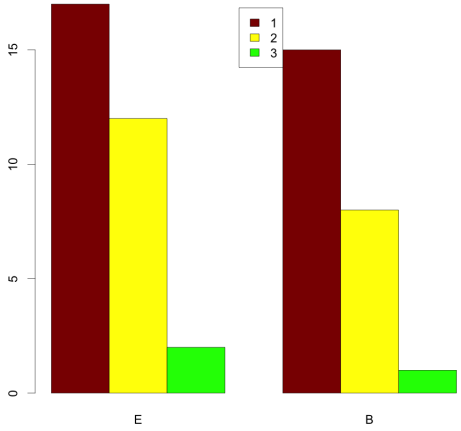



Abbildung 5.41: Vergleich $E$ mit $B$

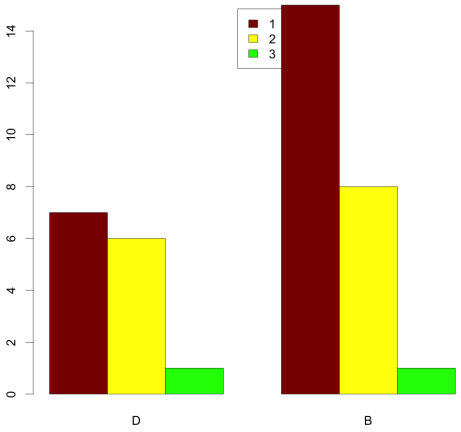



Abbildung 5.42: Vergleich $D$ mit $B$

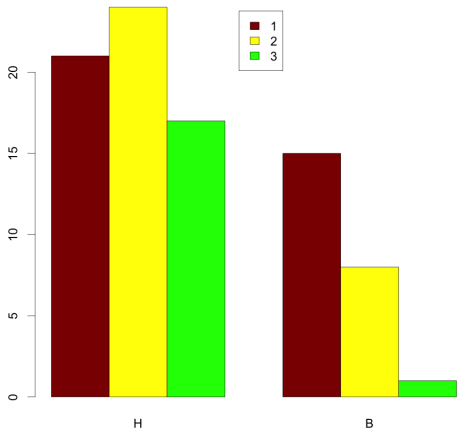



Abbildung 5.43: Vergleich $H$ mit $B$

Die Ergebnisse des Eingangstestes unterscheiden sich zwischen $E$ und $K$ sehr signifikant zugunsten von $K$, d. h. der gerechnete Fisher-Test ergibt einen $p$-Wert von $p$ kleiner gleich $0,01$. Offensichtlich ist die Kontrollgruppe $K$ deutlich leistungsstärker als die Experimentalgruppe $E$. Innerhalb der Experimentalgruppe $E$ treten keine signifikanten Unterschiede auf. In der Kontrollgruppe $K$ fällt auf, dass die Gruppe $K^1$ im Vergleich mit $K^2$ und $K^3$ signifikant schlechter ist. Zwischen den Hamburger Schülern und den Brandenburger Schülern besteht kein signifikanter Unterschied.

## 5.6 Hypothesentests

Bei der Auswertung des Fragebogens ging es im Kern um den Vergleich der Ergebnisse der Experimentalgruppe $E$ und der Kontrollgruppe $K$. Zusätzlich wurden die Ergebnisse der beiden Kurse $E^1$ und $E^2$ der Experimentalgruppe miteinander verglichen, um gegebenenfalls signifikante Unterschiede zu erkennen. Aus dem gleichen Grund wurden die Kurse $K^1$, $K^2$ und $K^3$ der Kontrollgruppe paarweise untereinander verglichen. Um zu überprüfen, in wie weit Probleme mit der Software inES bzw. inGE die Ergebnisse beeinflusst haben könnten, wurden des Weiteren die Gruppe $E$ mit $D$, $K$ mit $D$, $E$ mit $B$ sowie $D$ mit $B$ verglichen.

### 5.6.1 H1: Die Mensch-Maschine-Kommunikation mit gesprochener Sprache ist für SuS ein Phänomen aus ihrer Lebenswirklichkeit und motiviert sie, sich mit Informatik zu beschäftigen.

**Lebenswirklichkeit**

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ | $B$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 3 | 1 |
| 3 | 5 | 11 | 0 | 5 | 3 | 5 | 3 | 5 | 7 |
| 4 | 17 | 15 | 7 | 10 | 2 | 8 | 5 | 5 | 14 |
| 5 | 12 | 6 | 7 | 5 | 1 | 3 | 2 | 1 | 9 |
| 6 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| $n$ | $n_E = 35$ | $n_K = 36$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 11$ | $n_D = 14$ | $n_B = 31$ |

Tabelle 5.18: Kontingenztabelle Lebenswirklichkeit

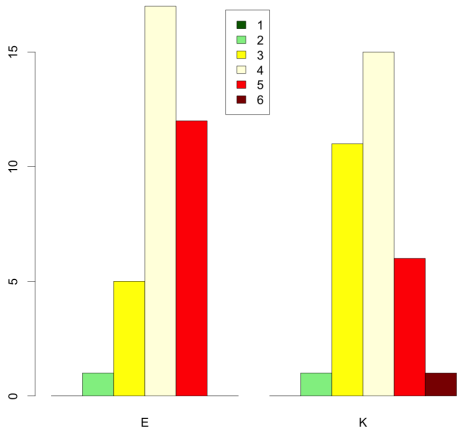



Abbildung 5.44: Vergleich $E$ mit $K$

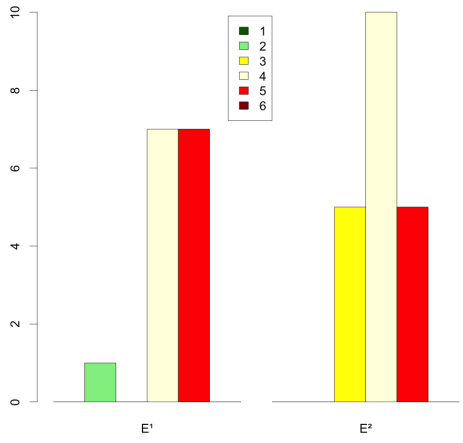



Abbildung 5.45: Vergleich $E^1$ mit $E^2$

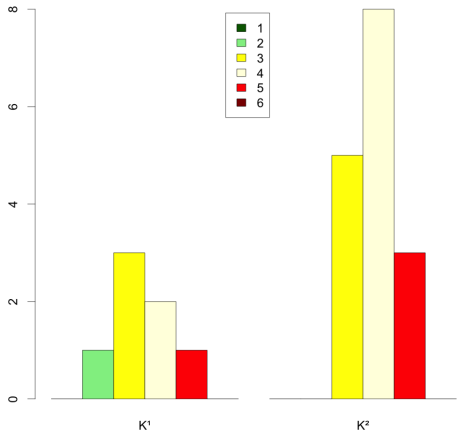



Abbildung 5.46: Vergleich $K^1$ mit $K^2$

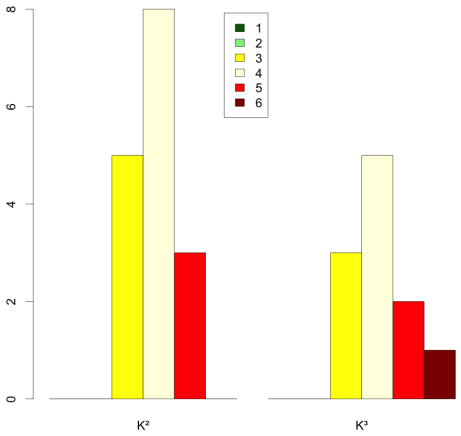



Abbildung 5.47: Vergleich $K^2$ mit $K^3$

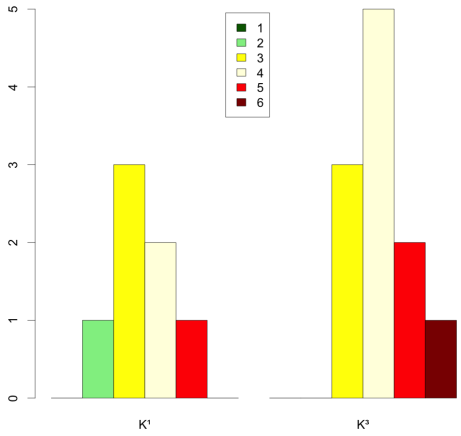



Abbildung 5.48: Vergleich $K^1$ mit $K^3$

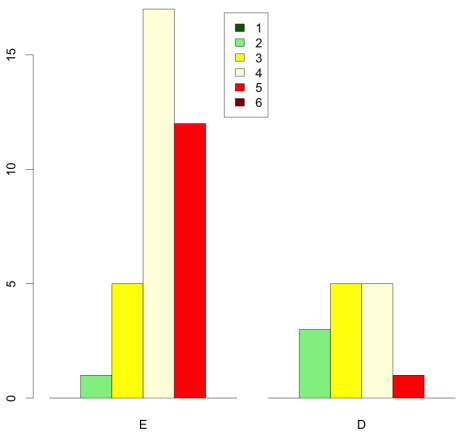



Abbildung 5.49: Vergleich $E$ mit $D$

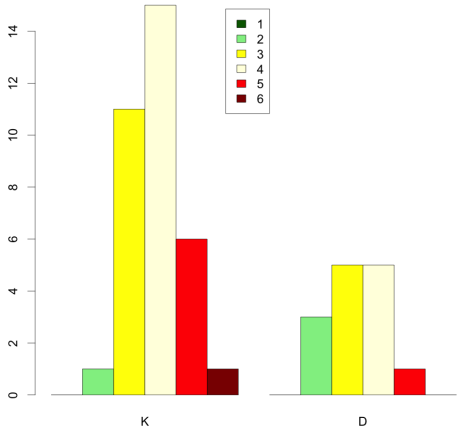



Abbildung 5.50: Vergleich $K$ mit $D$

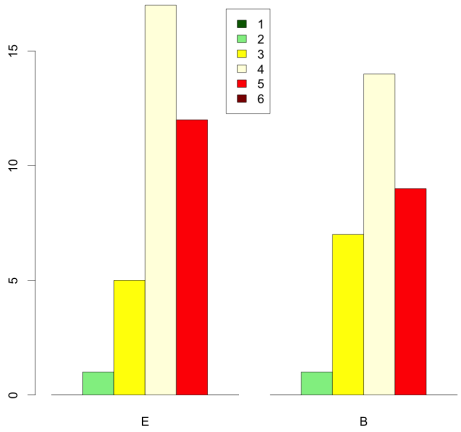



Abbildung 5.51: Vergleich $E$ mit $B$

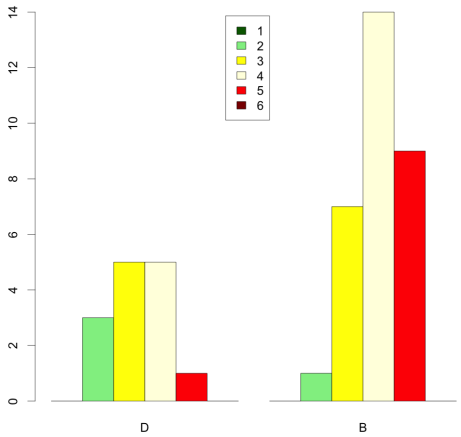



Abbildung 5.52: Vergleich $D$ mit $B$

Einem gerechneter Fisher-Test für alle Gruppen zeigt, dass sich nur die Ergebnisse der Selbsteinschätzungen der Gruppen $E$ und $D$ mit einem $p$-Wert von $p$ kleiner gleich $0,05$ signifikant voneinander unterscheiden.

### Motivation für Informatik

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ | $B$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 0 | 2 | 3 | 2 | 0 | 4 | 1 |
| 2 | 9 | 8 | 2 | 7 | 3 | 3 | 2 | 6 | 8 |
| 3 | 10 | 8 | 5 | 5 | 1 | 3 | 3 | 2 | 12 |
| 4 | 8 | 11 | 4 | 4 | 0 | 6 | 4 | 1 | 8 |
| 5 | 5 | 2 | 3 | 2 | 0 | 2 | 2 | 1 | 2 |
| 6 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $n$ | $n_E = 35$ | $n_K = 34$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 11$ | $n_D = 14$ | $n_B = 31$ |

Tabelle 5.19: Kontingenztabelle Motivation für Informatik

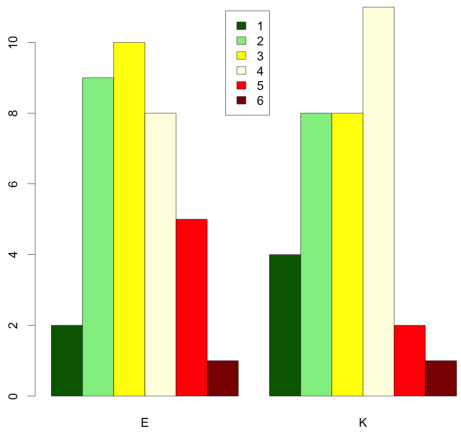



Abbildung 5.53: Vergleich $E$ mit $K$

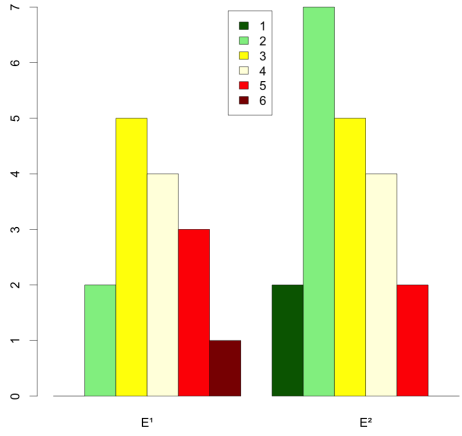



Abbildung 5.54: Vergleich $E^1$ mit $E^2$

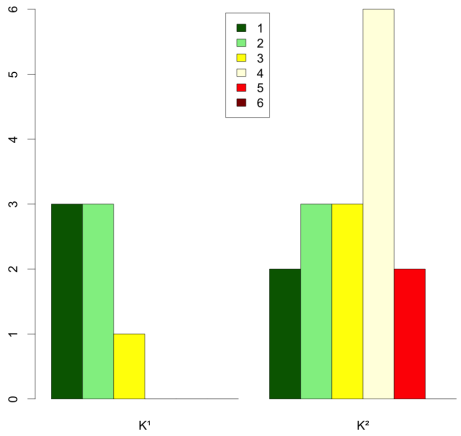



Abbildung 5.55: Vergleich $K^1$ mit $K^2$

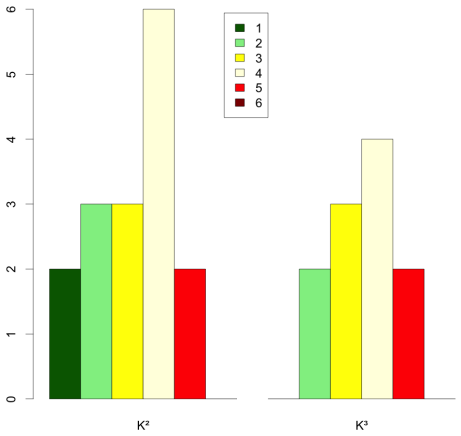



Abbildung 5.56: Vergleich $K^2$ mit $K^3$

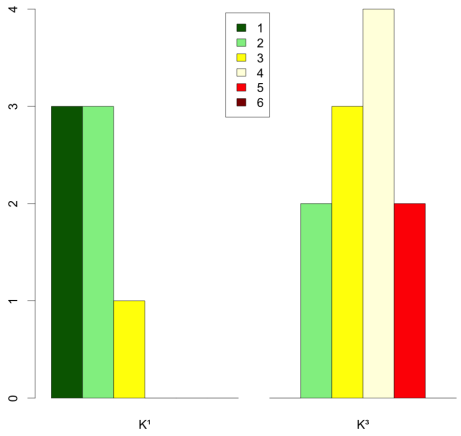



Abbildung 5.57: Vergleich $K^1$ mit $K^3$

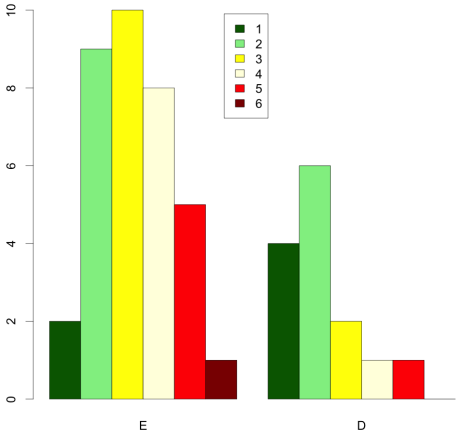



Abbildung 5.58: Vergleich $E$ mit $D$

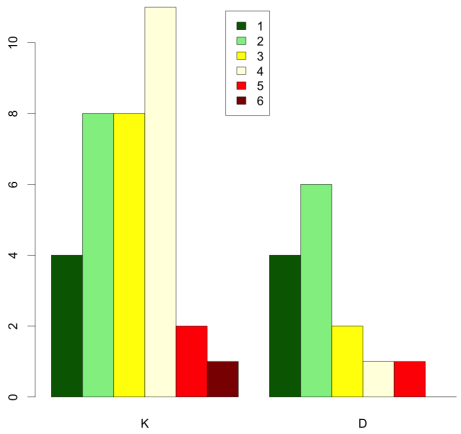



Abbildung 5.59: Vergleich $K$ mit $D$

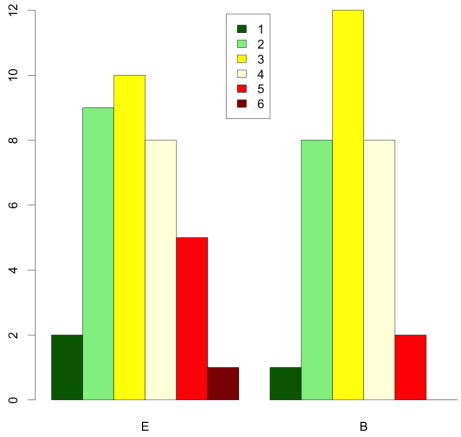



Abbildung 5.60: Vergleich $E$ mit $B$

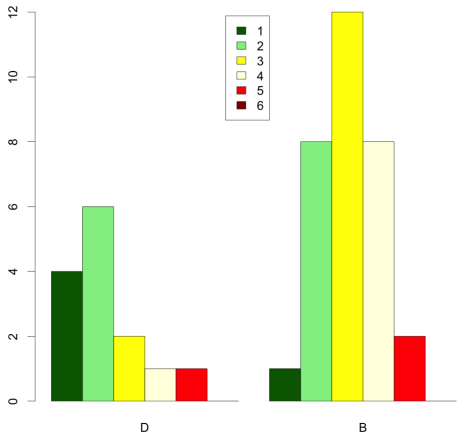



Abbildung 5.61: Vergleich $D$ mit $B$

Nur die Selbsteinschätzungen der Gruppen $K^1$ und $K^3$ sowie $D$ und $B$ unterscheiden sich mit einem $p$-Wert von $p$ kleiner gleich $0,05$ signifikant.

### 5.6.2 H2: Mit den digitalen Lernumgebungen inES und inGE eignen sich die SuS theoretische Konzepte der Informatik handlungsorientiert an.

**Ganzheitlichkeit**

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ | $B$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 3 | 0 | 0 | 3 | 0 | 0 | 0 | 1 |
| 2 | 5 | 6 | 0 | 5 | 2 | 4 | 0 | 2 | 2 |
| 3 | 3 | 11 | 0 | 3 | 2 | 6 | 3 | 5 | 14 |
| 4 | 12 | 7 | 4 | 8 | 0 | 4 | 3 | 5 | 6 |
| 5 | 9 | 5 | 6 | 3 | 0 | 2 | 3 | 2 | 8 |
| 6 | 6 | 2 | 5 | 1 | 0 | 0 | 2 | 0 | 0 |
| $n$ | $n_E = 35$ | $n_K = 34$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 11$ | $n_D = 14$ | $n_B = 31$ |

Tabelle 5.20: Kontingenztabelle Ganzheitlichkeit

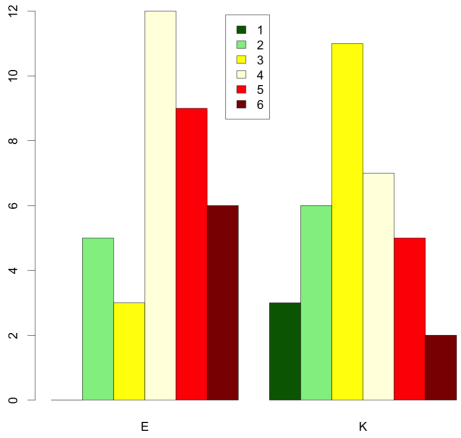



Abbildung 5.62: Vergleich $E$ mit $K$

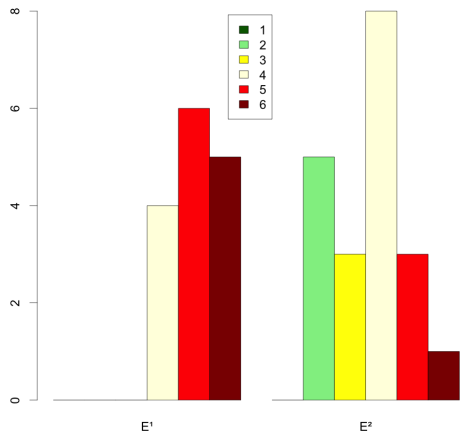



Abbildung 5.63: Vergleich $E^1$ mit $E^2$

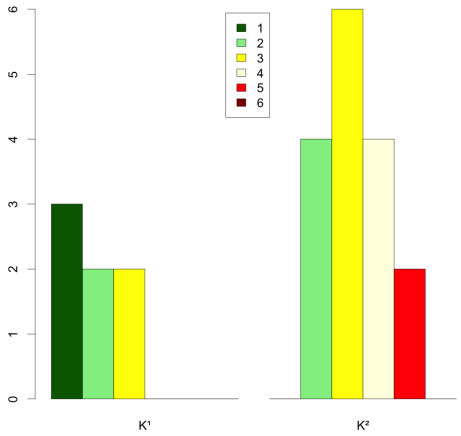



Abbildung 5.64: Vergleich $K^1$ mit $K^2$

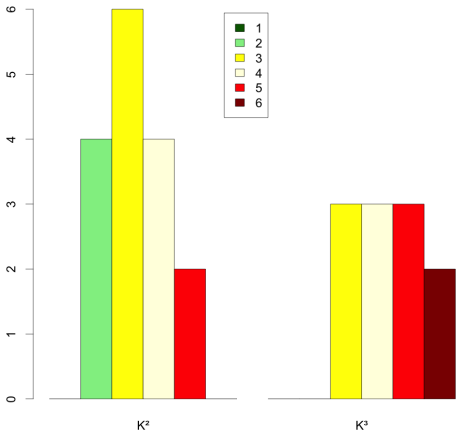



Abbildung 5.65: Vergleich $K^2$ mit $K^3$

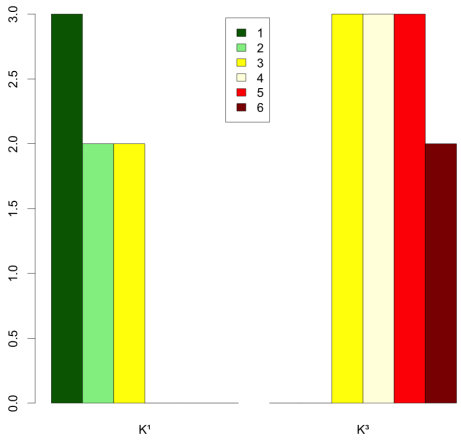



Abbildung 5.66: Vergleich $K^1$ mit $K^3$

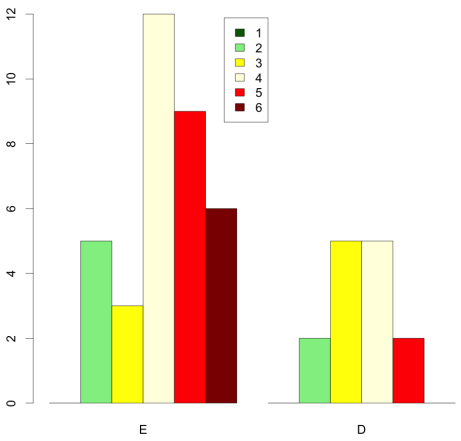



Abbildung 5.67: Vergleich $E$ mit $D$

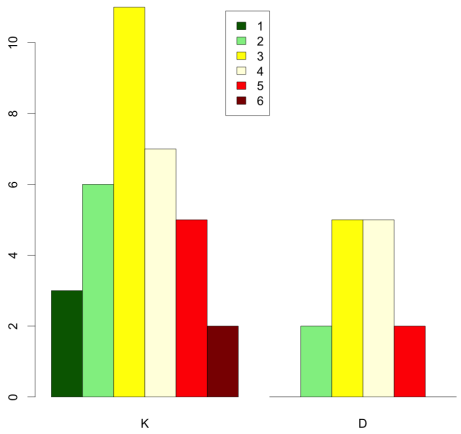



Abbildung 5.68: Vergleich $K$ mit $D$

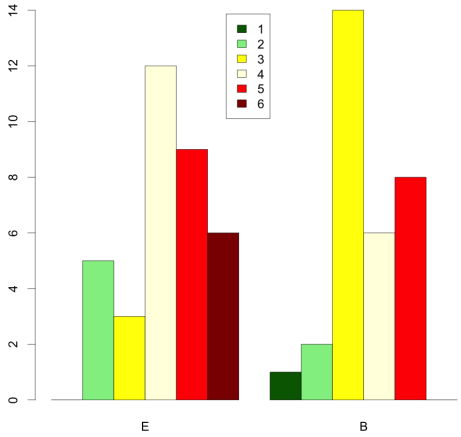



Abbildung 5.69: Vergleich $E$ mit $B$

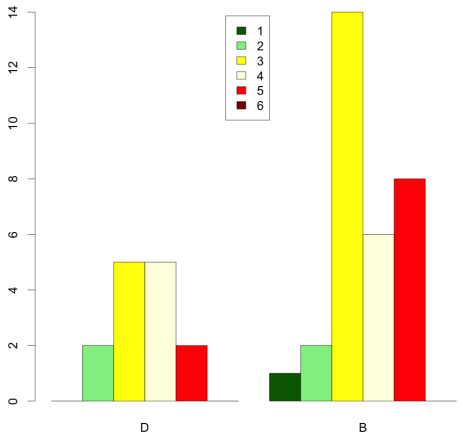



Abbildung 5.70: Vergleich $D$ mit $B$

Die Gruppen $E$ und $K$ sowie $K^1$ und $K^3$ unterscheiden sich mit einem $p$-Wert kleiner gleich $0,05$ signifikant. Die Unterschiede der Gruppen $E^1$ und $E^2$, $E$ und $B$ sind mit $p$ kleiner gleich $0,01$ sehr signifikant. Die Gruppen $K^1$ und $K^2$, $K^2$ und $K^3$, $E$ und $D$, $K$ und $D$ sowie $D$ und $B$ unterscheiden sich nicht signifikant.

**Schüleraktivität**

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ | $B$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 0 | 1 | 2 | 0 | 0 | 0 | 1 |
| 2 | 4 | 4 | 1 | 3 | 2 | 2 | 0 | 5 | 6 |
| 3 | 7 | 8 | 2 | 5 | 2 | 5 | 1 | 4 | 12 |
| 4 | 11 | 13 | 3 | 8 | 1 | 8 | 4 | 5 | 8 |
| 5 | 10 | 4 | 7 | 3 | 0 | 0 | 4 | 0 | 4 |
| 6 | 2 | 3 | 2 | 0 | 0 | 1 | 2 | 0 | 0 |
| $n$ | $n_E = 35$ | $n_K = 34$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 11$ | $n_D = 14$ | $n_B = 31$ |

Tabelle 5.21: Kontingenztabelle Schüleraktivität

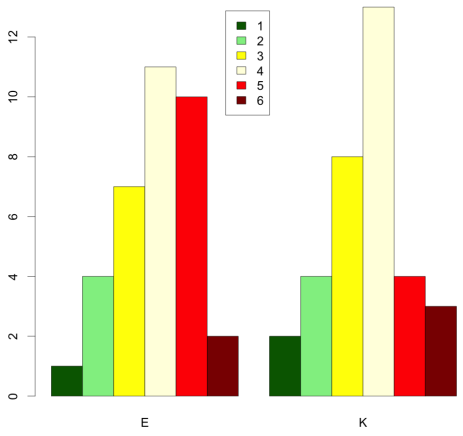



Abbildung 5.71: Vergleich $E$ mit $K$

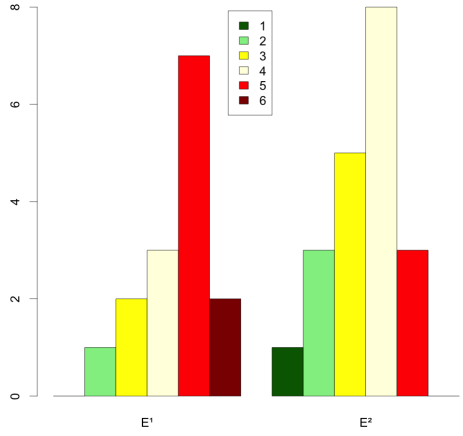



Abbildung 5.72: Vergleich $E^1$ mit $E^2$

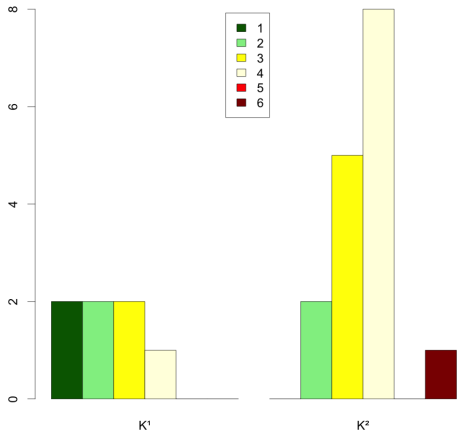



Abbildung 5.73: Vergleich $K^1$ mit $K^2$

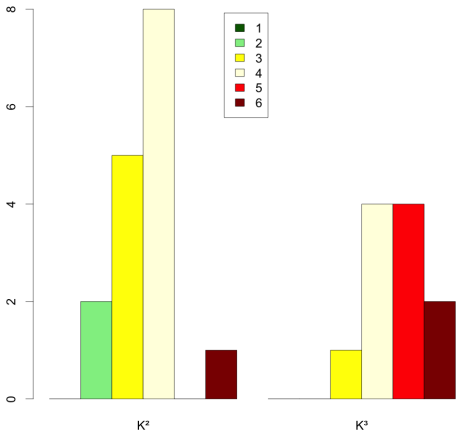



Abbildung 5.74: Vergleich $K^2$ mit $K^3$

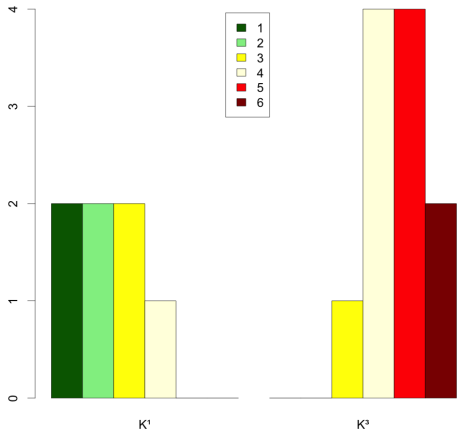



Abbildung 5.75: Vergleich $K^1$ mit $K^3$

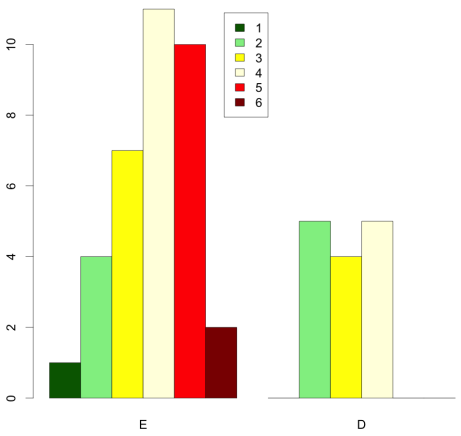



Abbildung 5.76: Vergleich $E$ mit $D$

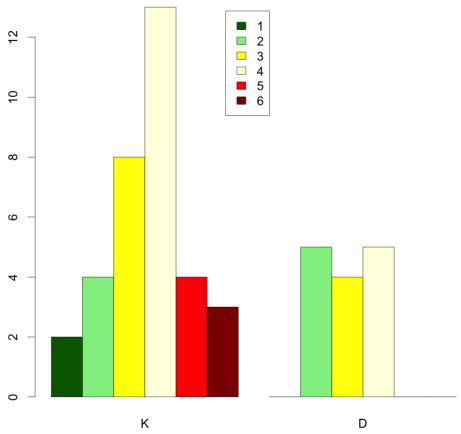



Abbildung 5.77: Vergleich $K$ mit $D$

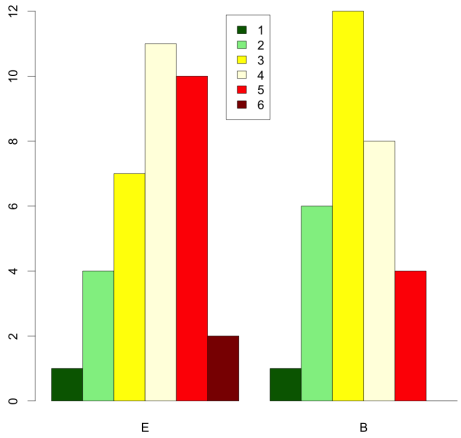



Abbildung 5.78: Vergleich $E$ mit $B$

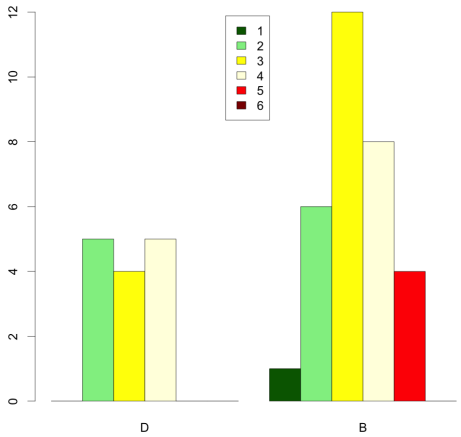



Abbildung 5.79: Vergleich $D$ mit $B$

Nur die Selbsteinschätzungen der Gruppen $K^2$ und $K^3$ sowie $K^1$ und $K^3$ zeigen hier signifikante Unterschiede, denn der $p$-Wert des gerechneten Fisher-Tests ist jeweils kleiner gleich $0,05$.

**Reflexion**

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ | $B$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 3 | 0 | 0 | 3 | 0 | 0 | 2 | 1 |
| 2 | 4 | 5 | 0 | 4 | 3 | 5 | 0 | 8 | 4 |
| 3 | 6 | 10 | 2 | 4 | 1 | 3 | 2 | 1 | 11 |
| 4 | 12 | 13 | 5 | 7 | 0 | 5 | 7 | 3 | 10 |
| 5 | 11 | 3 | 7 | 4 | 0 | 3 | 2 | 0 | 5 |
| 6 | 2 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| $n$ | $n_E = 35$ | $n_K = 34$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 11$ | $n_D = 14$ | $n_B = 31$ |

Tabelle 5.22: Kontingenztabelle Reflexion

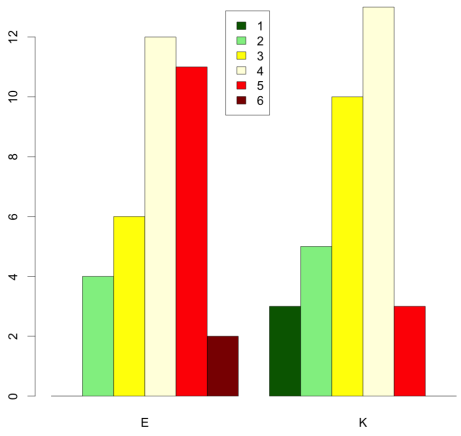



Abbildung 5.80: Vergleich $E$ mit $K$

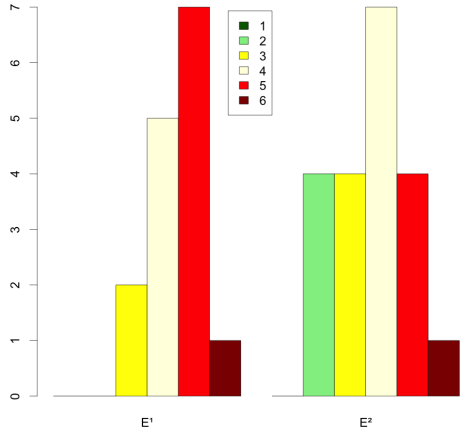



Abbildung 5.81: Vergleich $E^1$ mit $E^2$

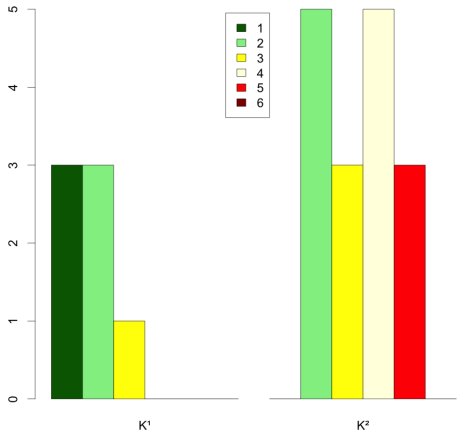



Abbildung 5.82: Vergleich $K^1$ mit $K^2$

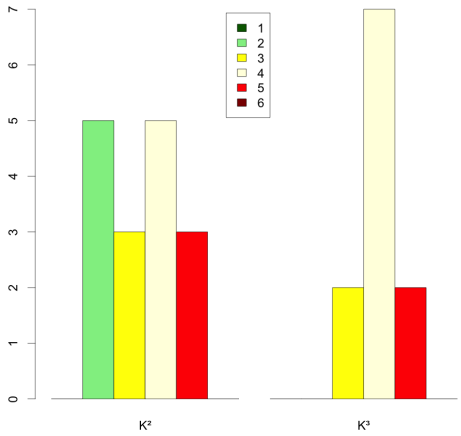



Abbildung 5.83: Vergleich $K^2$ mit $K^3$

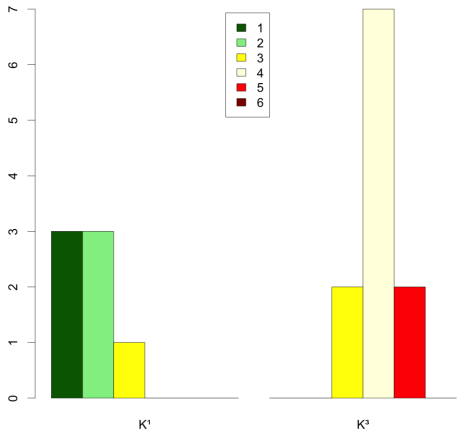



Abbildung 5.84: Vergleich $K^1$ mit $K^3$

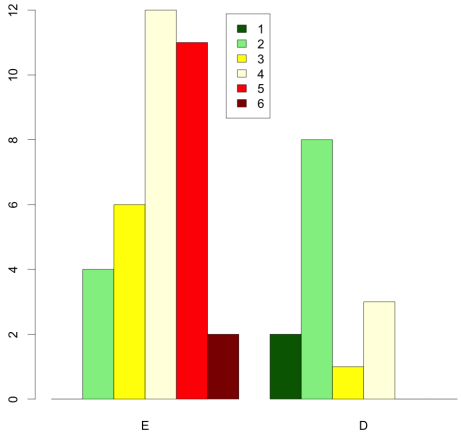



Abbildung 5.85: Vergleich $E$ mit $D$

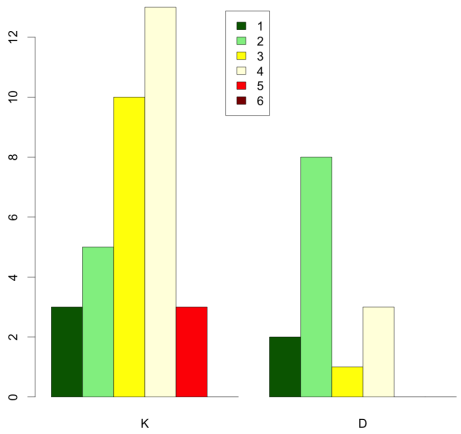



Abbildung 5.86: Vergleich $K$ mit $D$

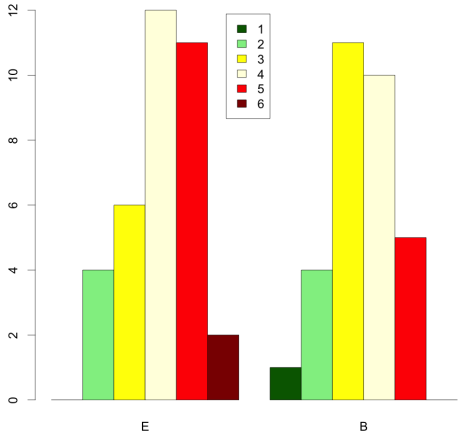



Abbildung 5.87: Vergleich $E$ mit $B$

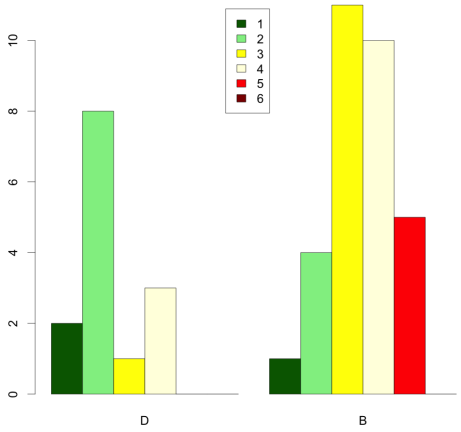



Abbildung 5.88: Vergleich $D$ mit $B$

Die Selbsteinschätzungen der Gruppen $E$ und $K$, $K^1$ und $K^2$ sowie $K$ und $D$ unterscheiden sich mit einem $p$-Wert von $0,05$ signifikant. Der Unterschied der Gruppe $D$ und $B$ ist mit $p$ kleiner gleich $0,01$ sehr signifikant. Des Weiteren ist der Unterschied zwischen $K^1$ und $K^3$ sowie $E$ und $D$ mit einem $p$-Wert von kleiner gleich $0,001$ hoch signifikant. Die restlichen Gruppen unterscheiden sich nicht signifikant.

### 5.6.3 H3: Das selbstständige Implementieren und Testen eigener Grammatiken fördert die Kreativität der SuS und ermöglicht verschiedenartige Projekte.

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ | $B$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 2 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 2 | 2 | 4 | 0 | 2 | 2 | 2 | 0 | 3 | 3 |
| 3 | 10 | 7 | 2 | 8 | 4 | 3 | 0 | 8 | 14 |
| 4 | 14 | 10 | 6 | 8 | 0 | 5 | 5 | 3 | 9 |
| 5 | 8 | 9 | 6 | 2 | 0 | 4 | 5 | 0 | 5 |
| 6 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| $n$ | $n_E = 35$ | $n_K = 33$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 10$ | $n_D = 14$ | $n_B = 31$ |

Tabelle 5.23: Kontingenztabelle Kreativität

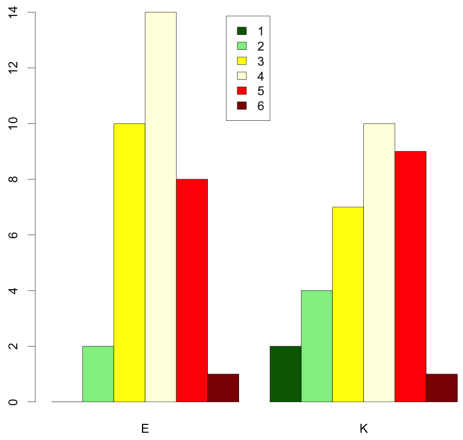



Abbildung 5.89: Vergleich $E$ mit $K$

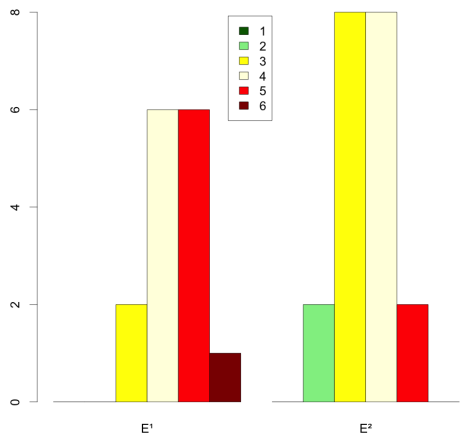



Abbildung 5.90: Vergleich $E^1$ mit $E^2$

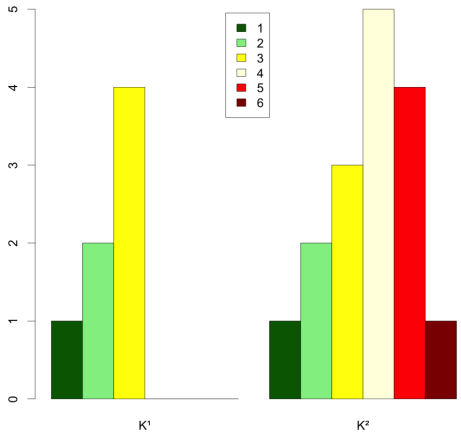



Abbildung 5.91: Vergleich $K^1$ mit $K^2$

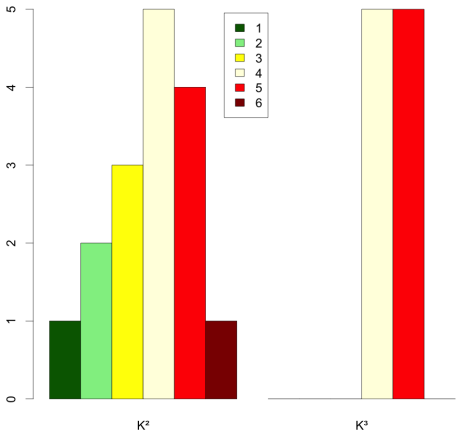



Abbildung 5.92: Vergleich $K^2$ mit $K^3$

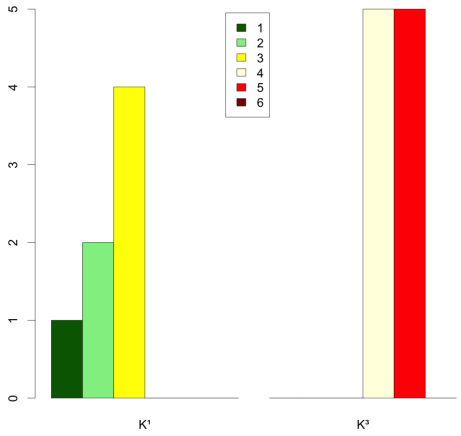



Abbildung 5.93: Vergleich $K^1$ mit $K^3$

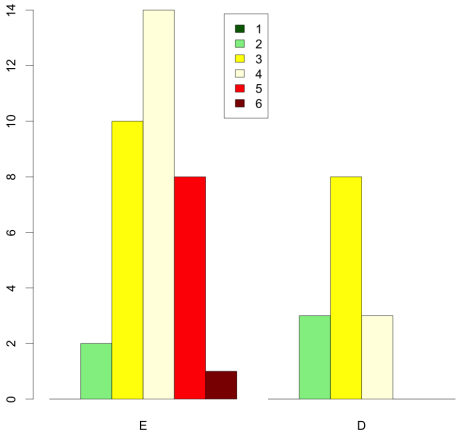



Abbildung 5.94: Vergleich $E$ mit $D$

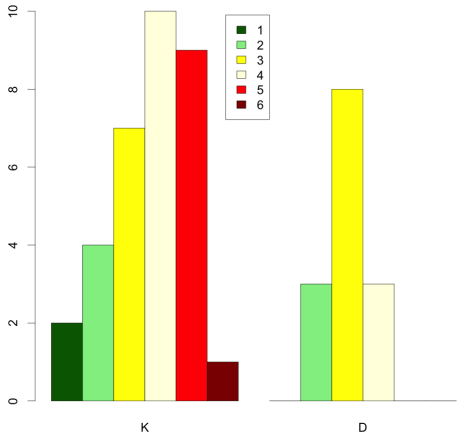



Abbildung 5.95: Vergleich $K$ mit $D$

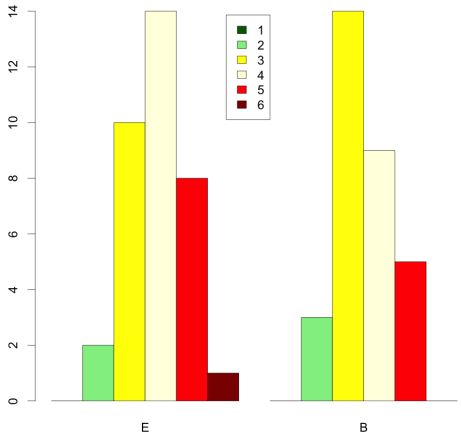



Abbildung 5.96: Vergleich $E$ mit $B$

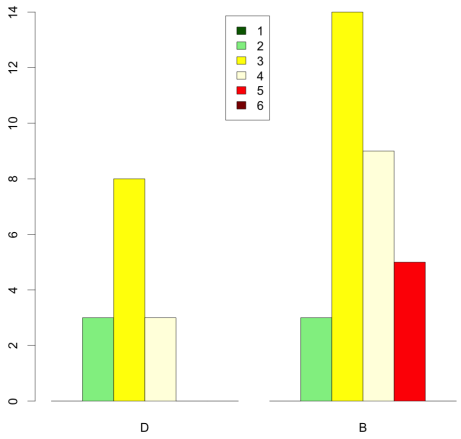



Abbildung 5.97: Vergleich $D$ mit $B$

Die Unterschiede der Gruppen $E$ und $D$ sowie $K$ und $D$ sind mit einem $p$-Wert kleiner gleich $0,05$ signifikant. Die Gruppe $K^1$ und $K^3$ unterscheidet sich mit einem $p$-Wert kleiner gleich $0,001$ hoch signifikant. Die übrigen Gruppen unterscheiden sich nicht signifikant.

Die folgende Tabelle zeigt eine Übersicht ausgewählter hochgeladener inES-Projekte mit einer kurzen Inhaltsbeschreibung.

| **Projekttitel** | **Inhalt** |
|---|---|
| Videothek | Eine telefonische Bestellung von Filmtiteln. |
| Flugbuchung | Eine telefonische Reservierung von Flugtickets. Der Anrufer wird des Weiteren über den Preis, die Flugroute sowie über technische Details des Flugzeuges informiert. |
| Sightseeing (Potsdam & Umgebung) | Ein Tourist kann sich über diese Hotline über verschiedene Sehenswürdigkeiten in Berlin, Potsdam und Umgebung informieren. |
| Länderinfo | Eine telefonische Auskunftmöglichkeit über Länder und Kontinente unseres Planeten. |
| Stundenplan | Ein Auskunftsystem über den Stunden- sowie Vertretungsplan der Schule. |
| Schnitzelgrube | Ähnlich dem inES-Beispiel kann hier ein Anrufer verschiedene klassische Gerichte telefonisch bestellen. |
| Hundezwinger | Ein Anrufer kann sich hier bei einem Tierheim nach verlorengegangenen Hunden erkundigen. |
| Weekendcar | Anrufer können sich hier ihren Traumwagen für ein Wochenendtrip ausleihen. |
| H2O Kundenservice | Ein Sprachdialog für die Abwicklung von Problemen der Wasserversorgung eines städtischen Wasserversorgers. |
| COD Shop | Ein Sprachdialog für die telefonische Bestellung von Plugins für ein Computerspiel. |
| Computer | Ein Sprachdialog zur Steuerung des eigenen Computers. |
| Freizeit | Ein Sprachdialog, der einen Anrufer Vorschläge zur Freizeitgestaltung (Sportcenter, Kochkurs) gibt. |
| Geschichtenhotline | Dieser Sprachdialog stellt für Kinder verschiedene Geschichten zum Vorlesen zur Verfügung. |
| Partnerbörse | Telefonische Partnervermittlung. |

Tabelle 5.24: abgegebene inES-Projekte

### 5.6.4 H4: Mit einer zustandsorientierten Modellierung verstehen die SuS das theoretische Konzept eines Parsers.

| Codesystem | GRUPPE = E | GRUPPE = K | GRUPPE = B | GRUPPE = D | SUM |
|---|---|---|---|---|---|
| Zustände | 3 | 1 | 1 | | 5 |
| Alphabet | 1 | | | | 1 |
| Kelleralphabet | 3 | | 1 | | 4 |
| Zustandsübergänge | | | | | |
| Startzustand | | | | | |
| Anfangszustand_Keller | | | | | |
| finale_Zustände | 2 | | 1 | | 3 |
| Verwendungszweck | 12 | | | | 12 |
| Beschreibung | 10 | 3 | 9 | 4 | 26 |
| SUM | 31 | 4 | 12 | 4 | 51 |
| N (Dokumente) | 28 | 10 | 26 | 5 | |

Abbildung 5.98: Kreuztabelle der Kodierung und zugehörigen Gruppen

Die Abbildung 5.98 zeigt die Kodierung der Schülerbeschreibungen des Kellerautomaten (T09).
In der Experimentalgruppe $E$ beschreiben 28 von 35 Schülern und in der Kontrollgruppe $K$ beschreiben 10 von 35 Schüler den Begriff des Kellerautomaten. Von den 28 Schülern der Gruppe $E$ beschreiben 22 Schüler den Begriff bzw. den Verwendungszweck korrekt. In der Gruppe $K$ beschreiben nur drei Schüler den Kellerautomaten korrekt. Auffallend ist, dass die Schüler der Gruppe $E$ viel häufiger als die Schüler der Gruppe $K$ Fachbegriffe wie Zustände, Alphabet, Kelleralphabet und finale Zustände benutzen. In der Gruppe $K$ wird nur in einer Beschreibung der Begriff Zustände erwähnt.
Bei der Gruppe $D$ wurden fünf von vierzehn Beschreibungen eingereicht. Davon sind vier der abgegebenen Beschreibungen nachvollziehbar. Die Schüler der Gruppe $D$ benutzen keine Fachbegriffe. In der Gruppe $B$ benutzt ein Schüler die Fachbegriffe Zustände, Kelleralphabet und finale Zustände. Von den 26 abgegebenen Beschreibungen der Gruppe $B$ enthalten neun nachvollziehbare Beschreibungen dieses Begriffes.

### 5.6.5 H5: Durch die Implementierung von formalen Grammatiken mithilfe verschiedener Darstellungsformen verstehen die SuS das theoretische Konzept einer formalen Grammatik.

| Codesystem | GRUPPE = E | GRUPPE = K | GRUPPE = D | GRUPPE = B | SUM |
|---|---|---|---|---|---|
| Verwendungszweck | 16 | 7 | 3 | 1 | 27 |
| Beschreibung | 7 | 2 | | | 9 |
| Nonterminale | 7 | | | | 7 |
| Terminale | 7 | | 1 | | 8 |
| Alphabet | 4 | | | | 4 |
| Startsymbol | 7 | | | | 7 |
| Produktionsregeln | 5 | | | | 5 |
| SUM | 53 | 9 | 4 | 1 | 67 |
| N (Dokumente) | 30 | 14 | 6 | 13 | |

Abbildung 5.99: Kreuztabelle der Kodierung und zugehörigen Gruppen

Die Abbildung 5.99 zeigt die Kodierung der Schülerbeschreibungen des Begriffes der formalen Grammatiken (T08).
In der Experimentalgruppe $E$ beschreiben 30 von 35 Schülern und in der Kontrollgruppe $K$ beschreiben 14 von 35 Schüler den Begriff der Formalen Grammatik. Von den 30 Schülern der Gruppe $E$ beschreiben 23 Schüler den Begriff bzw. den Verwendungszweck korrekt. In der Gruppe $K$ beschreiben neun Schüler den Begriff der Formalen Grammatik korrekt. Auffallend ist auch hier wieder, dass die Schüler der Gruppe $E$ viel häufiger als die Schüler der Gruppe $K$ Fachbegriffe benutzen. In der Gruppe $K$ benutzt kein Schüler die zur Formalen Grammatik gehörenden Fachbegriffe.
Sechs von vierzehn Schülern der Gruppe $D$ beschreiben den Begriff der formalen Grammatik. Davon sind drei der abgegebenen Beschreibungen fachlich korrekt. Nur ein Schüler der Gruppe $D$ benutzte den Fachbegriff Terminal. Die Schüler der Gruppe $B$ benutzen ebenfalls keine Fachbegriffe. Von den dreizehn abgegebenen Beschreibungen der Gruppe $B$ war nur eine Beschreibung richtig.

### 5.6.6 H6: Die strikte Visualisierung des gesamten Problemlöseprozesses unterstützt einen konstruktiven Umgang mit Fehlern.

**Selbstständigkeit**

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ | $B$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 2 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 2 | 3 | 4 | 0 | 3 | 2 | 2 | 0 | 4 | 8 |
| 3 | 16 | 9 | 4 | 12 | 4 | 5 | 0 | 5 | 9 |
| 4 | 13 | 15 | 9 | 4 | 0 | 6 | 9 | 5 | 13 |
| 5 | 3 | 2 | 2 | 1 | 0 | 2 | 0 | 0 | 1 |
| 6 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| $n$ | $n_E = 35$ | $n_K = 33$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 10$ | $n_D = 14$ | $n_B = 31$ |

Tabelle 5.25: Kontingenztabelle Selbstständigkeit

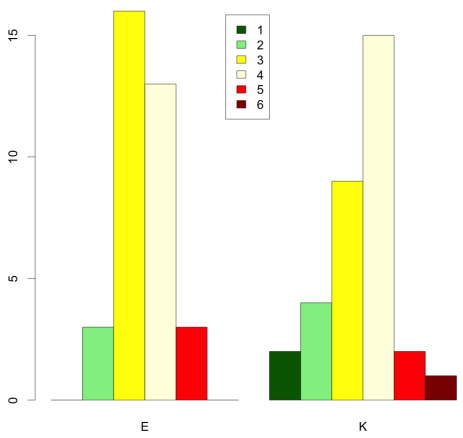



Abbildung 5.100: Vergleich $E$ mit $K$

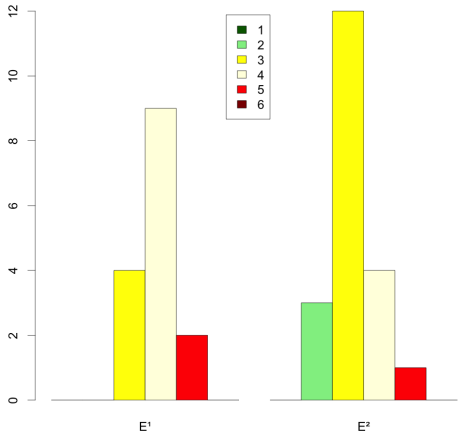



Abbildung 5.101: Vergleich $E^1$ mit $E^2$

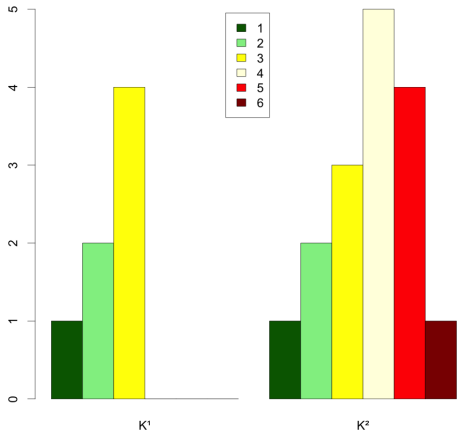



Abbildung 5.102: Vergleich $K^1$ mit $K^2$

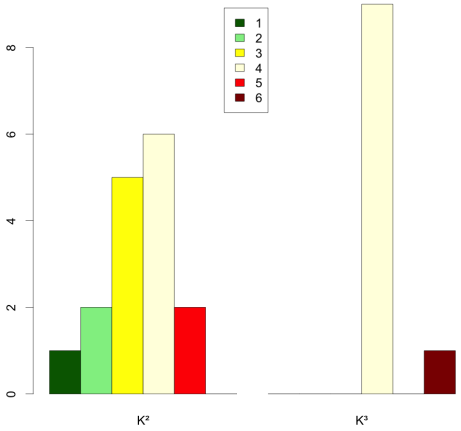



Abbildung 5.103: Vergleich $K^2$ mit $K^3$

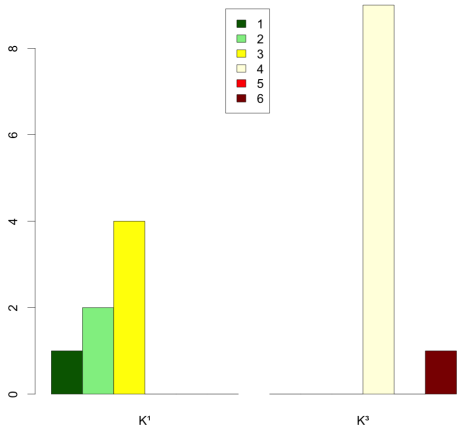



Abbildung 5.104: Vergleich $K^1$ mit $K^3$

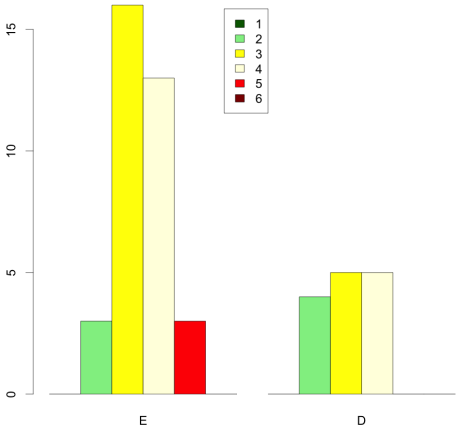



Abbildung 5.105: Vergleich $E$ mit $D$

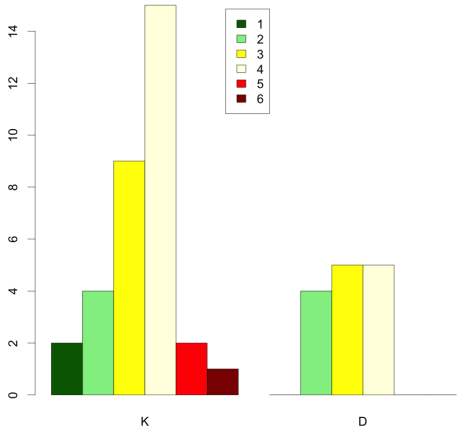



Abbildung 5.106: Vergleich $K$ mit $D$

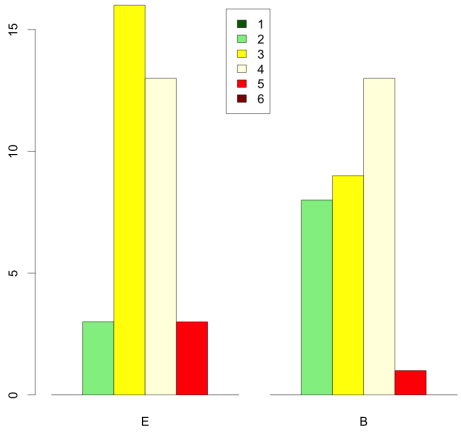



Abbildung 5.107: Vergleich $E$ mit $B$

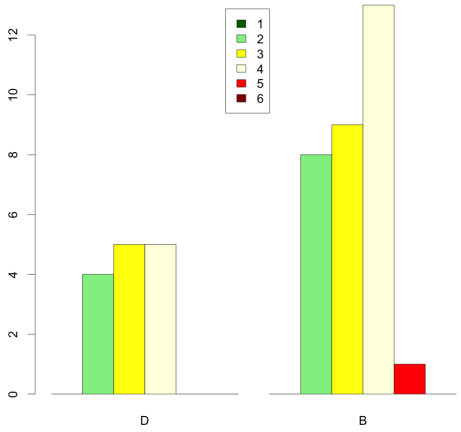



Abbildung 5.108: Vergleich $D$ mit $B$

Zwischen $K^1$ und $K^3$ bestehen mit einem $p$-Wert von $p$ kleiner gleich $0,01$ sehr signifikante Unterschiede. Die Unterschiede zwischen $E^1$ und $E^2$ sowie $K^2$ und $K^3$ sind mit einem $p$-Wert von $p$ kleiner gleich $0,05$ signifikant zugunsten von $E^1$ bzw. $K^3$. Zwischen den restlichen Gruppen bestehen keine signifikanten Unterschiede.

**Softwareunterstützung**

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ | $B$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 3 | 1 | 0 | 3 | 1 | 0 | 0 | 1 | 4 |
| 2 | 8 | 7 | 2 | 6 | 1 | 5 | 1 | 7 | 5 |
| 3 | 10 | 10 | 1 | 9 | 3 | 4 | 3 | 4 | 11 |
| 4 | 11 | 11 | 9 | 2 | 2 | 5 | 4 | 2 | 10 |
| 5 | 3 | 4 | 3 | 0 | 0 | 2 | 2 | 0 | 1 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| $n$ | $n_E = 35$ | $n_K = 33$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 10$ | $n_D = 14$ | $n_B = 31$ |

Tabelle 5.26: Kontingenztabelle Softwarehilfe

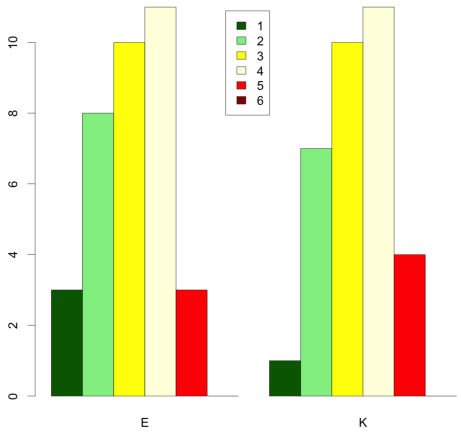



Abbildung 5.109: Vergleich $E$ mit $K$

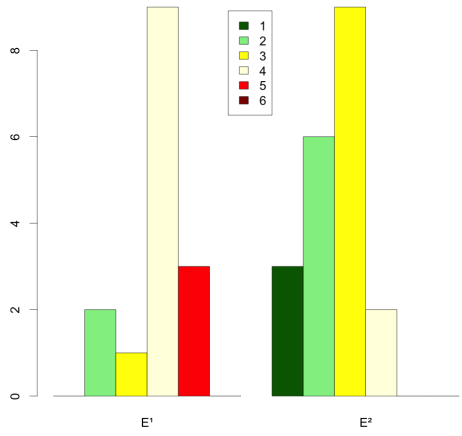



Abbildung 5.110: Vergleich $E^1$ mit $E^2$

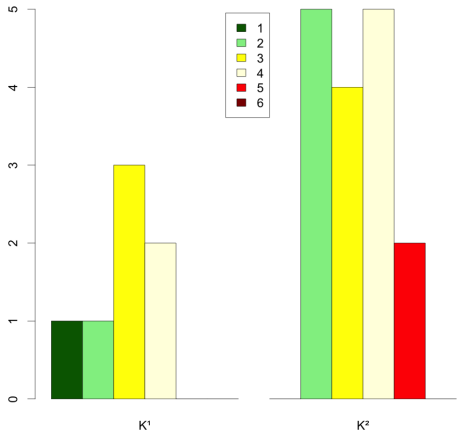



Abbildung 5.111: Vergleich $K^1$ mit $K^2$

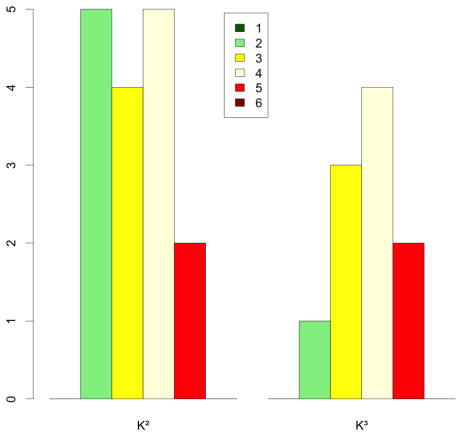



Abbildung 5.112: Vergleich $K^2$ mit $K^3$

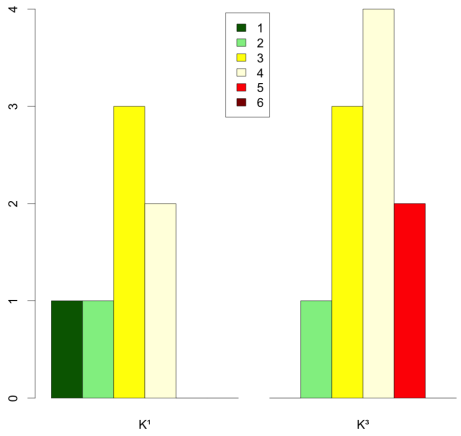



Abbildung 5.113: Vergleich $K^1$ mit $K^3$

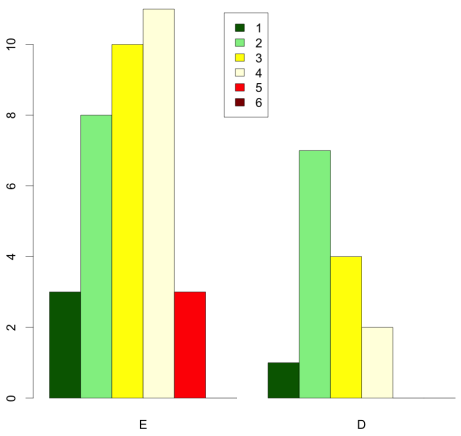



Abbildung 5.114: Vergleich $E$ mit $D$

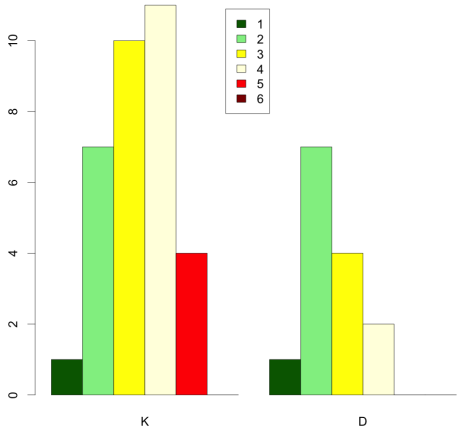



Abbildung 5.115: Vergleich $K$ mit $D$

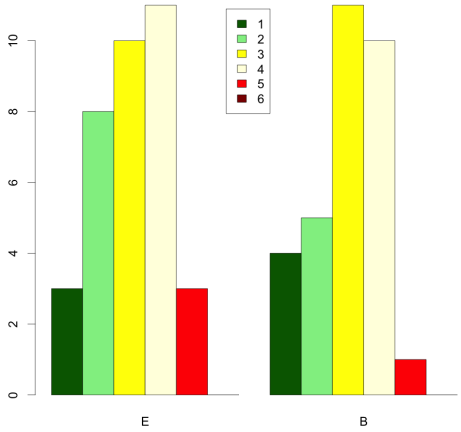



Abbildung 5.116: Vergleich $E$ mit $B$

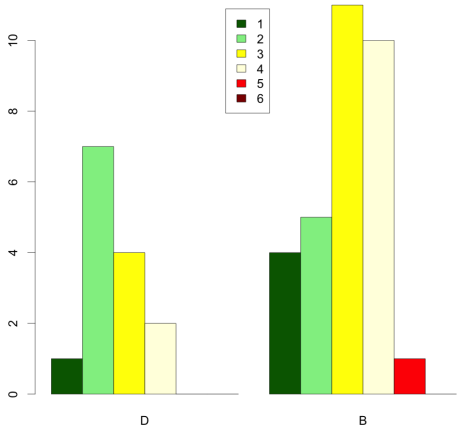



Abbildung 5.117: Vergleich $D$ mit $B$

Nur die Gruppen $E^1$ und $E^2$ unterscheiden sich durch einen $p$-Wert von $p$ kleiner gleich $0,01$ hoch signifikant zugunsten der Gruppe $E^1$. Alle anderen Gruppen zeigen keine signifikanten Unterschiede.

### 5.6.7 H7: Der Kontext Mensch-Maschine-Kommunikation mit gesprochener Sprache spricht SuS gleichermaßen an.

**Lebenswirklichkeit**

| | $E^w$ | $E^m$ | $K^w$ | $K^m$ |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 0 | 1 |
| 3 | 2 | 3 | 5 | 6 |
| 4 | 3 | 14 | 7 | 8 |
| 5 | 3 | 9 | 3 | 3 |
| 6 | 0 | 0 | 0 | 1 |
| $n$ | $n_{E^w} = 8$ | $n_{E^m} = 27$ | $n_{K^w} = 15$ | $n_{K^m} = 19$ |

Tabelle 5.27: Kontingenztabelle Lebenswirklichkeit nach Geschlecht

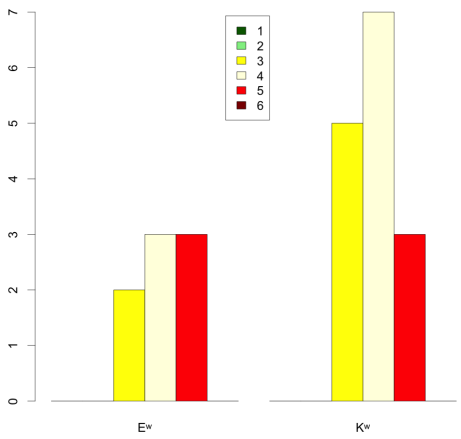



Abbildung 5.118: Vergleich $E^w$ mit $K^w$

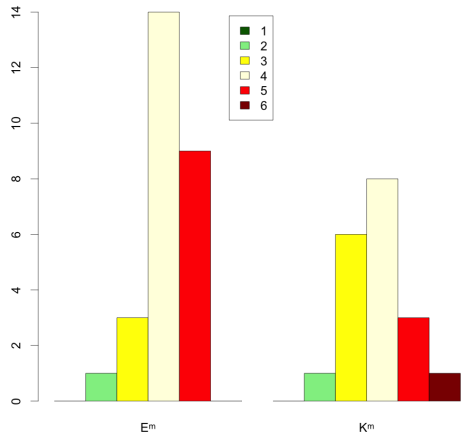



Abbildung 5.119: Vergleich $E^m$ mit $K^m$

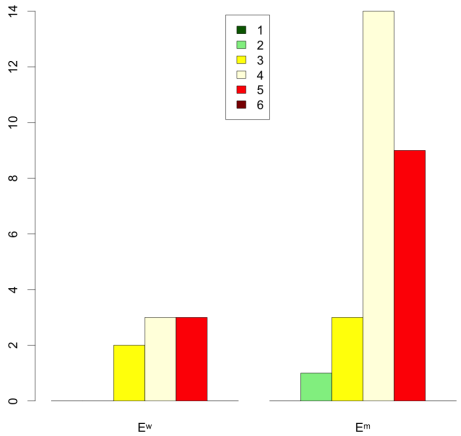



Abbildung 5.120: Vergleich $E^w$ mit $E^m$

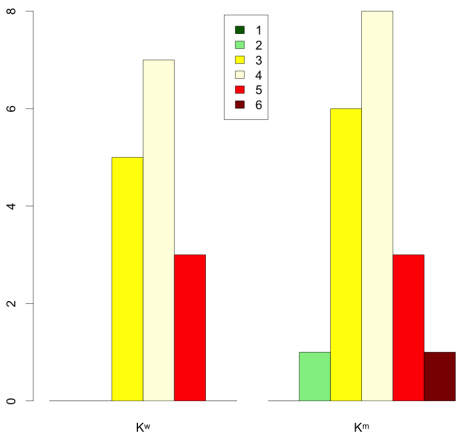



Abbildung 5.121: Vergleich $K^w$ mit $K^m$

Es bestehen keine signifikanten Unterschiede zwischen den einzelnen Gruppen.

### Motivation für Informatik

| | $E^w$ | $E^m$ | $K^w$ | $K^m$ |
|---|---|---|---|---|
| 1 | 1 | 1 | 2 | 1 |
| 2 | 1 | 8 | 4 | 5 |
| 3 | 2 | 8 | 4 | 9 |
| 4 | 1 | 7 | 4 | 1 |
| 5 | 2 | 3 | 1 | 2 |
| 6 | 1 | 0 | 0 | 1 |
| $n$ | $n_{E^w} = 8$ | $n_{E^m} = 27$ | $n_{K^w} = 15$ | $n_{K^m} = 19$ |

Tabelle 5.28: Kontingenztabelle Motivation für Informatik nach Geschlecht

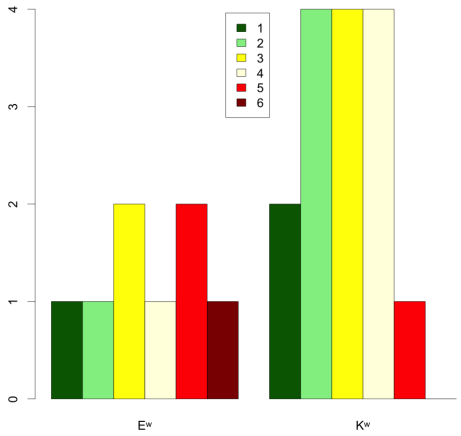



Abbildung 5.122: Vergleich $E^w$ mit $K^w$

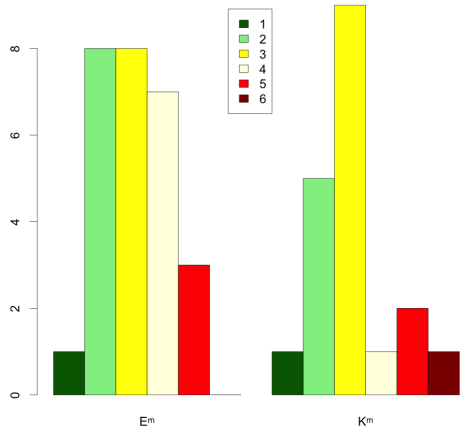



Abbildung 5.123: Vergleich $E^m$ mit $K^m$

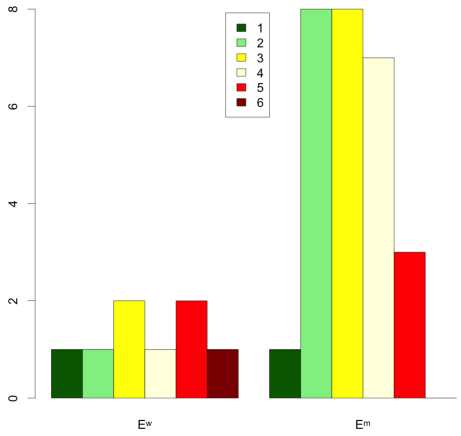



Abbildung 5.124: Vergleich $E^w$ mit $E^m$

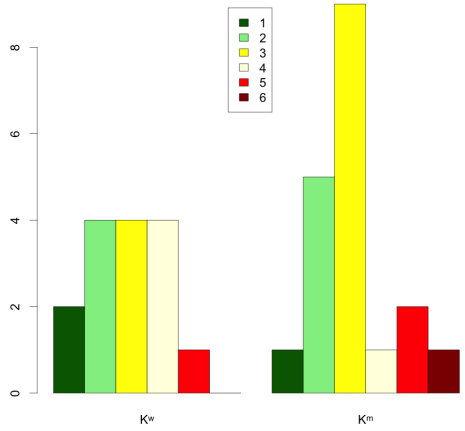



Abbildung 5.125: Vergleich $K^w$ mit $K^m$

Es bestehen keine signifikanten Unterschiede zwischen den einzelnen Gruppen.

**Ganzheitlichkeit**

| | $E^w$ | $E^m$ | $K^w$ | $K^m$ |
|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 2 |
| 2 | 3 | 2 | 2 | 4 |
| 3 | 1 | 2 | 5 | 6 |
| 4 | 1 | 11 | 3 | 4 |
| 5 | 0 | 9 | 3 | 2 |
| 6 | 3 | 3 | 1 | 1 |
| $n$ | $n_{E^w} = 8$ | $n_{E^m} = 27$ | $n_{K^w} = 15$ | $n_{K^m} = 19$ |

Tabelle 5.29: Kontingenztabelle Ganzheitlichkeit nach Geschlecht

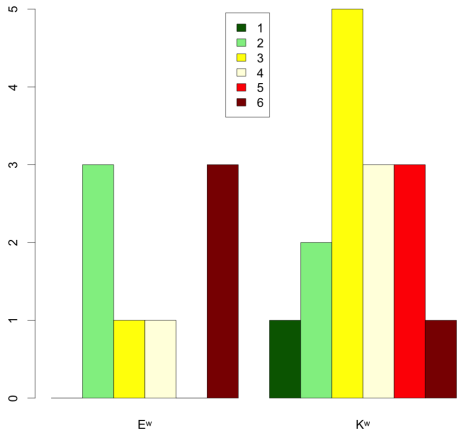



Abbildung 5.126: Vergleich $E^w$ mit $K^w$

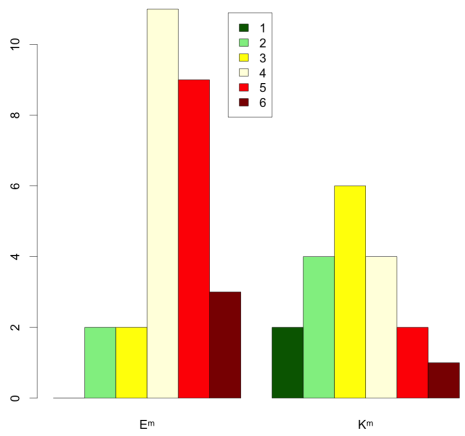



Abbildung 5.127: Vergleich $E^m$ mit $K^m$

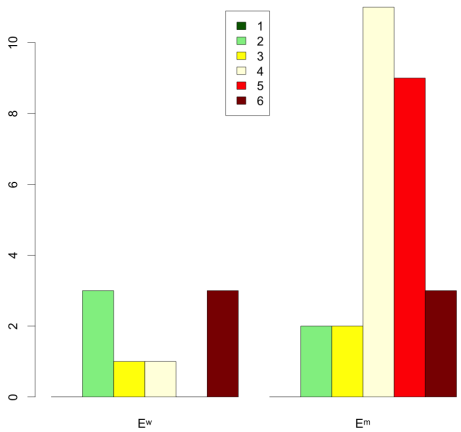



Abbildung 5.128: Vergleich $E^w$ mit $E^m$

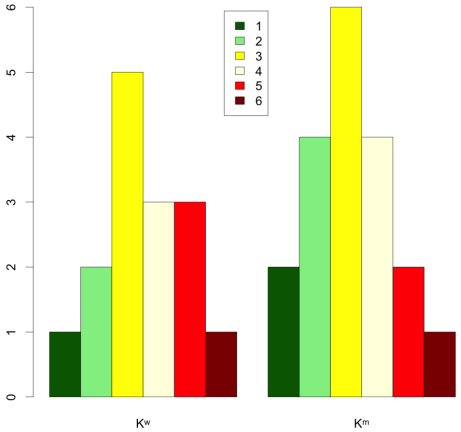



Abbildung 5.129: Vergleich $K^w$ mit $K^m$

Der Unterschied zwischen den Gruppen $E^m$ und $K^m$ sowie $E^w$ und $E^m$ ist mit einer Irrtumswahrscheinlichkeit von jeweils kleiner als $0,05$ signifikant. Dabei fällt der Unterschied zwischen $E^w$ und $E^m$ zugunsten von $E^m$ aus.

**Schüleraktivität**

| | $E^w$ | $E^m$ | $K^w$ | $K^m$ |
|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 2 |
| 2 | 2 | 2 | 1 | 3 |
| 3 | 2 | 5 | 4 | 4 |
| 4 | 0 | 11 | 7 | 6 |
| 5 | 2 | 8 | 2 | 2 |
| 6 | 1 | 1 | 1 | 2 |
| $n$ | $n_{E^w} = 8$ | $n_{E^m} = 27$ | $n_{K^w} = 15$ | $n_{K^m} = 19$ |

Tabelle 5.30: Kontingenztabelle Schüleraktivität nach Geschlecht

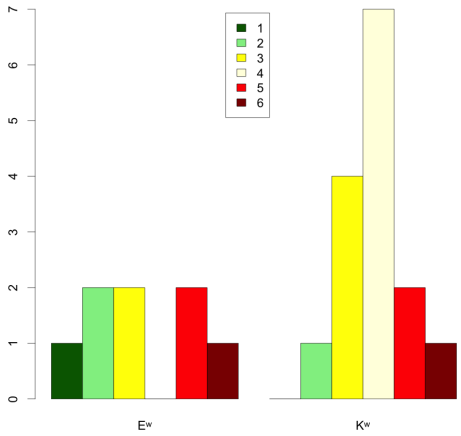



Abbildung 5.130: Vergleich $E^w$ mit $K^w$

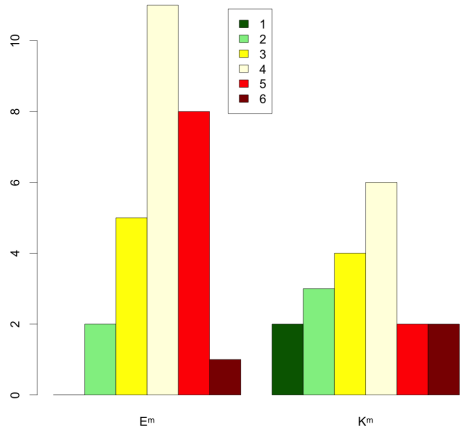



Abbildung 5.131: Vergleich $E^m$ mit $K^m$

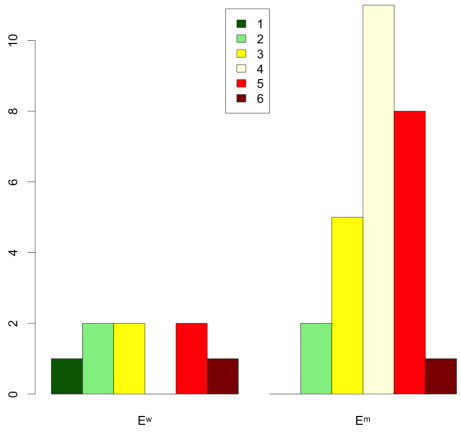



Abbildung 5.132: Vergleich $E^w$ mit $E^m$

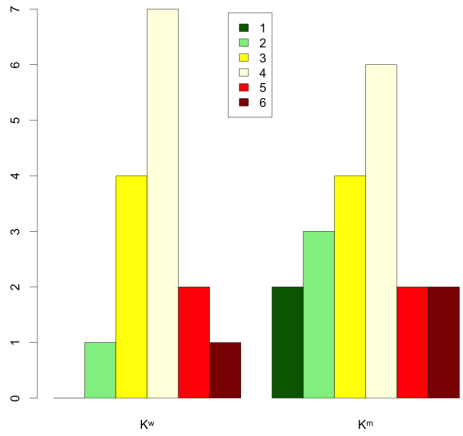



Abbildung 5.133: Vergleich $K^w$ mit $K^m$

Der Unterschied zwischen $E^w$ und $E^m$ ist mit einer Irrtumswahrscheinlichkeit von $p$ kleiner als $0,05$ zugunsten von $E^m$ signifikant.

**Reflexion**

| | $E^w$ | $E^m$ | $K^w$ | $K^m$ |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 3 |
| 2 | 2 | 2 | 5 | 1 |
| 3 | 1 | 5 | 2 | 9 |
| 4 | 3 | 9 | 5 | 5 |
| 5 | 1 | 10 | 3 | 1 |
| 6 | 1 | 1 | 0 | 0 |
| $n$ | $n_{E^w} = 8$ | $n_{E^m} = 27$ | $n_{K^w} = 15$ | $n_{K^m} = 19$ |

Tabelle 5.31: Kontingenztabelle Reflexion nach Geschlecht

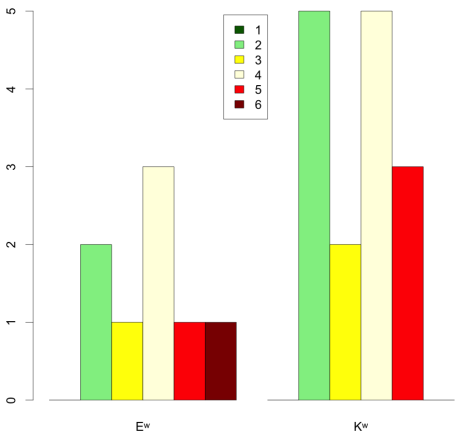



Abbildung 5.134: Vergleich $E^w$ mit $K^w$

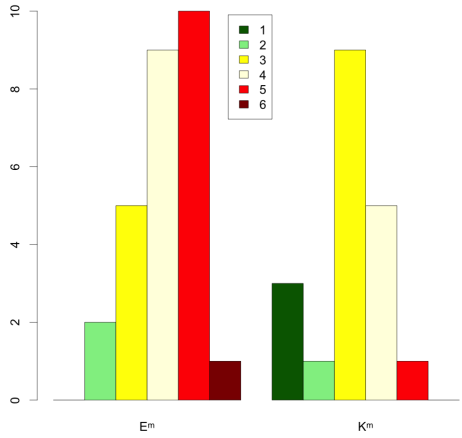



Abbildung 5.135: Vergleich $E^m$ mit $K^m$

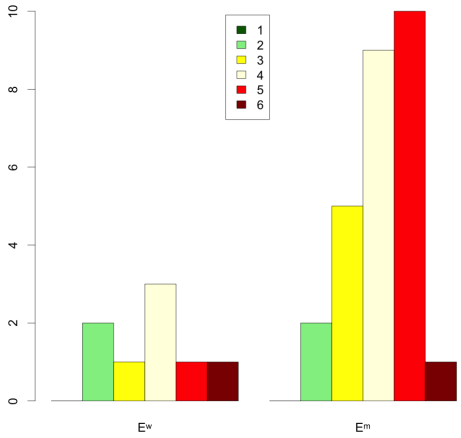



Abbildung 5.136: Vergleich $E^w$ mit $E^m$

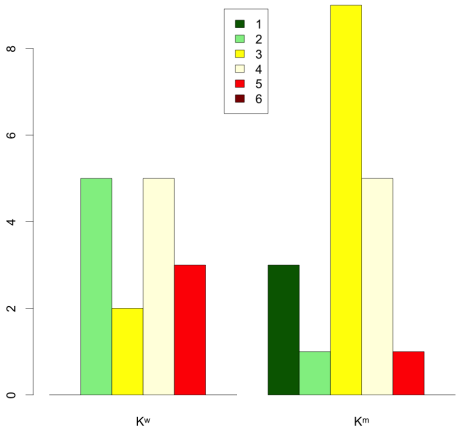



Abbildung 5.137: Vergleich $K^w$ mit $K^m$

Der Unterschied der Gruppe $E^m$ und $K^m$ ist mit einer Irrtumswahrscheinlichkeit von $p$ kleiner als $0,05$ zugunsten von $E^m$ signifikant. Des Weiteren unterscheidet sich die Gruppe $K^w$ und $K^m$ mit einer Irrtumswahrscheinlichkeit $p$ kleiner als $0,05$ zugunsten von $K^w$ signifikant.

**Kreativität**

| | $E^w$ | $E^m$ | $K^w$ | $K^m$ |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 2 |
| 2 | 0 | 2 | 1 | 3 |
| 3 | 5 | 5 | 5 | 2 |
| 4 | 1 | 13 | 6 | 4 |
| 5 | 1 | 7 | 2 | 7 |
| 6 | 1 | 0 | 1 | 0 |
| $n$ | $n_{E^w} = 8$ | $n_{E^m} = 27$ | $n_{K^w} = 15$ | $n_{K^m} = 18$ |

Tabelle 5.32: Kontingenztabelle Kreativität nach Geschlecht

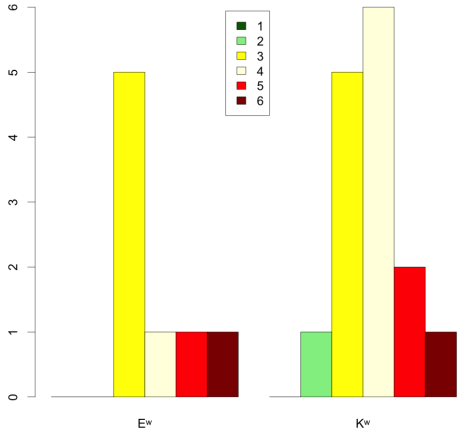



Abbildung 5.138: Vergleich $E^w$ mit $K^w$

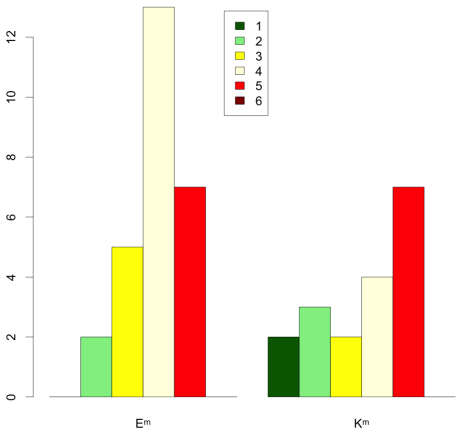



Abbildung 5.139: Vergleich $E^m$ mit $K^m$

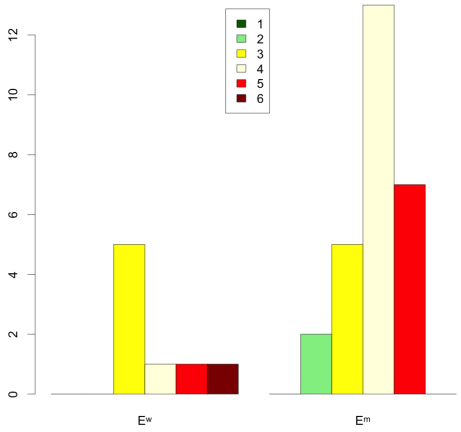



Abbildung 5.140: Vergleich $E^w$ mit $E^m$

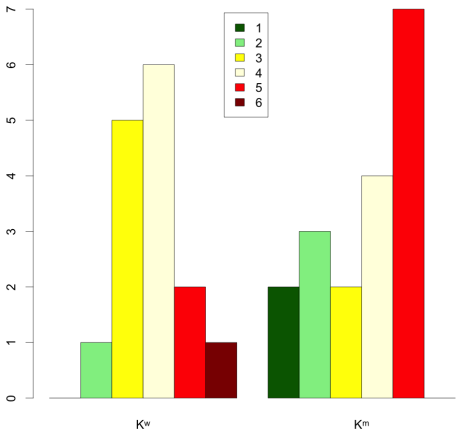



Abbildung 5.141: Vergleich $K^w$ mit $K^m$

Der Unterschied zwischen $E^w$ und $E^m$ ist mit einer Irrtumswahrscheinlichkeit von $p$ kleiner als $0,05$ zugunsten von $E^m$ signifikant.

### 5.6.8 H8: Die kontextorientierte Unterrichtseinheit zur Mensch-Maschine-Kommunikation mit gesprochener Sprache basierend auf IniK und unter Einsatz der Unterrichtssoftware inES hat einen pädagogischen Mehrwert.

Wie eingangs beschrieben, wurden für die Untersuchung dieser Hypothese die Ausprägungen der Kompetenzindikatoren des Bildungsplanes und die bei dem Test insgesamt erreichten Punktezahlen des im vierten Teil des Fragebogens durchgeführten Tests herangezogen. Im Test konnten höchstens fünfzig Punkte erreicht werden, dementsprechend ergab sich gemäß der im Abitur gültigen Bewertungsgrundlage zu jeder erreichten Punktzahl eine entsprechende Schulnote (1-6). Die Note 1 gab es ab 43, die Note 2 ab 35, die Note 3 ab 28, die Note 4 ab 20 und die Note 5 ab 10 erreichten Punkten. Ergebnisse, die schlechter als 10 Punkte ausfielen, wurden mit der Note 6 bewertet. Für die im Folgenden dargestellten Kontingenztabellen ergeben sich entsprechend sechs Cluster. Der besseren Übersichtlichkeit wegen wurde mit Schulnoten statt der üblichen Notenpunkte (0 - 15 Punkte) gerechnet, d. h. die Note 1 umfasst das Intervall von 13 bis 15 Punkten, die Note 2 von 10 bis 12 usw.

#### Die Ergebnisse des abschließenden Tests

Die folgende Kontingenztabelle zeigt die Verteilung der Noten für die verschiedenen Gruppen.

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $B$ | $D$ |
|---|---|---|---|---|---|---|---|---|---|
| 6 | 1 | 25 | 0 | 1 | 7 | 14 | 4 | 16 | 9 |
| 5 | 7 | 4 | 4 | 3 | 0 | 0 | 4 | 6 | 2 |
| 4 | 4 | 3 | 1 | 3 | 0 | 2 | 1 | 5 | 0 |
| 3 | 13 | 1 | 3 | 10 | 0 | 0 | 1 | 4 | 1 |
| 2 | 9 | 3 | 6 | 3 | 0 | 0 | 3 | 0 | 3 |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $n$ | $n_E = 35$ | $n_K = 36$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 13$ | $n_B = 31$ | $n_D = 15$ |

Tabelle 5.33: Kontingenztabelle der Testaufgaben T01-T07 aller Gruppen

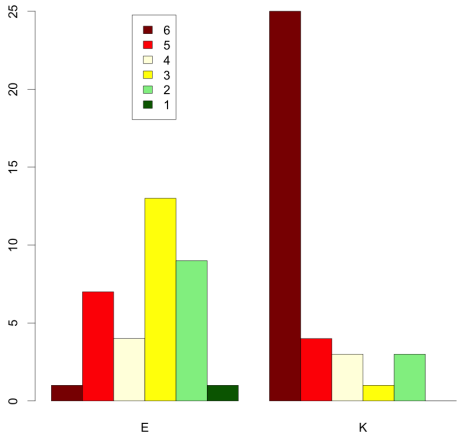



Abbildung 5.142: Vergleich $E$ mit $K$

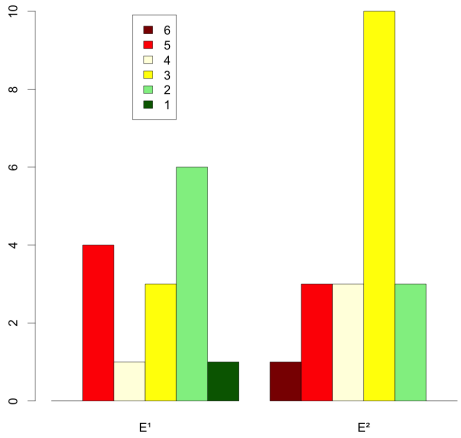



Abbildung 5.143: Vergleich $E^1$ mit $E^2$

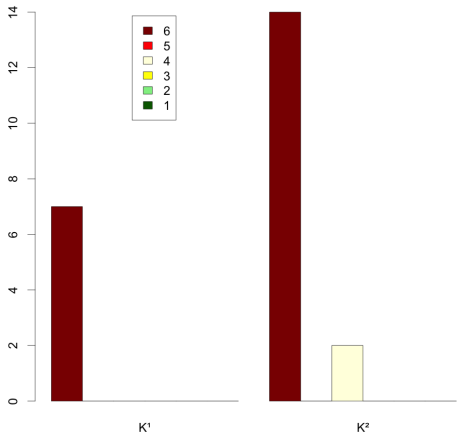



Abbildung 5.144: Vergleich $K^1$ mit $K^2$

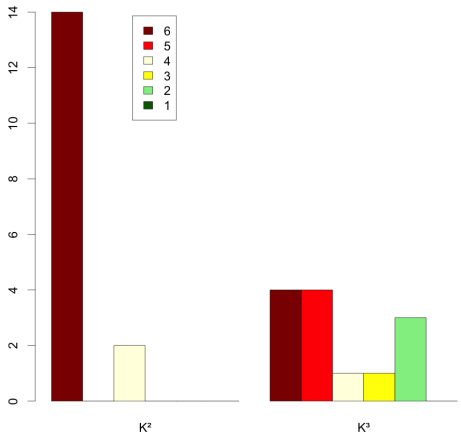



Abbildung 5.145: Vergleich $K^2$ mit $K^3$

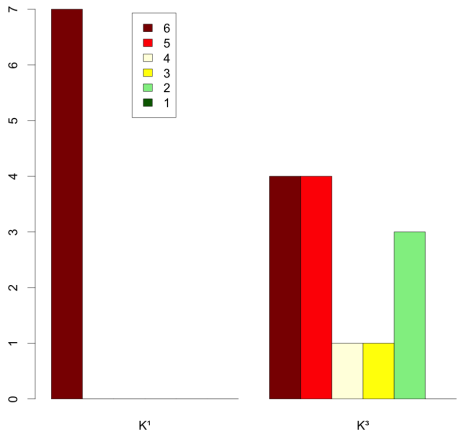



Abbildung 5.146: Vergleich $K^1$ mit $K^3$

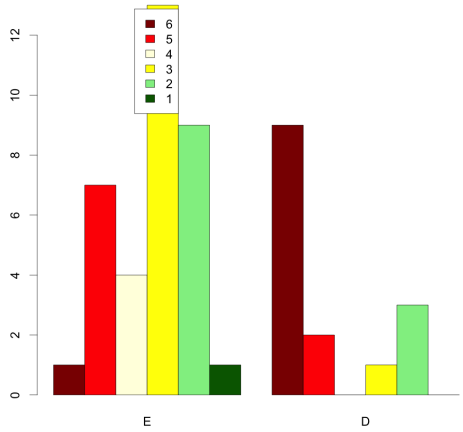



Abbildung 5.147: Vergleich $E$ mit $D$

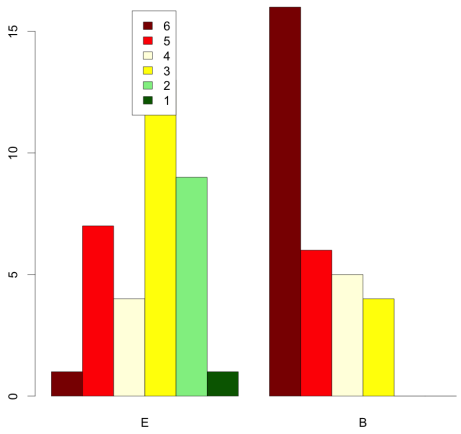



Abbildung 5.148: Vergleich $E$ mit $B$

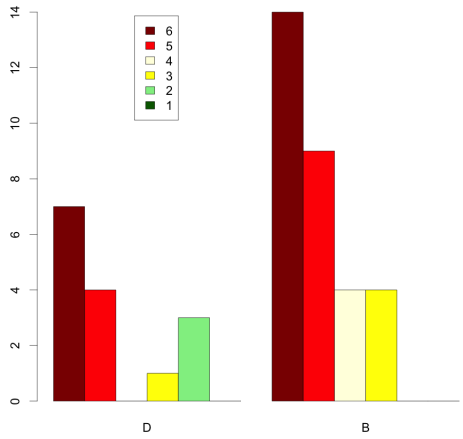



Abbildung 5.149: Vergleich $D$ mit $B$

Für die Gruppen $E$ und $K$ ergibt ein gerechneter Fisher-Test einen $p$-Wert von $p$ kleiner gleich $0,001$. Die Unterschiede der jeweils erreichten Punktzahl zwischen beiden Gruppen sind hoch signifikant.
Die Gruppen $E_1$ und $E_2$ unterscheiden sich durch einen $p$-Wert von $p$ größer als $0,05$ nicht signifikant. Des Weiteren unterscheiden sich auch die Gruppen $K_1$ und $K_2$ nicht signifikant. Vergleicht man die Gruppe $K_3$ jeweils mit $K_1$ und $K_2$, ergeben sich in beiden Fällen sehr signifikante und signifikante Unterschiede. Die $p$-Werte liegen einmal bei $p$ kleiner gleich $0,01$ und $p$ kleiner gleich $0,05$. Auch die Gruppen

$E$ und $D$ sowie $E$ und $B$ unterscheiden sich beide mit einem $p$-Wert von jeweils $p$ kleiner gleich $0,01$ sehr signifikant. Ein Fisher-Test für die Gruppen $D$ und $B$ liefert hingegen einen nicht signifikanten $p$-Wert von $p$ größer als $0,05$.

**Die Fachkompetenz „Informatiksysteme gestalten“**

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ | $B$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 2 | 0 | 0 | 2 | 0 | 0 | 0 | 0 |
| 2 | 2 | 7 | 1 | 1 | 4 | 6 | 0 | 2 | 7 |
| 3 | 2 | 8 | 1 | 1 | 1 | 4 | 0 | 6 | 13 |
| 4 | 16 | 13 | 7 | 9 | 0 | 6 | 7 | 2 | 9 |
| 5 | 12 | 6 | 5 | 7 | 0 | 0 | 6 | 4 | 2 |
| 6 | 3 | 0 | 1 | 2 | 0 | 0 | 0 | 0 | 0 |
| $n$ | $n_E = 35$ | $n_K = 36$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 13$ | $n_D = 14$ | $n_B = 31$ |

Tabelle 5.34: Kontingenztabelle Informatiksysteme gestalten

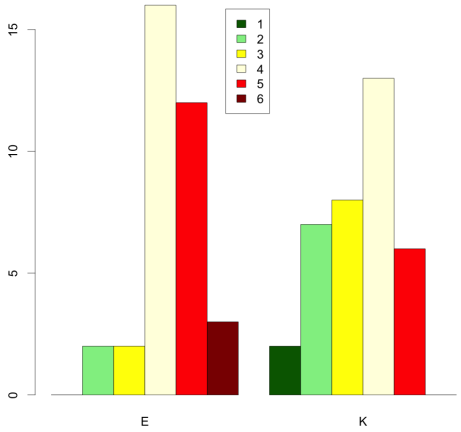



Abbildung 5.150: Vergleich $E$ mit $K$

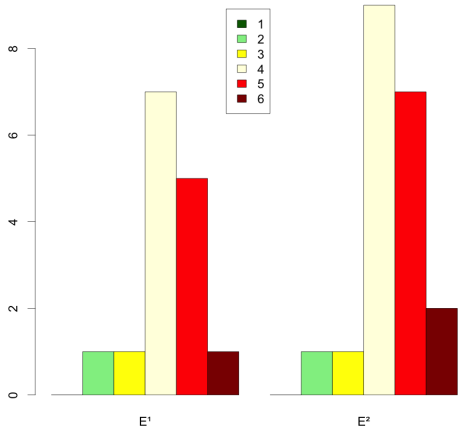



Abbildung 5.151: Vergleich $E^1$ mit $E^2$

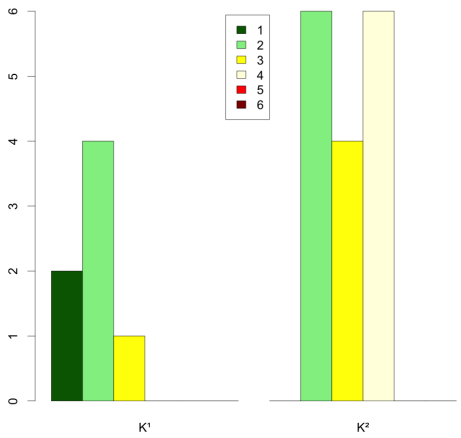



Abbildung 5.152: Vergleich $K^1$ mit $K^2$

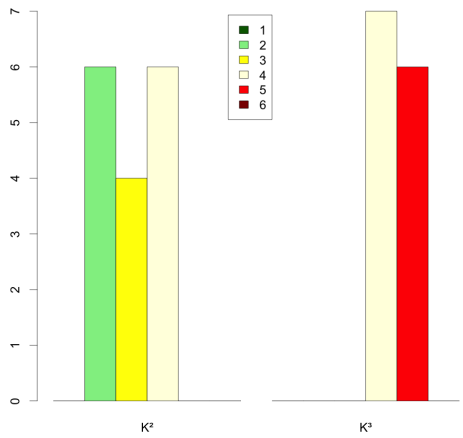



Abbildung 5.153: Vergleich $K^2$ mit $K^3$

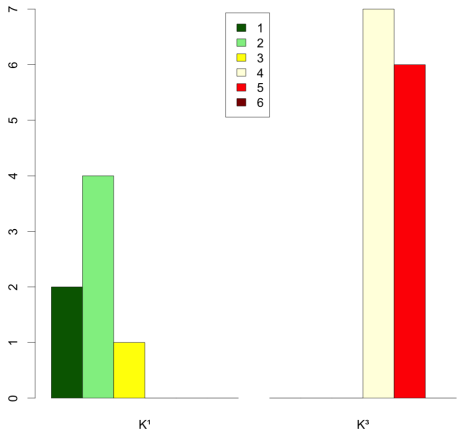



Abbildung 5.154: Vergleich $K^1$ mit $K^3$

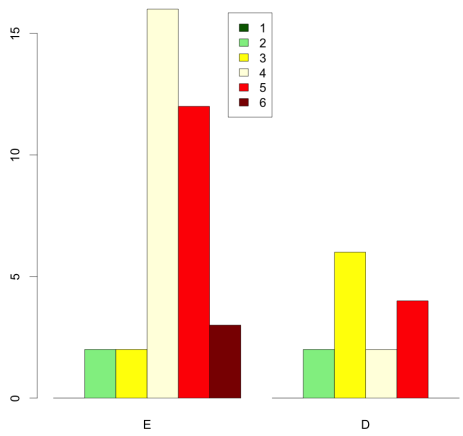



Abbildung 5.155: Vergleich $E$ mit $D$

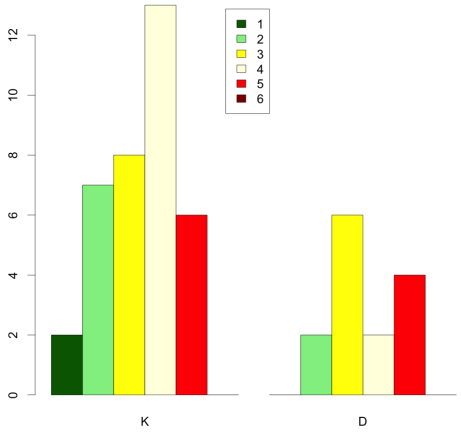



Abbildung 5.156: Vergleich $K$ mit $D$

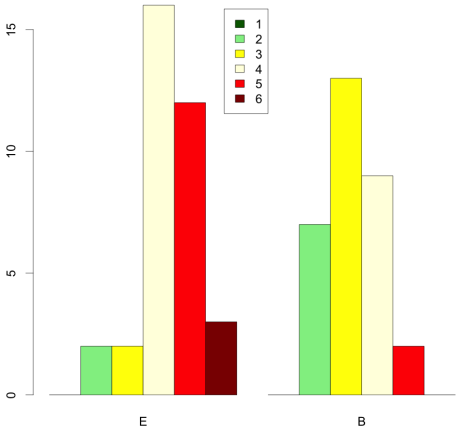



Abbildung 5.157: Vergleich $E$ mit $B$

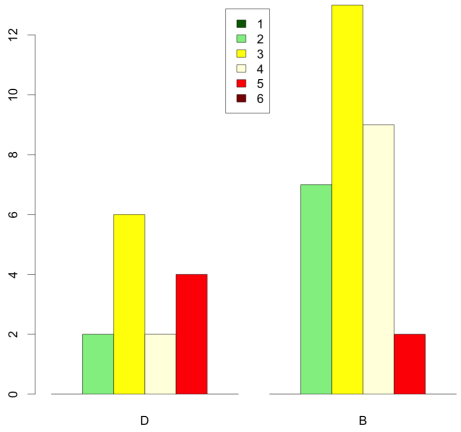



Abbildung 5.158: Vergleich $D$ mit $B$

Die Selbsteinschätzung dieser Kompetenz unterscheidet sich bei den Gruppen $E$ und $K$ durch einen $p$-Wert von $p$ kleiner gleich $0,01$ und ist dadurch sehr signifikant. Die Gruppen $E_1$ und $E_2$ unterscheiden sich nicht signifikant. Die Gruppen $K_1$ und $K_2$ unterscheiden sich mit einem $p$-Wert von $p$ kleiner gleich $0,05$ signifikant, die Gruppen $K_2$ und $K_3$ sowie $K_1$ und $K_3$ weisen mit einem $p$-Wert von $p$ kleiner gleich $0,01$ sehr signifikante Unterschiede auf. Zwischen den Gruppen $E$ und $D$, $D$ und $B$ sowie zwischen $K$ und $D$ gibt es keine signifikanten Unterschiede. Sehr signifikant, weil der $p$-Wert bei $p$ kleiner gleich $0,01$ ist, unterscheiden sich die Selbsteinschät-

zungen der Gruppen $E$ und $B$.

**Die Fachkompetenz „Informatiksysteme analysieren und verstehen“**

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ | $B$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 2 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 2 | 2 | 4 | 1 | 1 | 5 | 2 | 0 | 3 | 9 |
| 3 | 7 | 10 | 3 | 4 | 1 | 6 | 1 | 3 | 10 |
| 4 | 13 | 15 | 7 | 6 | 0 | 8 | 7 | 5 | 9 |
| 5 | 12 | 5 | 3 | 9 | 0 | 0 | 5 | 2 | 2 |
| 6 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| $n$ | $n_E = 35$ | $n_K = 36$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 13$ | $n_D = 14$ | $n_B = 31$ |

Tabelle 5.35: Kontingenztabelle Informatiksysteme analysieren und verstehen

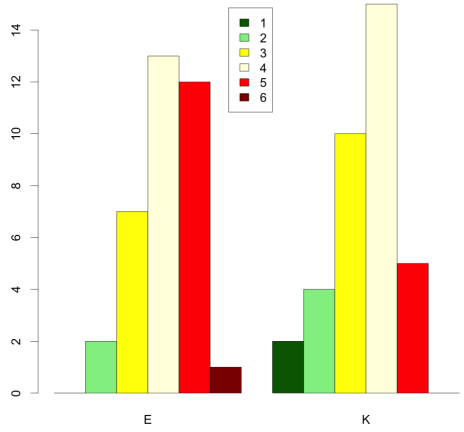



Abbildung 5.159: Vergleich $E$ mit $K$

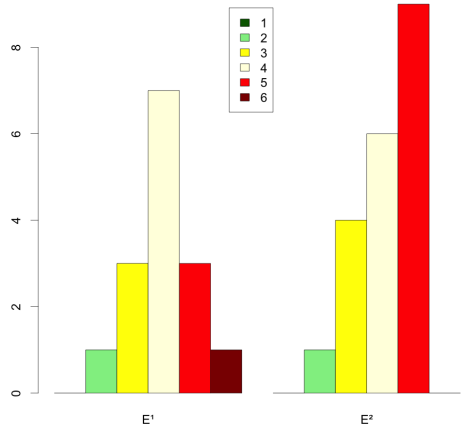



Abbildung 5.160: Vergleich $E^1$ mit $E^2$

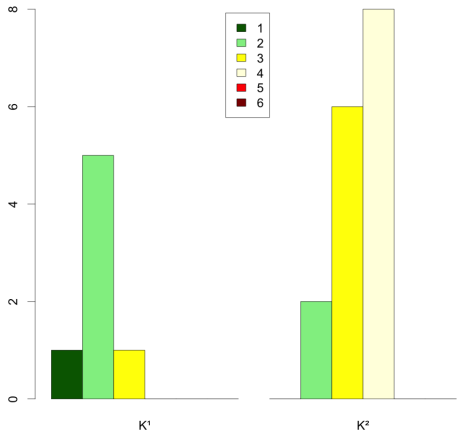



Abbildung 5.161: Vergleich $K^1$ mit $K^2$

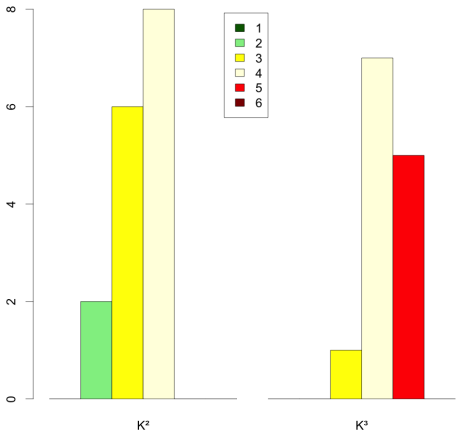



Abbildung 5.162: Vergleich $K^2$ mit $K^3$

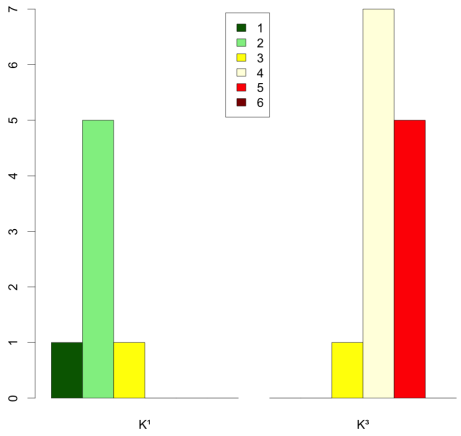



Abbildung 5.163: Vergleich $K^1$ mit $K^3$

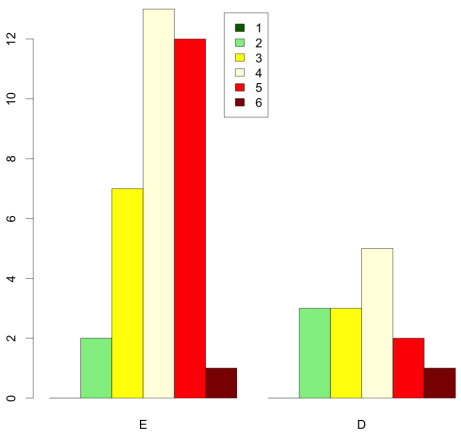



Abbildung 5.164: Vergleich $E$ mit $D$

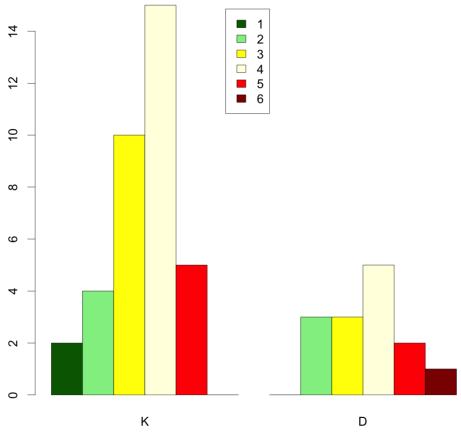



Abbildung 5.165: Vergleich $K$ mit $D$

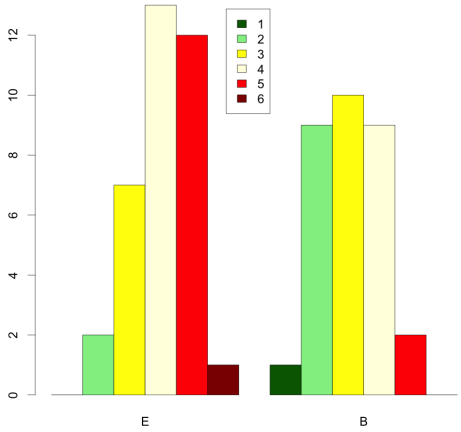



Abbildung 5.166: Vergleich $E$ mit $B$

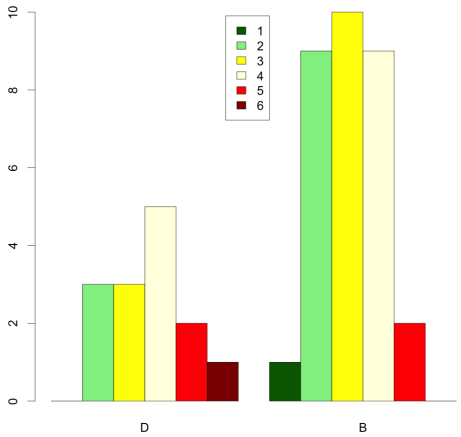



Abbildung 5.167: Vergleich $D$ mit $B$

Die Gruppen $E$ und $K$, $E^1$ und $E^2$, $E$ und $D$, $K$ und $D$ sowie $D$ und $B$ unterscheiden sich nicht signifikant. Die Unterschiede der Gruppen $K^1$ im Vergleich zu $K^2$ und im Vergleich zu $K^3$ sind signifikant, denn der $p$-Wert ist $p$ kleiner gleich $0,05$. Bei den Gruppen $K^1$ und $K^3$ sowie $E$ mit $B$ liegt der $p$-Wert mit $p$ kleiner gleich $0,01$ im sehr signifikanten Bereich.

**Die Fachkompetenz „Darstellen und Interpretieren“**

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ | $B$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 2 | 0 | 0 | 2 | 0 | 0 | 0 | 0 |
| 2 | 1 | 7 | 0 | 1 | 3 | 4 | 0 | 4 | 7 |
| 3 | 2 | 8 | 1 | 1 | 2 | 4 | 1 | 2 | 12 |
| 4 | 11 | 16 | 6 | 5 | 0 | 7 | 9 | 6 | 9 |
| 5 | 18 | 3 | 7 | 11 | 0 | 1 | 3 | 2 | 3 |
| 6 | 3 | 0 | 1 | 2 | 0 | 0 | 0 | 0 | 0 |
| $n$ | $n_E = 35$ | $n_K = 36$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 13$ | $n_D = 14$ | $n_B = 31$ |

Tabelle 5.36: Kontingenztabelle Informatiksysteme analysieren und verstehen

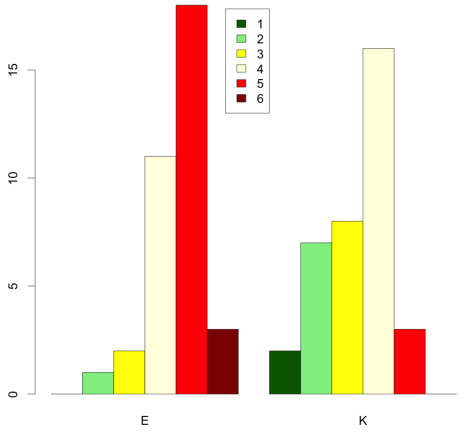



Abbildung 5.168: Vergleich $E$ mit $K$

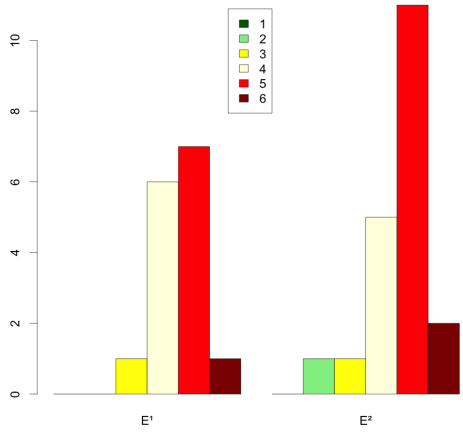



Abbildung 5.169: Vergleich $E^1$ mit $E^2$

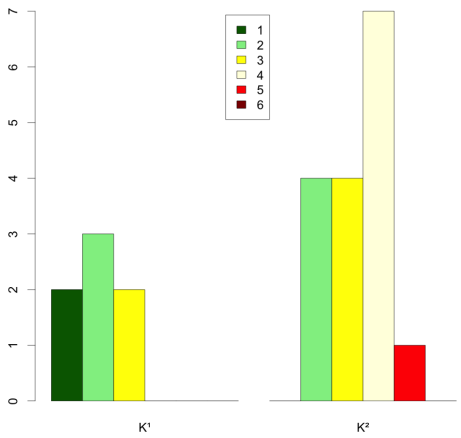



Abbildung 5.170: Vergleich $K^1$ mit $K^2$

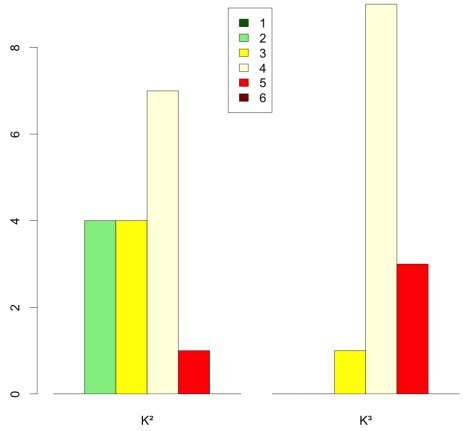



Abbildung 5.171: Vergleich $K^2$ mit $K^3$

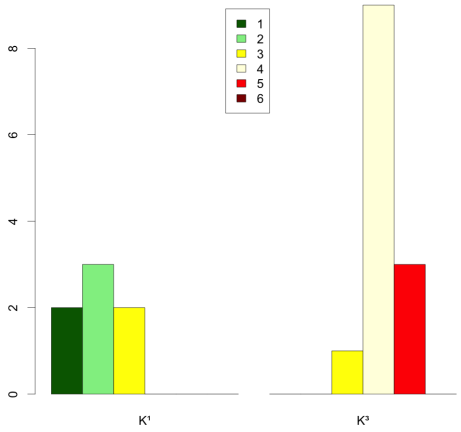



Abbildung 5.172: Vergleich $K^1$ mit $K^3$

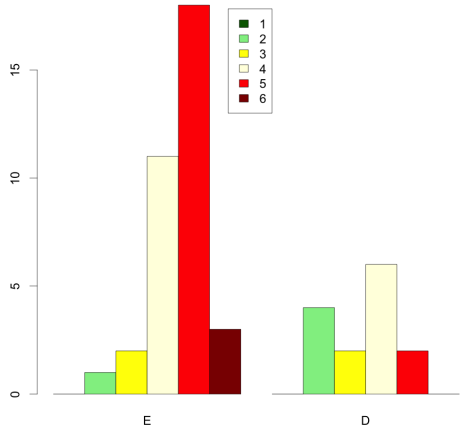



Abbildung 5.173: Vergleich $E$ mit $D$

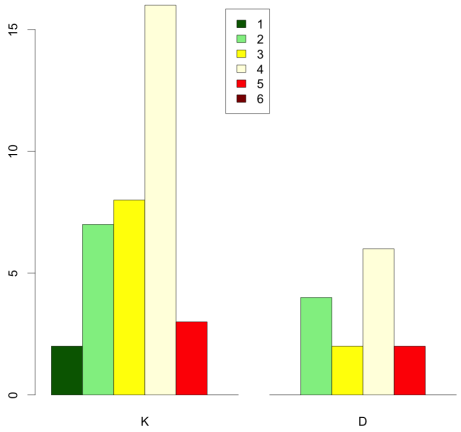



Abbildung 5.174: Vergleich $K$ mit $D$

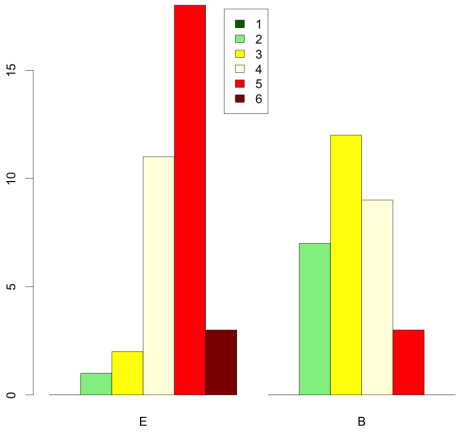



Abbildung 5.175: Vergleich $E$ mit $B$

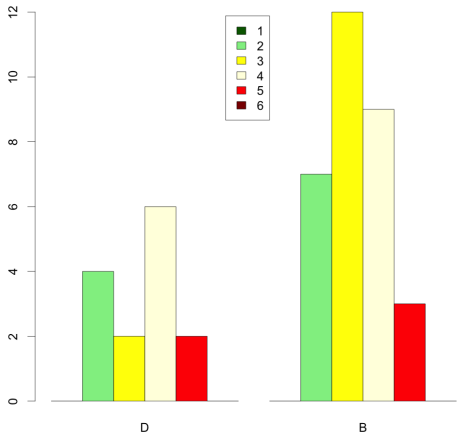



Abbildung 5.176: Vergleich $D$ mit $B$

Die Unterschiede der Gruppen $E$ und $K$, $K^1$ und $K^3$, $E$ und $D$, $D$ und $B$ sowie $E$ und $B$ sind nach einem gerechneten Fisher-Test mit einem $p$-Wert von $p$ kleiner gleich $0,01$ sehr signifikant. Die Gruppen $K^1$ und $K^2$ unterscheiden sich mit einem $p$-Wert von $p$ kleiner gleich $0,05$ signifikant voneinander. Die Gruppen $E^1$ und $E^2$, $K^2$ und $K^3$, $K$ und $D$ sowie $D$ und $B$ unterscheiden sich nicht signifikant.

**Die Fachkompetenz „Begründen und Bewerten“**

| | $E$ | $K$ | $E^1$ | $E^2$ | $K^1$ | $K^2$ | $K^3$ | $D$ | $B$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 0 | 2 | 0 | 0 | 0 | 1 |
| 2 | 0 | 9 | 0 | 0 | 4 | 5 | 0 | 2 | 4 |
| 3 | 3 | 12 | 1 | 2 | 0 | 6 | 1 | 3 | 14 |
| 4 | 11 | 10 | 4 | 7 | 1 | 5 | 8 | 5 | 9 |
| 5 | 14 | 4 | 8 | 6 | 0 | 0 | 4 | 3 | 3 |
| 6 | 7 | 0 | 2 | 5 | 0 | 0 | 0 | 1 | 0 |
| $n$ | $n_E = 35$ | $n_K = 36$ | $n_{E^1} = 15$ | $n_{E^2} = 20$ | $n_{K^1} = 7$ | $n_{K^2} = 16$ | $n_{K^3} = 13$ | $n_D = 14$ | $n_B = 31$ |

Tabelle 5.37: Kontingenztabelle Begründen und Bewerten

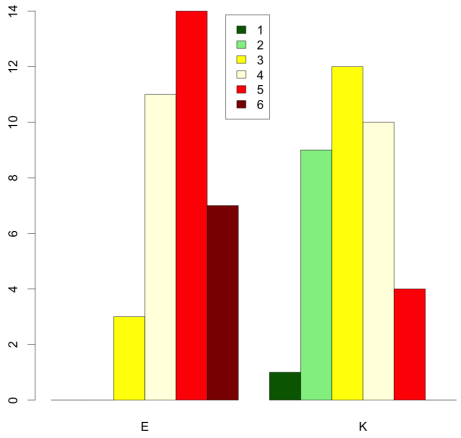



Abbildung 5.177: Vergleich $E$ mit $K$

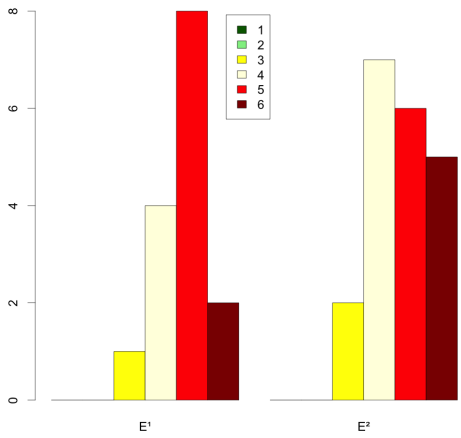



Abbildung 5.178: Vergleich $E^1$ mit $E^2$

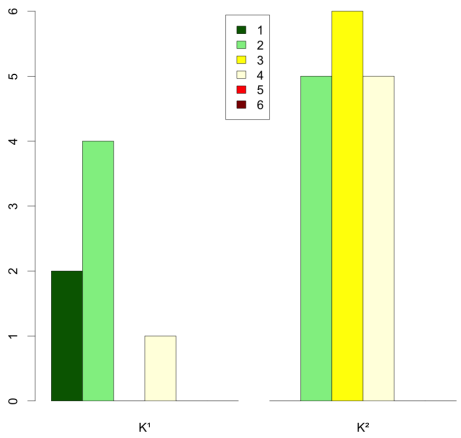



Abbildung 5.179: Vergleich $K^1$ mit $K^2$

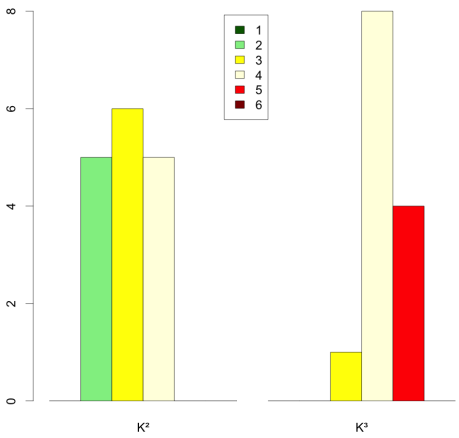



Abbildung 5.180: Vergleich $K^2$ mit $K^3$

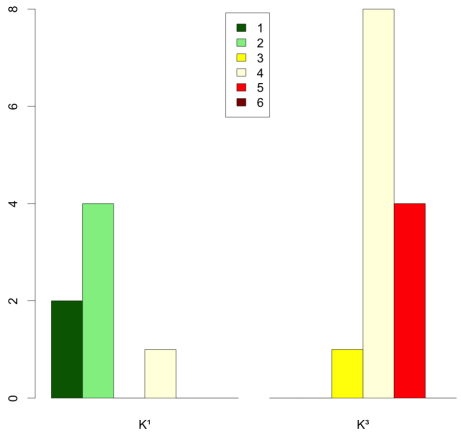



Abbildung 5.181: Vergleich $K^1$ mit $K^3$

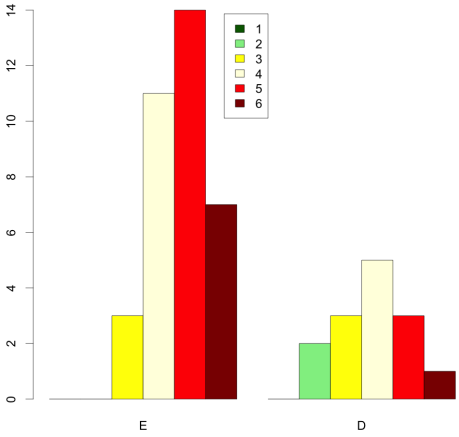



Abbildung 5.182: Vergleich $E$ mit $D$

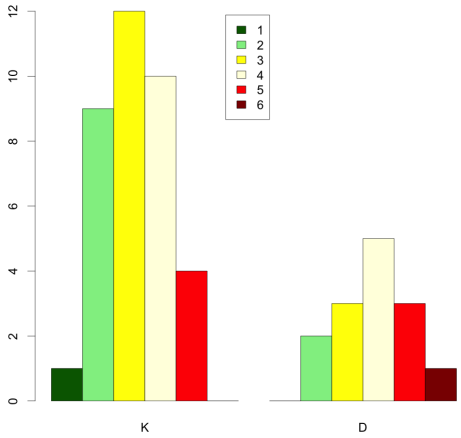



Abbildung 5.183: Vergleich $K$ mit $D$

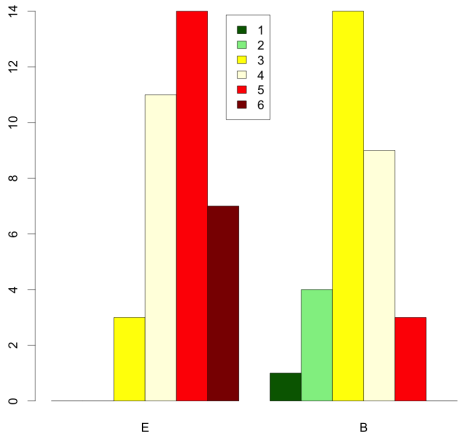



Abbildung 5.184: Vergleich $E$ mit $B$

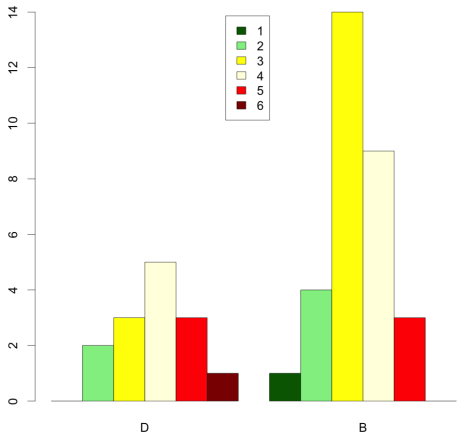



Abbildung 5.185: Vergleich $D$ mit $B$

Die Unterschiede in den Gruppen $E$ und $K$, $K^2$ und $K^3$, $K^1$ und $K^3$ sowie $E$ und $B$ sind nach einem gerechneten Fisher-Test und einem $p$-Wert von $p$ kleiner gleich $0,01$ sehr signifikant. Die Gruppen $K^1$ und $K^2$ unterscheiden sich mit einem $p$-Wert von $p$ kleiner gleich $0,05$ signifikant. Zwischen $E^1$ und $E^2$, $E$ und $D$, $K$ und $D$ sowie $D$ und $B$ gibt es keine signifikanten Unterschiede.

## 5.7 Diskussion der Ergebnisse

Die Untersuchung zeigte, dass sowohl die Mensch-Maschine-Kommunikation mit gesprochener Sprache als auch die Sprachübersetzung zur Lebenswelt der Schüler gehören und sie zur Beschäftigung mit Informatik motivieren (Hypothese H1). Probleme mit der Software inES bzw. inGE, wie sie bei der Gruppe $D$ und der Gruppe $B$ auftraten, beeinträchtigen sowohl die Motivation als auch das Interesse am Unterrichtsfach Informatik.

Die Untersuchung zeigte zudem, dass sich die Schüler die theoretischen Konzepte der Informatik mithilfe der Lernumgebungen inES und inGE handlungsorientiert aneignen können (Hypothese H2), denn:

- Die Handlungsorientierung, überprüft an der Variablen „Ganzheitlichkeit“, ist bei der Experimentalgruppe $E$ signifikant stärker ausgeprägt als bei der Kontrollgruppe $K$, d. h. dass die neue Unterrichtseinheit bei den Schülern der Experimentalgruppe $E$ mehr Sinne ansprach als bei der alten Unterrichtseinheit. Die neue Unterrichtseinheit unterstützt das Lernen im Einklang mit Kopf-, Herz- und Handarbeit besser.
- Die Schüler der Experimentalgruppe $E$ schätzten sich bezüglich ihrer Reflexionskompetenz besser ein, d. h. sie konnten diese theoretischen Informatikthemen leichter auf neue Sachverhalte übertragen und anwenden, denn die Lerngegenstände sind durch den flexibleren Umgang mit den Kontexten der neuen Unterrichtseinheit vielfältiger und nicht nur an einen speziellen Kontext wie bei der Sprachübersetzung gebunden. Des Weiteren fühlten sie sich im Umgang mit den theoretischen Themen der Informatik sicherer.
- Der Unterschied bezüglich der Variablen „Schüleraktivität“ ist zwar beim Vergleich der Gruppen $E$ und $K$ nicht signifikant, jedoch zeigte sich, dass innerhalb der Gruppe $K$, genauer zwischen $K^1$ und $K^3$ sowie $K^2$ und $K^3$ signifikante Unterschiede bestehen, d. h. während die Schüler der Experimentalgruppe $E$ sich bei allen Gruppen aktiv beteiligen konnten, bestehen Unterschiede in der Kontrollgruppe $K$, in der möglicherweise der Unterricht nicht immer alle Schüler zur selbstständigen Arbeit am Lerngegenstand motivieren konnte.

Starken Einfluss auf die Variablen „Ganzheitlichkeit“ und „Reflexion“ hat dabei die Funktionstüchtigkeit der Software inES bzw. inGE. Insbesondere die Software inES

ermöglicht ein „Lernen mit allen Sinnen“. Wird der störende Einfluss einer nicht oder nur teilweise funktionierenden Software mitberücksichtigt, lassen sich keine signifikanten Unterschiede zwischen den Gruppen $E$ und $K$ erkennen. Der Vergleich von $K$ mit $D$ (vgl. Abb. 5.86) zeigte sogar, dass durch Funktionsbeeinträchtigungen von inES die Selbsteinschätzungen der Schüler hinsichtlich der Variablen „Reflexion“ zugunsten von $K$ ändern.

Sowohl die alte als auch die neue Unterrichtseinheit fördert durch das selbstständige Implementieren und Testen eigener Grammatiken die Kreativität (Hypothese H3). Diesbezüglich konnte kein signifikanter Unterschied zwischen der Experimentalgruppe und der Kontrollgruppe nachgewiesen werden. In der neuen Unterrichtseinheit wurden aber deutlich mehr verschiedenartige Projekte bearbeitet als in der alten Unterrichtseinheit. Bei der alten Unterrichtseinheit zeigte sich, dass innerhalb der Kontrollgruppe hoch signifikante Unterschiede zwischen der Gruppe $K^1$ und $K^3$ bestehen. Das könnte auf eine starke Lenkung der Lehrkraft und wenige Freiheiten für die Entfaltung von Schülerideen zurückzuführen sein.
Dass die bereits beschriebenen Einschränkungen bezüglich der Software inES bzw. inGE starke Einflüsse auf die untersuchten Variablen haben können, zeigte sich auch hier beim Vergleich der Gruppen $E$ mit $D$.

Die in der neuen Unterrichtseinheit praktizierte zustandsorientierte Modellierung hat sich bewährt (Hypothese H4). Hoch signifikante Unterschiede zwischen der Experimentalgruppe und der Kontrollgruppe bei den Ergebnissen der Testfragen T01-T07 zeigen, dass die Schüler der Experimentalgruppe das theoretische Konzept eines Parsers besser verstanden haben. Letztere benutzten viel häufiger als die Schüler der Kontrollgruppe die korrekten Fachbegriffe für ihre Beschreibung eines Kellerautomaten. Die qualitativen Daten zeigen wie die quantitativen Daten auch, dass nur unter der Voraussetzung einer funktionierenden Software inES bzw. inGE diese Ergebnisse ermöglicht werden.

Auch bei der Implementierung von formalen Grammatiken mithilfe verschiedener Darstellungsformen (Hypothese H5) gab es hoch signifikante Unterschiede zwischen der Experimentalgruppe und der Kontrollgruppe. Wie bei der Hypothese H4 zeigen die Ergebnisse der Testfragen T01-T07, dass die Schüler der Gruppe Experimen-

talgruppe $E$ im Vergleich zur Kontrollgruppe $K$ alle im Rahmenplan und A-Heft aufgeführten Ziele erreichen. Von den 35 Schülern der Experimentalgruppe $E$ konnten 30 Schüler den Begriff bzw. den Verwendungszweck einer Formalen Grammatik korrekt beschreiben. Die korrekten Fachbegriffe benutzten die Schüler der Experimentalgruppe analog zu H4 viel mehr als die Schüler der Kontrollgruppe. Wieder zeigte sich dabei, dass die Ergebnisse nur unter der Voraussetzung einer funktionierenden Software inES bzw. inGE erreicht werden können.

Keine signifikanten Unterschiede zwischen der Experimentalgruppe und der Kontrollgruppe konnten hinsichtlich der Visualisierung des gesamten Problemlöseprozesses (Hypothese H6) nachgewiesen werden. Innerhalb beider Gruppen gibt es jedoch signifikante bis hoch signifikante Unterschiede. Die Schüler der Experimentalgruppe $E^1$ fühlten sich von der Software signifikant besser unterstützt als die Schüler der Experimentalgruppe $E^2$. Worin die Ursachen dafür liegen, müsste in einer weiteren Forschungsarbeit genauer untersucht werden.

Die Unterrichtseinheit Mensch-Maschine-Kommunikation mit gesprochener Sprache spricht Schülerinnen und Schüler nicht gleichermaßen an (Hypothese H7). Der Kontext gehört für beide gleichermaßen zu ihrer Lebenswirklichkeit und motiviert sie gleichermaßen zur Beschäftigung mit diesem Thema. Signifikante Unterschiede zugunsten der Schüler zeigten sich aber hinsichtlich des ganzheitlichen Lernens, der empfundenen Eigenaktivität und der Kreativität. Sowohl die Schüler als auch die Schülerinnen empfanden die Möglichkeit des ganzheitlichen Lernens, der Eigenaktivität, der Kreativität durchaus positiv. Die Schülerinnen hätten sich diesbezüglich aber vielleicht noch mehr Gestaltungsspielraum gewünscht. Die Fähigkeit, Erlerntes auf neue Sachverhalte zu übertragen und anwenden zu können, gelingt den Schülerinnen der Kontrollgruppe signifikant häufiger als den Schülern dieser Gruppe.

Zusammenfassend hat die empirische Untersuchung gezeigt, dass die kontextorientierte Unterrichtseinheit zur Mensch-Maschine-Kommunikation mit gesprochener Sprache basierend auf IniK und unter Einsatz der Unterrichtssoftware inES einen pädagogischen Mehrwert hat (Hypothese H8).

- Mit der neuen Unterrichtseinheit werden alle im aktuellen A-Heft angeführten Zielsetzungen erreicht.
- Die Ergebnisse des Tests sind in der Gruppe $E$ hoch signifikant besser ausgefallen als in der Gruppe $K$.

- Die Testergebnisse der Experimentalgruppe $E$ sind besser, obwohl die Schüler im Vergleich zur Kontrollgruppe $K$ signifikant häufiger fehlten und damit nicht alle Unterrichtsstunden wahrnehmen konnten.
- Die Gruppe $E$ kommt, obwohl sie im Vergleich zur Gruppe $K$ im Eingangstest deutlich schlechter abgeschnitten hat, zu hoch signifikant besseren Ergebnissen.
- In den Bereichen „Informatiksysteme gestalten“, „Darstellen und Interpretieren“ und „Begründen und Bewerten“ schätzten die Schüler der Experimentalgruppe $E$ signifikant häufiger als die Schüler der Kontrollgruppe $K$ ein, dass die Kompetenzen erreicht wurden. Im Kompetenzbereich „Informatiksysteme analysieren und verstehen“ ergaben die Selbsteinschätzungen keine signifikanten Unterschiede.
- Die Ergebnisse beider Experimentalgruppen $E^1$ und $E^2$ sind signifikant besser als die der besten Kontrollgruppe $K^3$, die ihrerseits signifikant bessere Ergebnisse erreicht hat als die Gruppen $K^1$ und $K^2$.
- Die Analyse zeigte zudem, dass der pädagogische Mehrwert nicht auf Kosten von Unterrichtszeit erreicht wurde. Im Gegenteil: Die Lehrkräfte der Experimentalgruppen setzten deutlich weniger Unterrichtsstunden ein als die Lehrkräfte der Kontrollgruppen.

Die Ergebnisse zeigten sehr deutlich, dass der pädagogische Mehrwert der Unterrichtseinheit sowohl vom Unterrichtskonzept IniK als auch vom Einsatz der Software inES einschließlich des Plugins inGE abhängt.

# Kapitel 6

# Zusammenfassung und Ausblick

Ausgangspunkt für die in der Arbeit vorgelegte Studie war die persönliche Erkenntnis, dass der bis dato praktizierte Informatikunterricht zum Themenbereich Sprachverarbeitung nicht den Anforderungen genügte und dazu führte, dass die Schüler teilweise demotiviert aufgaben, in Klausuren und anderen Leistungsnachweisen nicht überzeugten, und der Unterricht im schlimmsten Fall dazu beitrug, das teilweise noch vorherrschende Negativimage der Informatik zu verstärken. In einem ersten Schritt sollte die Studie herausfinden, worin die konkreten Ursachen für das Scheitern der Ziele des bisherigen Curriculums über die Sprachverarbeitung liegen. Dazu wurden Experteninterviews geführt, die zeigten dass die Ursache vor allem im anwendungsorientierten Informatikunterricht mit starker Betonung der Algorithmik und der Besonderheit zwingend funktional programmieren zu müssen, liegt (vgl. Kapitel 3, S. 44 uff). Es zeigte sich, dass im Extremfall reine Programmierkurse stattfanden, die unabhängig vom anwendungsorientierten Ansatz deplaciert sind. Diese Überbetonung des funktionalen Programmierens führte dazu, dass andere Dimensionen des Anwendungskontextes Sprachübersetzung nur ansatzweise oder gar nicht berücksichtigt werden konnten und die Schüler kein tiefes Verständnis der Konzepte formaler Sprachen und Grammatiken im Allgemeinen und der Arbeitsweise eines Parsers im Besonderen erwerben konnten.

Auf der Grundlage dieser Erkenntnisse sowie der Analyse der aktuellen fachdidaktischen Literatur wurde in einem zweiten Schritt eine neue überarbeitete Unterrichtseinheit sowie eine eigens dafür entwickelte Lernumgebung vorgestellt und deren pädagogischer Mehrwert in einer quantitativen Studie mit quasi-experimentellem Charakter nachgewiesen (vgl. Kapitel 5, S. 98 uff). Die neue Unterrichtseinheit be-

rücksichtigt konsequent die aktuellen fachdidaktischen Entwicklungen und genügt den von Alisch und Breier postulierten Merkmalen eines zeitgemäßen Informatikunterrichts (vgl. [AB14]). Dass der Informatikunterricht

- informationsorientiert,
- allgemeinbildend,
- kompetenzorientiert,
- kontextorientiert,
- medienaffin,
- gendersensibel,
- sprachfördernd,
- fächerverbindend und
- projektorientiert

durchgeführt wird, ist für den erfolgreichen Einsatz der neuen Unterrichtseinheit essentiell. In der Studie wurde nachgewiesen, dass mit dem neuen Unterrichtskonzept in Kombination mit den Lernumgebungen inES und inGE die Lernziele früherer Rahmenpläne besser im Sinne von effizienter und nachhaltiger erreicht werden und im Gegensatz zum alten Ansatz mit der neuen Unterrichtseinheit alle im aktuellen A-Heft angeführten Zielsetzungen nun wirklich erreicht werden. Das macht den pädagogischen Mehrwert aus.
Allerdings wird ein pädagogischer Mehrwert nur dann erreicht, wenn sich Lernumgebungen, wie z. B. inES mit vollem Funktionsumfang nutzen lassen und stabil im Unterricht eingesetzt werden können. Die Lernumgebung inES mit inGE existiert derzeitig nur als Desktopversion für die Betriebssysteme MacOS X, Windows und Linux. Durch die starke Abhängigkeit von externen Spracherkennungs- und Sprachsynthese-Bibliotheken der jeweils zugrunde liegenden Betriebssysteme und die damit verbundene Abhängigkeit der unterschiedlichen Entwicklungsstände zeigen, dass der zuverlässige Einsatz noch nicht überall gewährleistet ist. Hier muss in zukünftigen Entwicklungen die Lernumgebung sukzessive verbessert werden. Vorstellbar ist die vollständige Portierung dieser Lernumgebung ins Web, eine Möglichkeit, die es zu Beginn der Entwicklungsarbeit an inES mit inGE noch nicht gab. Beispielsweise wäre eine Anbindung an Googles Spracherkennung und Sprachsynthese möglich und würde es zulassen, dass sich zukünftig eine derartige Lernumgebung völlig unabhängig von jedem Betriebssystem einsetzen lässt und aufwendige

Installationen entfallen. Dadurch könnte, wie im Strategiepapier der KMK gefordert, die Lernumgebung inES mit inGE unabhängig vom genutzten Gerätetyp oder der eingesetzten Plattform, genutzt werden (vgl. [Kul16], S. 32). Außerdem ergeben sich weitere interessante fachdidaktische Anwendungen. Beispielsweise könnten unter Einbeziehung in bestehende didaktische Contentmanagementsysteme, z. B. moodle oder it's Learning, in Form von Anbindungsmodulen neue Prüfungssituationen entstehen (vgl. [Kul16], S. 14).
In dieser Arbeit wurde zudem erstmals der empirisch gesicherte Nachweis erbracht, dass und wie sich Schüler die in den Bildungsstandards für die Sekundarstufe II im Inhaltsbereich „Automaten und Sprache" formulierten Kompetenzen im Informatikunterricht aneignen können, die deutlich über die im A-Heft formulierten Anforderungen hinausgehen. In der Studie wurde außerdem gezeigt, dass das Unterrichtskonzept Informatik im Kontext (IniK) auch in der gymnasialen Oberstufe erfolgreich genutzt werden kann. Es kam bisher nur in der Sekundarstufe I zum Einsatz und wurde in den Bildungsstandards für die Sekundarstufe II nicht explizit erwähnt. Bei geeigneter Wahl des Kontextes könnte dieses Vorgehen auch für andere Themenbereiche in der gymnasialen Oberstufe geeignet sein.

Das KMK Strategiepapier sieht derartige Bildungsinhalte nicht vor. Der im Dokument verwendete Bildungsbegriff führt in der Argumentation des Strategiepapiers zu der Kernforderung, digitale Medien in den Unterrichtsprozessen aller Fächer zu verorten. „Dieser Forderung kann - ebenso wie der nach einer grundlegenden Transformation von Schulen im Lichte der fortschreitenden Digitalisierung - zwar grundsätzlich zugestimmt werden, allerdings ist sie nicht weitreichend genug und bereitet Lernende damit unzureichend auf die Herausforderungen der ‚digitalen Welt' vor" (vgl. [Bri16], S. 2). Die im Strategiepapier der KMK sichtbare Auffassung, man könne die informatische Bildungsinhalte auf andere allgemeinbildende Fächer auslagern und von Nichtinformatikern unterrichten lassen ist nicht zeitgemäß. Die neue Unterrichtseinheit erfordert ein Fach Informatik und ausgebildete Lehrkräfte, die über die notwendigen fachlichen und fachdidaktischen Voraussetzungen verfügen, um diesen Unterricht durchführen zu können. Insbesondere der Punkt 5.5 auf Seite 18 des vorliegenden Strategiepapiers:

> „**Algorithmen erkennen und formulieren**
> 5.5.1 Funktionsweisen und grundlegende Prinzipien der digitalen Welt kennen und verstehen,

5.5.2 Algorithmische Strukturen in genutzten digitalen Tools erkennen und formulieren,
5.5.3 Eine strukturierte, algorithmische Sequenz zur Lösung eines Problems planen und verwenden (vgl. [Kul16], S. 18)“

ist wichtig, jedoch nicht ausreichend. Sollen Schüler Funktionsweisen und grundlegende Prinzipien der digitalen Welt verstehen, benötigen sie informatisches Konzeptwissen, das weit über algorithmische Strukturen hinausgeht und nur von Lehrern gelehrt werden kann, die dem Personenkreis der grundständig studierten Lehrkräfte angehören. Des Weiteren zeigt sich, dass dieses informatische Konzeptwissen nur in einem eigenständigen Schulfach Informatik unterrichtet werden kann. Wie das gehen könnte zeigt beispielsweise die Schweiz mit dem vielversprechenden Lehrplan 21 (vgl. [Kon]).
Die Ergebnisse dieser Arbeit könnten auch Impulse für die Hochschuldidaktik liefern. In der Regel finden bundesweit Lehrveranstaltungen zu formalen Grundlagen der Informatik statt, die Wissen überwiegend deduktiv vermitteln. Nicht selten liegen die Durchfallquoten von Lehrveranstaltungen zur theoretischen Informatik bei 50% (vgl. [KK12], S. 21). Der frühzeitige Einbau einiger induktiver Lernphasen (Lernen vom Speziellen zum Allgemeinen), wie sie in der hier dokumentierten, kontextorientierten Unterrichtseinheit zur Mensch-Maschine-Kommunikation mit gesprochener Sprache erfolgreich praktiziert werden, könnten unter Einbeziehung der Lernumgebungen inES und inGE durchaus auch hochschuldidaktisch erfolgversprechend sein und helfen, die hohen Abbrecherquoten zu senken.

# Literaturverzeichnis


[AB14] Sven Alisch und Norbert Breier. Zehn Thesen zu einem zeitgemäßen Informatikunterricht. `http://d-64.org/gastbeitrag-zehn-thesen-zu-einem-zeitgema%CC%88sen-informatikunterricht/#more-1519`, zuletzt überprüft am 26.03.2017, März 2014.

[AJS11] Claus Albowski, Michael Janneck, und Monika Seiffert. Bildungsplan Informatik - Sekundarstufe I, 2011.

[AJSS09] Claus Albowski, Michael Janneck, Jan Schöttler, und Monika Seiffert. Bildungsplan Gymnasiale Oberstufe. http://www.hamburg.de/contentblob/1475204-/data/informatik-gyo.pdf, 2009.

[Ali13] Sven Alisch. Results of an expert interview as foundation for a study about the pedagogical added value by informatics in context. In *WiPSE '13: Proceedings of the 8th Workshop in Primary and Secondary Computing Education*. ACM Digital Library, 2013.

[AU01] Uwe Aßmann und Theo Ungerer. Informatik in der Schule. In *Informatik Spektrum*, 2001.

[BBF+06] Norbert Breier, Torsten Brinda, Michael Fothe, Steffen Friedrich, Bernhard Körber, und Hermann Puhlmann. Neuer Wein in alten Schläuchen? In *Computer und Unterricht - Heft 63*, 2006.

[BFO+04] Hartmut Bluhm, Uwe Fricke, Torsten Otto, Tammo Ricklefs, und Christian Siegel. *Bildungsplan Gymnasiale Oberstufe*. Freie und Hansestadt Hamburg, 2004.

[Bil08] Arbeitskreis Bildungsstandards. Grundsätze und Standards für die Informatik in der Schule - Bildungsstandards Informatik für die Sekundarstufe 1. http://www.informatikstandards.de/docs/bildungsstandards_2008.pdf, zuletzt überprüft am 26.03.2017, 2008.

[Bil16] A. Bildungsstandards. Bildungsstandards Informatik für die Sekundarstufe II. http://www.informatikstandards.de/docs/Bildungsstandards_SII.pdf, zuletzt überprüft am 26.03.2017, 2016.

[Bor05] Jürgen Bortz. *Statistik für Sozialwissenschaftler*. Springer, 2005.

[BPI05] Bildungsplan Informatik - Baden-Württemberg, zuletzt geprüft am 26.03.2017. http://www.lehrer.uni-karlsruhe.de/~za171/FS_Informatik/gy_s_inf.pdf, 2005.

[BPI13] Bildungsplan Informatik - Bayern. http://www.isb-gym8-lehrplan.de/contentserv/3.1.neu/g8.de/-index.php?StoryID=26193, zuletzt überprüft am 26.03.2017, 2013.

[Bre95] Norbert Breier. Vorlesungsskript Theoretische Informatik. http://www.oberstufeninformatik.de/theorie/Vorl_SS95.pdf, zuletzt überprüft am 26.03.2017, 1995.

[Bre03] Norbert Breier. Quo vadis Informatikunterricht, und welchen Beitrag kann die Informatik für die künftigen Informatiklehrer leisten. https://www.ew.uni-hamburg.de/ueber-die-fakultaet/personen/breier/files/vortrag271003.pdf, zuletzt überprüft am 26.03.2017, 2003.

[Bri16] Thorsten Brinda. Stellungnahme zum KMK-Strategiepapier "Bildung in der digitalen Welt". https://fb-iad.gi.de/fileadmin/stellungnahmen/gi-fbiad-stellungnahme-kmk-strategie-digitale-bildung.pdf, zuletzt überprüft am 26.03.2017, 07 2016.

[Cla95] Volker Claus. Informatik in der Schule als Sprachen-Unterricht. In Sigrid Schubert, editor, *Innovative Konzepte für die Ausbildung*, volume Informatik aktuell. Springer, 1995.

[Coy05] Wolfgang Coy. Informatik ... im Großen und Ganzen. *LOG IN Nr. 136/137*, 2005.

[DKW11] Ira Diethelm, Jochen Koubek, und Helmut Witten. IniK - Informatik im Kontext. *LOG IN*, 2011.

[EGH+03] Mario Eschrich, Hannes Gutzer, Henry Herper, Hans Lehmann, und Jörn Zuber. Rahmenrichtlinien Gymnasium - Sachsen-Anhalt, zuletzt geprüft am 26.03.2017. `http://www.bildung-lsa.de/pool/RRL_Lehrplaene/infogym.pdf`, zuletzt überprüft am 26.03.2017, 2003.

[Eng05] Dieter Engbring. Informatik im Kontext. *LOG IN Nr. 136/137*, 2005.

[EP02] Katrin Erk und Lutz Priese. *Theoretische Informatik - Eine umfassende Einführung.* Springer, 2002.

[EPA04] EPA Informatik für Niedersachsen. `http://db2.nibis.de/1db/cuvo/datei/epa_informatik.pdf`, zuletzt überprüft am 26.03.2017, 2004.

[fBuS11] Sächsischen Staatsinstitut für Bildung und Schulentwicklung. Lehrplan Gymnasium - Sachsen. `http://www.bildung.sachsen.de/apps/lehrplandb-/downloads/lehrplaene/lp_gy_informatik_2011.pdf`, zuletzt überprüft am 26.03.2017, 2011.

[fIeV99] GI Gesellschaft für Informatik e. V. Empfehlung Informatische Bildung und Medienerziehung. *LOG IN 19*, 1999.

[FKM16] Michael Fothe, Bernhard Körber, und Jürgen Müller. Von Leitbildern und Pionierinnen. *LOG IN Nr. 183/184*, 2016.

[For90] Hermann J. Forneck. Entwicklungstendenzen und Problemlinien der Didaktik der Informatik. In *Beiträge zur Didaktik der Informatik*, 1990.

[FvKS05] Uwe Flick, Ernst von Kardorff, und Ines Steinke. *Qualitative Forschung: Ein Handbuch.* rowohlts enzyklopädie, 2005.

[Gei90] Georg Geiser. *Mensch-Maschine-Kommunikation.* Oldenbourg, 1990.

[GI200] Empfehlungen für ein Gesamtkonzept zur informatischen Bildung an allgemeinbildenden Schulen Empfehlungen für ein Gesamtkonzept zur informatischen Bildung an allgemeinbildenden

Schulen. https://www.gi.de/fileadmin/redaktion/empfehlungen/gesamtkonzept_26_9_2000.pdf, zuletzt überprüft am 26.03.2017, 09 2000.

[Hau00] Roland Hausser. *Grundlagen der Computerlinguistik : Mensch-Maschine-Kommunikation in natürlicher Sprache.* Springer, 2000.

[HB02] Peter Hubwieser und Norbert Breier. An Information-Oriented Approach to Informatical Education. In *Informatics in Education - Volume 1*, 2002.

[HB08] Sabrina Hilger und Norbert Breier. inES - eine integrierte Entwicklungsumgebung für Sprachdialogsysteme inES – eine integrierte Entwicklungsumgebung für Sprachdialogsysteme inES - eine integrierte Entwicklungsumgebung für Sprachdialogsysteme. In *e-Learning Baltics 2008*, 2008.

[Her94] Michael Herczeg. *Software-Ergonomie: Grundlagen der Mensch-Maschine-Kommunikation.* Addison-Wesley, 1994.

[Hey97] Hans Werner Heymann. Allgemeinbildung als Aufgabe der Schule und als Maßstab für Fachunterricht Allgemeinbildung als Aufgabe der Schule und als Maßstab für Fachunterricht. In *Pädagogik '49 (1997)*, 1997.

[HH10] Jutta Heckhausen und Heinz Heckhausen. *Motivation und Handeln.* Springer, 2010.

[Hil08] Sabrina Hilger. Konzeption und Implementierung einer didaktischen Entwicklungsumgebung für Sprachdialogsysteme. 2008.

[His12] Wolfgang Hissenauer. *Der Weg vom Qualitätsziel zu Standards guten Unterrichts*, 2012.

[HLS+10] Peter Hubwieser, Patrick Löffler, Petra Schwaiger, Mathias Spohrer, Markus Steinert, Siglinde Voß, und Ferdinand Winhard. *Informatik 5.* Klett, 2010.

[Hub07] Peter Hubwieser. *Didaktik der Informatik.* Springer, 2007.

[Ini14] Informatik im Kontext (IniK). http://www.informatik-im-kontext.de, zuletzt überprüft am 26.03.2017, 2014.

[JA02] Margolis J. und Fisher A. *Unlockig the Clubhouse - Women in Computing*. Cambridge (MA, USA), 2002.

[JM11] Werner Jank und Hilbert Meyer. *Didaktische Modelle*. Cornelsen, 2011.

[KK12] Maria Knobelsdorf und Christoph Kreitz. Konstruktivistisch- und Kompetenzorientierte Lehre der Theoretischen Informatik. In Peter Forbig, Detlev Rick, und Axel Schmolinsky, editors, *HDI 2012 - Informatik für eine nachhaltige Zukunft*, 2012.

[Kla06] Wolfgang Klafki. Die bildungstheoretische Didaktik im Rahmen kritisch-konstruktiver Erziehungswissenschaft. In Herbert Gudjons, editor, *Didaktische Theorien*. Bergmann+Helbig Verlag Hamburg, 2006.

[Kla16] Frederike Klan. Informatik reine Männersache? *LOG IN Nr. 183/184*, 2016.

[Kon] Deutschschweizer Erziehungsdirektoren Konferenz. Lehrplan 21. http://v-ef.lehrplan.ch/index.php?code=b|10|0&la=yes, zuletzt überprüft am 26.03.2017.

[Kul10] Hessisches Kultusministerium. Lehrplan Informatik - Hessen. http://verwaltung.hessen.de/irj/HKM_Internet?-cid=48a34f21388de135d056cf8266b8b151, zuletzt überprüft am 26.03.2017, 2010.

[Kul16] Kultusminister. Bildung in der digitalen Welt. https://www.kmk.org/fileadmin/Dateien/pdf/PresseUndAktuelles/2016/Bildung_digitale_Welt_Webversion.pdf, zuletzt überprüft am 26.03.2017, Taubenstraße 10, 10117 Berlin, 2016.

[KWSS09] Jochen Koubek, Helmut Witten, Peter Schulze, und Carsten Schulte. Informatik im Kontext (IniK). In *INFOS 2009*, 2009.

[Löw09] Wolfgang Löwer. Bildungsplan Informatik - Bremen. http://www.lis.bremen.de/sixcms/media.php/13-/INF_GyQ_2009.pdf, zuletzt überprüft am 26.03.2017, 2009.

[Mag] Johannes Magenheim. Informatische Bildung und Medienbildung. http://www.cs.uni-paderborn.de/uploads/media/informatische_bildung_medienbildung_01.pdf, zuletzt überprüft am 26.03.2017.

[McT04] Michael F. McTear. *Spoken dialogue technology: toward the conversational user interface*. Springer, 2004.

[MfB02] Forschung und Kultur des Landes Schleswig-Holstein Ministerium für Bildung, Wissenschaft. Lehrplan Informatik - Schleswig-Holstein. http://lehrplan.lernnetz.de/index.php?wahl=109, zuletzt überprüft am 26.03.2017, 2002.

[MfB10] Jugend und Kultur Ministerium für Bildung, Wissenschaft. Rahmenplan Informatik - Rheinland-Pfalz. http://informatik.bildung-rp.de/lehrplaene.html, 2010.

[MfB11] Jugend und Sport Land Brandenburg Ministerium für Bildung. Rahmenlehrplan Informatik - Brandenburg. http://bildungsserver.berlin-brandenburg.de/fileadmin/bbb/unterricht/-rahmenlehrplaene_und_curriculare\protect\discretionary{\char\hyphenchar\font}{}{}_materialien/-gymnasiale_oberstufe/curricula/2011/Informatik-VRLP_GOST_2011_Brandenburg.pdf, zuletzt überprüft am 26.03.2017, 2011.

[MfSuW99] Wissenschaft und Forschung des Landes Nordrhein-Westphalen Ministerium für Schule und Weiterbildung. Rahmenrichtlinien Gymnasium - NRW. http://www.standardsicherung.schulministerium.nrw.de-/lehrplaene/upload/lehrplaene_download/-gymnasium_os/4725.pdf, zuletzt überprüft am 26.03.2017, 1999.

[PRF08] Ilka Parchmann, Bernd Ralle, und David S. Fuccia. Entwicklung und Struktur der Unterrichtskonzeption Chemie im Kontext. In *Chemie im Kontext*. Waxmann, 2008.

[RMe04] *Rahmenplan Medienerziehung*. Ministerium für Bildung, Wissenschaft und Kultur Mecklenburg-Vorpommern, 2004.

[Rom08] Ralf Romeike. *Kreativität im Informatikunterricht.* PhD thesis, Universität Potsdam, 2008.

[Ros14] Norbert Rosenboom, editor. *Abitur 2014.* Hamburger Behörde für Bildung und Sport, 2014.

[RS12] M. Riepke und J. Siegeris. Informatik ein Männerfach!? Monoedukative Lehre als Alternative. *Informatik Spektrum 35 Jg., Heft 5*, 2012.

[RSB10] Elisabeth Raab-Steiner und Michael Benesch. *Der Fragebogen.* UTB, 2010.

[Ryc03] D. S. Rychen. Key Competencies. In *For a succesful life and well-functioning society.* Hogrefe and Huber, 2003.

[Sch08] Uwe Schöning. *Theoretische Informatik - kurzgefasst.* Spektrum Akademischer Verlag, 2008.

[Sch10a] Robert Scharner. Stand der Informatik in der Schule in Hessen, Rheinland Pfalz und dem Saarland. http://ddi.cs.uni-potsdam.de/Lehre/ddi2/Dossiers2010/Scharner2010.pdf, 2010.

[Sch10b] Heidi Schelhowe. *Kompetenzen in einer digital geprägten Kultur.* BMBF, 2010.

[Sch13] Ingrid Schirmer. Technik „wofür“ anstelle Technik „an sich“ - Kontextorientierung als Rahmen für einen genderbewussten Informatikunterricht. In *GI-Edition Lecture Notes in Informatics - Informatik erweitert Horizonte*, 2013.

[Sei03] Monika Seiffert. Vom Gesamtkonzept zum Curriculum. Planung von Kurssequenzen. *LOG IN*, 2003.

[SfB06] Jugend und Sport Berlin Senatsverwaltung für Bildung. Rahmenlehrplan Informatik - Berlin. `http://www.berlin.de/imperia/md/content/sen-bildung/unterricht/lehrplaene/sek2_informatik.pdf-?start&ts=1283429474&file=sek2_informatik.pdf`, zuletzt überprüft am 26.03.2017, 2006.

[Sig16] Das menschliche Gehör - Akustik. http://www.mu-sig.de/Theorie/Akustik/Akustik06.htm, zuletzt überprüft am 26.03.2017, 03 2016.

[SK11] Monika Seiffert und Alexandra Kück. KI - automatische Sprachverarbeitung. http://www.informatik-hamburg.de/pmwiki.php/M%f6glichkeitenUndGrenzen/Sprachverarbeitung, zuletzt überprüft am 26.03.2017, 2011.

[SS11] Siegrid Schubert und Andreas Schwill. *Didaktik der Informatik*. Spektrum, 2011.

[Str07] Jörg Stratmann. *Pädagogischer Mehrwert und Implementierung von Notebooks an der Hochschule*. Waxmann, 2007.

[TMfB12] Wissenschaft und Kultur Thüringer Ministerium für Bildung. Lehrplan Informatik - Thüringen. https://www.schulportal-thueringen.de/media/detail?tspi=3657, zuletzt überprüft am 26.03.2017, 2012.

[Uni13] Duke University. JFLAP - Java Formal Languages and Automata Theory. http://www.jflap.org, 07 2013.

[V1416] Deutsches akustisches Modell - Projekt VoxForge. http://www.voxforge.org/de/downloads, zuletzt überprüft am 26.03.2017, 03 2016.

[W3C16a] Dokumente der W3C Voicegroup. https://www.w3.org/Voice/Guide/, zuletzt überprüft am 26.03.2017, 04 2016.

[W3C16b] Voice Extensible Markup Language (VoiceXML) Version 2. https://www.w3.org/TR/voicexml20/, zuletzt überprüft am 26.03.2017, 03 2016.

# Anhang A

# Tabellen über die Erfassung der Kompetenzen des Bildungsplanes 2008

Legende: + Kompetenz erreicht, o Kompetenz teilweise erreicht, - Kompetenz nicht erreicht

Tabelle A.1: Informatiksysteme analysieren und verstehen

| Kompetenzen | L1 | L2 | L3 | L4 | L5 | L6 | $\sum$ |
|---|---|---|---|---|---|---|---|
| *Die Schülerinnen und Schüler* | | | | | | | |
| beschreiben gleichartig strukturierte Elemente in Benutzungsschnittstellen und verwenden geeignete Vorstellungen, um sich in unbekannten Informatiksystemen zu orientieren, | | | | | | + | 1 |
| erläutern die prinzipielle Funktionsweise und das Zusammenwirken der wesentlichen Hardware- Komponenten eines Computers produktunabhängig, | | | | | | | 0 |
| untersuchen Algorithmen und vergleichen sie hinsichtlich Effizienz und Qualität der Lösung, | + | | o | + | + | + | 5 |
| analysieren Informatiksysteme hinsichtlich der zugrunde liegenden Strukturen und Prozesse sowie der Aufgabenteilung zwischen Mensch und Maschine, | + | + | | + | + | + | 5 |
| identifizieren die wesentlichen Schichten und Komponenten der Architektur größerer Informatiksysteme, | | | | o | | | 1 |
| untersuchen, wo Daten in verteilten Systemen real gespeichert und verarbeitet werden und wie sie darauf zugreifen können. | | | | | | | 0 |

Tabelle A.2: Informatiksysteme gestalten

| Kompetenzen | L1 | L2 | L3 | L4 | L5 | L6 | $\sum$ |
|---|---|---|---|---|---|---|---|
| *Die Schülerinnen und Schüler* | | | | | | | |
| strukturieren Inhalte und bereiten diese produktorientiert sowohl hierarchisch gegliedert als auch vernetzt für unterschiedliche Zielgruppen angemessen auf, | o | | | + | | | 2 |
| kennen Methoden der evolutionären und partizipativen Gestaltung von Informatiksystemen und wenden sie in kleinen Entwicklungsvorhaben an, | | | | + | | | 1 |
| berücksichtigen bei Gestaltungsvorhaben universelle und medienspezifische Gestaltungskriterien sowie rechtliche Rahmenbedingungen, | | | | | | | 0 |
| wählen bei der Gestaltung von Informatiksystemen passende Algorithmen aus, | + | + | + | + | + | + | 6 |
| implementieren Modelle sowohl mit Hilfe grafischer Entwicklungsumgebungen als auch mit einer höheren Programmiersprache, | + | + | | + | + | + | 5 |
| identifizieren automatisierbare Sachverhalte der realen Welt und modellieren sie mit mindestens drei unterschiedlichen Modellierungsansätzen, und zwar mit objektorientierter Modellierung und jeweils mindestens einem weiteren Modellierungsansatz | | + | + | + | + | | 4 |
| vergleichen die Eignung verschiedener Modellierungsansätze für unterschiedliche Problemstellungen und wählen für gegebene Problemstellungen einen geeigneten Ansatz aus, | | | | + | + | + | 3 |
| implementieren Modelle sowohl mithilfe grafischer Entwicklungsumgebungen als auch mit höheren Programmiersprachen unterschiedlicher Paradigmen, | | | + | | | | 1 |
| implementieren einen Interpreter für eine selbst entwickelte einfache formale Sprache. | + | - | o | | | + | 4 |

Tabelle A.3: Darstellen und Interpretieren

| Kompetenzen | L1 | L2 | L3 | L4 | L5 | L6 | $\sum$ |
|---|---|---|---|---|---|---|---|
| *Die Schülerinnen und Schüler* | | | | | | | |
| unterscheiden zwischen Daten und Information, | | | | + | | + | 2 |
| beschreiben Modelle und Algorithmen sowohl grafisch als auch verbal, | + | + | | + | + | + | 5 |
| verwenden verschiedene digitale Repräsentationsformen multimedialer Daten und wählen für unterschiedliche Anwendungsfälle geeignete Repräsentationen aus, | | | | | + | + | 2 |
| können Information mithilfe einer Dokumentenbeschreibungssprache darstellen, | | | | | | + | 1 |
| unterscheiden natürliche von formalen Sprachen, | + | o | o | + | + | o | 6 |
| differenzieren formale Sprachen hinsichtlich ihrer Interpretierbarkeit. | - | - | - | - | o | o | 6 |

Tabelle A.4: Begründen und Bewerten

| Kompetenzen | L1 | L2 | L3 | L4 | L5 | L6 | $\sum$ |
|---|---|---|---|---|---|---|---|
| *Die Schülerinnen und Schüler* | | | | | | | |
| begründen, vergleichen und bewerten informatische Modellierungen in Bezug auf ihren Anwendungskontext und formale Kriterien, | o | | | + | + | | 3 |
| bewerten die prinzipielle und praktische Realisierbarkeit von Informatiksystemen, ohne dass dabei die Mathematik im Zentrum steht, | + | + | | + | + | + | 5 |
| beurteilen die Gebrauchstauglichkeit von Informatiksystemen auf der Grundlage von Gestaltungskriterien, | | + | | + | | | 2 |
| bewerten die Auswirkungen von Informatiksystemen auf die betroffenen Menschen, | + | | + | + | + | + | 5 |
| nehmen eine begründete Position zu Automatisierungsvorhaben ein und beziehen dabei auch die rechtlichen Rahmenbedingungen ein. | | | | | | | 0 |

Tabelle A.5: Kommunizieren und Kooperieren

| Kompetenzen | L1 | L2 | L3 | L4 | L5 | L6 | $\sum$ |
|---|---|---|---|---|---|---|---|
| *Die Schülerinnen und Schüler* | | | | | | | |
| organisieren und koordinieren ihre Arbeit in Projektgruppen und wenden dazu Methoden des Projektmanagements an, | | | | + | | | 1 |
| nutzen Informatiksysteme zur Kooperation und reflektieren die Kommunikationsprozesse, | + | + | | + | + | | 4 |
| erarbeiten sich Inhalte auch anhand englischsprachiger Dokumentation, | o | | | + | | + | 3 |
| beschreiben Sachverhalte mithilfe von Texten, Bildern und Diagrammen, | + | | | + | + | + | 4 |
| verwenden die informatische Fachsprache angemessen, | + | + | | + | + | + | 5 |
| dokumentieren Lernergebnisse, Arbeitsabläufe und Arbeitsergebnisse, | o | | | + | + | + | 4 |
| präsentieren wesentliche Ergebnisse adressatengerecht. | + | + | | + | + | + | 5 |

Tabelle A.6: Möglichkeiten und Grenzen von Informatiksystemen

| Kompetenzen | L1 | L2 | L3 | L4 | L5 | L6 | $\sum$ |
|---|---|---|---|---|---|---|---|
| *Die Schülerinnen und Schüler* | | | | | | | |
| bewerten Verfahren hinsichtlich Effizienz und Bedeutung aufgrund der Einsatzmöglichkeiten, | | | + | + | + | + | 4 |
| kennen prinzipielle und praktische Grenzen der Berechenbarkeit, | | | + | + | + | o | 4 |
| reflektieren gesellschaftliche, ethische und rechtliche Aspekte, | | | + | o | o | | 3 |
| reflektieren über Möglichkeiten und Grenzen von Informatiksystemen, z. B. durch eine fachkundige Diskussion der Frage „Welche Teile der geistigen Tätigkeiten des Menschen können Maschinen übernehmen?“ | o | o | o | o | o | o | 6 |

# Anhang B

# Der Fragebogen

# Mensch-Maschine-Kommunikation

### Feedback zur Unterrichtseinheit "Mensch-Maschine-Kommunikation".

### Fragebogen zur Unterrichtseinheit "Mensch-Maschine-Kommunikation"

Herzlich Willkommen zur Beantwortung des Fragebogens zur Mensch-Maschine-Kommunikation. Zunächst einen ganz herzlichen Dank für Ihre Bereitschaft zur Beantwortung einiger Fragen.

Dieser Fragebogen besteht aus drei Teilen. Im ersten Teil werden persönliche Daten abgefragt. Dazu gehören auch drei kurze Testfragen, die aus unterschiedlichen Teilgebieten der Informatik kommen. Keine Sorge! Wenn Sie diese Fragen nicht beantworten können, ist das überhaupt nicht schlimm. Halten Sie sich auch nicht zu lange an diesen Fragen auf.

Der zweite Teil besteht aus allgemeinen Feedbackfragen zur Unterrichtseinheit "Mensch-Maschine-Kommunikation". Beachten Sie hier bitte, dass es sich bei der Skala von 1 bis 6 **nicht um Schulnoten** handelt!

**Die Skala geht immer von 1 ("trifft nicht zu") bis 6 ("trifft vollkommen zu").**

Der dritte Teil des Fragebogens besteht aus zehn Testfragen. Versuchen Sie diese Testfragen so gut wie es geht zu lösen.

**Für alle hier erhobenen Daten gilt: Alle Daten werden streng vertraulich behandelt und nicht veröffentlicht. Des Weiteren werden diese Daten auch nicht den jeweiligen Fachlehrern für eine Benotung, etc. zur Verfügung gestellt werden. Sie können und sollen hier ganz offen und ehrlich ohne Angst antworten. Nur damit können Sie zur Verbesserung der Unterrichtseinheit "Mensch-Maschine-Kommunikation" beitragen.**

Ich bitte Sie jede einzelne Frage wahrheitsgemäß zu beantworten.

Hinweis: Manche Fragestellungen klingen sehr ähnlich. Das ist beabsichtigt.

Herzliche Grüße,

Sven Alisch

Diese Umfrage enthält 53 Fragen.

## Persönliche Daten

Fragen zur Ermittlung des Vorwissens sowie die Erhebung persönlicher Daten.

**1 [P01]Geschlecht ***

Bitte wählen Sie nur eine der folgenden Antworten aus:

- ○ weiblich
- ○ männlich

**2 [P02]Bitte geben Sie die Informatik Vornote (Zensur) des letzten Halbjahres an. ***

Bitte wählen Sie nur eine der folgenden Antworten aus:

- ○ 1
- ○ 2
- ○ 3
- ○ 4
- ○ 5

**3 [P03]Unter welchem der aufgeführten Betriebssysteme arbeiten Sie hauptsächlich im Unterricht. ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | Windows XP | Windows 7 | Windows 8 (oder 8.1) | Linux Derivate | Mac OS X |
|---|---|---|---|---|---|
| Betriebssystem | ○ | ○ | ○ | ○ | ○ |

**4 [P04]Die nachfolgenden Fragen beschäftigen sich mit der Funktionstüchtigkeit der eingesetzten Software. Dabei bedeutet eine 1 "trifft nicht zu" und eine 6 "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Die im Unterricht eingesetzte Software lief fehlerfrei. | ○ | ○ | ○ | ○ | ○ | ○ |
| Es gab keine Audio-Probleme (z.B. Headset, Medienwiedergabe, etc.). | ○ | ○ | ○ | ○ | ○ | ○ |
| Die im Unterricht eingesetzte Software stürzte ab. | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich konnte die Probleme der eingesetzten Software nachvollziehen. | ○ | ○ | ○ | ○ | ○ | ○ |

**5 [P05]Bitte geben Sie die Anzahl der Unterrichtsversäumnisse (in Stunden) während der Unterrichtseinheit "Mensch-Maschine-Kommunikation" ein. ***

Bitte geben Sie Ihre Antwort hier ein:

[ ]

**6 [P06]Bitte geben Sie Ihr Informatikinteresse an. Dabei bedeutet "1" kein Interesse und "6" ein sehr großes Interesse. ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich interessiere mich in meiner Freizeit für Informatik. | ○ | ○ | ○ | ○ | ○ | ○ |

## Aufgabenstellung 1

Im Folgenden bitte ich Sie, diese allgemeinen Aufgabenstellungen der Informatik zu lösen. Es ist nicht schlimm, wenn Sie diese Aufgaben nicht lösen können!

**7 [P201]Wissensfrage 1**

**Setze die Reihe fort: 11; 110; 1001; 1100; 1111; ...**

Bitte wählen Sie nur eine der folgenden Antworten aus:

- ○ 11010
- ○ 10000
- ○ 11100
- ○ 10010
- ○ 10001

## Aufgabenstellung 2

Im Folgenden bitte ich Sie, diese allgemeine Aufgabenstellungen der Informatik zu lösen. Es ist nicht schlimm, wenn Sie diese Aufgaben nicht lösen können!

**8 [P301]Was fehlt hier?**

```
long fakultaet(long x)
{
  if ( x==0)
    return 1;
  else
    return _______ ;
}
```

Bitte wählen Sie nur eine der folgenden Antworten aus:

- ○ fakultaet(x)*x
- ○ x*fakultaet(x-1)
- ○ x
- ○ -1
- ○ x*fakultaet(x)
- ○ x*x

## Aufgabenstellung 3

Im Folgenden bitte ich Sie, diese allgemeine Aufgabenstellungen der Informatik zu lösen. Es ist nicht schlimm, wenn Sie diese Aufgaben nicht lösen können!

**9 [P401]Ein Biber möchte seinem Freund, dem Hasen, geheime Nachrichten zukommen lassen.
Die beiden haben sich dafür einen Geheimcode ausgedacht. Mit dem werden ihre
Nachrichten verschlüsselt, damit niemand mitlesen kann.
Bei ihrem Geheimcode bleiben die Vokale (A, E, I, O, U) und die Satzzeichen unverändert.
Die Konsonanten werden durch den jeweils folgenden Konsonanten im Alphabet ersetzt.
Z wird dabei durch B ersetzt.
Wie lautet Bibers Nachricht „HALB ACHT IM WALD" im Geheimcode?**

Bitte wählen Sie nur eine der folgenden Antworten aus:

- ○ HELB ECHT OM WELD
- ○ JEMC EDJV ON XEMF
- ○ GAKZ ABGS IL VAKC
- ○ JAMC ADJV IN XAMF

## Fragekomplex 1

**10 [F01]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann die von mir entwickelten Programme meinen Mitschülerinnen und Mitschülern erklären. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann den Begriff Algorithmus beschreiben. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann ein grafisches Modell zur Sprachverarbeitung verbal beschreiben. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann meine grafischen Modelle zur Sprachverarbeitung begründen. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 2

**11 [F02]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann die Algorithmen zur Sprachverarbeitung meinen Mitschülerinnen und Mitschüler erklären. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann Algorithmen zur Sprachverarbeitung hinsichtlich ihrer Effizienz untersuchen. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann ein Modell zur Sprachverarbeitung grafisch beschreiben. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann meine grafischen Modelle zur Sprachverarbeitung mit denen meiner Mitschülerinnen und Mitschüler nach Gütekriterien vergleichen. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 3

**12 [F03]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann die Algorithmen zur Sprachverarbeitung anderer Schülerinnen und Schüler mit meiner eigenen Lösung nach Gütekriterien vergleichen. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann Algorithmen zur Sprachverarbeitung hinsichtlich ihrer Korrektheit untersuchen. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann einen Algorithmus verbal beschreiben. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann meine modellierten formalen Grammatiken mit denen meiner Mitschülerinnen und Mitschüler nach Gütekriterien vergleichen. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 4

**13 [F04]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann aus verschiedenen Algorithmen unterschiedlicher Güte, einen für meine Problemstellung geeigneten Algorithmus auswählen. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann Problemlösungen der Sprachverarbeitung mit Hilfe eines Algorithmus beschreiben. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann einen Algorithmus grafisch beschreiben. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann meine modellierten formalen Grammatiken begründen. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 5

**14 [F05]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann Algorithmen zur Sprachverarbeitung mit Hilfe eines endlichen Automaten oder eines Kellerautomaten (Parser) modellieren. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich unterscheide verschiedene Gütekriterien für einen Algorithmus. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann formale Sprachen mit formalen Grammatiken grafisch darstellen. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann Beispiele für reguläre und kontextfreie Sprachen nennen. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 6

**15 [F06]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann zu Problemen der Sprachverarbeitung grafische Modelle entwerfen. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann die Bestandteile der im Unterricht eingesetzten Software nennen. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann Beispiele für die Unterschiede zwischen einer natürlichen und einer formalen Sprache nennen. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann die Begriffe reguläre und kontextfreie Sprachen erklären. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 7

**16 [F07]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann Modelle zur Sprachverarbeitung mit Hilfe einer Programmiersprache (z.B. VoiceXML, Haskell oder Scheme) implementieren. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann die Funktionalität der im Unterricht eingesetzten Software erklären. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann Unterschiede zwischen einer formalen Sprache und einer natürlichen Sprache erklären. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann erklären, für welche Sprachklasse der endliche Automat geeignet ist. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 8

**17 [F08]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann Problemlösungen zur Sprachverarbeitung in einer geeigneten Entwicklungsumgebung grafisch modellieren. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann den Aufbau der im Unterricht eingesetzten Software beschreiben. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann mehrere Automatentypen nennen. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann erklären, für welche Sprachklasse der Kellerautomat (Parser) geeignet ist. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 9

**18 [F09]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann formale Grammatiken mit Hilfe eines Syntaxdiagramms darstellen. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann Algorithmen mit Hilfe der im Unterricht eingesetzten Software implementieren. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann erklären, welcher Automatentyp für eine formale Sprache benötigt wird. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann die Realisierbarkeit einer Grammatik für eine Sprache bewerten. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 10

**19 [F10]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann formale Grammatiken (z.B. mit Produktionen, ABNF, XML) darstellen. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann einen vorgegebenen Quelltext zur Sprachverarbeitung (z.B. Sprachdialogsystem, Übersetzer) hinsichtlich der Strukturen untersuchen. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann den Unterschied zwischen einer regulären und einer kontextfreien Grammatik beschreiben. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann die Realisierbarkeit eines endlichen Automaten für die Verarbeitung einer Sprache bewerten. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 11

**20 [F11]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann Problemlösungen der Sprachverarbeitung unterschiedlich modellieren. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann einen vorgegebenen Quelltext zur Sprachverarbeitung (z.B. Sprachdialogsystem, Übersetzer) hinsichtlich der Prozesse untersuchen. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann ein Beispiel für eine formale Sprache angeben, die durch eine kontextfreie Grammatik jedoch nicht mit einer regulären Grammatik erzeugt werden kann. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann die Realisierbarkeit eines Kellerautomaten für die Verarbeitung einer Sprache bewerten. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 12

**21 [F12]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann grafisch entworfene Modelle mit Hilfe einer Programmiersprache (z.B. mit VoiceXML, Haskel oder Scheme) implementieren. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich benutze Geräte, wie z.B. Smartphones, Handys oder Spielekonsolen. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Bevor ich einen endlichen Automaten entwickle, kann ich bewerten, ob dieser für eine gegebene Sprache geeignet ist. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich will mich auch in Zukunft mit dem Thema Sprachverarbeitung auseinandersetzen. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 13

**22 [F13]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich unterscheide verschiedene Gütekriterien für formale Grammatiken. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich habe in meiner Freizeit mit Geräten wie Handys, Smartphones oder Spielekonsolen mit gesprochener Sprache kommuniziert. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann die Auswirkungen der maschinellen Sprachverarbeitung auf die betroffenen Menschen beschreiben. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Die vergangene Unterrichtseinheit hat mit dazu beigetragen, dass ich mich stärker für Informatik interessiere. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 14

**23 [F14]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann die verschiedenen Implementierungsansätze für formale Grammatiken (z.B. Syntaxdiagramme, XML oder ABNF) hinsichtlich ihrer Eignung vergleichen. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann die Strukturen eines vorgegebenen Quelltextes zur Sprachverarbeitung (z.B. eines Sprachdialogsystems oder eines Übersetzers) erklären. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann die positiven und negativen Auswirkungen der maschinellen Sprachverarbeitung erklären. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich finde das Thema Sprachverarbeitung interessant. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 15

**24 [F15]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann aus verschiedenen Implementationen formaler Grammatiken (z.B. Syntaxdiagramm, XML, ABNF, Produktionen) einen geeigneten Ansatz nach Gütekriterien auswählen. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann die Prozesse aus einem vorgegebenen Quelltext zur Sprachverarbeitung (z.B. eines Sprachdialoges oder eines Übersetzers) erklären. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann die Auswirkungen der maschinellen Sprachverarbeitung auf die betroffenen Menschen bewerten. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich wollte wissen, wie die Kommunikation zwischen Mensch und Maschine funktioniert. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 16

**25 [F16]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann überprüfen, ob ein Satz von einem endlichen Automaten akzeptiert wird. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Die im Unterricht behandelten Probleme kenne ich aus dem Alltag. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Das Thema Sprachverarbeitung hat mich sehr angesprochen und motiviert. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 17

**26 [F17]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann überprüfen, ob ein Satz von einem Kellerautomaten (Parser) akzeptiert wird. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Das Thema Sprachverarbeitung hat mit dazu beigetragen, dass ich mich später auch beruflich mit Informatik beschäftigen möchte. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Die eingesetzte Software half mir die Prinzipien der Sprachverarbeitung zu verstehen. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 18

**27 [F18]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann Beispielsätze, die eine formale Grammatik erzeugt, aufschreiben. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Die Visualisierungsmöglichkeiten der eingesetzten Software erleichterten mir das Lernen. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Die im Unterricht eingesetzte Software war leicht zu bedienen. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich habe durch die im Unterricht eingesetzte Software viel über menschliche Sprache gelernt. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 19

Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu".

**28 [F19]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann Beispielsätze, die ein endlicher Automat akzeptiert, aufschreiben. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Durch das Basteln an Grammatiken kam ich auf eigene Ideen. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich konnte die Aufgabenstellungen mit Hilfe der eingesetzten Software selbstständig lösen. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich habe durch die im Unterricht eingesetzte Software viel über künstliche (Computer-) Sprache gelernt. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 20

**29 [F20]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann Beispielsätze, die ein Kellerautomat (Parser) akzeptiert, aufschreiben. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich hatte die Möglichkeit grafische Modelle zu erstellen. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich war nicht auf die Hilfe meines Lehrers, bzw. meiner Lehrerin angewiesen. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich habe auch außerhalb des Unterrichts über die Aufgaben nachgedacht. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 21

**30 [F21]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann erklären, wie eine formale Sprache von einem Automaten akzeptiert wird. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich konnte mit Hilfe der Software experimentieren. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Die im Unterricht eingesetzte Software erleichterte es mir, die Aufgaben zu lösen. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich habe das Gefühl, für mich etwas dazugelernt zu haben. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 22

**31 [F22]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann für eine reguläre Sprache einen geeigneten endlichen Automaten entwerfen. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich konnte mit Hilfe der Software kreativ sein. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Die im Unterricht eingesetzte Software veranschaulichte mir Problemstellungen der Sprachverarbeitung. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann mir vorstellen, dass ich das erworbene Wissen in Zukunft gebrauchen kann. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 23

**32 [F23]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich kann für eine selbst entwickelte formale Sprache einen Kellerautomaten (Parser) entwerfen. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich konnte mit Hilfe der eingesetzten Software eigene Ideen ausprobieren. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Es hat mir Spaß gemacht, mein Verständnis für dieses Thema Sprachverarbeitung zu vertiefen. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 24

**33 [F24]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich konnte eigene Projektideen mit Hilfe der eingesetzten Software umsetzen. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Die Darstellung der Lerninhalte war verständlich. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Die Software markierte mir die fehlerhaften Stellen. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 25

**34 [F25]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich benutzte die Software zum Modellieren und Implementieren. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| In der Unterrichtseinheit konnte ich kreativ sein. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Mein Produkt lief auch mit fehlerhaftem Code. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 26

**35 [F26]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Für das praktische Üben war ausreichend Zeit. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Die Software half mir dabei, meine eigenen Fehler nachzuvollziehen. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 27

**36 [F27]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| In der Unterrichtseinheit wurde viel experimentiert. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Durch grafisches Modellieren machte ich weniger Fehler. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Durch die in der Software benutzten Darstellungsweise konnte ich meine Fehler finden. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Die gestellten Aufgaben waren lösbar. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 28

**37 [F28]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich konnte meine Ideen im Unterricht einbringen und umsetzen. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich konnte meine eigenen Fehler finden. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Die Fehlermeldungen der im Unterricht benutzten Software waren nachvollziehbar. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Die Inhalte sind für den Umgang mit Computersystemen sehr nützlich. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 29

Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu".

**38 [F29]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Der Nutzen formaler Sprachen wurde deutlich. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich fand meine Fehler ohne die Hilfe meines Lehrers, bzw. meiner Lehrerin. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Die Hilfsmittel zur Unterstützung des Lernens (z.B. Arbeitsblätter, Software) waren ausreichend und in guter Qualität vorhanden. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich habe oft die gleichen Fehler gemacht. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 30

**39 [F30]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich konnte im Unterricht etwas Neues entdecken. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich konnte meine Fehler schnell selbst finden. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Einige Themen haben mich besonders interessiert. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich habe auch außerhalb des Unterrichts über die Aufgaben nachgedacht. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 31

**40 [F31]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich konnte mich leicht auf die Sache konzentrieren. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich konnte Alternativen ausprobieren. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich habe das Gefühl, für mich etwas dazugelernt zu haben. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Meine syntaktischen Fehler waren nachvollziehbar. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

## Fragekomplex 32

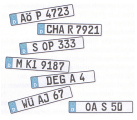
**41 [F32]Bitte beantworten Sie die folgenden Fragen. Die "1" bedeutet immer "trifft nicht zu" und die "6" bedeutet "trifft vollkommen zu". ***

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| Ich machte überwiegend syntaktische Fehler. **(SQ001)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Ich kann mir vorstellen, dass ich das erworbene Wissen in Zukunft gebrauchen kann. **(SQ002)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Es hat mir Spaß gemacht, mein Verständnis für dieses Thema zu vertiefen. **(SQ003)** | ○ | ○ | ○ | ○ | ○ | ○ |
| Im Unterricht hatte ich Erfolgserlebnisse. **(SQ004)** | ○ | ○ | ○ | ○ | ○ | ○ |

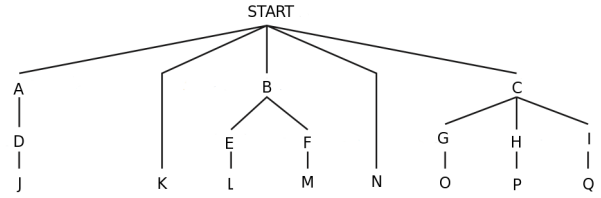
## Testfrage 1

**42 [T01]KFZ Zeichen können z.B. mit Hilfe der folgenden formalen Grammatik gebildet werden:**




```
root $KENNZEICHEN;
$KENNZEICHEN = $ZULASSUNGSBEZIRK ' ' $BUCHSTABENKOMBINATION ' ' $ZAHL;
$ZULASSUNGSBEZIRK = $BUCHSTABE | $BUCHSTABE $BUCHSTABE | $BUCHSTABE $BUCHSTABE $BUCHSTABE
$BUCHSTABENKOMBINATION = $BUCHSTABE | $BUCHSTABE $BUCHSTABE;
$ZAHL = $ZIFFER | $ZIFFER $ZIFFER | $ZIFFER $ZIFFER $ZIFFER | $ZIFFER $ZIFFER $ZIFFER $ZIFFER;
$BUCHSTABE = 'A' | 'B' | 'C' | 'D' | 'E' | 'F' | 'G' | 'H' | ... | 'X' | 'Y' | 'Z' | 'Ä' | 'Ö' | 'Ü';
$ZIFFER = '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9';
```

**Das Nummernschild "P AS 123" ergibt unter Anwendung der obigen Grammatik folgenden Ableitungsbaum:**




**Hinter jedem Buchstaben steckt ein Terminal oder eine Variable. Ordne jedem Buchstaben entweder ein Terminal oder eine Variable zu. Die einfachen Anführungszeichen (' ') stehen immer für ein Leerzeichen.**

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | $KENNZEICHEN | $ZIFFER | $ZULASSUNGSBEZIRK | $BUCHSTABENKOMBINATION | $ZAHL | $BUCHSTABE | ' ' | P | A | S | 1 | 2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| B | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| C | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| D | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| E | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| F | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| G | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| H | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| I | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| J | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| K | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| L | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| M | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| N | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| O | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| P | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| Q | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| START | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ○ |

## Testfrage 2

**43 [T02]Natürliche Zahlen**
**Mit der folgenden formalen Grammatik können natürliche Zahlen erzeugt werden. Die Startregel sei immer $ZAHL.**
**$ZAHL = $ZIFFER | $ZIFFER $ZAHL**
**$ZIFFER = '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'**
**Die Zahlen können jedoch beliebig viele führende Nullen aufweisen (z.B. 01234 oder 007652). Welcher der folgenden Lösungsvorschläge erzeugt die natürlichen Zahlen in ihrer üblichen (ohne führende Nullen) Schreibweise?**

**Lösungsvorschlag 1:**

```
$ZAHL = $ZIFFER | $ZIFFERNICHTNULL $ZIFFERNFOLGE
$ZIFFERNFOLGE = $ZIFFER | $ZIFFER $ZIFFERNFOLGE
$ZIFFERNNICHTNULL = ’1’ | ’2’ | ’3’ | ’4’ | ’5’ | ’6’ | ’7’ | ’8’ | ’9’
$ZIFFER = ’0’ | $ZIFFERNICHTNULL
```

**Lösungsvorschlag 2:**

```
$ZAHL = (’1’ | ’2’ | ’3’ | ’4’ | ’5’ | ’6’ | ’7’ | ’8’ | ’9’) | $ZIFFERNICHTNULL $ZIFFERNFOLGE
$ZIFFERNFOLGE = $ZIFFER | $ZIFFER $ZIFFERNFOLGE
$ZIFFERNNICHTNULL = ’1’ | ’2’ | ’3’ | ’4’ | ’5’ | ’6’ | ’7’ | ’8’ | ’9’
$ZIFFER = ’0’ | $ZIFFERNICHTNULL
```

**Lösungsvorschlag 3:**

```
$ZAHL = $ZIFFER | (’1’ | ’2’ | ’3’ | ’4’ | ’5’ | ’6’ | ’7’ | ’8’ | ’9’) $ZIFFERNFOLGE
$ZIFFERNFOLGE = $ZIFFER | $ZIFFER $ZIFFERNFOLGE
$ZIFFER = ’0’ | ’1’ | ’2’ | ’3’ | ’4’ | ’5’ | ’6’ | ’7’ | ’8’ | ’9’
```

**Lösungsvorschlag 4:**

```
$ZAHL = $ZIFFER | $ZIFFERNICHTNULL $ZIFFERNFOLGE
$ZIFFERNFOLGE = $ZIFFER $ZIFFERNFOLGE | $ZIFFER
$ZIFFERNNICHTNULL = ’1’ | ’2’ | ’3’ | ’4’ | ’5’ | ’6’ | ’7’ | ’8’ | ’9’
$ZIFFER = $ZIFFERNICHTNULL | '0'
```

**Für welche oder welchen Lösungsvorschlag entscheidest du dich? Wähle aus!**

Bitte wählen Sie alle zutreffenden Antworten aus:

- ☐ Lösungsvorschlag 1
- ☐ Lösungsvorschlag 2
- ☐ Lösungsvorschlag 3
- ☐ Lösungsvorschlag 4

## Testfrage 3

**44 [T03]Die Ureinwohner einer bislang unerforschten Insel reden eine seltsame Sprache, die nur aus den Lauten "Da", "Li" und "Mo" bestehen. Dabei liegen all ihren Wörtern folgende Ableitungsregeln zugrunde. $S bezeichnet hierbei die Startvariable.**

```
$S = 'Da'|'Li'
$S = 'DaDa'$S|'Da'$S'Li'|$L'Mo'
$L = 'Li' | $L$S'Mo'
```

**Welche der folgenden Wörter gehört zur Sprache der Inselbewohner?**

Bitte wählen Sie alle zutreffenden Antworten aus:

- ☐ DaLiLiMo
- ☐ LiDaMoMo
- ☐ DaMoLiMo
- ☐ DaDaDaDaLi
- ☐ DaDaLiMo
- ☐ DaDaDaLiLi
- ☐ LiMo
- ☐ DaLiDaDaMo

## Testfrage 4

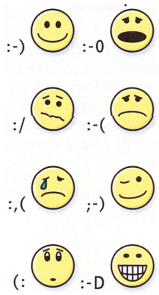
**45 [T04]**

**Gegeben sei die folgende Startregel einer EBNF (engl. ABNF) für die Erzeugung von Smileys.**

**Dabei steht S für Smiley, H für Hut, N für Nase, A für Augen und M für Mund.**

**$S = H A N M | A N M | H A M | A M;**

**Welche der folgenden EBNF Umformungen sind dazu äquivalent?**




Bitte wählen Sie alle zutreffenden Antworten aus:

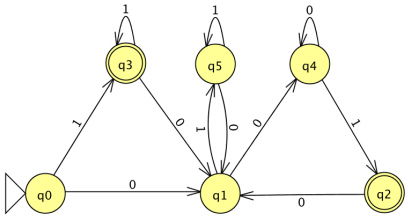
- [ ] $S = A N M | H A M | A M;
- [ ] $S= [H] A [N] M;
- [ ] $S = (H | A | N | M);
- [ ] $S = H* A* N M*;

## Testfrage 5

**46 [T05]**

### Endlicher Automat

Gegeben sei folgender endlicher Automat.




Welche folgenden Wörter werden akzeptiert?

Bitte wähle aus der Liste alle Wörter aus, die von dem endlichen Automaten akzeptiert werden.

Bitte wählen Sie alle zutreffenden Antworten aus:

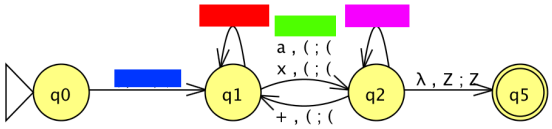
- [ ] 1111
- [ ] 10101
- [ ] 00101001
- [ ] 1001
- [ ] 001101
- [ ] 0101010

## Testfrage 7

**48 [T07]**

### Der Kellerautomat (Parser)

Gegeben sei der folgende Kellerautomat:




Erkannt werden sollen die Terme:

- (a+b)
- ((a+b))
- ((((a+x+a+b))))

Bitte ordne den Kästchen ihren Inhalt zu.

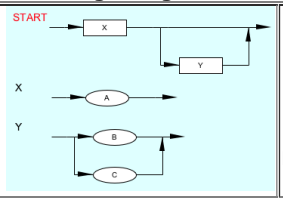
Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | (; (; (( | b; (; ( | (; Z; (Z | ); (; λ | -; (; (( | y; (; ( | ); ); )) |
|---|---|---|---|---|---|---|---|
| Kästchen rot | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| Kästchen grün | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| Kästchen blau | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| Kästchen lila | ○ | ○ | ○ | ○ | ○ | ○ | ○ |

## Testfrage 8

**49 [T08]Jeweils zwei Grammatiken gehören zusammen (immer eine aus A-D gehört zu einer aus 1-4). Finde die zusammengehörigen Grammatiken.**

**A**




**B**

```
root $START;
$START= ( $A  $B  $C ) ;
$A= X ;
$B= Y | W ;
$C= ( T  $C ) ;
$C=;
```

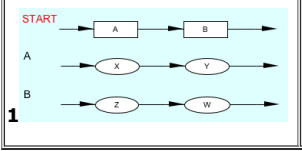
**C**

```
<grammar langid="409" root="START">
<rule id="START" scope="public"><one-of><item>X</item><item>Y</item>
</one-of><token>A</token><one-of><item>Z</item><item>T</item></one-of>
</rule></grammar>
```

**D**

```
root $START;
$START= ( $A  $B
$A= ( X  Y ) ;
$B= ( Z  W ) ;
```

**1**




**2**

```
root $START;
$START= (( X |  Y ) A ( Z |  T )) ;
```

**3**

```
<grammar langid="409" root="START"><rule id="START" scope="public">
<ruleref uri="#A"/><ruleref uri="#B"/><ruleref uri="#C"/></rule>
<rule id="A"><token>X</token></rule><rule id="B">
<one-of><item>Y</item><item>W</item></one-of></rule>
<rule id="C"><token>T</token><ruleref uri="#C"/></rule><rule id="C"/>
</grammar>
```

**4**

```
root $START;
$START= ( $X ( $NULL |  $Y
$X= A ;
$Y= B |  C ;
```

Bitte wählen Sie die zutreffende Antwort für jeden Punkt aus:

| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| A | ○ | ○ | ○ | ○ |
| B | ○ | ○ | ○ | ○ |
| C | ○ | ○ | ○ | ○ |
| D | ○ | ○ | ○ | ○ |

## Testfrage 9

**50 [T09]Erkläre den Begriff der "formalen Grammatik". Bitte schreibe nicht mehr als 10 Sätze!**

Bitte geben Sie Ihre Antwort hier ein:

## Testfrage 10

**51 [T10]Erkläre den Begriff "Kellerautomat". Bitte benutze nicht mehr als 10 Sätze!**

Bitte geben Sie Ihre Antwort hier ein:

## Projekt

# Anhang C

# Handreichung zur Benutzung der Lernsoftware inES und inGE im Informatikunterricht

## C.1 inES - integrierte Entwicklungsumgebung für Sprachen

### C.1.1 Die Oberfläche von inES

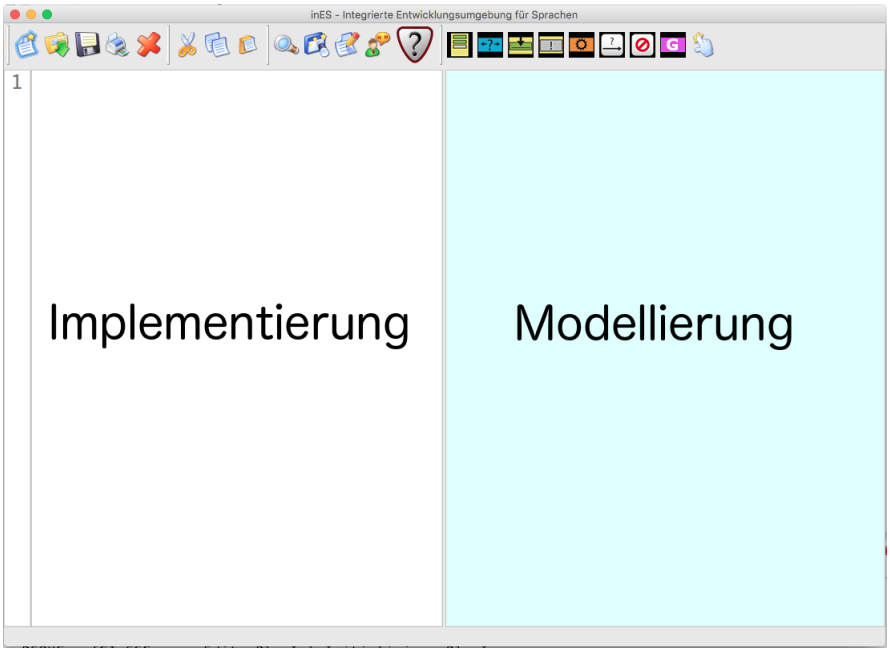



Abbildung C.1: **inES Hauptfenster mit Beispieldatei**

Die Abbildung C.1 zeigt die Oberfläche von inES kurz nach dem Programmstart. Die Arbeitsfläche unterhalb der Werkzeugleiste ist in einen Modellierungs- und in einen Implementierungsbereich aufgeteilt. Auf der rechten, türkis gefärbten Hälfte, werden Sprachdialoge grafisch modelliert. Die grafischen Modellobjekte werden hierfür über die obige Werkzeugleiste, direkt über dem Modellierungsbereich, hinzugefügt. Die linke Seite zeigt den Quelltexteditor, mit dem ein Sprachdialog in *VoiceXML* bearbeitet werden kann. Für die Bearbeitung der Quelltexte werden Kopier-, Ausschneide- bzw. Einfügewerkzeuge über die Werkzeugleiste sowie durch allgemein bekannte Tastenkombinationen (engl. Short-Cuts) bereitgestellt. Darüber hinaus können Quelltexte validiert werden. Syntaktische Fehler im Quelltext werden bei den betreffenden Stellen farbig (rot) markiert. Zeilennummern links neben dem Quelltexteditor dienen zur besseren Orientierung.
Der vertikal geteilte Arbeitsbereich kann flexibel angepasst werden. Der mittlere vertikale Balken kann nach links bzw. rechts, je nachdem ob gerade modelliert oder implementiert wird, verschoben werden.

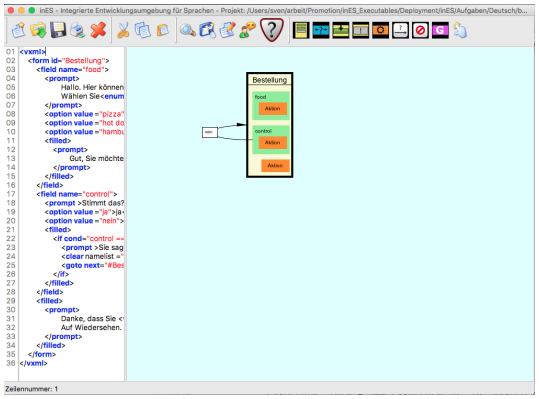



Abbildung C.2: Ansicht 1

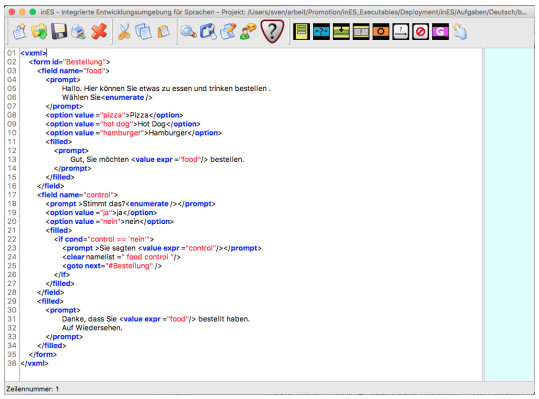


Abbildung C.3: Ansicht 2

Sprachdialoge werden durch Drücken des entsprechenden Buttons („das Männchen mit der Sprechblase“) gestartet.

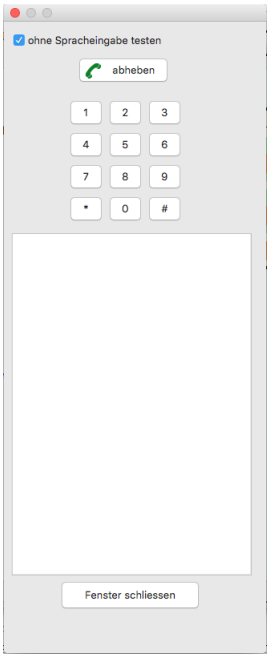



Abbildung C.4: **Sprachdialoge starten**

Sprachdialoge können mit bzw. ohne gesprochene Sprache getestet werden. Insbesondere während der Entwicklungsphase eines Sprachdialoges ist es oft sinnvoll, ohne eine Spracherkennung zu arbeiten, denn durch das Auftreten von Fehlerkennungen kann wertvolle Unterrichtszeit verloren gehen und die Motivation der Schüler sinken. Sprachdialoge können so zügiger getestet werden. Werden Sprachdialoge ohne eine Spracherkennung durchgeführt, erscheinen die Antwortmöglichkeiten im Protokollfenster als Hyperlink, der mithilfe einer Maus angewählt werden kann. Die in der Abbildung dargestellten Buttons mit Ziffern entsprechen der auf Mobil- und DECT-Telefonen üblichen Ziffernanordnung und dienen der Eingabe von Zahlen. Der weiße Bereich stellt das aufgezeichnete Protokoll des Gesprächs zwischen Mensch und Maschine dar. Nach dem Beenden des Sprachdialogs kann dadurch der gesamte Gesprächsverlauf nachträglich analysiert werden. Für die Weiterarbeit am Sprachdialog kann das Fenster geschlossen werden.

### Kopplung zwischen Modell und Code

Die Modellierung ist von der Implementierung eines Sprachdialoges unabhängig, d. h. es müssen sich nicht alle Modellobjekte im Quelltext eines Sprachdialoges, im Folgenden VoiceXML-Quelltext genannt, wiederfinden. Umgekehrt gilt das Gleiche.

Die Elemente des VoiceXML-Quelltextes können durchaus Bestandteile enthalten, die nicht im Modell implementiert bzw. gar nicht implementierbar sind.
Anders als in der prototypischen Entwicklung von Hilger besteht bei der Neuentwicklung die Möglichkeit, sowohl das Modell mit dem VoiceXML-Quelltext als auch den VoiceXML-Quelltext mit dem Modell, also in umgekehrter Richtung zu synchronisieren. Für die Synchronisierung stehen zwei Buttons in der Werkzeugleiste zur Verfügung.

Abbildung C.5: Modell erstellen

Abbildung C.6: Code erstellen

Das Drücken des Buttons „Modell erstellen“ erstellt auf Grundlage des VoiceXML-Quelltexts ein passendes Modell. Umgekehrt löst das Drücken des Buttons „Code erstellen“ die Synchronisierung des VoiceXML-Quelltextes auf Basis des Modells aus. Im letzteren Fall wird, sollte der Implementierungsbereich leer sein, eine VoiceXML-Quelltextvorlage erstellt. Damit sollen die Schüler bevor sie in VoiceXML implementieren zunächst zur Modellierung eines Sprachdialoges motiviert werden.

### C.1.2 Die Modellierung eines Sprachdialoges in VoiceXML

Im Kapitel 4.1.6 wurde ein VoiceXML-Modell schemenhaft skizziert. Für inES entwickelte Hilger im Rahmen ihrer Diplomarbeit eine geeignete Modelldarstellung für VoiceXML-Dialoge und implementierte diese in ihrem Prototypen. Diese Modelldarstellung wurde auch für die neue Implementation übernommen.
Im Folgenden wird die Entwicklung eines VoiceXML-Dialoges mithilfe der inES-Modellobjekte am Beispiel eines interaktiven Flugbuchungssystem vorgestellt.

### Die Modellobjekte des VoiceXML-Modells

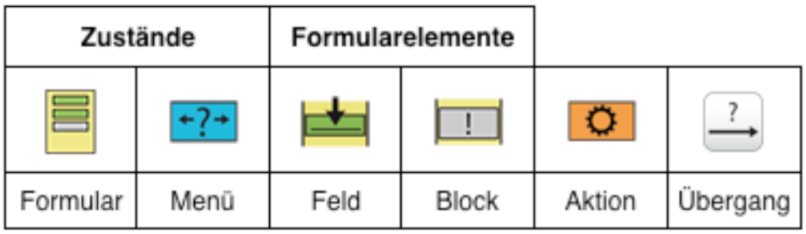



Abbildung C.7: **inES Modellobjekte**

VoiceXML-Modelle sind zustandsbasiert und können aus den in Abbildung C.7 dargestellten Modellobjekten bestehen. VoiceXML-Modelle unterscheiden zwei Zustände. Das ist das Menü bzw. das Formular. Mithilfe des Modellobjektes *Übergang* können die verschiedenen Zustände zu einem Dialogmodell verbunden werden, wodurch die im VoiceXML-Standard definierten Sprünge repräsentiert werden.
Formulare in inES können sowohl Blöcke als auch Felder beinhalten. Während Blöcke mehrere VoiceXML-Anweisungen bündeln können, handelt es sich bei Feldern um sogenannte Feldvariablen, die abgefragte Daten eines Benutzers für die spätere Auswertung speichern können. Sowohl bei Feldern als auch bei Blöcken können Sprünge (Übergänge) zu anderen Modellobjekten implementiert werden. Des Weiteren unterstützt inES in der neuen Implementation auch das Springen innerhalb eines Formulars.
Das <filled>-Tag wird durch das Modellobjekt *Aktion* repräsentiert. Felder und Formulare können solche *Aktion*s-Objekte besitzen.

### Attribute der Modellobjekte

| | Symbol | Objekt | Attribute |
|---|---|---|---|
| Zustände |  | Formular | Beschreibung<br>Titel<br>Formularelemente<br>Aktion<br>Übergänge |
| |  | Menü | Beschreibung<br>Titel<br>Übergänge |
| Formularelemente |  | Feld | Beschreibung<br>Titel<br>Aktion<br>Übergänge |
| |  | Block | Beschreibung<br>Übergänge |
| |  | Aktion | Beschreibung |
| |  | Übergang | Zielzustand<br>Bedingung |

Abbildung C.8: **inES Modellobjekte mit ihren Attributen**

Die Abbildung C.8 zeigt die Modellobjekte und ihre dazugehörigen Attribute. Bis auf das Modellobjekt Übergang besitzen alle Modellobjekte das Attribut *Beschreibung*. Mithilfe dieses Attributes können Schüler Notizen bzw. Gedanken formulieren. Das können z. B. Begründungen für das Wählen eines bestimmten Modellobjektes sowie seine Aufgabe im Kontext des zu modellierenden Sprachdialoges sein.
Die Modellobjekte Formular, Menu und Feld besitzen das Attribut *Titel*. Die Vergabe eines eindeutigen Namens für das Attribut *Titel* durch die Schüler ist die Grundvoraussetzung für die Nutzung von Übergängen von Feld zu Feld innerhalb eines Formulars oder Übergänge zu weiteren Formularen bzw. Menüs.
Formulare können aus mehreren Formularelementen unterschiedlichen Typs, das sind Feld oder Block, bestehen. Des Weiteren können den Formular- und den Feld-Modellobjekten das Attribut *Aktion* explizit hinzugefügt werden. Mit dem Modellobjekt Aktion werden ausgehende Kanten, die Übergänge zu anderen Feld-, Formular- und Menü-Modellobjekten modelliert. Diese Übergänge werden als ausgehende Kanten dargestellt und besitzen die Attribute Zielzustand und Bedingung. Der Zielzustand gibt das Formular, Feld oder Menü an, zu dem gesprungen wird. Bei der Bedingung kann es sich um eine konkrete Benutzeräußerung oder einen unbedingten „goto“- Sprung handeln. Die Bedingungen für einen Übergang werden zum einen an der Kante beschriftet und zum anderen während des Bearbeitens eines Modellobjek-

tes in der Spalte „Bedingung“ in der Liste der Aktionen dargestellt. Die Bearbeitung eines bestimmten Modellobjektes erfolgt durch Doppelklick. Formular-, Feld- und Menü-Modellobjekte speichern alle Übergänge in einer Liste.

### Zustände und Übergänge

Im Folgenden wird ein Sprachdialog schrittweise modelliert werden, der über bestimmte Fluggesellschaften informiert und die Buchung eines Fluges ermöglicht. Die Buchung eines Fluges soll von den bloßen Informationsansagen über Fluggesellschaften getrennt werden. Hierfür eignet sich ein Menü.
Ein Menü kann durch Drücken des entsprechenden Buttons aus der Werkzeugleiste erstellt werden. Danach wird nach einem eindeutigen Namen, einem Attributwert für den Titel des Menüs, gefragt. Wird kein Name vergeben, zeichnet das Programm ein nicht beschriftetes blau gefülltes Rechteck in den Modellierungsbereich. Die Attributwerte des Menüs können auch im Nachhinein durch doppeltes Anklicken bearbeitet werden. Des Weiteren kann dadurch dem Menü auch eine Notiz hinzugefügt werden (siehe Abbildung C.9).

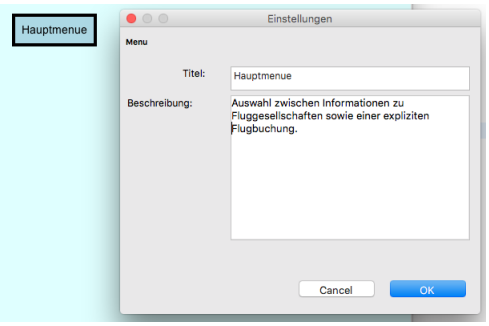



Abbildung C.9: **inES Bearbeiten von Menüs**

Für den Sprachdialog werden zwei Formulare, das sind *infoF* und *buchungF*, benötigt. Sie werden genau wie das Modellobjekt Menü über die Werkzeugleiste hinzugefügt. Die Zustände des Sprachdialoges, hier die Modellobjekte, werden durch Anklicken stärker umrandet. Dadurch weiß ein Schüler, welchen Zustand bzw. welches Modellobjekt er bearbeitet. Des Weiteren können diese Modellobjekte auf dem Implementierungsbereich mithilfe der gedrückten linken Maustaste frei verschoben werden.
Als nächstes werden zwei Übergänge benötigt, das ist der Übergang vom Haupt-

menü zum Formular *infoF* und der Übergang vom Hauptmenü zum Formular *buchungF*. Hierfür wird zunächst das Hauptmenü mit der linken Maustaste angeklickt. Anschließend wird aus der Werkzeugleiste der Button Übergang gedrückt und das entsprechende Formular, z. B. *infoF*, angeklickt. Es erscheint eine Kante mit der Beschriftung „Eingabe“. Für die Modellierung der Bedingung ergeben sich zwei Möglichkeiten. Entweder man klickt die Beschriftung doppelt an oder bearbeitet das Hauptmenü, indem auf das Modellobjekt des Hauptmenüs doppelt geklickt wird (siehe Abbildung C.10 und Abbildung C.11).

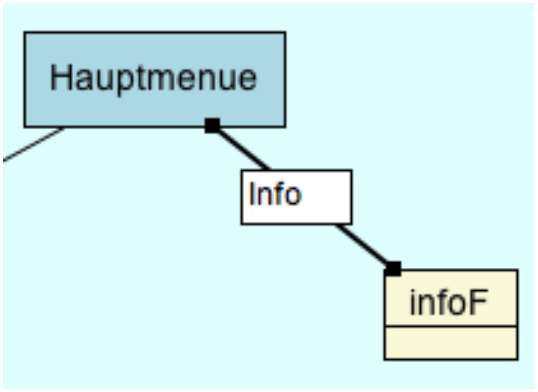



Abbildung C.10: Bedingung direkt bearbeiten

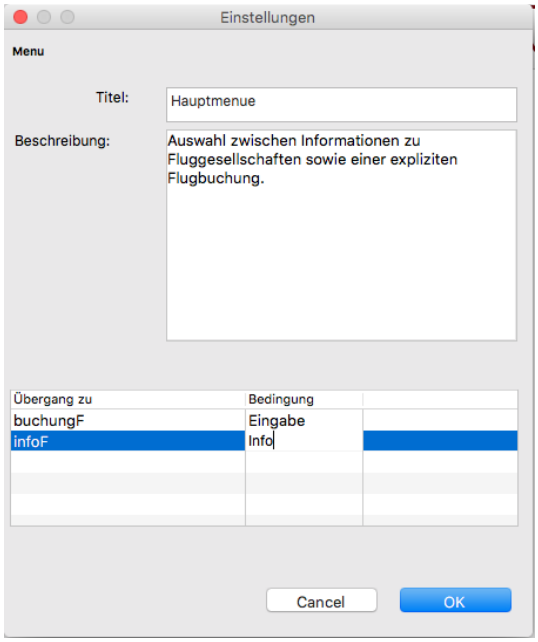



Abbildung C.11: Bedingung über Eigenschaftenfenster bearbeiten

Der Übergang vom Hauptmenü zum Formular *buchungF* wird analog modelliert. Die einzugebende Bedingung richtet sich nach den Benutzereingaben. Hier ist es zum einen das Wort „Info“ für den Übergang zum Formular *infoF* und das Wort „Buchung“für den Übergang zum Formular *buchungF*.
Alle Modellobjekte können durch anwählen und anschließendem drücken der „Entf“-Taste gelöscht werden. Das gilt auch für die Übergänge (Kanten). Alternativ kann nach Anwahl eines Modellobjektes auch der Löschen-Button aus der Werkzeugleiste gedrückt werden.

### Felder und Aktionen

Der Benutzer soll Informationen über die Fluggesellschaften erhalten, bei denen eine Flugbuchung möglich ist. Hierfür sollen ihm verschiedene Gesellschaften zur Auswahl angeboten werden. Das folgende Beispiel soll sich auf zwei fiktive Fluggesellschaften, das ist „Luftgänse“ und „Aliair“, beschränken. Die Auswahl selbst wird über ein Feld „gesellschaften“ entgegengenommen. Felder werden durch das Anwählen des Formulars und das anschließende Drücken des Feld-Buttons der Werkzeugleiste hinzugefügt. Es öffnet sich wieder ein Dialog über den der Attributwert des Titels eingegeben werden kann. Entweder durch Drücken der Enter-Taste oder des „OK“-Buttons wird die Eingabe bestätigt. Wie bei den Formular- bzw. Menü-Modellobjekten können auch die Attributwerte der Feld-Modellobjekte durch doppeltes Anklicken nachträglich bearbeitet werden.

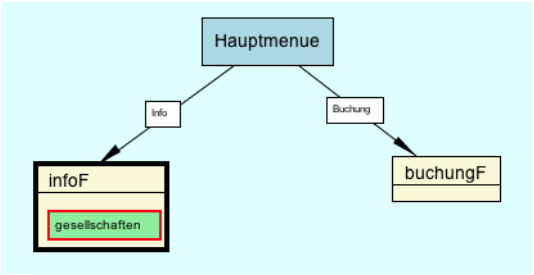



Abbildung C.12: inES Felder hinzufügen

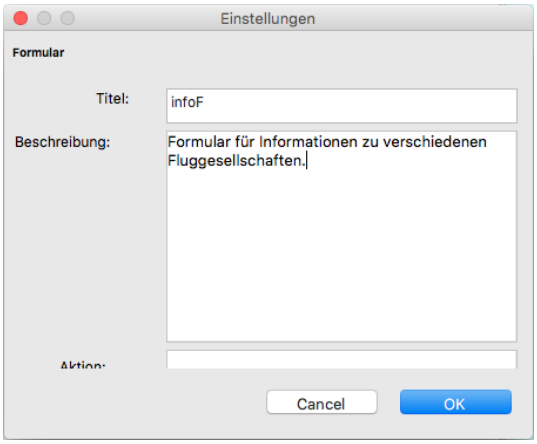



Abbildung C.13: Attributwerte des Formulars infoF

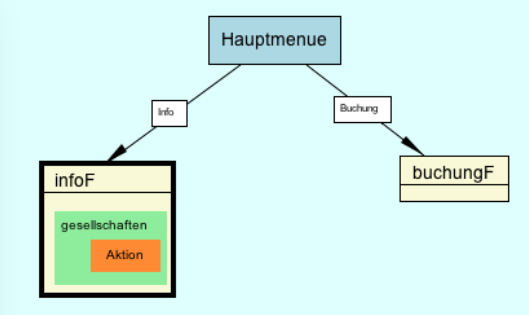



Abbildung C.14: inES Aktion hinzufügen

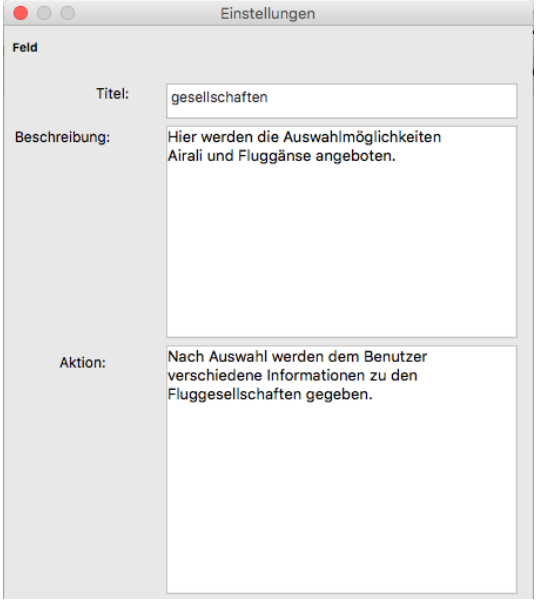



Abbildung C.15: Attributwerte des Feldes gesellschaften

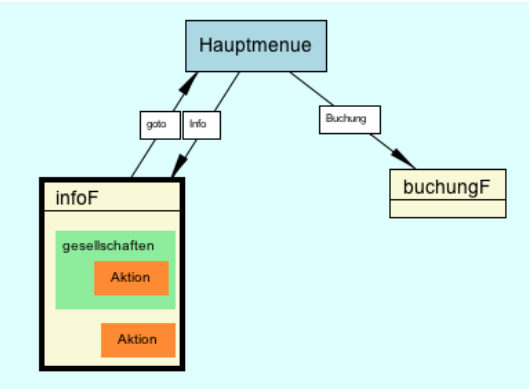



Abbildung C.16: inES Aktion hinzufügen

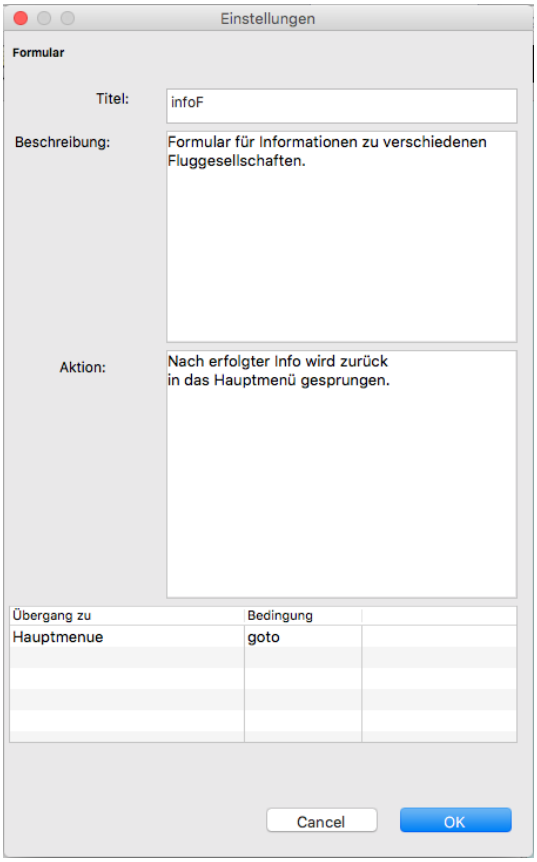



Abbildung C.17: Attributwerte des Formulars infoF

Die Benutzereingabe für die Auswahl der Fluggesellschaft soll gleich ausgewertet werden. Dafür wird für das Feld „gesellschaften“ eine Aktion benötigt. Analog zur Modellierung von Formularen wird zunächst wieder das betreffende Modellobjekt angewählt und anschließend der Button Aktion gedrückt. Die Abbildungen C.14 und C.16 zeigen, wie sowohl ein Feld als auch ein Formular um eine Aktion erweitert wurden. Des Weiteren zeigen die Abbildungen C.15 und C.17, dass auch Aktionen mit Kommentaren versehen werden können. Außerdem zeigt die Abbildung C.16 einen Rücksprung zum Hauptmenü. Dieser Rücksprung soll nach der erfolgreichen Abarbeitung des Formulars erfolgen. Alle gespeicherten Werte werden nach erfolgtem Rücksprung gelöscht. Der Rücksprung wird im VoiceXML-Quelltext mit einem <goto>-Tag implementiert.
Nach Rückkehr im Hauptmenü hat der Benutzer des Sprachdialoges erneut die Möglichkeit, sich über Fluggesellschaften zu erkundigen oder einen Flug zu buchen.

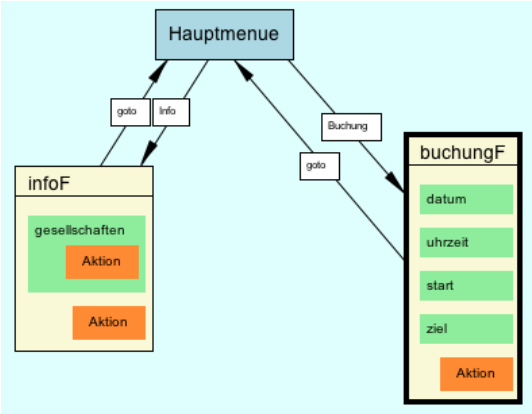



Abbildung C.18: inES mit Formular buchungF

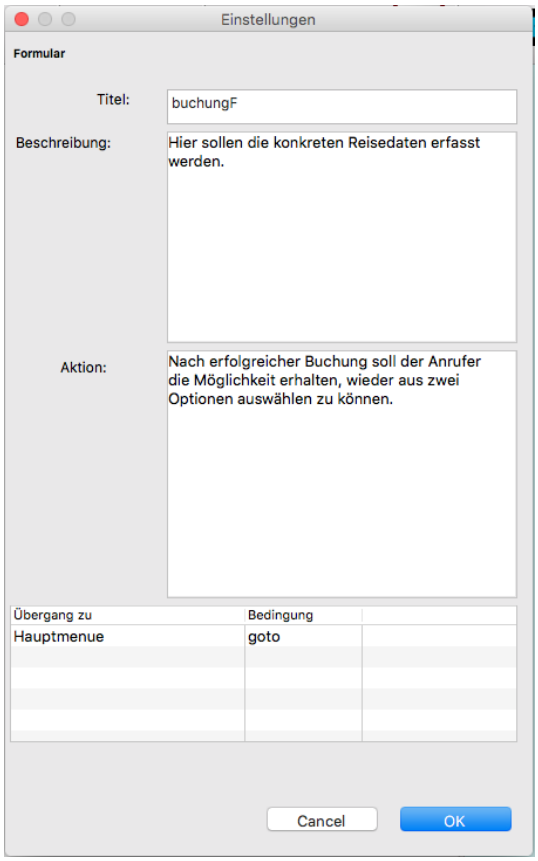



Abbildung C.19: Attributwerte des Formulars buchungF

Die Abbildungen C.18 und C.19 zeigen das Formular buchungF mit allen dazuge-

hörigen Eigenschaften. Dieses Formular verarbeitet alle Benutzereingaben erst am Ende, deshalb werden für die Felder keine weiteren Aktionen benötigt. Die Aktion des Formulars buchungF wird erst dann ausgelöst, wenn der Benutzer alle Felder mit gültigen Werten belegt bzw. zulässige Worte gesprochen hat.
Bei der Ausführung eines VoiceXML-Dialoges ist die Reihenfolge der Formulare und Menüs sehr wichtig. Durch das Anwählen eines Formular- bzw. Menü-Modellobjektes und anschließendem Drücken der Hand (letzter Button der Werkzeugleiste) markiert man das Modellobjekt, das ganz oben im VoiceXML-Quelltext steht und als erstes interpretiert werden soll.

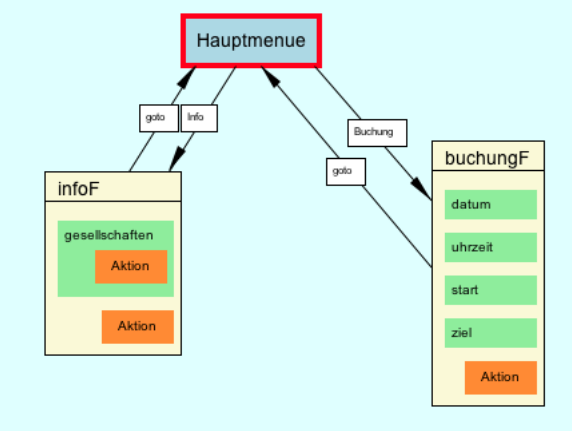



Abbildung C.20: **inES Festlegen des Startmenüs**

### Blöcke und Übergänge von Formularelementen

Damit ein Benutzer dieses Sprachdialoges beim Rücksprung in das Hauptmenü nicht immer wieder die Begrüßung des Sprachdialoges hören muss, ist die Modellierung eines Formulars zur Begrüßung sinnvoll. Da die Begrüßung lediglich aus einer Sprachausgabe besteht und keine Spracheingabe benötigt wird, ist die Modellierung eines Blockes angemessen. Blöcke speichern keine Spracheingaben, deshalb bleiben sie anonym.

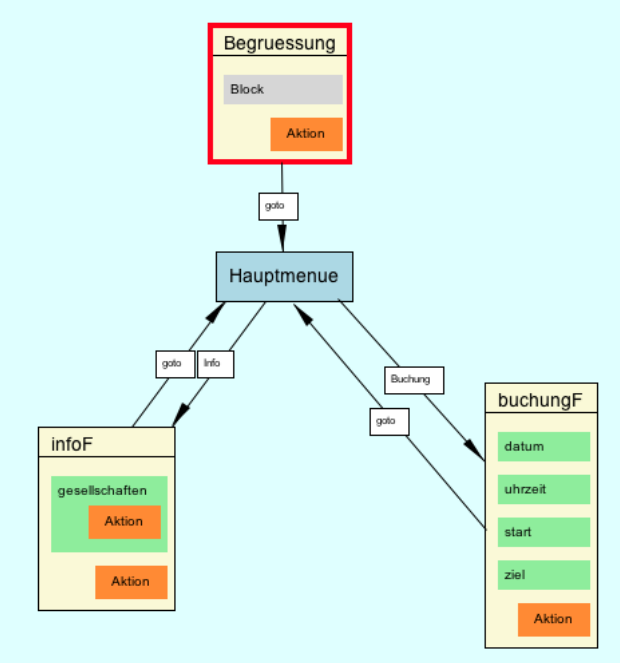



Abbildung C.21: **inES Blöcke für reine Sprachausgaben**

Sollte der Benutzer während des Buchens eines Fluges feststellen, dass bestimmte Daten, etwa der Start- oder der Zielort, falsch erkannt worden sind, dann besteht die Möglichkeit, bestimmte Felder innerhalb eines Formulars anzuspringen. Analog zur Modellierung von Übergängen zwischen Formularen und Menüs können auch Übergänge zwischen Feldern und Übergänge von Feldern zu Formularen modelliert werden. Durch die Implementierung zusätzlicher und auf Korrektheit prüfender Feld-Modellobjekte kann die Richtigkeit der Benutzereingaben durch den Benutzer selbst bestätigt werden und im Falle einer fehlerhaften Eingabe das zuletzt gesprochene wiederholt eingegeben werden (siehe Abbildung C.22).

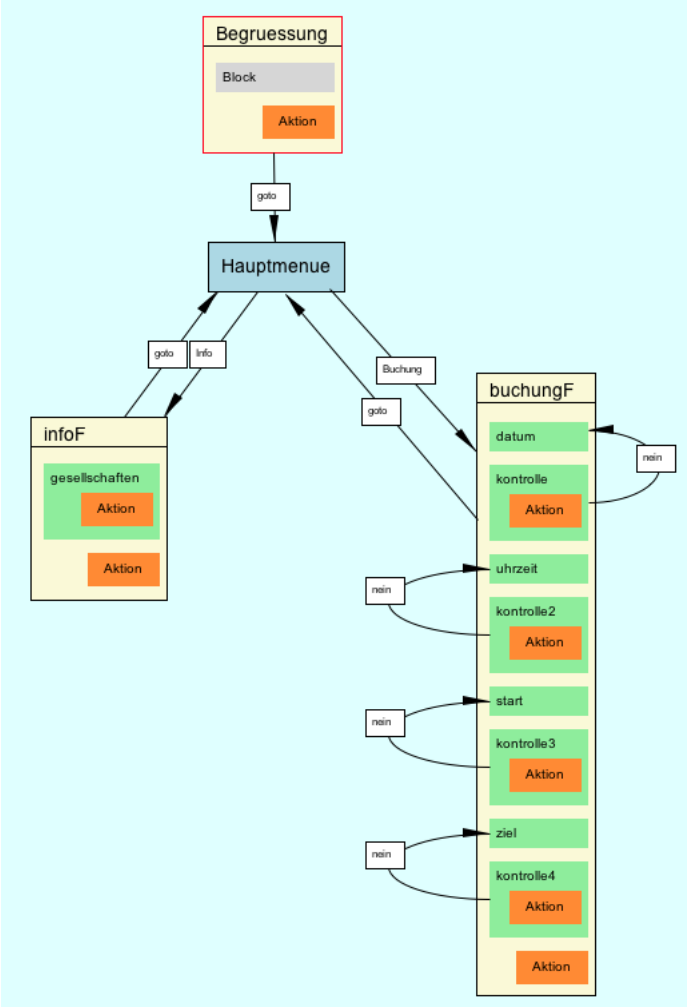



Abbildung C.22: **inES Sprünge innerhalb eines Formulars**

### C.1.3 Editieren und Validieren von VoiceXML-Quelltexten

inES stellt für das Implementieren, Bearbeiten und Überprüfen von VoiceXML-Quelltexten einen Editor zur Verfügung. Er unterstützt die Schüler mit einer Syntaxhervorhebung beim Schreiben des VoiceXML-Quelltextes. Des Weiteren stehen im Editor auch die üblichen Werkzeuge für das Kopieren, Ausschneiden und Einfügen zur Verfügung. VoiceXML-Anweisungen können in der enthaltenen Onlinehilfe nachgeschlagen werden. Der Button „Code-Vorlage“ der Werkzeugleiste erzeugt ein VoiceXML-Quelltextvorlage im Editor. Mithilfe des validierenden VoiceXML-Parsers kann die Syntax vor der Ausführung des Sprachdialoges getestet werden.

### Syntaxhervorhebung

Zur besseren Lesbarkeit des VoiceXML-Quelltextes unterstützt der Editor die Syntaxhervorhebung (engl. syntax highlighting). Wörter und Zeichen erscheinen in Abhängigkeit ihrer Bedeutung in unterschiedlichen Farben. Die Baumstruktur des VoiceXML-Quelltextes verlangt das Öffnen und Schließen von VoiceXML-Tags in der richtigen Reihenfolge. Jeder öffnende Tag muss in umgekehrter Reihenfolge wieder geschlossen werden. Beachten Schüler diese Reihenfolge nicht, ändern sich die Farben der Syntaxhervorhebungen nicht mehr und es können deshalb Fehler schneller gefunden und beseitigt werden. Kommentare werden ebenfalls in einer sich vom restlichen Quelltext abgesetzten Farbe dargestellt.

### Code-Vorlage

Jedes Modellobjekt besitzt eine textuelle Repräsentation im VoiceXML-Quelltext. Die Abbildung C.23 zeigt den Zusammenhang zwischen jedem grafischen Modellobjekt und seiner textuellen Repräsentation.

| Modellobjekt | Code-Vorlage |
|---|---|
| Menü | `<menu id="menuTitel">`<br>`  <!--Programmiere hier das Menue menuTitel aus.-->`<br>`</menu>` |
| Formular ohne Aktion | `<form id="formTitel">`<br>`  <!--Fuege hier die Formularelemente von formTitel ein.-->`<br>`</form>` |
| Formular mit Aktion | `<form id="formTitel">`<br>`  <!--Fuege hier die Formularelemente von formTitel ein.-->`<br>`  <filled>`<br>`  <!--Programmiere hier die Aktion.-->`<br>`  </filled>`<br>`</form>` |
| Feld ohne Aktion | `<field name="feldTitel">`<br>`  <!--Programmiere hier das Feld feldTitel aus.-->`<br>`</field>` |
| Feld mit Aktion | `<field name="feldTitel">`<br>`  <!--Programmiere hier das Feld feldTitel aus.-->`<br>`  <filled>`<br>`  <!--Programmiere hier die Aktion.-->`<br>`  </filled>`<br>`</field>` |
| Block | `<block>`<br>`  <!--Programmiere hier den anonymen Block aus.-->`<br>`</block>` |

Abbildung C.23: **Zusammenhang von Modellobjekt und VoiceXML-Quelltext**

Der VoiceXML-Quelltext wird zwischen dem Toplevel-Tag <vxml> und </vxml> ausprogrammiert. inES unterstützt keine automatische Vervollständigung der eingegebenen Anweisungen. Dafür kann der Schüler, anders als im Prototypen implementiert, durch Drücken des „Code-Vorlage“ aus der Werkzeugleiste eine vollständige VoiceXML-Quelltextvorlage erzeugen, die ihm die Tipparbeit erleichtert. Es müssen nur die nicht im Modell abgebildeten Anweisungen, z. B. Sprachausgaben oder sonstige Bedingungen, implementiert werden.
Die Abbildung C.24 zeigt den Zusammenhang zwischen dem Modell auf der rechten und dem VoiceXML-Quelltext auf der linken Seite.

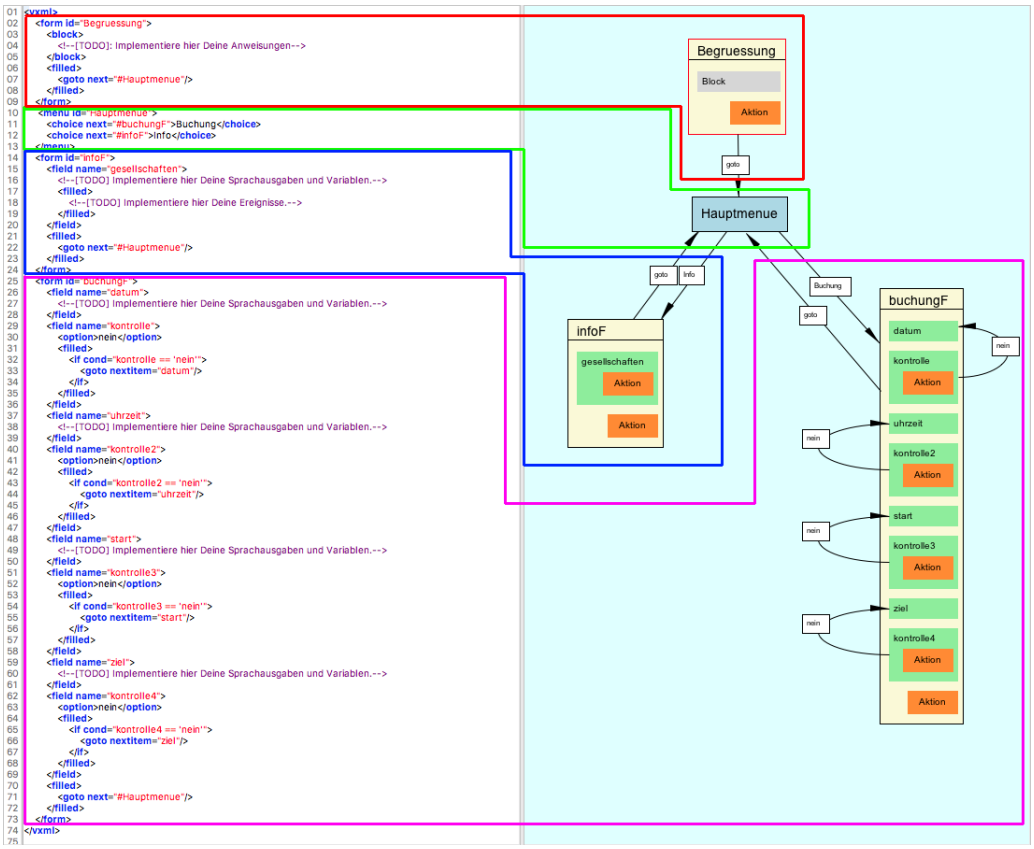



Abbildung C.24: **Code- und Modellblöcke**

Der ursprünglich objektorientierte Ansatz der prototypischen Implementierung von Hilger wird auch in der neuen inES-Anwendung verfolgt. Jedes Modellobjekt wird durch den von einem geöffneten Tag und einem geschlossenen Tag eingeschlossenen VoiceXML-Quelltext repräsentiert. Die im Modellierungsbereich dargestellte Kapselung findet sich dementsprechend im VoiceXML-Quelltext wieder.

Beispielsweise werden alle Form-Elemente von einem geöffneten und geschlossenen <form> eingebettet, wodurch man bereits am Modell erkennen kann, wie die Sprache VoiceXML aufgebaut ist.

### Hilfe zur Sprache VoiceXML

Den Schülern steht während ihrer Arbeit mit inES ein Hilfesystem zur Verfügung. Dort können alle in inES implementierten Tags nachgeschlagen werden. Die folgenden Informationen gehören zum Hilfesystem:

- allgemeine Informationen über inES
- Beschreibung der Tags
- Beschreibung der Modellobjekte
- Beschreibung der Attribute
- Informationen über die Verschachtelungsebenen

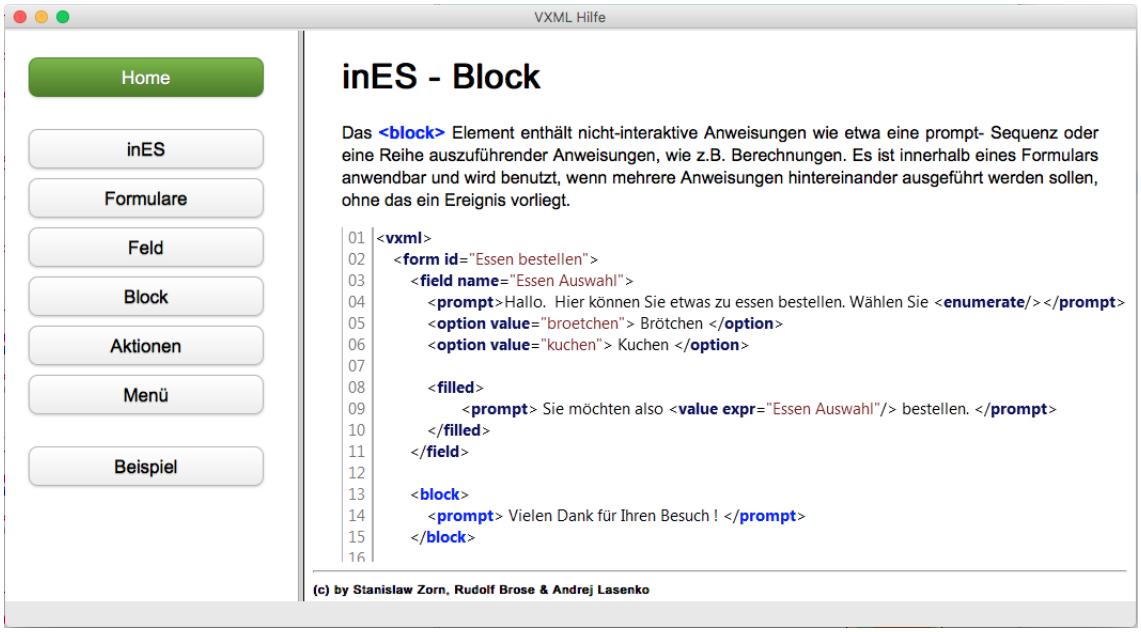



Abbildung C.25: **inES Onlinehilfesystem**

### Abgleich Code und Modell

Der Abgleich von Modell und VoiceXML-Quelltext ist im Vergleich zum Prototypen von Hilger in der aktuellen inES Version anders implementiert. Die aktuelle inES

Version kann in beide Richtungen, d. h. vom Modell zum Code und umgekehrt, synchronisieren. Damit entfällt der Zwang des grafischen Modellierens, denn es gibt Schüler, die lieber zuerst programmieren bzw. implementieren möchten. Sollten diese Schüler im Laufe ihrer Arbeit auf Probleme stoßen, können sie sich das Modell jederzeit erzeugen lassen.

#### Der Parser

inES kann einen VoiceXML-Quelltext auf Syntaxfehler überprüfen. Dabei wird der VoiceXML-Quelltext auf

- Wohlgeformtheit des XML Dokumentes
- Zulässigkeit von Tags
- Zulässigkeit der Verschachtelung von Tags
- Zulässigkeit der Attribute für Tags
- Zulässigkeit leerer Tags

überprüft. Für diese Überprüfung wird mithilfe des Quelltextes ein endlicher Automat erzeugt und vom VoiceXML-Interpreter ausgeführt. Man spricht in diesem Zusammenhang auch von einem sogenannten *validierenden Parser*.
Sobald ein Fehler auftritt, beendet der Parser die Ausführung und es wird die betreffende fehlerhafte Codezeile farbig markiert. Des Weiteren erscheint in der Statuszeile des inES Hauptfensters eine Fehlermeldung.

Der VoiceXML-Quelltext des Flugbuchungssystems befindet sich im Anhang.

## C.2 Das Plugin inGE - integrierter Grammatikeditor für inES

inGE, der **in**tegrierte **G**rammatikeditor für in**E**S, ist eine Erweiterung für die didaktische Lernumgebung inES. Mit inGE können zusätzlich formale Grammatiken für die VoiceXML-Sprachdialoge modelliert werden, sodass ganze natürlich gesprochene Sätze mit mehr als nur einem Wort erkannt werden können. Des Weiteren können aus den erkannten Sätzen bestimmte Daten extrahiert und den entsprechenden Feldvariablen für die Weiterverarbeitung zugewiesen werden.

### C.2.1 Die Ausführungsmodi von inGE

inGE kann aus didaktischen Gründen, auf die in den folgenden Kapiteln näher eingegangen wird, auf zwei Arten genutzt werden: entweder als PlugIn in inES integriert oder als separate Anwendung. Die Bedienung unterscheidet sich dabei nicht. Wird inGE als PlugIn benutzt, dann befindet sich ein entsprechendes Symbol in der Werkzeugleiste von inES (siehe Abbildung C.26).


Abbildung C.26: **Grammatik-Button**

Genau wie beim Hinzufügen von Aktionen können sowohl den VoiceXML-Formularen als auch den Feldvariablen formale Grammatiken durch Drücken des inGE-Buttons hinzugefügt werden. Die entsprechenden Formulare und Feldvariablen erhalten ein entsprechendes Label (siehe Abbildung C.27).


Abbildung C.27: **Feldvariable mit formaler Grammatik**

Alternativ kann inGE direkt und ohne inES gestartet werden. Das kann ggf. sinnvoll sein, wenn bestimmte formale Grammatiken nach Eigenschaften untersucht werden sollen und das Modellieren eines Sprachdialoges nicht im Mittelpunkt steht.

### C.2.2 Die Oberfläche von inGE

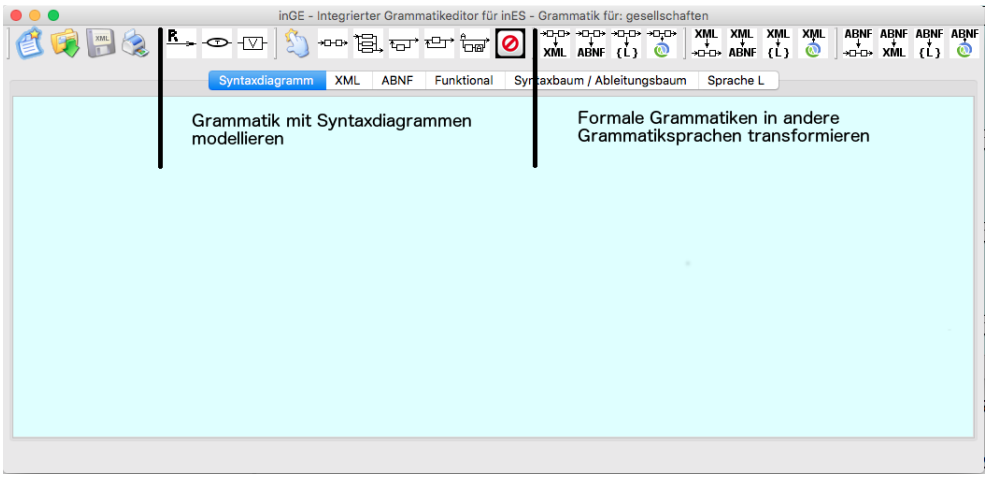



Abbildung C.28: **Oberfläche von inGE**

Die Oberfläche von inGE zeigt eine Werkzeugleiste und einen Arbeitsbereich, der durch einen Karteireiter direkt über dem Arbeitsbereich das Implementieren von formalen Grammatiken mithilfe von Syntaxdiagrammen, der XML und der ABNF zulässt. Des Weiteren lassen sich im Arbeitsbereich formale Grammatiken durch Ableitungsbäume oder dem exemplarischen Erzeugen von Sätzen testen. Die Werkzeugleiste besteht aus den Modellierungswerkzeugen für die Syntaxdiagramme und Buttons für das Transformieren einer formalen Grammatik in andere Darstellungsformen (z. B. der ABNF). Die folgende Tabelle stellt die Buttons der Werkzeugleiste sowie die Beschreibung ihrer Funktion dar.

| inGE - Die Werkzeugleiste - Teil 1 | |
|---|---|
| Button | Beschreibung der Funktion |
|  | Erstellt eine neue Regel. |
|  | Erstellt ein neues Terminal. In inGE können das Zeichen bzw. ganze Wörter sein. |
|  | Erstellt eine neue Variable (Nonterminal). |
|  | Wenn eine Regel markiert ist und dieser Button gedrückt wird, wird diese Regel zur Startregel (bzw. Startsymbol). |
|  | Erstellt eine neue Sequenz. Entspricht der UND-Konstruktion. |
|  | Erstellt eine neue Alternative. Entspricht der ODER-Konstruktion. |
|  | Erstellt eine neue Iteration. Wiederholungsstruktur für ein Wort oder eine Sequenz. Das Wort bzw. die Sequenz muss nicht zwingend gesprochen werden. |
|  | Erstellt eine neue Iteration. Wiederholungsstruktur für ein Wort oder eine Sequenz. Das Wort bzw. die Sequenz muss mindestens einmal gesprochen werden. |
|  | Modelliert eine neue Rekursion. |
|  | Löscht ein ausgewähltes Element. |

Tabelle C.1: Buttons der Werkzeugleiste in inGE

| inGE - Die Werkzeugleiste - Teil 2 | |
|---|---|
| Button | Beschreibung der Funktion |
|  | Überführt eine formale Grammatik von der Syntaxdiagrammdarstellung in die XML-Darstellung. |
|  | Überführt eine formale Grammatik von der Syntaxdiagrammdarstellung in die ABNF-Darstellung. |
|  | Erzeugt die zu einer formalen Grammatik in Syntaxdiagrammdarstellung gehörende Sprache, indem einzelne beispielhafte Sätze erzeugt werden. |
|  | Überführt eine formale Grammatik von der Syntaxdiagrammdarstellung in eine funktionale Darstellung. |
|  | Überführt eine formale Grammatik von der XML-Darstellung in die Syntaxdiagrammdarstellung. |
|  | Überführt eine formale Grammatik von der XML-Darstellung in die ABNF-Darstellung. |
|  | Erzeugt die zu einer formalen Grammatik in XML-Darstellung gehörende Sprache, indem einzelne beispielhafte Sätze erzeugt werden. |
|  | Überführt eine formale Grammatik von der XML-Darstellung in eine funktionale Darstellung. |
|  | Überführt eine formale Grammatik von der ABNF-Darstellung in die Syntaxdiagrammdarstellung. |
|  | Überführt eine formale Grammatik von der ABNF-Darstellung in die XML-Darstellung. |
|  | Erzeugt die zu einer formalen Grammatik in ABNF-Darstellung gehörende Sprache, indem einzelne beispielhafte Sätze erzeugt werden. |
|  | Überführt eine formale Grammatik von der ABNF-Darstellung in eine funktionale Darstellung. |

Tabelle C.2: Buttons der Werkzeugleiste in inGE

### C.2.3 Die Implementierung einer formalen Grammatik

Mit der Implementierung formaler Grammatiken wird beispielsweise das Formular *buchungF* des VoiceXML-Dialogs aus dem Kapitel 4.2.3 vereinfacht, denn es können alle implementierten Kontrollen, die für das Nachfragen und Bestätigen von Informationen benötigt wurden, weggelassen werden. Das Formular *buchungF* benötigt nur die Feldvariablen *datum*, *uhrzeit*, *start* und *ziel*, wie in Abbildung C.29 dargestellt.
Für die Implementierung einer formalen Grammatik muss das entsprechende Formular *buchungF* ausgewählt und danach der Grammatik-Button gedrückt werden. Dem markierten Formular wird daraufhin ein pink-farbiges Symbol *G*, wie in der folgenden Abbildung zu sehen, hinzugefügt.


Abbildung C.29: buchungF mit formaler Grammatik

Ein Doppelklick auf das pink-farbige Symbol G öffnet den Grammatikeditor *inGE*. Vor der eigentlichen Implementierung formaler Grammatiken müssen nun mögliche Antworten auf die Frage nach dem Flugziel überlegt werden.
Folgende Antwortmöglichkeiten wären denkbar:

- Ich möchte gern übermorgen 20.00 Uhr von München nach Frankfurt fliegen.
- Ich möchte von Berlin nach Hamburg fliegen.
- Ich will am Donnerstag nach Berlin fliegen.
- Ich möchte am Freitag um 8.00 Uhr von Berlin nach Hamburg fliegen.

Beim Betrachten der Antwortmöglichkeiten könnten zunächst folgende Merkmale entdeckt werden.

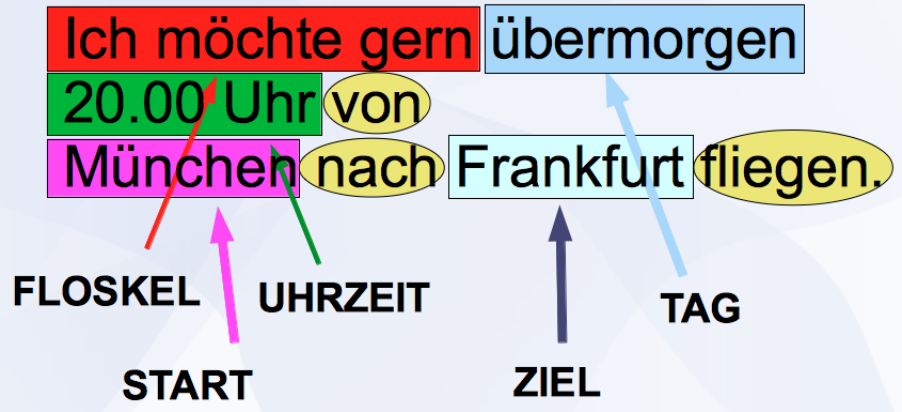



Abbildung C.30: Entwickeln einer formalen Grammatik

Phrasen wie z. B. „*Ich möchte*“ oder „*Ich will*“ sind Floskeln, mit denen höfliche Menschen einen Satz beginnen könnten. Sie enthalten keine nützliche Information für die Buchung eines Fluges, müssen jedoch bei der Erkennung natürlich gesprochener Sprache berücksichtigt werden. Nach Sprechen einer solchen Floskel könnte eine Zeitangabe folgen. Zeitangaben könnten aus Wochentagen, Daten bzw. aus einem simplen „*morgen*“ oder „*übermorgen*“ bestehen. Die Antworten sollten des Weiteren eine Start- und Zielangabe enthalten. Die Abbildung C.30 zeigt eine exemplarische Antwortmöglichkeit. Die rechteckig umrahmten Wörter bzw. Wortgruppen sind die später modellierten Nonterminale (Variablen) und die oval umrahmten Wörter sind mögliche auftretende Terminale. Für die Implementierung von Nonterminalen einer formalen Grammatik für die Buchung von Flügen könnten die folgenden Mengen definiert werden.

$$
\begin{aligned}
FLOSKEL &:= \{\text{„Ich möchte“}, \text{„Ich will“}, \dots\} \\
TAG &:= \{\text{„Montag“}, \dots, \text{„Sonntag“}, \text{„morgen“}, \text{„übermorgen“}, \dots\} \\
UHRZEIT &:= \{\text{„8:00“}, \text{„16:00“}, \text{„20:00“}\} \\
START &:= \{\ \text{„Berlin“}, \text{„Hamburg“}, \text{„München“}, \text{„Frankfurt“}\ \} \\
ZIEL &:= \{\ \text{„Berlin“}, \text{„Hamburg“}, \text{„München“}, \text{„Frankfurt“}\ \}
\end{aligned}
$$

Für die natürlich gesprochenen Antwortmöglichkeiten kann die erste Regel, die Startregel bzw. das Startsymbol, in *inGE* modelliert werden. Mit dem „Regel“-Button der Werkzeugleiste wird eine Regel hinzugefügt. Als Bezeichner für diese Regel wählt man beispielsweise „S“.

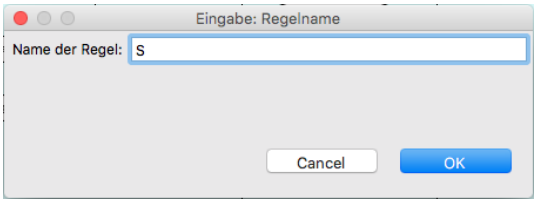



Abbildung C.31: inGE Dialog neue Regel

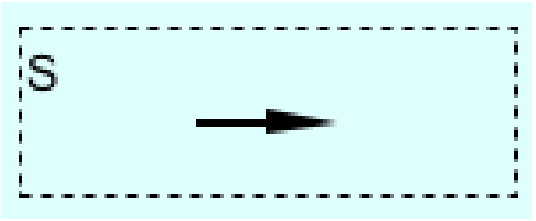



Abbildung C.32: inGE Die leere START-Regel

Nun können weitere Elemente, das sind Terminale oder Nonterminale, dieser „S“-Regel hinzugefügt werden. Da die Antwortmöglichkeiten mit einer Floskel beginnen und von einer Zeitangabe folgen sowie mit der Angabe eines Start- bzw. Zielortes enden, können die Nonterminale und Terminale direkt aufeinander folgen. Das hintereinander anführen von Nonterminalen und Terminalen entspricht einer UND-Konstruktion. Des Weiteren erzeugt jedes neu hinzugefügte Nonterminal automatisch eine neue Regel, die anschließend definiert werden muss. Zunächst wird dafür die „S“-Regel ausgewählt und danach auf den Variablenbutton gedrückt. Wird ein neues Nonterminal hinzugefügt, stellt *inGE* das durch ein Rechteck mit einem in der Mitte befindlichen *V* dar. Der Bezeichner lässt sich durch einen Doppelklick bearbeiten. Sobald ein Variablenbezeichner bearbeitet wird, ändert sich automatisch auch der Bezeichner der neu hinzugefügten Regel. Ein Terminal wird auf die gleiche Art und Weise hinzugefügt. *inGE* erstellt jedoch für ein neu hinzugefügtes Terminal keine neue Regel.

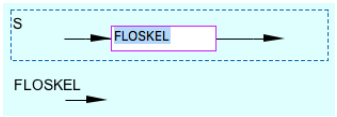



Abbildung C.33: Entwickeln einer formalen Grammatik

Um ein weiteres Nonterminal der „S“-Regel hinzuzufügen, muss eine neue Sequenz hinzugefügt werden. Hierfür muss wieder die vorhandene „S“-Regel ausgewählt und anschließend der entsprechende Sequenzbutton (⇢□□⇢) gedrückt werden. Daraufhin wird eine anwählbare schwarze Linie der Regel hinzugefügt. Fährt man mit der Maus über diese Linie, verfärbt sie sich pink. Wird im pinken Zustand die linke Maustaste gedrückt, ist die Linie angewählt und es können weitere Elemente, z. B. ein neues Nonterminal, hinzugefügt werden.

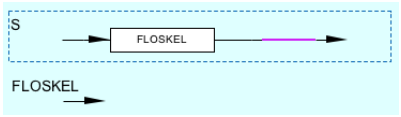



Abbildung C.34: Entwickeln einer formalen Grammatik

Diese Schritte werden solange wiederholt, bis alle notwendigen Nonterminale hinzugefügt sind. Zuletzt wird mithilfe des Handbuttons (  ) aus der Werkzeugleiste die „S“-Regel als Startsymbol gekennzeichnet.

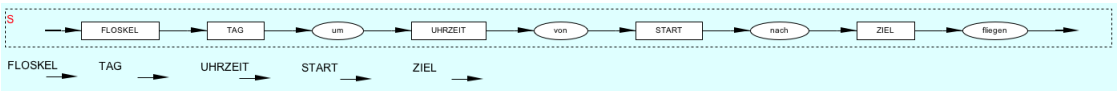



Abbildung C.35: Entwickeln einer formalen Grammatik

Nach Implementierung der Startregel werden die Regeln „*FLOSKEL* “, „*TAG*“, „*START*“
und „*ZIEL*“ modelliert. Ein Anrufer wird, wenn er mit einer Floskel beginnt, entweder „*Ich möchte*“ oder „*Ich will*“ sprechen. Er wird nicht beides nacheinander sagen. Entsprechend wird für die Implementierung eine Entweder-ODER-Konstruktion benutzt. Dafür wählt man die Regel *FLOSKEL* an und drückt den „Alternative“-Button (  ). Standardmäßig erzeugt *inGE* zwei Alternativen. Werden mehr Alternativen benötigt, muss eine Alternative ausgewählt und der entsprechende Button erneut gedrückt werden.

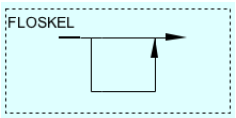



Abbildung C.36: Entwickeln einer formalen Grammatik

Das Hinzufügen von Terminalen geht analog zum Hinzufügen von Nonterminalen. Man klickt mit der Maus auf eine der Linien, an denen ein Terminal oder Nonterminal hinzugefügt werden soll und drückt die entsprechenden Button. Die Abbildung C.37 zeigt die fertige *FLOSKEL*-Regel.

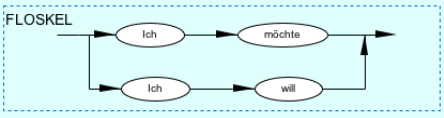



Abbildung C.37: Entwickeln einer formalen Grammatik

Die Regel für den Tag, den Start und das Ziel bestehen ebenfalls aus solchen Alternativen. Die Abbildung C.38 zeigt eine mögliche formale Grammatik für die Antwortmöglichkeiten.

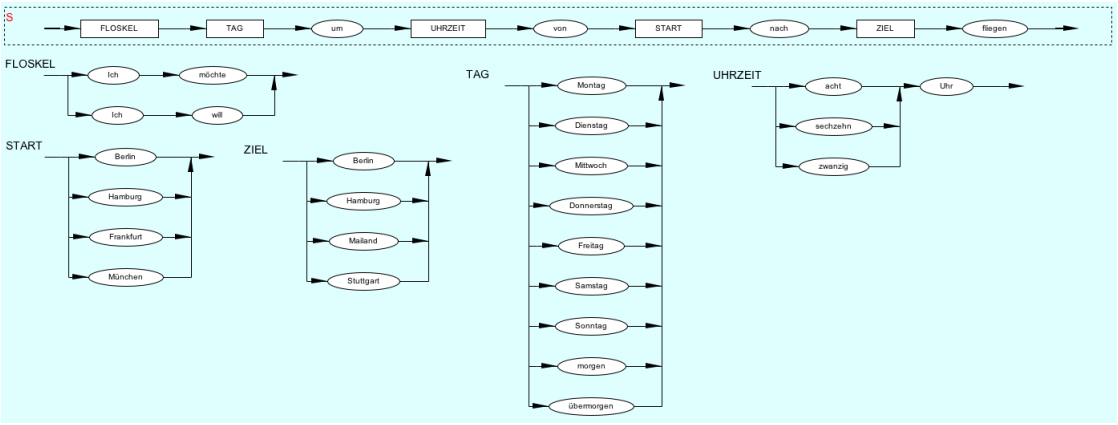



Abbildung C.38: Entwickeln einer formalen Grammatik

## C.2.4 Formale Grammatiken untersuchen

Für die Überprüfung einer modellierten formalen Grammatik ergeben sich zwei Möglichkeiten. Enthalten wie in diesem Beispiel die formalen Grammatiken sehr wenige Regeln und kaum Wiederholungsstrukturen (z. B. Rekursionen), kann die zu einer formalen Grammatik gehörende Sprache erzeugt werden. *inGE* stellt hierfür drei Button zur Verfügung. Je nachdem ob aus einem Syntaxdiagramm, einer ABNF- oder XML-Darstellung die Sprache erzeugt werden soll, muss hierfür der entsprechende Button gedrückt werden ( , ABNF, XML). Nach Drücken eines der Buttons zeigt *inGE* eine exemplarische Auswahl an Sprachbeispielen.

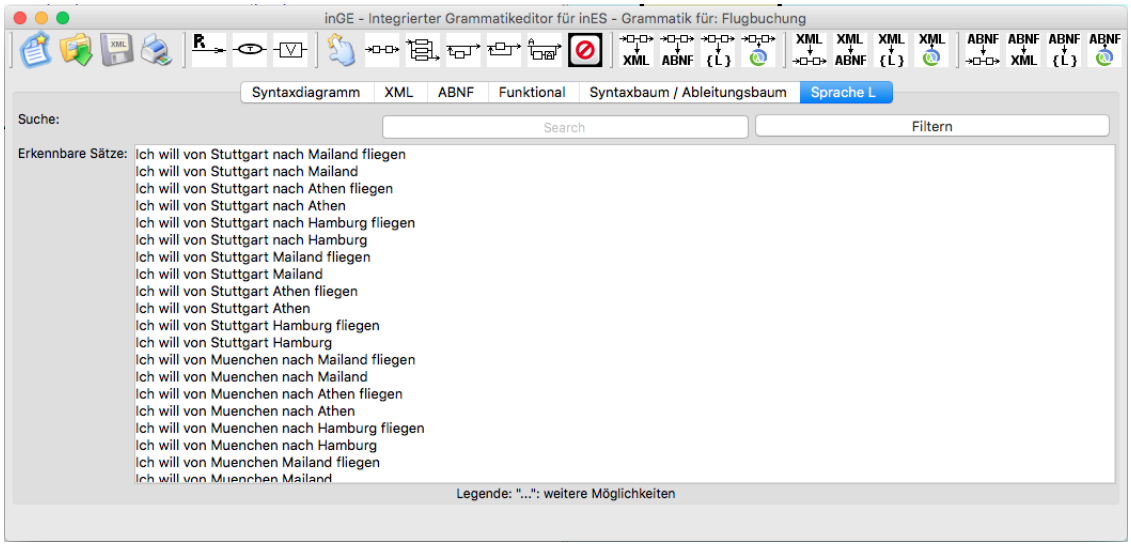



Abbildung C.39: Sprache erzeugen

Der Nachteil dieser Variante ist, dass der entsprechende Satz gesucht werden muss. Des Weiteren kann es passieren, dass aufgrund unendlicher Kombinationsmöglichkeiten nur eine beschränkte Auswahl angezeigt werden kann und der gewünschte Satz nicht mit aufgeführt wird. Es ist besser, den gewünschten Satz per Tastatur einzugeben und parsen zu lassen. Die Abbildung C.40 zeigt den zu dem Satz „*Ich möchte morgen von Berlin nach Hamburg*“ zugehörigen Ableitungsbaum, der aufgrund der modellierten Grammatik gebildet werden kann. Sobald der Ableitungsbaum fehlerhaft oder gar nicht erzeugt werden kann, muss der Fehler in der formalen Grammatik gesucht werden.

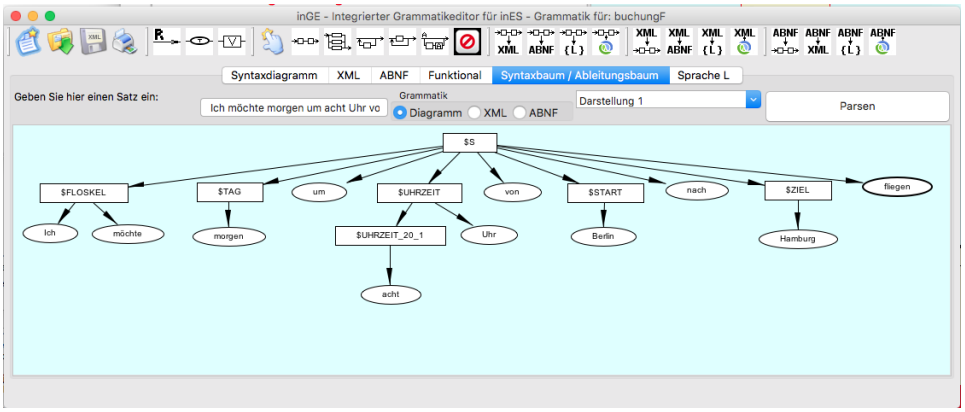



Abbildung C.40: Ableitungsbäume erzeugen

### C.2.5 ABNF- und XML-Grammatiken

Die in Kapitel 4.3.4 modellierte formale Grammatik liegt bisher nur als Syntaxdiagramm vor. Syntaxdiagramme sind für den Menschen gut verständlich, jedoch für die Verarbeitung in einem VoiceXML Sprachdialog nicht geeignet. Für VoiceXML Sprachdialoge muss die formale Grammatik in einer XML- oder ABNF-Darstellung vorliegen. *inGE* kann deshalb Syntaxdiagramme in die XML- bzw. die ABNF-Darstellung transferieren. Im Übrigen geht das auch in umgekehrter Richtung, d. h. formale Grammatiken in XML- bzw. ABNF-Darstellung können mithilfe von Syntaxdiagrammen dargestellt werden.
Für die Transformation einer formalen Grammatik in Syntaxdiagrammdarstellung in XML- oder ABNF-Darstellung stehen in *inGE* die entsprechenden Button (  ) zur Verfügung.

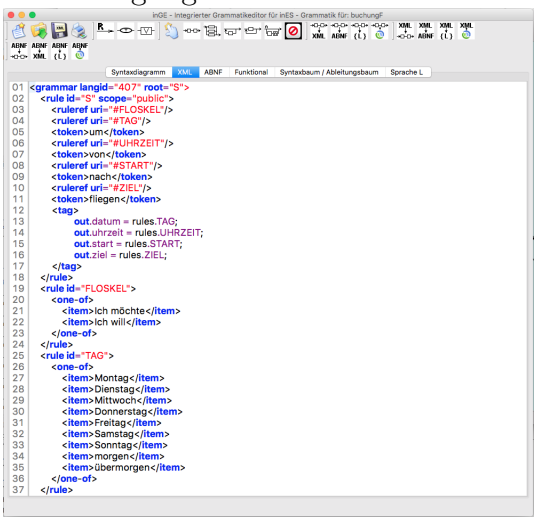



Abbildung C.41: inGE Grammatik in XML Darstellung

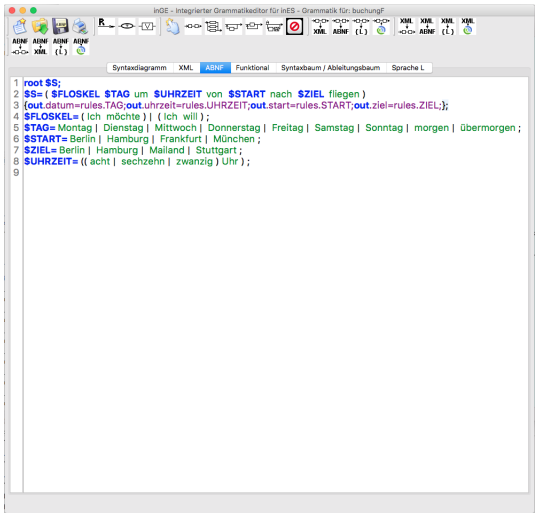



Abbildung C.42: inGE Grammatik in ABNF Darstellung

Damit aus einem natürlich gesprochenen Satz die für diesen VoiceXML Sprachdialog notwendigen Daten extrahiert werden können, reicht ein Transferieren des Syntaxdiagramms allein nicht aus. Es müssen für den Dialog wichtige Metadaten, sogenannte Tags, hinzugefügt werden. Der VoiceXML Sprachdialog enthielt die Feldvariablen *datum*, *uhrzeit*, *start* und *ziel*. Dementsprechend müssen diesen Feldvariablen die entsprechenden Ergebnisse des natürlich gesprochenen Satzes zugewiesen werden. Wurde die formale Grammatik nach XML transferiert, muss dem XML Quellcode bei der Startregel folgendes hinzugefügt werden:

```
1  <tag>
```

```
2    out.datum=rules.TAG;
3    out.uhrzeit=rules.UHRZEIT;
4    out.start=rules.START;
5    out.ziel=rules.ZIEL;
6  </tag>
```

Über das *out*-Objekt des XML-Quellcodes wird das VoiceXML-Formular angesprochen, dem diese formale Grammatik zugeordnet ist. In diesem Beispiel ist es das Formular *buchungF*. Mithilfe der Punktnotation kann auf die entsprechenden Feldvariablen eines VoiceXML-Formulars zugegriffen werden. Mit dem *rules*-Objekt des XML-Quellcodes können die Ergebnisse des natürlich gesprochenen Satzes abgefragt werden. Das *rules*-Objekt kann über die Punktnotation gezielt die Daten des für eine Regel Gesprochenen abfragen. Z. B. liefert *rules.TAG* beim gesprochenen Antwortsatz *Ich möchte morgen um acht Uhr von Berlin nach Hamburg fliegen* das Wort *morgen*. Die Zeile eins des obigen Quelltextauszuges weist dementsprechend den Wert *morgen* der Feldvariablen *datum* des VoiceXML Formulars *buchungF* zu. In der ABNF-Darstellung können die Tags analog eingebunden werden, wie der folgende ABNF-Quelltext zeigt.

```
1  root $S;
2  $S= ( $FLOSKEL  $TAG  um  $UHRZEIT  von  $START  nach  $ZIEL  fliegen )
3  {out.datum=rules.TAG;out.uhrzeit=rules.UHRZEIT;out.start=rules.START;out.ziel=rules.ZIEL;};
4  $FLOSKEL= ( Ich  möchte ) |  ( Ich  will ) ;
5  $TAG= Montag |  Dienstag |  Mittwoch |  Donnerstag |  Freitag |  Samstag |  Sonntag |
   morgen |  übermorgen;
6  $START= Berlin |  Hamburg |  Frankfurt |  München;
7  $ZIEL= Berlin |  Hamburg |  Mailand |  Stuttgart;
8  $UHRZEIT= (( acht |  sechzehn |  zwanzig ) Uhr );
```

### C.2.6 Die formale Grammatik in den VoiceXML Sprachdialog einbinden

Zunächst muss man sich für eine der zwei Darstellungsformen formaler Grammatiken, entweder XML- oder ABNF-Darstellung, entscheiden. Die formale Grammatik in der gewünschten Darstellungsform muss mit dem Diskettensymbol separat gespeichert werden. Die Diskettensymbole tragen entsprechend der gewählten Darstellungsform die Bezeichnung XML bzw. ABNF. Wird beispielsweise die formale Grammatik in XML-Darstellung gespeichert, erzeugt *inGE* eine Datei, die den Namen der gerade geöffneten *inES*-Datei gefolgt vom Formular dessen Grammatik bearbeitet wird und der Dateiendung *xml* trägt. Über den Code-Vorlage Button (  ) in *inES*

kann der Pfad zur Datei automatisch im Quelltext des VoiceXML-Sprachdialoges eingefügt werden. Der folgende Quelltext-Ausschnitt des VoiceXML-Sprachdialoges zeigt das Ergebnis des automatischen Einfügens exemplarisch.

```
1  <form id="buchungF">
2    <grammar src="/Users/sven/Flugbuchung_mit_Grammatik-ines.buchungF.xml" type="text/xml"/>
3    <initial>
4      <prompt>Bitte nennen Sie Start und Zielort der gewünschten Verbindung.</prompt>
5    </initial>
6    <field name="datum">
7    </field>
8    <field name="uhrzeit">
9    </field>
10   <field name="start">
11   </field>
12   <field name="ziel">
13   </field>
14   <filled>
15     <prompt>Sie wollen <value expr="datum" /> um
16       <value expr="uhrzeit" /> von <value expr="start" />
17       nach <value expr="ziel" /> fliegen.</prompt>
18       <goto next="#Hauptmenue"/>
19   </filled>
20 </form>
```

Eine besondere Aufmerksamkeit sei dem benutzten <initial>-Tag der Zeile drei bzw. Zeile fünf gewidmet, denn dort zeigt sich, dass dieser VoiceXML-Sprachdialog ein Dialog mit gemischter Initiative ist und ein Sprechen natürlich gesprochener Sätze als Eingabe erwartet wird.

### C.2.7 Das Testen des Sprachdialoges unter Verwendung formaler Grammatiken

Das Ausführen des VoiceXML-Sprachdialoges unter Verwendung einer formalen Grammatik funktioniert nach demselben Schema wie bei einem Dialog ohne einer eigenen formalen Grammatik. Die Abbildung C.43 zeigt den Testlauf des VoiceXML-Sprachdialoges. Die Abbildung C.44 zeigt, dass auch hier ein Benutzen des Sprachdialoges ohne Spracherkennung möglich ist. Der Satz kann über die Tastatur eingegeben werden.

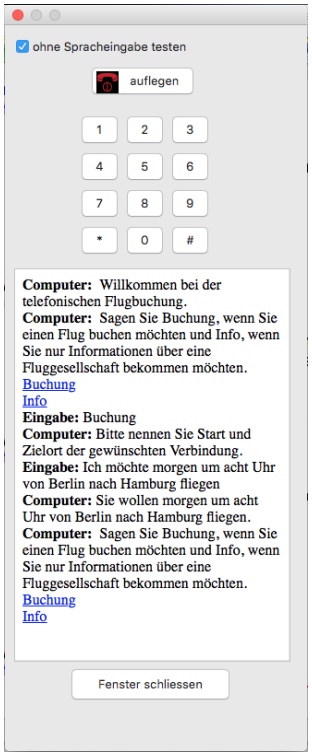



Abbildung C.43: Test des Sprachdialoges über Spracherkennung

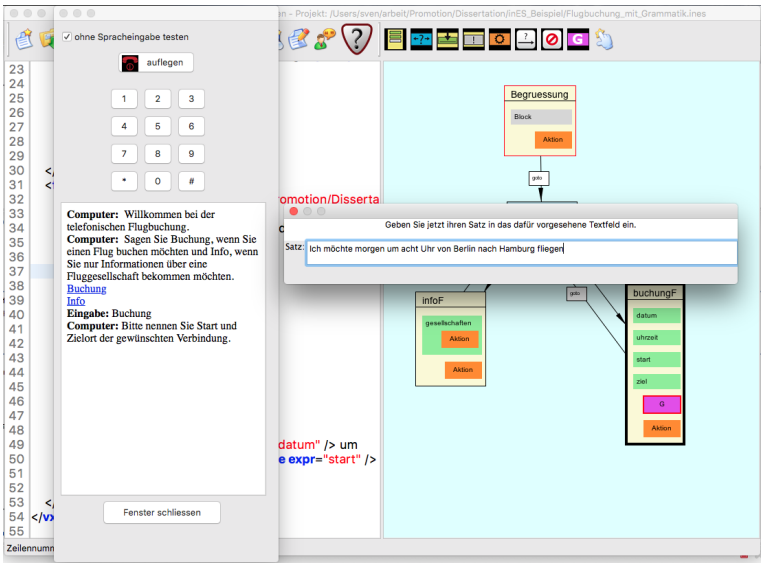



Abbildung C.44: Test des Sprachdialoges ohne Spracherkennung

# Anhang D

# Results of an expert interview as foundation for a study about the pedagogical added value by informatics in context

# Results of an expert interview as foundation for a study about the pedagogical added value by informatics in context

Sven Alisch
Gymnasium Lohbrügge
Binnenfeldredder 5
21031 Freie und Hansestadt Hamburg, Germany
s.alisch@li-hamburg.de

## ABSTRACT

The topic of this article is an interface between didactics and theoretical concepts of informatics. It describes *Pedagogical Added Value* and investigates how it can be defined, characterized and quantified. The basis of this research is a theoretical semester focusing on language translation. The analysis uses a guided interview to qualitatively describe the experiences of six Hamburg teachers. Based on these results, future research steps are proposed.

## Categories and Subject Descriptors

K.3.2 [**Computer and Information Science Education**]: Accreditation, Computer science education, Curriculum, Information systems education

## General Terms

Experimentation, Human Factors, Legal Aspects

## Keywords

informatics in context, language translation, functional programming, speech, formal grammar, formal language

## 1. INTRODUCTION

The informatics curriculum in Hamburg uses a typical application-oriented approach. This approach strongly emphasises the process of modelling, and is based on professional software design methods. The objective of application-orientated approach is ability to accomplish real life tasks in modern digital society (cmp. [1], pp. 10). Here application-orientated means the use of fundamental informatics theory for pratical purposes.

The application-orientated approach to language translation utilizes formal grammar theory and automata theory. During the semester students program with Scheme, a functional programming language. After implementing a word-by-word translator, they have to use formal grammars to improve performance. Therefore they have to model a formal grammar for their own native language. Afterwards students have to implement a parser for their native language. During five years of teaching these theoretical concepts, the author recognized many weak spots within the curriculum. In particular, functional programming methods do not emphasize important concepts from formal grammars and automata theory. For example, language parser automata are a challenge to implement using functional programming. The following questions were especially interesting to the author:

1. Is the application-oriented approach responsible for the weakness in the curriculum?
2. Is language translation a feasible application context?
3. Is it proper to use a functional programming language in this application context?
4. Did the author teach the curriculum in the right way?
5. What are the reasons for not achiving the objectives of the curriculum?
6. How could a new revised curriculum be drafted to improve the teaching of theoretical concepts?
7. How could a new revised curriculum be evaluated?

The last question leads to the objectives of this research. Therefore the term *Pedagogical Added Value* (PAV) is suitable, because question five is fundamental for the author, the term is introduced here. Stratmann defines in the following way: "[The *PAV*] is divided into two categories [...] first, do the new methods and media achive the **prior objectives**. Second, do the new methods successfully accomplish **new objectives** (cmp. [5]). A new teaching unit has an *PAV* if it increases efficiency and effectivness or improves the learning experiences. Stratmann uses this term in connection with a study of the deployment of notebook computers at a college. In his work he verified the use of notebooks led to an improvement in digital learning environments.

## 2. RESEARCH DESIGN

A qualitative approach is suitable because of the small numbers of teachers that teach this curriculum in Hamburg. The curriculum in Hamburg was recently reformed and few

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
*WiPSCE '13,* November 11–13, 2013, Aarhus, Denmark.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2455-7/11/13
http://dx.doi.org/10.1145/2532748.2532779

courses are taught at a high level. Theoretical concepts are only a component of these courses.
The author used guided interviews for his investigation. The questions, the code design, and the analysis of the interviews followed reference [4].

## 2.1 The interviewed teachers

The interviewed teachers have experience teaching the topic of language translation. They have a university degree in informatics. They have experience teaching the application-orientated approach to language translation, and experience with teaching secondary education students. The candidates were found by advertising on a mailing list reaching every Hamburg informatics teacher. Six candidates were found.

## 2.2 The code-system

Analysing a guided interview needs a definition of codes. Figure 1 shows the four main codes: objectives,

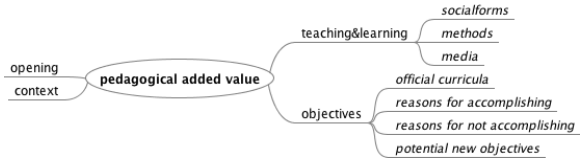



**Figure 1: The codes and subcodes of the guided interview**

teaching and learning, opening, and context. These terms are derivated from the definition of the *PAV*.

**objectives:** This code is very important, because it is a fundamental component of the *PAV* definition. The author wanted to explore the question: What objectives did other teachers accomplish at the end of that curriculum? What elements helped reach the objectives? What kind of obstacles prevented them from reaching objectives? What are the root causes for not reaching objectives? Are their experiences similar to the experiences of the author?

**teaching and learning:** When the author taught this curriculum, the integrated development environment was DrRacket, which has no visual elements. This made activity-orientated learning more difficult. What are the experiences of other teachers? Did the students work alone or in groups? Which kind of software (e.g. learning environments) did they use?

**opening, and context:** How do the teachers initiate the curriculum? Is language translation a suitable task? What kind of experiences did they have with language translation?

## 2.3 Results

The result of the interviews was, that for most of the teachers, language translation is an interesting task with a relatively small scope, but it has lots of limitations:

- no student motivation when entering into the topic
- no sense of achievement from creativity and innovation because it is not a free style project
- complexity of a functional programming language
- the use of a primitive development environment, which does not make problems visible
- considerations in connection with mathematical perspectives and the official curriculum (because of the functional paradigm)
- no modelling and "playing" with formal grammars and formal languages
- insofar as teachers implement a parser, they do it without learning the background of theoretical concepts of informatics
- a jumping to other topics within this curriculum confuses students
- not all of the theoretical informatics objectives in the official curriculum are reached

This investigation provides evidence of both good aspects and deficits in the application-orientated approach. A result is that the current unit is a functional programming language course, and has some point of contacts with language translation. The danger of a programming language course is focussing to much on an algorithm approach like in the 1970's. This would conflict with the official curriculum itself. The theoretical concepts of informatics, formal languages and formal grammars are not tought in depth.

# 3. CONCLUSION

These problems and deficits of the application-orientated approach are the basis for a new approach based on Informatics in Context (IniK), which is fundamentally an approach focussed on a real world phenomena (cmp. [3], pp. 102) and a multidimensional context. The next steps are the development of a new teaching unit based on IniK, and the development of a digital learning environment as a kind of an integrated development environment for speech (inES[1]). Furthermore the *PAV* of the new approach should be quantitatively compared to the older approach.

# 4. REFERENCES

[1] C. Albowski, M. Janneck, J. Schöttler, and M. Seiffert. Bildungsplan gymnasiale oberstufe. http://www.hamburg.de/contentblob/1475204-/data/informatik-gyo.pdf, 2009.

[2] S. Alisch. inES - integrierte Entwicklungsumgebung für Sprachen. http://www.ines-ide.de, 07 2013.

[3] I. Diethelm, J. Koubek, and H. Witten. Inik - informatik im kontext. *LOG IN*, 2011.

[4] U. Kuckartz, T. Dresing, S. Rädiker, and C. Stefer. *Qualitative Evaluation - Der Einstieg in die Praxis*. VS Verlag für Sozialwissenschaften, 2008.

[5] J. Stratmann. *Pädagogischer Mehrwert und Implementierung von Notebooks an der Hochschule*. Waxmann, 2007.

[1]First versions of that learning environment can be downloaded under [2]

# Anhang E

# eigenständige Veröffentlichungen

- Results of an expert interview as foundation for a study about the pedagogical added value by informatics in context, Veröffentlichung siehe Quelle: [Ali13b] und Anhang D.

# Anhang F

# Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Dissertationsschrift selbst verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.