

Benedikt Neuhold

Adaptives Informationssystem zur Erlernung mehrstelliger Addition und Subtraktion

Diplomarbeit

Technische Universität Graz

Institut für Informationssysteme und Computer Medien
Leiter: Univ.-Prof. Dipl.-Ing. Dr.techn. Frank Kappe

Betreuer: Univ.-Doz. Dipl.-Ing. Dr.techn. Martin Ebner

Graz, Juli 2013

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Graz, _____
Date Signature

Eidesstattliche Erklärung¹

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am _____
Datum Unterschrift

¹Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008; Genehmigung des Senates am 1.12.2008

Kurzfassung

Durch den rasanten Aufstieg des Internets in den letzten Jahren stehen immer mehr digitale Daten von Benutzern zur Verfügung. Dies stellt eine Chance für den Bildungsbereich dar, mit diesen Daten zu arbeiten, diese zu analysieren und daraus wertvolle Schlüsse ziehen zu können. Mit der Thematik der Datensammlung und Analyse beschäftigt sich der Forschungsbereich „Learning Analytics“. Dieser stellt Tools und Techniken zur Auswertung von Daten, welche durch Lernmanagementsysteme generiert werden, zur Verfügung. In vielen Fällen sind diese Tools jedoch sehr komplex und für das Lehrpersonal kaum praktisch anwendbar.

Das Hauptziel dieser Arbeit ist die Konzeption und die Implementierung eines adaptiven Informationssystems für Schülerinnen und Schüler der dritten bis fünften Schulstufe. Dieses System mit dem Namen „Plusminus-Trainer“ bietet die Möglichkeit, Addition und Subtraktion zu trainieren und eine Echtzeitfehleranalyse zu erhalten. Diese unmittelbare Rückmeldung ist nicht nur für Schülerinnen und Schüler wertvoll, sondern ist insbesondere für Lehrpersonen nützlich, da der Plusminus-Trainer Überblick über den aktuellen Wissensstand eines jeden Lernenden gibt und mögliche Schwierigkeiten und Fehlermuster aufzeigt.

Die Arbeit gliedert sich im Wesentlichen in zwei Abschnitte. Der erste Abschnitt umfasst einen Überblick über aktuelle Forschung im Bereich „Learning Analytics“, sowie eine detaillierte Untersuchung der beiden schriftlichen Rechenverfahren der Addition und Subtraktion und deren häufigsten Fehler. Zusätzlich wurde mathematische Software, welche dem Plusminus-Trainer ähnlich ist, untersucht. Der zweite Abschnitt ist zur Gänze der entwickelten Software gewidmet. Zunächst werden die technische Konzeption sowie die Implementierung des Informationssystems erläutert. Außerdem wurde die Software an einer österreichischen Volksschule (Grundschule) getestet. Daten von insgesamt 34 Schülerinnen und Schüler wurden gesammelt und analysiert. Diese Daten ließen Fehlermuster und durchwegs unterschiedliche Lernkurven der jeweiligen Schülerinnen und Schüler erkennen. Ebenso konnten durch diesen Test einige Mängel in der Software gefunden sowie neue Ideen für die Weiterentwicklung der Software gewonnen werden.

Abstract

The increasing usage of the internet in a variety of areas has provided digital data of its users, which can be of particular interest for the area of education. Specific data of students can be of advantage for educational institutions as they allow insight into students' learning processes and thought patterns. The field of research called "Learning Analytics" has devoted itself to gathering and analyzing such data. In particular, tools and methods for the examination and interpretation of data generated by learning management systems have been developed in the last years. Most tools, however, are highly complex and thus not applicable for teachers.

The central aim of this thesis is the conception and implementation of an adaptive information system called "Plusminus-Trainer" designed for students of year three to five. The main advantage of this system is the possibility to practice written additions and subtractions and to receive real-time analysis of errors. This feedback is not only valuable for students but most importantly for teachers, who receive information about each students' level of knowledge as well as possible difficulties and error patterns.

This thesis is divided into two main parts. The first part is devoted to an overview of research within the field of Learning Analytics, as well as an overview of written calculation methods used in primary schools and in particular students' common error patterns. This is followed by an outline of mathematical learning software similar to the developed Plusminus-Trainer. The second part of the thesis is entirely devoted to the developed information system. Firstly, the technical design and implementation of the Plusminus-Trainer are described. Secondly, the developed system was tested in an Austrian primary school. Data of 34 students were gathered and analyzed. Interestingly, the data revealed common error patterns and highly differing learning curves of students. The testing of the system was also crucial in order to identify further ideas for improvement.

Danksagung

Ein großer Dank gilt meinem Betreuer der Diplomarbeit Herrn Martin Ebner. Er stand mir bei der Erstellung dieser Arbeit und bei etwaige Fragen stets mit Rat und Tat zur Seite. Bei Herrn Martin Schön möchte ich mich für die anregenden Diskussionen und das konstruktive Feedback rund um die Entwicklung des Trainers bedanken.

Weiterer Dank geht an die Schüler, sowie deren Klassenlehrerinnen, mit denen ich die Evaluation der entwickelten Software durchführen durfte und an die Direktion der Volksschule, die mir dieses Projekt ermöglichte. Weiters möchte ich mich bei der Hauptschule bedanken, die den Computerraum für meine Evaluation zur Verfügung stellte.

Für die Korrektur meiner Diplomarbeit möchte ich mich bei Herrn Helmut Holzschuster sehr herzlich bedanken.

Ebenso möchte ich mich bei allen Kolleginnen und Kollegen bedanken, die mich in meinem Studium unterstützt und zur Seite gestanden haben. Vor allem bei Barbara Rossegger, Sabrina Huber und Jennifer Frey für die schöne gemeinsame Zeit im Informatikstudium.

Ganz besonders möchte ich mich bei meinen Eltern Ingrid und Franz bedanken, die mir dieses Studium durch finanzielle und vor allem emotionale Unterstützung erst ermöglicht haben.

Zuletzt möchte ich mich bei Sandra, der wichtigsten Person in meinem Leben, für die großartige Unterstützung, die Geduld und die aufbauenden Worte recht herzlich bedanken.

Graz, im Juli 2013

Benedikt Neuhold

Inhaltsverzeichnis

1	Einleitung	1
1.1	Struktur der Arbeit	2
2	Learning Analytics	3
2.1	Learning-Management-Systeme	7
2.1.1	Tools für Beurteilung und Auswertung	8
2.1.2	Kritik an diesen Tools	16
2.2	Tutoring Systems	17
2.3	Learning Analytics Cycle	18
2.3.1	Lerntheorie	20
3	Schriftliche Rechenverfahren	22
3.1	Addition	23
3.1.1	Verfahren	23
3.1.2	Schwierigkeiten und Fehler	25
3.2	Subtraktion	26
3.2.1	Verfahren	27
3.2.2	Schwierigkeiten und Fehler	31
3.3	Diagnostische Tests	33
3.3.1	Addition	35
3.3.2	Subtraktion	37
3.4	Österreichischer Lehrplan	39
4	Mathematische Lernsoftware	41
4.1	Buggy	41
4.1.1	Diagnostische Modelle	42
4.1.2	Erweiterungen von BUGGY	43
4.2	Mathematik heute - Bruchrechnung	46
4.2.1	Fehlerbehandlung	48
4.3	ASSISTments	49
4.4	Zusammenfassung	52

5 Technische Umsetzung	53
5.1 Verwendete Technologien	53
5.1.1 PHP	54
5.1.2 Zend Framework	54
5.1.3 MySql	55
5.1.4 Doctrine	56
5.1.5 CSS und HTML	56
5.1.6 jQuery	57
5.2 Konzept	57
5.2.1 Addition	58
5.2.2 Subtraktion	59
5.3 Aufbau und Komponenten	60
5.4 Beispielgenerierung	63
5.4.1 Struktur und Aufbau	65
5.5 Programmablauf	65
5.5.1 Beispielzuteilung	67
5.5.2 Validierung und Sicherung der Daten	68
5.5.3 Fehleranalyse	70
5.6 Layout	73
5.7 Einstellungsmöglichkeiten	76
5.8 Statistische Auswertung	79
5.9 Datenbankmodell	80
5.9.1 Entity UserAnswer	82
5.9.2 Entities Error und ErrorUserAnswerReference	83
5.9.3 Entity Problem	84
5.10 Webservice	84
6 Evaluation	85
6.1 Vorbereitungen	85
6.2 Durchführung	85
6.3 Ergebnisse	86
6.3.1 Unterschiedliche Lernertypen	89
6.3.2 Fehlermuster	92
6.3.3 Feedback	94
7 Diskussion	96
8 Zusammenfassung und Ausblick	99
Literatur	101

Abbildungsverzeichnis

2.1	Der Prozess von Learning Analytics nach Siemens [2013]	6
2.2	LAe-R Rubrik Administration in Moodle [aus J. Dimopoulos, 2013]	10
2.3	Ablauf von LAe-R [nach I. Dimopoulos et al., 2013, S. 197]	11
2.4	Monitoring Ansicht für den Kurs „Programmierung für Alle (Java)“ [aus Anna Lea Dyckhoff et al., 2012, S. 64]	13
2.5	Analyse Ansicht für den Indikator Aktivitätsverhalten („Activity behavior“) [aus Anna Lea Dyckhoff et al., 2012, S. 64]	13
2.6	Eine Analyseansicht eines Studenten in der Grockit Plattform	16
2.7	Forschungsbereich von ITS und Cognitive Science [nach Nwana, 1990, S. 253]	18
2.8	Learning Analytics Cycle [nach Clow, 2012]	19
3.1	schriftliches Additionsverfahren	24
3.2	Beispiel der Fehlertypen bei der schriftlichen Addition [in Anlehnung an Padberg, 2005]	26
3.3	Unterschiedliche Sprechweisen bei Abziehen und Ergänzen [in Anlehnung an Padberg, 2005, S. 223]	27
3.4	Entbündelungstechnik [nach Padberg, 2005, S. 225]	29
3.5	Erweiterungsverfahren [nach Padberg, 2005, S. 225]	30
3.6	Beispiel einiger Fehlertypen bei der schriftlichen Subtraktion [in Anlehnung an Padberg, 2005]	32
3.7	Diagnostischer Test für die Addition [nach Gerster, 1982, S. 25]	36
3.8	Diagnostischer Test für die Subtraktion [nach Gerster, 1982, S. 49]	38
4.1	Zerlegung der Addition in ihre Teiloperationen [aus J. S. Brown und Burton, 1978, S. 160]	43
4.2	Aufgabenseite nach Validierung der Lösung [aus Schumann, 2013, S. 7]	47
4.3	Konfiguration der Schwierigkeitsfaktoren [aus Schumann, 2013, S. 7]	48
4.4	Programm für Lehrer um die Statistik der Schüler abzurufen [von Stiftung Universität Hildesheim, 2013]	49
4.5	Frage 19 aus dem 2003 Massachusetts Comprehensive Assessment System [aus Razzaq et al., 2005]	50
4.6	Tutoring Beispiel für Frage 19 mit Hilfen und Fehlermeldungen [aus Razzaq et al., 2005]	51

5.1	MVC Structure	55
5.2	Vereinfachte Darstellung des Zusammenspiels von Controller und Servicelayer über Interfaces	62
5.3	Grafische Oberfläche des Problem Generator	64
5.4	Ausgabe des Problem Generators	64
5.5	Programmablauf des Plusminus-Trainers im Übungsmodus	67
5.6	Vereinfachter Prozess der Beispielzuteilung des Plusminus-Trainers	68
5.7	Anzeige nach Validierung eines richtigen Ergebnisses, jedoch ohne ausgefüllte Überträge	70
5.8	Formulare des Plusminus-Trainers	73
	5.8a Loginformular für den Benutzer	73
	5.8b Überprüfungsabfrage vor dem Start	73
5.9	Die Eingabemaske für eine Subtraktion	74
5.10	Anzeige nach Validierung eines richtigen Ergebnisses	75
5.11	Anzeige nach Validierung eines falschen Ergebnisses	75
5.12	Statistik eines Benutzers bis zur Ebene Kompetenzlevel aufgeklappt	76
5.13	Benutzereinstellungen für den Plusminus-Trainer	77
5.14	Einstellungen durch einen Lehrer für eine Klasse	78
5.15	Statistische Übersicht für den Lehrer	80
5.16	Zusammenfassende Ansicht über die Statistik der einzelnen Rechenoperationen	81
5.17	Statistische Auswertung der erkannten Fehler einer Rechenoperation	81
5.18	Entities im Plusminus-Trainer und ihre Beziehungen	82
5.19	Die wichtigsten Entities des Plusminus-Trainers und ihre Attribute	83
6.1	Grafische Darstellung der Antworten	87
6.2	Anstieg des Userlevels bei der Addition	87
6.3	Anstieg des Userlevels bei der Subtraktion	88
6.4	Maximal erreichtes Kompetenzlevel bei der Addition	88
6.5	Maximal erreichtes Kompetenzlevel bei der Subtraktion	89
6.6	Kompetenzniveauverlauf eines schwächeren, demotivierten Lernertyps (ID: 93)	90
6.7	Verlauf der Kompetenzlevel eines starken Lernertyps (ID: 89)	90
6.8	Verlauf der Kompetenzlevel eines durchschnittlichen Lernertyps (ID: 86)	91
6.9	Verlauf der Kompetenzlevel eines festgefahrenen Lernertyps (ID: 71)	91
6.10	Klassenstatistik von Norbert	93
6.11	Fehlerstatistik von Norbert	93
6.12	Klassenstatistik von Jana	94
6.13	Fehlerstatistik von Jana	94

Tabellenverzeichnis

2.1	Unterschiede von EDM und LAK [nach Siemens und Baker, 2012]	5
3.1	Verfahren für die schriftliche Subtraktion [nach Padberg, 2005]	31
5.1	Einteilung der Kompetenzlevel für die Addition.	58
5.2	Einteilung der Kompetenzlevel für die Subtraktion.	60
5.3	Implementierte Fehler für die Addition	72
5.4	Implementierte Fehler für die Subtraktion	72
5.5	Gemeinsame Fehler	73
6.1	Häufige Fehler in den Rechenoperationen	92

Verzeichnis der Quellcodes

5.1	Die Methode zur Generierung aller Zahlen für ein Pattern	65
5.2	ErrorInterface, das jeder Fehler implementieren muss	71

1 Einleitung

Der Arithmetikunterricht mit seinen vier Grundrechnungsarten ist in der Grundschule ein wesentlicher Bestandteil des Mathematikunterrichts. Schriftliche Rechenverfahren stellen, trotz der Einsatzmöglichkeit von Taschenrechner und Computer, eine wichtige Kompetenz für Schülerinnen und Schüler dar. Obwohl sehr viel Zeit für das Erlernen dieser Verfahren aufgewendet wird, gibt es immer wieder Schülerinnen und Schüler die auch nach dem Abschluss der Grundschule nicht korrekt oder nur mangelhaft mit diesen Rechnungen operieren können. Dies ist vor allem dann der Fall, wenn fehlerhafte Denk- und Rechenmuster in der Grundschule nicht erkannt und gezielt bearbeitet werden. Dadurch verankern sich diese falschen Denkstrukturen und bleiben über die Grenzen der Grundschule hinaus erhalten. Eine wichtige Aufgabe des Lehrenden besteht aus diesem Grund darin, Fehlermuster zu erkennen und diese gezielt zu beheben. Als Grundlage dafür müssen Schülerfehler analysiert und dokumentiert werden, was für die Lehrpersonen nicht nur einen fachlichen sondern insbesondere einen zeitlichen Aufwand darstellt.

Der Forschungsbereich von Learning Analytics beschäftigt sich unter anderem mit der Analyse von großen Datenmengen und bietet Möglichkeiten diese Daten einfach zu interpretieren. Ein wichtiger Aspekt dieses Forschungsbereichs ist die einfache und lesbare Visualisierung der Daten. Dies ermöglicht Benutzern ohne großen Aufwand Schlussfolgerungen aus einer großen Menge von Daten zu ziehen. Es gibt bereits eine Reihe von Tools, die versuchen einen besseren Überblick über die Lernleistung der Schüler zu verschaffen. Learning Analytics verfolgt den Ansatz den Lehrenden auf gezielt auf die Stärken und Schwächen der Schüler zu sensibilisieren und aufmerksam zu machen. Informationen und Lernimpulse sollen in erster Linie nicht von der Software vermittelt werden, sondern der aktuelle Wissensstand soll analysiert und für den Lehrenden aufbereitet zur Verfügung gestellt werden. Durch diese Statistiken und Visualisierungen kann der Lehrende Schülerinnen und Schüler individuell unterstützen und konkret auf fehlerhafte Denkmuster eingehen.

Die Arbeit möchte sich, auf Grundlage dieses Forschungsbereichs, mit der Entwicklung einer Software für die schriftliche Addition und Subtraktion beschäftigen. Um ein solches Programm für Schüler der dritten bis fünften Schulstufe zu entwickeln, müssen einige pädagogische und didaktische Aspekte berücksichtigt werden. Zusätzlich muss auf die unterschiedlichen Techniken der Differenzbildung und des Übertrags bei der schriftlichen Subtraktion eingegangen werden. Der Trainer soll eine detaillierte und erweiterbare

Fehleranalyse bereitstellen und soll einen adaptiven Ansatz verfolgen. Dadurch passt die Anwendung die Beispiele genau an den Lernfortschritt des Schülers an und unterstützt den Lernprozess. Ein weiterer Kernpunkt soll in der Analyse und Aufbereitung der Daten liegen. Lehrpersonen, Schülerinnen und Schüler selbst sollen auf einfache Weise einen Überblick über ihren Lernfortschritt erhalten und auf erkannte Fehlerstrukturen aufmerksam gemacht werden.

1.1 Struktur der Arbeit

Zu Beginn der Arbeit wird in Kapitel 2 kurz ein Einblick in den Forschungsbereich von Learning Analytics gegeben. Methoden und Tools für die Beurteilung und Auswertung von Daten, die aus verschiedenen Systemen stammen, werden kurz erläutert. Zusätzlich wird versucht den Prozess von Learning Analytics mit etablierten Lerntheorien in Verbindung zu bringen.

Kapitel 3 behandelt die beiden schriftlichen Rechenverfahren, die der neue Trainer unterstützt. Didaktische und pädagogische Überlegungen, sowie die einzelnen schriftlichen Verfahren werden näher betrachtet. Aus diesen Betrachtungen wird in weiterer Folge eine Liste von häufigen Fehlern der beiden Verfahren erstellt und erläutert.

Das Kapitel 4 gibt einen kurzen Überblick über bereits vorhandene Lernsoftware im Mathematikbereich. In dem Bereich, in dem sich diese mathematische Software ansiedelt, gibt es nur sehr wenige Programme die frei zugänglich sind. Aus diesem Grund wird zusätzlich eine Software vorgestellt, die bereits um 1978 entwickelt wurde und als eine der Ersten dieser Art gilt.

Die technische Umsetzung wird in Kapitel 5 beschrieben. Angefangen vom Konzept bis hin zur fertigen Software wird versucht den Entwicklungsprozess mit all seinen Überlegungen zu beschreiben und die Verhaltensweise des Trainers näher zu bringen.

Nach der Fertigstellung der ersten Version des Trainers konnte eine Evaluation der Software mit zwei 4. Klassen einer Volksschule durchgeführt werden. Die Vorbereitungen, der Ablauf sowie die gewonnen Ergebnisse werden in Kapitel 6 zusammengefasst.

Wichtige Punkte aus allen wissenschaftlichen und pädagogischen Theorien in Zusammenhang mit der Entwicklung der mathematischen Software werden im Kapitel 7 diskutiert.

2 Learning Analytics

Vor allem durch den raschen Anstieg der Internetnutzung hat sich in den letzten Jahren die Anzahl der Lernressourcen im Internet vervielfacht. Seien es Plattformen, die Lernunterlagen bereitstellen (wie MERLOT ¹), konkrete Einrichtungen wie die Khan Academy ², OpenCourseware Systeme oder andere Open Educational Resources (OER). Lernen passiert immer häufiger mithilfe des Internets [vgl. M. Ebner und S. Schön, 2012; Sicilia et al., 2013]. Aus diesem Grund stehen auch immer größere lernspezifische Datenmengen von Benutzern zur Verfügung. Für Bildungseinrichtungen und Lehrende ist es eine große Chance, diese Daten zu nutzen und aufgrund dieser Daten ihre Entscheidungen zu treffen. Die Forschung kann bereits belegen, dass diese Entscheidungen, die aufgrund von solchen Daten getroffen werden, die Produktivität und den Erfolg einer Firma erhöhen [vgl. Brynjolfsson et al., 2011]. Nach Siemens und Long [2011] gehen Bildungseinrichtungen sehr ineffizient mit diesen gewonnenen Daten um. Analysen werden oftmals nur jährlich durchgeführt und dies verhindert, dass Lehrende aktiv in den Lernprozess eingreifen und Aktionen zur Verbesserung während des Lehr- und Lernprozesses gesetzt werden können.

Durch die Nutzung der Internettechnologie für das Lernen wird die erfasste Datenmenge über Lernende immer größer. Jeder Tweet, jeder Facebookeintrag, jeder Webseitenbesuch kann bereits einen digitalen Fußabdruck hinterlassen. Diese, von Lernenden produzierten Daten, können zusammengeführt werden und geben einen Aufschluss darüber was gerade in ihren Lernprozessen geschieht und wie die Lehrperson sie bestmöglich unterstützen kann. Die Analyse dieser Lerndaten kann zu einer Früherkennung von Problemen in Lernprozessen von Schülerinnen und Schülern führen. Lehrenden wird zusätzlich ermöglicht, aktiv in den Lernprozess einzugreifen und gezielt Probleme zu beheben [vgl. Siemens und Long, 2011].

Durch das steigende Interesse an Daten und deren Analyse in Bildung, Lehre und Lernen wurde auch in der Forschung das Augenmerk auf die Entwicklung von Methoden und Techniken zur Analyse dieser Daten gelegt. Zwei Forschungsbereiche befassen sich sehr intensiv mit diesem Thema - zum einen Learning Analytics and Knowledge (LAK) und zum anderen Educational Data Mining (EDM) [vgl. Siemens und Baker, 2012]. Beide Forschungsbereiche befassen sich mit der Thematik von BigData im Bildungsbereich und haben einen großen überlappenden Bereich in ihrer Forschung. EDM wird von der

¹<http://www.merlot.org> [Zugriff am 09.07.2013]

²<https://www.khanacademy.org> [Zugriff am 09.07.2013]

International Educational Data Mining Society folgendermaßen definiert:

Educational Data Mining is an emerging discipline, concerned with developing methods for exploring the unique types of data that come from educational settings, and using those methods to better understand students, and the settings which they learn in. [International Educational Data Mining Society, 2013]

LAK ist nach Siemens und Baker [2012] die Messung, Speicherung, Analyse und Visualisierung von Lernenden und deren Kontext. Ziel von LAK ist es, den Lernprozess zu verstehen und ihn und das Umfeld, in dem er passiert, zu verbessern. Dabei rückt vor allem die Lehrperson stark in den Vordergrund. Beide Bereiche haben einen datenintensiven Ansatz und werden in der nächsten Zeit den Bildungssektor beeinflussen. Daten zu analysieren und daraus Hilfe für die Planung, Interventionen und Entscheidungen zu erhalten, ist ein neuer Ansatz. Durch diesen Ansatz werden auch Technologien und Methoden für die Verarbeitung dieser Daten benötigt. Beide, EDM und LAK, haben das Ziel, die Analyse von großen Datenmengen im Bildungsbereich zu verbessern und die Forschung sowie die praktische Verwendung voran zu treiben.

Obwohl sich die beiden Bereiche sehr stark überlappen, weisen sie doch feine Unterschiede in ihrer Ausrichtung auf. EDM fokussiert auf automatische Erkennung von Patterns, wohingegen LAK versucht, den Entscheidungsprozess von Stakeholdern zu unterstützen. Einen weiteren Unterschied kann man bei der Adaption und Personalisierung erkennen. Da EDM großen Wert auf automatisierte Erkennung legt, werden die erstellten Modelle meist von einem Computersystem (z.B.: intelligentes Computersystem) genutzt, um Entscheidungen zu treffen. Im Gegensatz dazu werden LAK Modelle eher dazu verwendet, um den Lehrenden und den Lernenden zu informieren. Ein wichtiger Unterschied liegt auch in der Erforschung von Systemen. EDM versucht meist das System in Komponenten zu unterteilen und diese gesondert zu analysieren. LAK hingegen versucht den Prozess als ein Ganzes in seiner gesamten Komplexität zu verstehen. Tabelle 2.1 zeigt noch einmal die wichtigsten Unterschiede der beiden Bereiche zusammengefasst.

Gerade durch die kleinen Unterschiede sehen Siemens und Baker [2012] einen großen Vorteil für den Forschungsbereich. EDM und LAK fokussieren auf verschiedene Dinge und können so immer wieder voneinander lernen. Die freundschaftliche Rivalität ist zudem ein Ansporn für beide Communities. Wichtig ist, dass die Kommunikation zwischen den beiden Communities forciert wird, um so die bestmöglichen Leistungen für den Bildungssektor und die Wissenschaft des Lernens zu erzielen. Nach Ryan Baker sollten LAK und EDM beste Freunde werden, um so diesen Austausch von Wissen zu gewährleisten [vgl. Baker et al., 2012].

	LAK	EDM
Forschung	Unterstützung von menschlichen Entscheidungen, automatisierte Erkennung wird als Tool verwendet um dieses Ziel zu erreichen	Fokus auf automatisierter Erkennung, Unterstützung von menschlichen Entscheidungen wird als Tool verwendet um dieses Ziel zu erreichen
Reduktion & Holismus	Verständnis des ganzen komplexen Systems	Reduzierung auf Komponenten, Analyse der Komponenten und deren Beziehungen
Herkunft	LAK stammt eher aus den Bereichen semantisches Web, „intelligent curriculum“, Ausgangsvorhersage und semantische Interventionen	EDM hat seine Wurzeln eher im Bereich von Bildungssoftware und Studentenmodellierung mit einer starken Tendenz zu Vorhersagen zu Kursausgängen
Adaption & Personalisierung	Fokussiert auf Informieren und Unterstützen der Lehrer und Lerner	Fokus auf automatischer Adaption
Techniken & Methoden	Soziale Netzwerk-Analyse, Stimmungsanalysen, Diskursanalysen, Lernerfolgsvorhersage, ..	Klassifizierung, Clustering, Bayesian Modellierung, „relationship mining“, ..

Tabelle 2.1: Unterschiede von EDM und LAK [nach Siemens und Baker, 2012]

Siemens [2013] definiert Learning Analytics (LA) folgendermaßen:

Learning analytics is the use of intelligent data, learner-produced data, and analysis models to discover information and social connections, and to predict and advise on learning.

Seiner Ansicht nach verwendet Learning Analytics die Daten, die von Lernenden durch ihre Arbeit produziert werden. Mit Hilfe von Analysemodellen aus diesem Bereich wird dann versucht, Informationen und soziale Verbindungen zu ermitteln, um Vorhersagen und Hilfen für das Lernen zu erstellen.

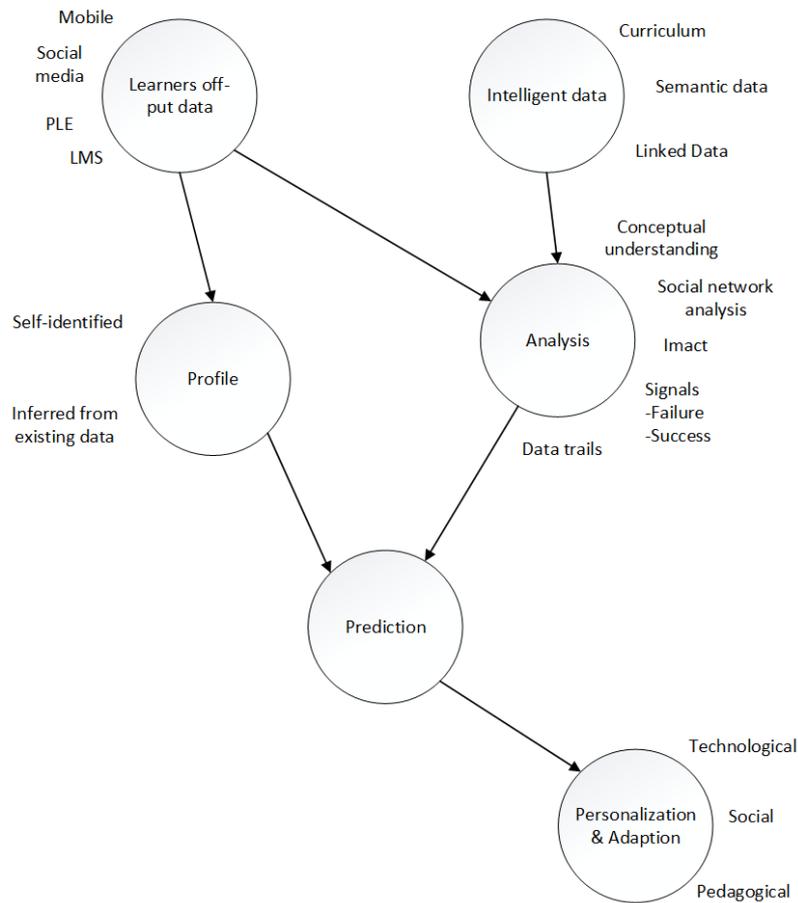


Abbildung 2.1: Der Prozess von Learning Analytics nach Siemens [2013]

In Abbildung 2.1 können wir einen möglichen Ansatz für den konkreten Ablauf von Learning Analytics sehen. Nach diesem Ansatz erzeugen Lernende durch ihren täglichen Lernprozess konstant Daten. Sei es ein Statusupdate in Facebook, ein neuer Tweet oder die Arbeit mit Lernmanagement-Systemen. Diese Daten werden in eine Datenbank zusammengeführt und warten dort auf ihre Analyse. Alle diese gewonnenen Inhalte werden zusätzlich um ein Profil des jeweiligen Benutzers bzw. der jeweiligen Benutzerin ergänzt. Dieses wird aus verschiedenen Quellen, die über das ganze Internet verteilt sind, zusammengesetzt und enthält Daten aus Netzwerken wie LinkedIn, Facebook, Twitter, Google und noch viele mehr. Durch semantische Prozesse werden aus diesen gesamten Inhalten „intelligenten Daten“. Diese werden der Analyse übergeben und in Bezug auf das aktuelle Lernziel ausgewertet. Die Auswertung versucht Prognosen für Lernende zu erstellen, mögliche Interventionen zu definieren, das Lernumfeld zu personalisieren und/oder zu adaptieren. Der effektive Einsatz von Learning Analytics an Schulen und Universitäten kann diesen helfen, dass sie individuelle Lernschwierigkeiten erkennen und

auf diese reagieren können [vgl. Siemens, 2013].

Baker et al. [2012] verstehen unter Learning Analytics das Sammeln von Spuren, welche von Lernern hinterlassen werden. Mit Hilfe dieser Spuren versucht Learning Analytics das Lernen selbst zu verbessern. Dies geschieht mit Hilfe einfacher visueller Repräsentationen und Feedback für Lehrende und Lernende. Ihrer Meinung nach ist es eine große Gefahr zu viele und unwichtige Informationen zu präsentieren, aus denen weder Lernende noch Lehrende wichtige Erkenntnisse ablesen können .

Die 1st International Conference Learning Analytics & Knowledge beschreibt Learning Analytics als

the measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimising learning and the environments in which it occurs. [1st International Conference Learning Analytics & Knowledge, 2013]

Diese Definition deckt einen weiten Bereich ab und lässt erkennen, wie groß das Forschungsfeld rund um Learning Analytics ist.

Bei L. Johnson et al. [2013] ist zu lesen, dass Learning Analytics speziell dazu verwendet werden soll, den Lehrenden die Möglichkeit zu geben, die Fortschritte der Lernenden zu verfolgen und individuelle Informationen darüber abzurufen. Es gibt bereits einige Learning Management Systeme (LMS), die dies ermöglichen. Ein solches Tool wird zum Beispiel an der Austin Peay State University in Tennessee eingesetzt. Der „Degree Compass“ versucht mit den Daten der Studierenden und analytischen Prognosetechniken die bestmöglichen Kurse für einen erfolgreichen Abschluss zu finden [vgl. L. Johnson et al., 2013, S. 28-30].

2.1 Learning-Management-Systeme

Die Analyse von Lernprozessen bei Lernenden ist eine wichtige Aufgabe für den ganzen Lehrkörper. Sehr viele Bildungseinrichtungen haben daher in der letzten Zeit begonnen, auf neue Technologien für Lehr- und Lernumgebungen zu setzen. Oftmals werden für diesen Zweck sogenannte „Learning-Management-Systeme“ (LMS) verwendet. Die bekanntesten dieser Systeme sind zur Zeit Moodle³ und Blackboard⁴. Diese Tools unterstützen Lehrende sowie Administratoren mit vielen Möglichkeiten. Angefangen bei der Bereitstellung von Lernressourcen, bis hin zu administrativen Tätigkeiten innerhalb des Kurses. Sehr viele dieser Systeme haben Kommunikationstools wie Chats, Foren und Emails integriert und sorgen so für eine Vernetzung von Studenten untereinander sowie auch von Lehrenden zu Studenten. Durch den Einsatz dieser Systeme wird eine

³<https://moodle.org> [Zugriff am 09.07.2013]

⁴<http://www.blackboard.com> [Zugriff am 09.07.2013]

große Menge von Daten generiert. Ziel der Bildungseinrichtungen ist es, diese Daten zu analysieren, um die Effizienz des Lernen und Lehrens zu verbessern. So könnte zum Beispiel das System den Lehrenden über Gemeinsamkeiten bei den Suchbegriffen der Studierenden informieren. Dies könnte er als Indikator sehen, dass er für diesen Bereich weitere Lernressourcen zur Verfügung stellen muss [vgl. Kalz et al., 2011; Niemann et al., 2011].

Auch Lockyer und Dawson [2011] sehen in den Learning-Management-Systemen eine Möglichkeit, alle Aspekte von Online-Lernen der Studierenden zu erfassen. Sie speichern Kommunikationen zwischen Lernenden sowie zwischen Lehrenden und Lernenden, halten fest, welche Lernressourcen aufgerufen werden und welche Aufgaben sie abschließen. Diese Daten und die Analysetechniken, die im Forschungsbereich von Learning Analytics entwickelt werden, ermöglichen es Bildungseinrichtungen und Lehrenden, den Lernprozess zu verstehen und aktiv in diesen einzugreifen. Durch diese gewonnenen Erkenntnisse können personalisierte Lernprozesse geschaffen werden.

2.1.1 Tools für Beurteilung und Auswertung

Lehrende aus allen Schulstufen nutzen in der heutigen Zeit Learning-Management-Systeme, um so den traditionellen Unterricht mit multimedialen Komponenten und neuen Technologien zu verbessern und eine umfangreichere Lernumgebung zu schaffen [vgl. I. Dimopoulos et al., 2013, S. 195].

Nach Dillenbourg et al. [2009] soll Lernen nicht als ein isolierter Prozess betrachtet werden. Lernen soll einen hohen Anteil an Interaktion zwischen Studierenden selbst, zwischen Studierenden und Lehrenden und zwischen Studierenden und Lernressourcen enthalten. Diese Interaktionen als Lehrer zu dokumentieren und zu bewerten stellt jedoch eine große Herausforderung da.

Methoden zur Beurteilung, die sich im Klassenzimmer über Jahre bewährt haben, sind für den Einsatz auf diesem Gebiet nicht zu gebrauchen [vgl. Quellmalz und Kozma, 2003, S. 389 - 390]. Die Lehrperson steht zunehmend vor der Herausforderung, mit den großen Datenmengen, die durch Technologieunterstützung gewonnen werden, umzugehen. Durch die Forschung rund um Learning Analytics und Educational Data Mining gibt es viele Tools und Methoden, diese Daten zu analysieren und zu interpretieren. Jedoch sind diese Tools für den Lehrenden oftmals zu komplex, um mit ihnen effektiv arbeiten zu können [vgl. Romero und Ventura, 2010, S. 611-612].

Dadurch wird die Forderung immer lauter, dass Learning Analytics Tools einfacher und direkt in die Lernumgebung integriert werden sollen. Es sollte zudem die Möglichkeit einer einfachen Konfiguration bestehen. Damit soll es der Benutzerin bzw. dem Benutzer ermöglicht werden, Daten einfach zu analysieren und zu interpretieren. Die Resultate dieser Analyse sollen in einem klaren und verständlichen Format visualisiert werden. Die

Anzeige der Ergebnisse soll auch für Laien zu interpretieren sein [vgl. Anna Lea Dyckhoff et al., 2012, S. 60].

Für die Lehrenden selbst sind solche Tools sehr wichtig, da Lehrpersonen durch sie den Lehr- und Lernprozess und die Lehrmethoden einfacher evaluieren und reflektieren können [vgl. A. L. Dyckhoff et al., 2013, S. 220].

2.1.1.1 LAe-R

LAe-R steht für Learning Analytics Enriched Rubric und wurde als Moodle Modul entwickelt. Dieses Modul erlaubt Lehrenden, dass sie so genannte „enriched rubrics“ erstellen. In diesen Rubriken kann man Kriterien und Benotungsskalen definieren. Diese Merkmale werden direkt mit den Ergebnissen der Analyse der Moodle Daten verknüpft. Diese Daten können zum Beispiel die Anzahl der Einträge im Forum, die Zeit, in der man sich mit Lernunterlagen beschäftigt hat oder Benotungen bei Moodletests sein. LAe-R kann dann mit Hilfe von Data-Mining-Methoden automatisch einen Wert für jedes Kriterium berechnen und anhand der Benotungsskala eine Note vergeben. Das Endergebnis wird durch das Summieren aller Kriterien berechnet [vgl. I. Dimopoulos et al., 2013, S. 195].

Die aktuelle Version von LAe-R⁵ ist zur Zeit für Moodle 2.2 und höher erhältlich und erweitert das bereits existierende Standard Rubric Module. Dem Lehrenden ist es nun möglich, weitere Rubriken zu erstellen, in denen er Bewertungskriterien mit der dazugehörigen Beschreibung sowie der Bewertungsskala eingeben kann (siehe Abbildung 2.2) [vgl. I. Dimopoulos et al., 2013, S. 196]. LAe-R unterstützt nach I. Dimopoulos et al. [2013, S. 196] folgende Learning Analytics Kriterien:

- **Zusammenarbeit** (collaboration)
Die Zusammenarbeit kann durch Kriterien wie Beiträge und Antworten in Foren, Chat-Nachrichten oder Anzahl von Anhängen in Forenbeiträgen bestimmt werden.
- **Bewertung von Aufgaben** (grades to assignments)
Hier kann die Benotung von diversen Aufgaben in Moodle herangezogen werden, um diese Kategorie zu definieren.
- **Nutzen von Lernressourcen** (study of learning resources)
Zum Bestimmen dieser Eigenschaft können die Anzahl der Ansichten bestimmter Lernressourcen gemessen werden.

LAe-R ermöglicht es Lehrenden, nicht nur das Gesamtergebnis für jeden Studierenden zu berechnen, sondern zusätzlich das Ergebnis jedes Kriteriums zum durchschnittlichen Ergebnis der Klasse bzw. des Kurses zu vergleichen [vgl. I. Dimopoulos et al., 2013, S. 196].

⁵http://docs.moodle.org/23/en/Learning_Analytics_Enriched_Rubric [Zugriff am 09.07.2013]

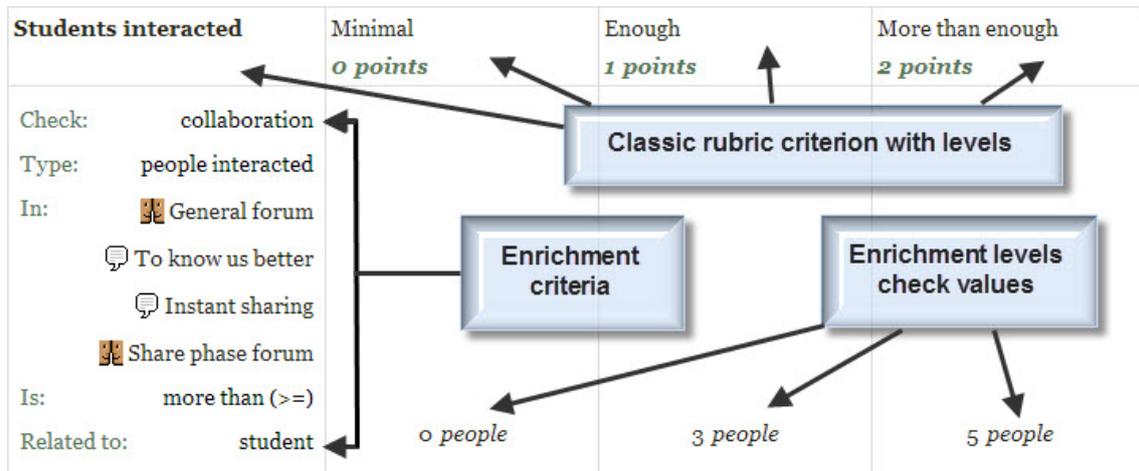


Abbildung 2.2: LAe-R Rubrik Administration in Moodle [aus J. Dimopoulos, 2013]

Die Auswertung wird automatisch durchgeführt und berechnet für jedes Kriterium die dazugehörigen Punkte. In Abbildung 2.3 sehen wir den Ablauf von LAe-R. Für jedes Kriterium wird mit Hilfe von Data-Mining und Learning-Analytics-Methoden aus den Daten der Moodle Datenbank ein Wert (Benchmark) berechnet. Dieser Wert wird mit den vorhandenen Kriteriumleveln verglichen. Hat eine Studentin oder ein Student zum Beispiel in Foren mit vier Personen interagiert, so wird in der Rubrik in Abbildung 2.2 eine Benchmark von vier berechnet. Diese Benchmark wird einem Level zugeordnet. Im diesem Fall würde der Student einen Punkt erhalten, da er in Level 2 „Enough“ landet [vgl. I. Dimopoulos et al., 2013, S. 196-198].

Das Moodle Modul wurde durch eine Studie auf zwei Gesichtspunkte evaluiert. Zum einen wurde der Sourcecode auf Einhaltung des Programmierstandards von Moodle evaluiert. Zum anderen wurde die Funktionalität und Usability von zwölf erfahrenen Lehrpersonen im primären und sekundären Bildungsbereich getestet. Die Testpersonen wiesen alle gute Kenntnisse mit Moodle auf. Das Feedback der Tester war sehr positiv. So meldeten 91,7 %, dass sie gerne mit diesem Tool arbeiten und es sehr nützlich finden. Rund 75 % waren mit dem Userinterface zufrieden und 83,3 % fanden die Handhabung sehr einfach und intuitiv [vgl. I. Dimopoulos et al., 2013].

2.1.1.2 eLAT

eLAT (exploratory Learning Analytics Toolkit) ist ein Tool bei dem in der Entwicklung speziell auf die Bedürfnisse von Lehrenden eingegangen wurde. Entwickelt wurde dieses Programm, das hauptsächlich in .NET geschrieben ist, an der RWTH Aachen⁶. Vorausgegangen ist dem Entwicklungsprozess eine gründliche Bedarfsanalyse. Diese

⁶<http://www.rwth-aachen.de> [Zugriff am 09.07.2013]

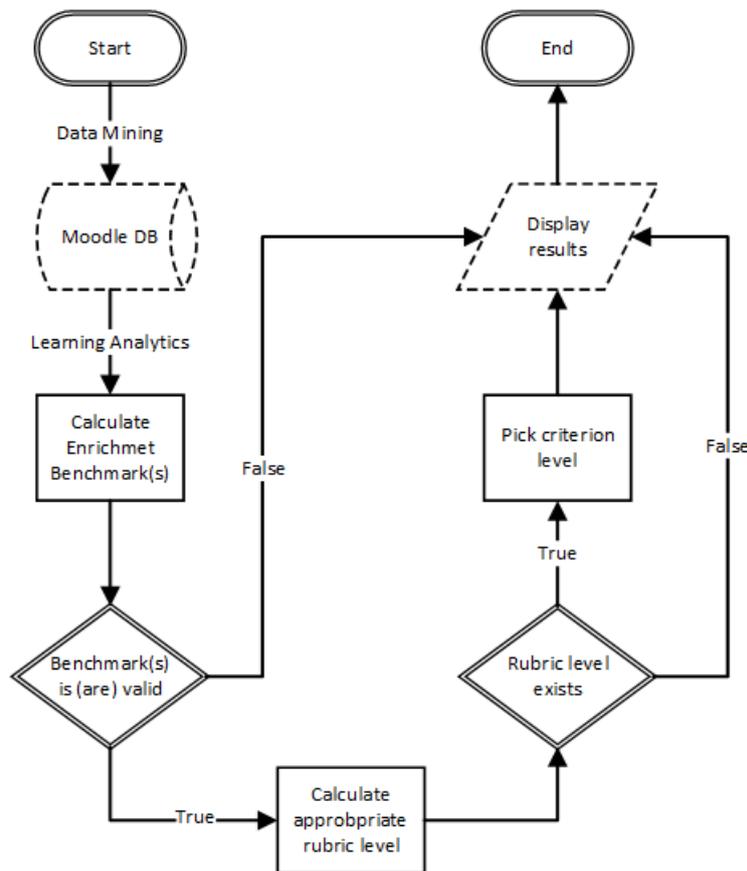


Abbildung 2.3: Ablauf von LAe-R [nach I. Dimopoulos et al., 2013, S. 197]

beschäftigte sich mit folgenden Punkten [vgl. Anna Lea Dyckhoff et al., 2012, S. 61-62]:

- **Bedienbarkeit und Brauchbarkeit** (usability and usefulness):
Jeder Kurs an der Universität und jede Schulstunde ist anders. Dies hängt von den Lehrenden und den Studierenden ab. Es kommen unterschiedlichste Lehr- und Lernstrategien zum Einsatz. Das Vorwissen auf dem Gebiet von Learning Analytics ist bei jedem Lehrenden unterschiedlich. Deshalb braucht es auf der einen Seite ein Learning Analytics Tool, das einfach zu verstehen und leicht zu bedienen ist, auf der anderen Seite ein Tool, das mächtig genug ist, auch spezifische Fragestellungen zu modellieren und tiefgreifende Analysen durchzuführen.
- **Kompatibilität, Erweiterbarkeit und Wiederverwendbarkeit** (interoperability, extensibility and reusability):
Die meisten Learning Analytics Tools wurden für eine spezifische virtuelle Lernumgebung entwickelt und können nicht einfach Daten von anderen Lernumgebungen verwenden. In Zukunft wird es sicher noch weitere Lernumgebungen geben, von denen man nützliche Daten zur Analyse erhalten

kann. eLAT soll deshalb leicht mit Lernumgebungen kommunizieren und deren Daten verwenden können.

- **Echtzeit-Verarbeitung** (real-time operation):
Probleme bei Studierenden können zu jedem Zeitpunkt des Semesters auftreten. Diese Probleme sollen vom System und vom Lehrenden schnell erkannt und gelöst werden. Daher soll ein Learning Analytics Tool jederzeit, und nicht nur am Ende des Semesters, aktuelle Daten und Analysen zur Verfügung stellen. Möglichkeiten zur Filterung und der Visualisierung sollen ebenfalls gegeben sein und in Echtzeit durchgeführt werden.
- **Datenschutz** (Data Privacy):
Persönliche Daten sollen jederzeit vor Missbrauch geschützt sein. Daher soll es die Möglichkeit geben, Daten anonymisiert oder nur in Subgruppen geordnet an das System zu übergeben.

Nach dieser Analysephase entstand die Software in einem iterativen und inkrementellen Prozess in zwei Phasen. In Phase 1 wurde die Implementierung und das Testen des Frameworks durchgeführt. Die Erstellung des Design und die Evaluation der grafischen Oberfläche wurden in Phase 2 berücksichtigt [vgl. Anna Lea Dyckhoff et al., 2012, S. 62-63].

In der ersten Phase wurden verschiedene Architekturansätze evaluiert, um Learning Analytics für verschiedene Lernumgebungen zu unterstützen. Im Wintersemester 2010/2011 konnten vier Kurse ausgewählt werden, um die neue Software zu testen. Über einen Zeitraum von drei Monaten speicherte man alle Daten der Kursteilnehmer. Durch die Nutzung von „echten Daten“ konnte man einerseits bessere Erkenntnisse über die bereits implementierten Indikatoren gewinnen und andererseits auch die Lehrenden aktiv in den Entwicklungsprozess mit einbinden. Die Phase 2 wurde parallel zur Phase 1 gestartet. Diese Phase wurde in drei Iterationen unterteilt. In der ersten Iteration wurde der Inhalt definiert und gesammelt. Die nächste Iteration beschäftigte sich hauptsächlich mit dem Layout und der Datenpräsentation der grafischen Oberfläche und die letzte fokussierte auf die Interaktion der Benutzer mit der Software [vgl. Anna Lea Dyckhoff et al., 2012, S. 62-63].

Das Ergebnis dieses Softwareentwicklungsprozesses war ein Learning Analytics Tool mit einer einem Launchpad ähnlichen grafische Oberfläche. Ein Launchpad ist nach Few [2006] sehr ähnlich einem Dashboard, allerdings mit mehr Möglichkeiten für weitere Analysen und virtuelle Darstellungen. Der Einstieg in das Programm erfolgt über eine Monitoring-Ansicht (siehe Abbildung 2.4). Diese zeigt sämtliche Indikatoren auf einen Blick und ermöglicht der Benutzerin bzw. dem Benutzer die Anzeige durch Klicken auf die einzelnen Indikatoren zu konfigurieren. Die Analyse-Ansicht arbeitet mit Widgets, welche die einzelnen Daten der Indikatoren darstellen. Aus diesem Grund ist es für die Benutzerin bzw. dem Benutzer möglich, die für sie bzw. ihn wichtigsten Indikatoren

auf einen Blick zu sehen. Dies geschieht einfach durch das Anzeigen und Ausblenden der einzelnen Widgets. Die Analyse-Ansicht des Indikators für Aktivitätsverhalten ist in Abbildung 2.5 zu sehen. In den Analyse-Ansichten kann die Benutzerin bzw. der Benutzer auf gewisse Daten filtern und Ansichten verändern [vgl. Anna Lea Dyckhoff et al., 2012, S. 63-65].

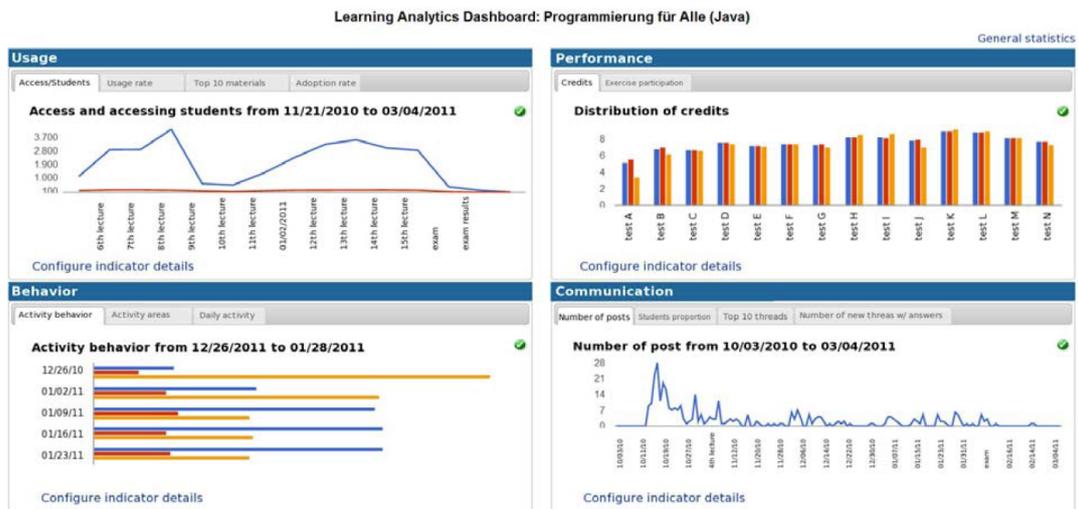


Abbildung 2.4: Monitoring Ansicht für den Kurs „Programmierung für Alle (Java)“ [aus Anna Lea Dyckhoff et al., 2012, S. 64]

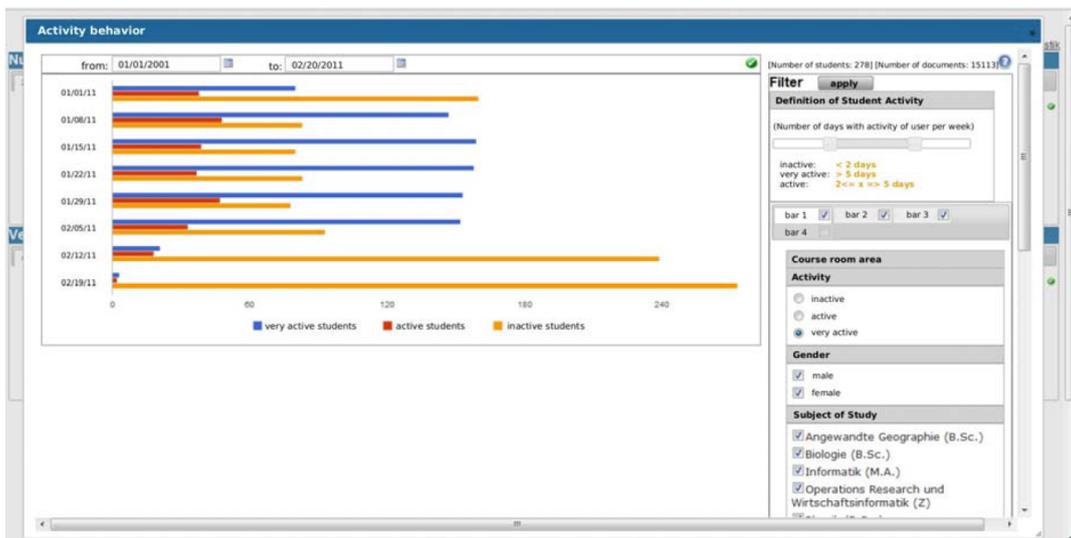


Abbildung 2.5: Analyse Ansicht für den Indikator Aktivitätsverhalten („Activity behavior“) [aus Anna Lea Dyckhoff et al., 2012, S. 64]

Einige wichtige Indikatoren der Software nach Anna Lea Dyckhoff et al. [2012] sind:

- **Aktivitätsverhalten** (activity behavior):
In Abbildung 2.5 sehen wir diesen Indikator. Hierbei werden die Studierenden in drei Gruppen aufgeteilt - „sehr aktive Studierende“ (blaue Balken), „aktive Studierende“ (rote Balken) und „inaktive Studierende“ (oranger Balken). In diesem Fall wurde als Zeitspanne eine Woche gewählt. Die Zuweisung zu diesen Gruppen erfolgt durch die Berechnung der durchschnittlichen Tage in einer Woche, wo der Student aktiv ist (z.B.: Logins in das System, etc.). Die Zeitspanne sowie die Definitionen der verschiedenen Gruppen kann vom Lehrenden einfach verändert werden.
- **Zugriff und zugreifende Studierende** (access and accessing students):
Dieser Indikator enthält die Anzahl der eingeloggtten Studierenden und deren Klicks über einen Zeitraum. Dieser kann von der Lehrperson bestimmt werden kann.
- **Aktivitätsbereiche** (activity areas):
Durch diesen Indikator ist es Lehrenden möglich, herauszufinden, welche und wann Lernumgebungen verwendet werden.
- **Top 10 Ressourcen** (top 10 resources):
Dieser Indikator listet die am meist verwendeten Lernressourcen und ihre totale Nutzungsanzahl auf.
- **Verwendung des Forums** (forum usage):
Hierbei werden Foreneinträge, geteilt in Beiträge und Antworten, über einen bestimmten Zeitraum gezählt und dargestellt.
- **Einführungszeit** (adoption rate):
Dieser Indikator speichert die Zeitspanne, die vergeht, wenn der Lehrende die Lernressource hochlädt bis, der Studierende sie verwendet.

2.1.1.3 Grockit

In ihrem Paper „Envolving a learning analytics platform“ beschreiben Bader-Natal und Lotze [2011] die Entwicklung einer Learning-Analytics-Komponente für die webbasierte Lernplattform Grockit⁷. Grockit ist eine kollaborative Lernumgebung mit starkem Interesse an der Datenanalyse, um die Effektivität der Plattform zu verstehen und verbessern zu können. Bader-Natal und Lotze [2011] fassen die Herausforderungen und Probleme bei der Entwicklung einer solchen Komponente sehr gut zusammen:

The challenges in creating an analytics platform for a web application – including learner analytics platforms – is in balancing the power and flexibility of an approach with the efficiency and scalability limitations of that approach. As the size of a dataset grows, this balance often shifts.

⁷<https://grockit.com> [Zugriff am 09.07.2013]

Mit diesen Voraussetzungen im Hinterkopf wurde ein Konzept für die Erstellung der Komponente erarbeitet. Das Ergebnis war der Aufbau einer Standard Pipeline bestehend aus Datensammlung, Datenselektierung, Datenanalyse, Visualisierung und letztendlich Verteilung. Die Datensammlung muss im System so integriert sein, dass Daten über das Benutzerverhalten und deren erledigte Aufgaben und Ziele gesammelt werden. Dies wurde direkt in die Webplattform von Grockit implementiert. Durch SQL Queries ist es möglich, Daten zusammenzuführen, die für eine Beantwortung einer speziellen Frage benötigt werden (Datenselektierung). Ein wichtiger Teil der Komponente ist die Analysephase. Hierbei werden die zuvor gefilterten Daten verwendet, um auf jede spezielle Frage eine Antwort zu finden. Hierzu verwendet die Komponente das Statistikpaket „R“⁸. Ebenfalls mit diesem Statistikpaket wird in der Visualisierungsphase gearbeitet. Hierbei ist es wichtig, dass eine einfache Visualisierung der Daten gefunden wird. Die Verteilung dieser Analysen und Visualisierungen wurde zu Beginn über den Emailversand ermöglicht. Mit zunehmender Benutzeranzahl und damit steigenden Daten wurde diese Methode jedoch durch ein webbasierendes Reportingsystem ersetzt. In diesem System werden alle generierten Reports und die verwendeten Daten archiviert und sind so für eine spätere Ansicht oder neue Analysen verfügbar [vgl. Bader-Natal und Lotze, 2011, S. 180-183].

Durch diesen modularen Aufbau ist es auf einfache Weise möglich, die Plattform um neue Analysen und Visualisierungen zu erweitern. Es muss lediglich eine SQL-Datei für das Abrufen der Daten und eine R-Datei für die Analyse und das Anzeigen der Daten erstellt werden. Das System übernimmt das Setup der benötigten Komponenten, das Sammeln der Daten und die Verteilung [vgl. Bader-Natal und Lotze, 2011, S. 181].

Immer mehr Studierende und Lehrende interessieren sich für die Daten, die sie beim Lernen und Lehren produzieren. Aus diesem Grund wurden viele Analyseergebnisse und Statistiken auch den Benutzern von Grockit zugänglich gemacht. Der Lehrer einer Klasse und jeder Student kann seinen aktuellen Fortschritt direkt in der Grockit Plattform abrufen (siehe Abbildung 2.6) [vgl. Bader-Natal und Lotze, 2011, S. 184-185].

⁸<http://www.r-project.org> [Zugriff am 09.07.2013]



Abbildung 2.6: Eine Analyseansicht eines Studenten in der Grockit Plattform

2.1.2 Kritik an diesen Tools

An dieser Stelle ist stets zu bedenken, dass all diese automatisierten Tools einen sehr großen Nachteil besitzen, denn sie können in den meisten Fällen nur quantitative Daten messen. Dies ist für die Beurteilung eines Lernerfolgs und die Bewertung von Schülerinnen und Schülern jedoch zu wenig. Daher sollten solche Auswertungen in erster Linie als Feedback für den Lehrer bzw. die Lehrerin gesehen werden und nicht in die Leistungsbeurteilung der Schülerinnen und Schüler einfließen. Lernen ist ein kognitiver Prozess und jeder Lernprozess wird von Lernenden in unterschiedlicher Weise durchlaufen [vgl. Holzinger, 2001]. Aus diesem Grund ist es nicht sinnvoll anhand dieser quantitativen Daten eine Leistungsbeurteilung zu erstellen. Lernende, die ihren Lernprozess mit Hilfe von Kommunikation zu Mitschülern und Mitschülerinnen bzw. zu Lehrenden durchlaufen, würden bei einer Bewertung der Anzahl der Foreneinträge zu einer großen Wahrscheinlichkeit eine bessere Beurteilung erhalten als jene Lernenden, die den Lernprozess ohne oder nur mit wenig Kommunikation nach Außen durchlaufen. Dies stellt jedoch in keiner Weise eine Aussage über die Leistung eines Schülers oder einer Schülerin dar.

Zusätzlich kann das Wissen um eine Bewertung, die Motivation der Schülerinnen und Schüler, die nach Holzinger [2001] von den neuen Medien selbst ausgeht, verringern.

2.2 Tutoring Systems

Nach Nwana [1990, S. 251-252] stammt die Idee künstliche Intelligenz und Bildung zusammenzubringen bereits aus den 1970iger Jahren. Einige Jahre später teilte sich dieses Feld in zwei Gruppen auf. Die kleinere der beiden Gruppen spezialisierte sich auf entdeckende Lernumgebungen bzw. „Learning by doing“. Das berühmteste Programm in diesem Feld ist wahrscheinlich die LOGO-Sprache. Entwickelt wurde diese Programmiersprache, um Studierenden die Welt der Geometrie näher zu bringen. Die zweite und größere der beiden Gruppen fokussierte die Entwicklung und Forschung auf Intelligent Tutoring Systems (ITS). In diesem Fall soll der Computer als Lehrer und Vermittler fungieren und dem Schüler so gezielte Informationen vermitteln. Diese Vorgehensweise ist dem klassischen Ansatz von der Lehre in den meisten Klassenräumen sehr ähnlich. Nwana [1990, S. 252] beschreibt ITS folgendermaßen:

Intelligent tutoring systems (ITSs) are computer programs that are designed to incorporate techniques from the AI community in order to provide tutors which know what they teach, who they teach and how to teach it.

Intelligent Tutoring Systems sollen so konzipiert sein, dass sie die menschlichen Lehrperson vollständig ersetzen können. Um solche Systeme zu entwickeln, ist die Zusammenarbeit von Computerwissenschaft, kognitiver Psychologie und Bildungsforschung essentiell. Dieses Feld der Forschung fasst man unter dem Begriff „Cognitive science“ zusammen (siehe Abbildung 2.7) [vgl. Nwana, 1990, S. 251-253].

Intelligente Tutoren-Systeme haben den Vorteil, dass sie, im Gegensatz zur Lehrperson in der Klasse, eine persönliche und individuelle Lernerfahrung bieten können. Nwana [1990, S. 253-254] schreibt, dass diese individuelle Unterstützung und die Lernumgebung, die auf die Bedürfnisse der Schülerinnen und Schüler zugeschnitten ist, eine sehr effektive Möglichkeit darstellt Informationen zu vermitteln. In einem Vergleich zwischen dem Unterricht in einer Klasse mit 30 Studenten und Studentinnen, die von Tutoren betreut wurden, fand Bloom [1984] heraus, dass 98% der Personen, die von einem Tutor betreut wurden, bessere Ergebnisse erzielten als der durchschnittliche Studierende in der Klasse. Dies war der Fall, obwohl jede Testperson die gleiche Anzahl an Lernzeit investierte.

ITS können hier eine Brücke schlagen und das konventionelle Unterrichten in Klassen mit persönlicher und individueller Unterstützung verbinden. Durch das schnelle Feedback des Systems wird der Lernerfolg des Studenten zusätzlich unterstützt [vgl. Nwana, 1990, S. 253-254].

In einer Studie wurde ein solches Intelligent Tutoring System genauer untersucht. Hauptaugenmerk wurde dabei auf den Zusammenhang von Faktoren wie Langeweile, Konfusion, Frustration und Konzentration beim Arbeiten mit dem System und dem Erfolg beim Abschlusstest am Ende des Schuljahres gelegt. Für die Analyse wurden automatische Detektoren für diese Zustände definiert und anhand von Logdaten des

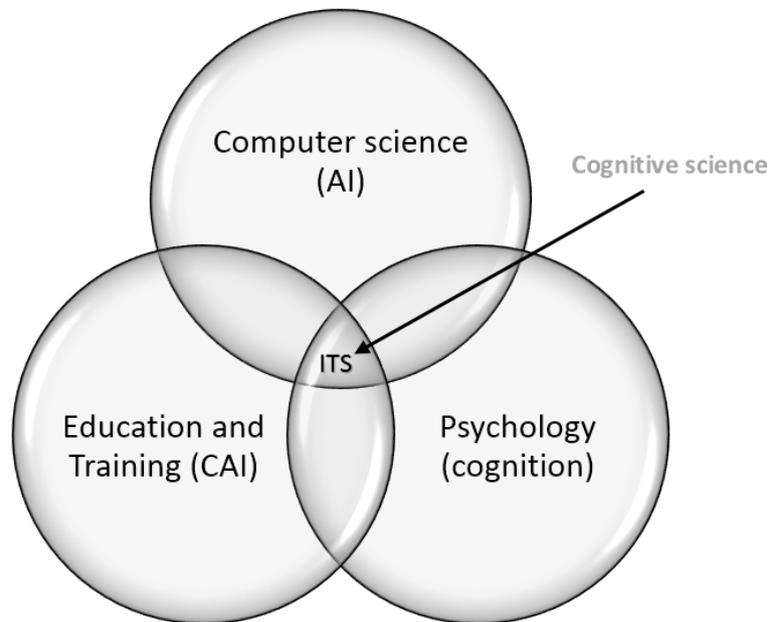


Abbildung 2.7: Forschungsbereich von ITS und Cognitive Science [nach Nwana, 1990, S. 253]

Systems ASSISTments ausgewertet. Zusätzlich wurde zwischen den Phasen des Lösens von Problem und dem Arbeiten mit dem virtuellen Tutor unterschieden. Es war möglich, diese Daten über einen Zeitraum von zwei Jahren und bei einer Beteiligung von über tausend Studenten und Studentinnen auszuwerten. Studierende, die beim Lösen der Probleme Langeweile oder Konfusion verspürten, konnten auch beim abschließenden Test, wie erwartet, nicht überzeugen. Im Gegensatz zu den Erwartungen ließen sich Langeweile und Konfusion beim arbeiten mit dem virtuelle Tutor mit guten Noten im Abschlusstest verknüpfen. Ebenso unerwartet war eine positive Beziehung zwischen Frustration und Lernen. Obwohl diese Studie nur für eine einzige Lernplattform durchgeführt wurde, können diese Zusammenhänge bereits erste Indikatoren für die Benachrichtigung von Lehrerinnen und Lehrern sein. [vgl. Pardos et al., 2013, S. 117-120].

2.3 Learning Analytics Cycle

Learning Analytics stellt bei der Analyse und Interpretation der Daten immer den Lehrenden in den Mittelpunkt. Es wird versucht diesen bei seinen Entscheidungen zu unterstützen und ihm einen Überblick über mögliche Interventionen zu ermöglichen. Nach Campell und Oblinger [2007] besteht ein Analyseprozess aus fünf Schritten: Sammeln (Capture), Berichten (Report), Prognose (Predict), Agieren (Act) und Weiterentwickeln

(Refine). Im Fall des Agierens ist es sehr wichtig, passende Interventionen zu finden und durchzuführen. Aufbauend auf diese Schritte definiert Clow [2012] den Learning Analytics Cycle (siehe Abbildung 2.8).

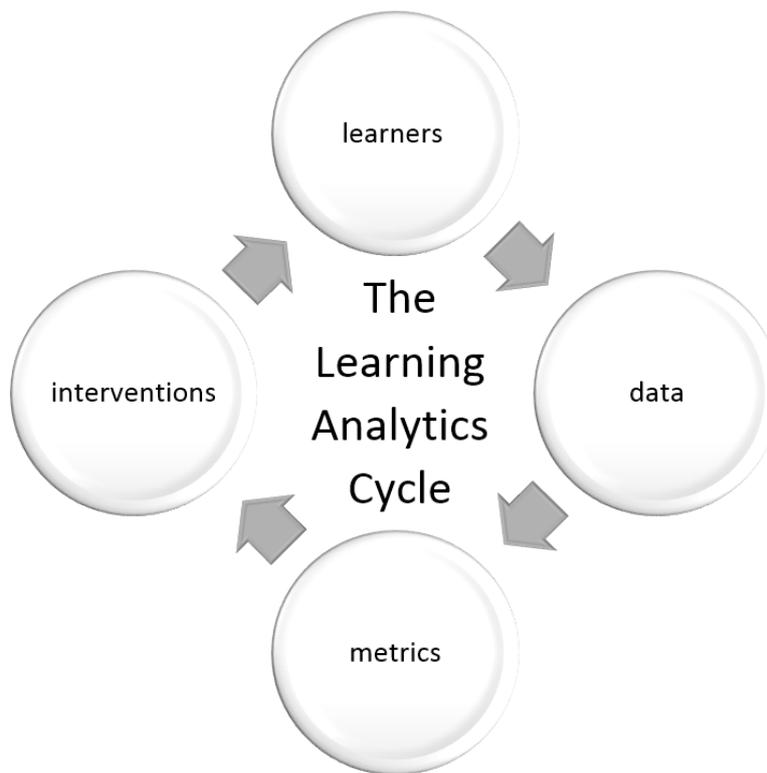


Abbildung 2.8: Learning Analytics Cycle [nach Clow, 2012]

Dieser geschlossene Prozess besteht nach Clow [2012] aus vier Hauptkomponenten:

Die erste Komponente sind die **Lernenden** selbst. Dies können Studierende einer Universität oder andere Lernende sein, die Online-Lernunterlagen oder Open Educational Resources (OER) verwenden.

Im nächsten Schritt steht das **Generieren und Sammeln** von Daten im Vordergrund. Diese Daten können Informationen des Lernalters bzw. der Lernerin, geografische Daten, Logdaten von Lernumgebungen, Postings in einem Forum, Onlineprüfungsergebnisse usw. sein. Eine große Anzahl dieser Daten kann bereits automatisch generiert und verarbeitet werden.

Die dritte Komponente in diesem Prozess ist das Herzstück von Learning Analytics - **die eigentliche Analyse**. Durch diesen Prozessschritt werden Daten über den Lernprozess des einzelnen Studierenden bzw. Lernenden gewonnen und Statistiken und visuelle Repräsentationen erstellt. Auf diesem Gebiet konnten bereits sehr

nützliche Entwicklungen, wie Dashboards, Netzwerk-Analyse und Vorhersagesysteme, hervorgebracht werden. Diese Tools sind für den Lehrenden von großer Bedeutung, da diese einen Einblick über den aktuellen Wissensstand der Studierenden geben.

Es ist jedoch sehr wichtig, dass dieser Prozess nicht bei diesem dritten Schritt endet, sondern dass aus dieser Analysephase konkrete **Interventionen** resultieren, die den Lerner bzw. die Lernerin auch wirklich erreichen. Dies kann durch Dashboards und Statistiken für die Lernenden selbst ermöglicht werden, um ihre Ergebnisse mit anderen Kollegen zu vergleichen und selbst zu reflektieren, oder durch die Intervention einer Lehrperson, die versucht auf die Probleme der Studierenden einzugehen und Missverständnisse zu beseitigen.

Learning Analytics muss diese vier Schritte nicht unbedingt beinhalten. Ein Projekt, das Analysen über das Lernverhalten der einzelnen Benutzer erstellt und diese Analysen nicht zur Verfügung stellt, wird jedoch nicht sehr erfolgreich sein [vgl. Clow, 2012, S. 135].

2.3.1 Lerntheorie

Der Learning Analytics Cycle aus Abbildung 2.8 hat die Grundlagen bereits in der fundamentalen Lerntheorie. Dieses Fachgebiet ist schon wesentlich älter als Learning Analytics selbst [vgl. Clow, 2012, S. 135].

Eine sehr wichtige Lerntheorie, die dem Learning Analytics Cycle zu Grunde liegt, ist der entdeckende Lernprozess nach Kolb [vgl. Kolb, 1984]. Er beruht in vielen Punkten auf den Ergebnissen von Dewey und Piaget und fügt das Konzept von 'Lernen durch Feedback' von Lewin hinzu. In diesem Prozess gibt es, ebenso wie beim Prozess von Learning Analytics, vier Komponenten. Zu Beginn des Prozesses kommt die konkrete Erfahrung (concrete experience), die dann durch den zweiten Schritt der reflektierenden Beobachtung analysiert wird. Der nächste Schritt besteht darin, diese Beobachtungen zu abstrahieren, dies nennt Kolb „abstract conceptualisation“. Aus diesem Konzept geht der Lerner in die Phase des aktiven Experimentierens (active experimentation) über. Durch dieses aktive Ausprobieren erhält er wieder eine konkrete Situation bzw. Erfahrung und der Kreis schließt sich [vgl. Clow, 2012; McCarthy, 2010].

Der Prozess von Learning Analytics besitzt zwei Beispiele, wo die Einflussnahme vom Lernprozess nach Kolb zu erkennen ist. Die erste Gemeinsamkeit kann man durch das Betrachten des ganzen Prozesses von Learning Analytics erkennen. Zu Beginn stehen der Lerner selbst und die Aktionen, die er ausführt (concrete experience). Danach werden Daten generiert und gespeichert (observation) und eine Metrik bzw. Analyse erstellt (abstract conceptualisation). Als Letztes folgt die Intervention (active experimentation). Einen weiteren Ansatz kann man erkennen, wenn man den Prozess von Learning Analytics auf einem individuellen Level betrachtet. Durch die Statistiken und einfachen Visualisierungen von Daten wird es Lernenden leichter gemacht, den eigenen Lernprozess

zu reflektieren und zu abstrahieren [vgl. Clow, 2012, S. 135].

Eine weitere Lerntheorie kommt von D. Schön [1983]. Für ihn ist das Reflektieren in Aktionen und das spätere Reflektieren dieser Aktionen ein wichtiger Prozess für das Lernen. Dieses Reflektieren kann durch einen ständigen und iterativen Feedbackprozess maßgeblich unterstützt werden. Durch die Komponente „Intervention“ im Learning Analytics Prozess kann dieses Feedback vom System direkt oder über den Umweg des Lehrers zu dem Lernenden gelangen. Auf beide Arten erhalten Lernende das benötigte Feedback, um ihren Lernprozess zu verbessern [vgl. Clow, 2012, S. 135].

Im Vereinigten Königreich ist eine weitere Lerntheorie weit verbreitet. Die Theorie stammt von Laurillard [2002] und basiert auf Kolbs Lernprozess und der „conversation theory“ von Pask [vgl. Pask, 1976]. Ihrer Ansicht nach geschieht Lernen durch eine Serie von Gesprächen zwischen Lehrenden und Studierenden und Studierenden untereinander, verknüpft mit Adaption und Reflexion. Dies geschieht auf zwei unterschiedlichen Ebenen: der Ebene der Aktion und der Ebene der Konzeption. Learning Analytics kann diesen Prozess dahingehend unterstützen, dass die Aktionen, die ein Student oder eine Studentin ausführt gespeichert und der Lehrperson zugänglich gemacht werden. Aus diesen Daten können Lehrende ihr Feedback für jeden Schüler adaptieren und verbessern [vgl. Clow, 2012, S. 135-136].

3 Schriftliche Rechenverfahren

Im Gegensatz zum halbschriftlichen Rechnen und zum Rechnen mit dem Kopf, wo mit Zahlen operiert wird, verwendet das schriftliche Rechenverfahren das Stellenwertsystem und führt Operationen mit einzelnen Ziffern durch. Dies ermöglicht die Zerlegung von komplexen Berechnungen in mehrere einfachere Teilschritte. Die Verwendung von schriftlichen Rechenverfahren erfolgt nach festen Regeln. Diese festen Regeln werden oft als Algorithmus zusammengefasst. Für Ziegenbalg et al. [2007, S. 23] ist ein Algorithmus „eine endliche Folge von eindeutig bestimmten Elementaranweisungen, die den Lösungsweg eines Problems exakt und vollständig beschreiben“. Dieser Lösungsweg soll durch die korrekte Abfolge einzelner Schritte zu einer Lösung führen (sofern diese existiert). Diese Algorithmen oder Verfahren sind nicht eindeutig und in vielen Fällen je nach Ländern unterschiedlich [vgl. Padberg, 2005, S. 203-205].

Die Stärken der schriftlichen Rechenverfahren liegen sicher in der breiten Einsetzbarkeit und der Förderung des Verständnisses für das dezimale Stellenwertsystem. Dadurch, dass Algorithmen nicht nur für einzelne Beispiele gelten, sondern auf Aufgaben von gleicher Struktur angewendet werden können, kann man mit ihnen bereits früh Beispiele mit beliebig großen Zahlen lösen. Durch das Stellenwertsystem wird dieses Lösen solcher komplizierten Aufgaben wesentlich einfacher und beschränkt sich oftmals nur auf Rechnungen aus dem kleinen Einspluseins oder Einmaleins. Zu erwähnen ist auch, dass diese Rechenverfahren ebenso außerhalb des dezimalen Stellenwertsystems (z.B.: im binären System) gelten und angewendet werden können. Ebenso bilden Algorithmen eine der zentralen Ideen im Bereich der Mathematik. Das Operieren mit Algorithmen kann also mit Hilfe der schriftlichen Rechenverfahren bereits in der Grundschule erlernt werden [vgl. Padberg, 2005, S. 204-206].

Es gibt jedoch auch einige Kritikpunkte bezüglich des Einsatzes von schriftlichen Rechenverfahren. Oftmals verlieren Schülerinnen und Schüler durch das Operieren mit Ziffern das Verständnis für die ganze Zahl und sehen Rechnen nur mehr als reine Ziffernmanipulation. Dies führt dazu, dass Ergebnisse mit leicht ersichtlicher falscher Größenordnung einfach akzeptiert werden. Damit verbunden, wird auch das Zahlenverständnis nicht geschult und es wird nur noch anhand von Algorithmen operiert. Dieser Meinung sind auch Kamii und Dominick [1997, S. 58]: „Algorithms thus make most children stop trying to make sense of numbers and lead them to focus on remembering only the steps“. Schülerinnen und Schüler neigen gerne dazu, durch die Rechensicherheit, die ihnen der Algorithmus vermittelt, die schriftlichen Rechenverfahren auch bei einfachen

Beispielen anzuwenden. In weiterer Folge führt das Operieren mit Algorithmen zu kognitiver Passivität und nach Kamii und Dominick [1997, S. 58] sogar dazu, dass „[students] give up their own thinking“. Die Gefahr bei zu früher Automatisierung der Rechenverfahren besteht auch darin, dass Einsichtsprozesse für das Verfahren nicht durchlaufen werden. Dies führt dazu, dass Algorithmen bei Vergessen nicht mehr rekonstruiert werden können [vgl. Padberg, 2005, S. 206-207].

Diesen Stärken und Schwächen der schriftlichen Rechenverfahren muss im Unterricht Rechnung getragen werden. Die daraus resultierenden Konsequenzen beschreibt Padberg [2005, S. 207-210] näher. In diesem Kapitel wird nun näher auf die schriftlichen Rechenverfahren der Addition und Subtraktion eingegangen, da diese Operationen auch im Plusminus-Trainer implementiert wurden.

3.1 Addition

Die Addition (oder in der Volksschule auch als Plus-Rechnung bekannt) ist für die meisten Kinder die leichteste der vier Grundrechnungsarten. Die durchschnittliche Fehlerquote im dritten bis sechsten Schuljahr liegt bei ca. 5% [vgl. Padberg, 2005, S. 214-215]. Für die schriftliche Addition gibt es im Gegensatz zu den anderen Grundrechnungsarten nur ein Verfahren [vgl. Padberg, 2005, S. 221]. Dieses wird im nächsten Kapitel kurz erläutert.

3.1.1 Verfahren

Bei der schriftlichen Addition werden die Summanden zunächst stellenweise untereinander geschrieben (Einer über Einer, Zehner über Zehnern, usw.). Anders als beim nicht normierten Rechnen (z.B. Kopfrechnen oder halbschriftlichen Rechnen) ist das sorgfältige und stellengerechte Aufschreiben der Zahlen bei Normalverfahren generell grundlegend. Auf kariertem Papier bietet es sich an, eine Ziffer pro Kästchen zu schreiben. Dann werden die gleichen Stellenwerte miteinander addiert, also „Einer + Einer“, „Zehner + Zehner“, usw. Begonnen wird immer beim kleinsten Stellenwert (d.h. bei Additionen mit Summanden in \mathbb{N} bei der Einer-Stelle). Es ist von Vorteil, wenn von „unten nach oben“ gerechnet wird, da die schriftliche Subtraktion nach dem gleichen Prinzip funktioniert. Die einzelnen Zwischenergebnisse werden in der richtigen Stellenwertspalte notiert. Wird allerdings der Wert 9 überschritten, ist es notwendig das Zwischenergebnis zu „entbündeln“ (z.B. 13 Einer = 1 Zehner und 3 Einer) und Überträge klein in der links benachbarten Spalte anzuschreiben [vgl. Padberg, 2005, S. 212-213].

In Abbildung 3.1 sehen wir die schriftliche Addition anhand des Beispiels $45 + 37$. Im ersten Schritt werden die Einer addiert. Durch Summieren von $5 + 7$ erhalten wir 12. Da wir uns an der Einerstelle befinden, schreiben wir die 2 in die Lösungsspalte und schreiben den verbliebenen Zehner als kleine 1 in die linke Spalte (= Zehner-Stelle). Diese 1 wird

$$\begin{array}{r}
 45 \\
 + 37 \\
 \hline
 2
 \end{array}
 \longrightarrow
 \begin{array}{r}
 45 \\
 + 37 \\
 \hline
 1 \\
 2
 \end{array}
 \longrightarrow
 \begin{array}{r}
 45 \\
 + 37 \\
 \hline
 1 \\
 82
 \end{array}$$

Abbildung 3.1: schriftliches Additionsverfahren

dann im nächsten Schritt in die Summe aufgenommen und man erhält $1 + 3 + 4 = 8$. Dieses wird an die Zehner-Stelle der Lösung vermerkt.

3.1.1.1 Wieso funktioniert dieses Verfahren?

Das Additionsverfahren beruht im Wesentlichen auf der Gültigkeit des Kommutativ-, Assoziativ- und Distributivgesetzes, sowie auf der Tätigkeit des Umbündelns, sobald ein Zwischenergebnis mehrstellig ist.

Betrachten wir dies anhand des Beispiels aus 3.1. Wegen der Gültigkeit des Kommutativ- und Assoziativgesetzes bezüglich der Addition gilt:

$$\begin{aligned}
 45 + 37 &= (40 + 5) + (30 + 7) \\
 &= (5 + 7) + (40 + 30)
 \end{aligned}
 \tag{3.1}$$

Die Gültigkeit des Distributivgesetzes bewirkt, dass wir beim Additionsverfahren die Ziffern spaltenweise addieren können, denn es gilt:

$$\begin{aligned}
 (5 + 7) + (40 + 30) &= (5 + 7) + (4 \cdot 10 + 3 \cdot 10) \\
 &= (5 + 7) + (4 + 3) \cdot 10
 \end{aligned}
 \tag{3.2}$$

Durch Umbündeln von 12E in 1Z und 2E (bzw. von 12 in $10 + 2$) ergibt sich:

$$\begin{aligned}
 (5 + 7) + (4 + 3) \cdot 10 &= (2 + 1 \cdot 10) + (4 + 3) \cdot 10 \\
 &= 2 + (1 + 4 + 3) \cdot 10 \\
 &= 2 + 8 \cdot 10 \\
 &= 82
 \end{aligned}
 \tag{3.3}$$

3.1.2 Schwierigkeiten und Fehler

Wie bereits in Kapitel 3.1 erwähnt, ist die schriftliche Addition für Schülerinnen und Schüler die einfachste Operation der vier Grundrechnungsarten. Die Fehler, die gemacht werden, konzentrieren sich sehr stark auf zwei Fehlertypen. Zum einen sind dies Fehler beim kleinen Einsundeins und zum anderen Fehler bei Überträgen. Fehler bei ungleicher Stellenanzahl und falsche Rechenregeln mit der 0 treten zwar auf, jedoch weit weniger häufig [vgl. Padberg, 2005, S. 214-216].

- **Fehler beim kleinen Einspluseins:**

Hierbei kommt es sehr häufig vor, dass die Summe gerade um 1 vom korrekten Ergebnis abweicht. Dies resultiert oftmals daraus, dass die Lernenden die Einsundeins-Aufgabe nicht auswendig wissen, sondern durch Abzählen lösen [vgl. Padberg, 2005, S. 214-216]. Durch dieses zählende Lösen passiert nach Gerster [1982, S. 29] vorwiegend einer von zwei typischen Fehlern. Beim ersten Fehlertyp zählt der Schüler bzw. die Schülerin fälschlicherweise die erste Zahl bereits mit und kommt dadurch auf ein falsches Ergebnis. Dies würde in Beispiel a) der Abbildung 3.2 beim Zusammenzählen der Zehnerstellen ($5 + 4$) bedeuten, der Schüler bzw. die Schülerin zählt 5, 6, 7, 8 und nennt 8 als Ergebnis. Der zweite Fehlertyp besteht darin, dass der Schüler bzw. die Schülerin zwar richtig zu zählen beginnt, zum Beispiel 6, 7, 8, 9, jedoch dann die um 1 größere Zahl 10 als Ergebnis angibt. Durch diese fehlerhaften Zählstrategien kommt es daher sehr oft zu einer Abweichung von 1 von der korrekten Antwort [vgl. Padberg, 2005, S. 214-216]. Ein Beispiel für diesen Fehlertyp ist in Abbildung 3.2 a) zu sehen.

- **Übertragsfehler:**

Padberg [2005, S. 215] unterscheidet bei diesem Fehlertyp zwischen einer falschen Übertragsstrategie, wie im Beispiel b) in Abbildung 3.2 zu sehen, und zwischen dem Nichtberücksichtigen des Übertrags. Beispiele für die Nichtberücksichtigung des Übertrags findet man in Abbildung 3.2 c) bis f). In diesem Fall differenziert Padberg [2005, S. 215] noch einmal genauer nach Lernenden, die generell den Übertrag vernachlässigen [Beispiel c)] und jenen, die dies nur in speziellen Fällen tun. Bei Beispiel d) wäre dies „kein Übertrag zur 0“, bei Beispiel e) „kein Übertrag zu einer leeren Stelle“ und bei Beispiel f) „kein Übertrag zur 9“. Gerster [1982, S. 28-35] differenziert Übertragsfehler sogar noch genauer und detaillierter.

- **Fehler bei ungleicher Stellenanzahl:**

Dieser Fehlertyp ist beispielsweise in Beispiel g) in Abbildung 3.2 zu erkennen. Der Schüler bzw. die Schülerin lässt die Hunderterstelle, wo nur noch eine Zahl zu addieren wäre, im Ergebnis einfach leer. Nach Padberg [2005, S. 216] verfahren die Lernenden hierbei nach dem Motto „zu 'Nichts' kann ich auch nichts addieren“. Die Ursache des Fehlers sieht Gerster [1982, S. 30-31] darin, dass bei der Einführung der schriftlichen Addition fast ausschließlich Beispiele mit gleicher Stellenanzahl

gerechnet werden.

- **Falsche Rechenregel mit der 0:**

Die Fehler in den Beispielen h) und i) in Abbildung 3.2 entstehen dadurch, dass Schülerinnen und Schüler eine falsche Rechenregel für die 0 besitzen oder die Operation mit der Multiplikation verwechseln. Sie sind der Meinung, dass $0 + a = 0 = a + 0$ [vgl. Gerster, 1982, S. 29].

<p>a) $\begin{array}{r} 442 \\ + 354 \\ \hline 786 \end{array}$</p>	<p>b) $\begin{array}{r} 267 \\ + 438 \\ \hline 7815 \end{array}$</p>	<p>c) $\begin{array}{r} 763 \\ + 859 \\ \hline 1512 \end{array}$</p>
<p>d) $\begin{array}{r} 578 \\ + 406 \\ \hline 974 \end{array}$</p>	<p>e) $\begin{array}{r} 578 \\ + 91 \\ \hline 569 \end{array}$</p>	<p>f) $\begin{array}{r} 247 \\ + 98 \\ \hline 335 \end{array}$</p>
<p>g) $\begin{array}{r} 536 \\ + 62 \\ \hline 98 \end{array}$</p>	<p>h) $\begin{array}{r} 276 \\ + 202 \\ \hline 408 \end{array}$</p>	<p>i) $\begin{array}{r} 301 \\ + 286 \\ \hline 507 \end{array}$</p>

Abbildung 3.2: Beispiel der Fehlertypen bei der schriftlichen Addition [in Anlehnung an Padberg, 2005]

Um die beschriebenen Fehlertypen zu vermeiden, ist es nach Gerster [1982, S. 37-38] hilfreich, bereits einige wichtige Punkte bei der Darstellung der Addition zu beachten:

- Übertragungsziffern sollen konsequent verwendet werden (entweder immer oder nie).
- Bei Überträgen sollte immer eine ganze Zeile für die Übertragungsziffer frei sein.
- Die Überträge sollen immer genau in die richtige Spalte geschrieben werden und nicht dazwischen.
- Bei Karopapier sollte in jedem Kästchen nur eine Zahl stehen.

3.2 Subtraktion

Im Gegensatz zur schriftlichen Addition gibt es bei der schriftlichen Subtraktion verschiedene Verfahren. Diese Verfahren sind mehr oder weniger für den Gebrauch in der Grundschule geeignet. Es gibt auch immer wieder Diskussionen über die Stärken

und Schwächen der einzelnen Verfahren (siehe dazu Lorenz und Radatz [1993]; Dröge et al. [1999]; Padberg [1994]).

3.2.1 Verfahren

Um die einzelnen Subtraktionsverfahren zu charakterisieren, geht Padberg [2005, 222 ff] von zwei Gesichtspunkten aus. Er unterscheidet die Verfahren nach der Art der Differenzbildung und nach der Art des Übertrags.

3.2.1.1 Arten der Differenzbildung

Bei der Art der Differenzbildung wird wiederum zwischen Abziehen und Ergänzen unterschieden. Beim Abziehen wird die Minussprechweise („9 minus 5 ist gleich“) angewandt, während beim Ergänzen die Plusprechweise („5 plus wie viel gleich 9“) verwendet wird [vgl. Gerster, 1982, 40 ff; Padberg, 2005, 222 ff].

Ob ein Schüler Abziehen oder Ergänzen verwendet, macht in der schriftlichen Form keinen Unterschied. In beiden Fällen ergibt sich das gleiche Ergebnis. Es ist jedoch ein großer Unterschied in der Art der Berechnung der Teildifferenzen und in der Sprechweise [vgl. Padberg, 2005, 222 ff]. In Abbildung 3.3 werden anhand eines Beispiels die unterschiedlichen Sprechweisen demonstriert. Aus diesen Sprechweisen ergeben sich jeweils Vor- sowie auch Nachteile.

			Sprechweise:	
			beim Abziehen:	beim Ergänzen:
8	9	3	3 minus 1 gleich <u>1</u>	1 plus <u>2</u> gleich 3
-	3	2	9 minus 2 gleich <u>7</u>	2 plus <u>7</u> gleich 9
<hr/>			8 minus 3 gleich <u>5</u>	3 plus <u>5</u> gleich 8
<u>5</u>	<u>7</u>	<u>2</u>		

Abbildung 3.3: Unterschiedliche Sprechweisen bei Abziehen und Ergänzen [in Anlehnung an Padberg, 2005, S. 223]

Die Vorteile des Abziehens bzw. Nachteile des Ergänzens nach Padberg [2005, 222 ff] sind die folgenden:

- Das Abziehen ist der natürlichen Sinngebung der Subtraktion näher, wohingegen das Ergänzen eher gekünstelt wirkt.
- Oftmals wird durch die Technik des Ergänzens die Operation verwechselt und addiert.
- Die Schreib- und Sprechweise gehen beim Ergänzen auseinander. So ist beim Ergänzen die Ergebniszahl in der Mitte (z.B.: 3 plus 4 gleich 7). Beim Abziehen

ist sie jedoch auf dem richtigen Platz (z.B.: 7 minus 3 gleich 4)

- In den meisten lebensnahen Situationen wird etwas weggenommen, sprich abgezogen.

Dagegen stehen einige Punkte, die für das Ergänzen und gegen das Abziehen sprechen [vgl. Padberg, 2005, 222 ff]:

- Beim Ergänzen wird nur das Einsundeins benötigt. Dies ist zumeist besser verankert als das Einsminuseins.
- Bei der Methode des Ergänzens wird zum Subtrahenden solange addiert bis man beim Minuenden angekommen ist. Dies entspricht dem Vorwärtszählen, welches meistens besser beherrscht wird als das Rückwärtszählen (Abziehen).
- Die Schüler können durch das Ergänzen schon sehr früh den Zusammenhang zwischen Subtraktion und Addition erkennen.

3.2.1.2 Berechnen von Überträgen

Bei Beispielen mit Überträgen wird hauptsächlich zwischen folgenden Techniken unterschieden: Die Entbündelungs- oder Borgetechnik und die sogenannte Erweiterungstechnik, eine Technik bei der Minuend und Subtrahend gleichsinnig verändert werden. Eine weitere Technik ist das Auffüllen des Subtrahenden zum Minuenden (Auffülltechnik). Da diese jedoch nicht sehr verbreitet ist, werden nur die beiden anderen Techniken kurz mit ihren Stärken und Schwächen erläutert.

Entbündelungstechnik

Beim Entbündelungsverfahren, welches in Abbildung 3.4 dargestellt ist, wird bei notwendiger Stellenüberschreitung der nächsthöhere Stellenwert des Minuenden entbündelt. Im Beispiel der Abbildung 3.4 würde einer der fünf vorhandenen Zehner des Minuenden in zehn Einer umgewandelt. Als Notation wird hier meist eine kleine 1 zwischen den Spalten angeschrieben. Diese muss bei der Berechnung des nächsten Stellenwerts mit abgezogen werden [vgl. Kühnhold und Padberg, 1986; Padberg, 2005].

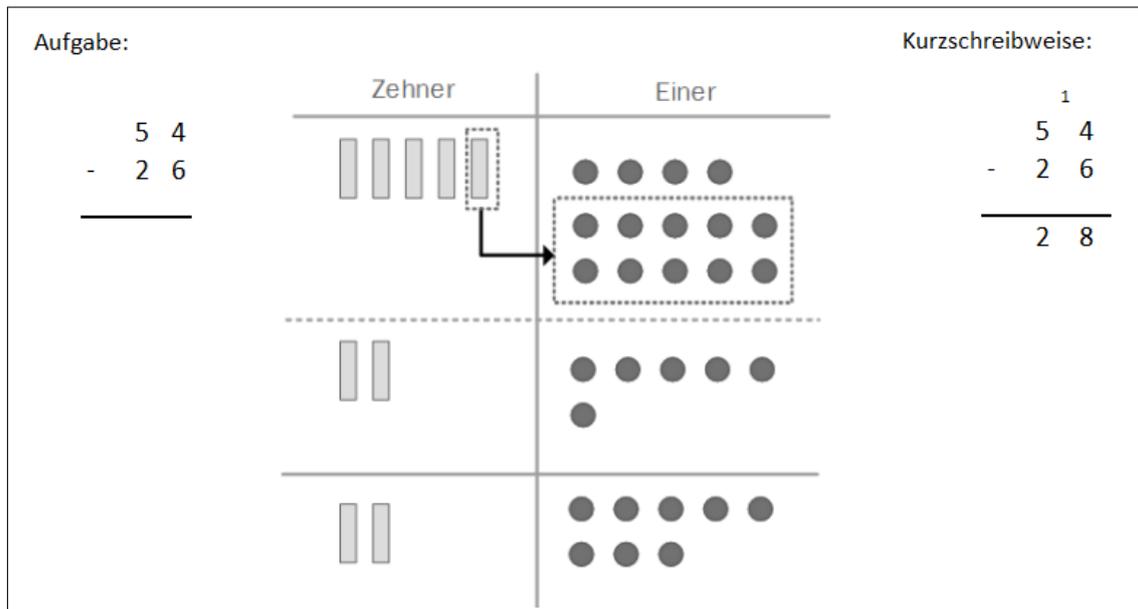


Abbildung 3.4: Entbündelungstechnik [nach Padberg, 2005, S. 225]

Die Entbündelungstechnik wird vor allem mit der Differenzbildung „Abziehen“ verbunden. Daraus ergeben sich nach Padberg [2005] einige Vor- und Nachteile:

Da diese Technik sehr gut visualisiert werden kann, ist sie für Schüler sehr einfach nachvollziehbar und wird meistens sehr gut verstanden. Das Umbündeln ist zumeist bereits vom Prinzip der Addition bekannt, was das Verständnis erleichtert. Außerdem sind keine „Tricks“ wie bei der Erweiterungstechnik (Konstanz der Differenz), notwendig.

Bei mehreren Subtrahenden ist diese Technik jedoch oftmals schwieriger anzuwenden, da bei ungeeigneten Zahlen mehr als ein nächster Stellenwert entbündelt werden muss. Minuenden, die aus mehreren Nullen bestehen, führen bei dieser Technik zu einem Spezialfall, da keine Elemente beim direkt vorhergehenden Stellenwert vorhanden sind. Hierbei muss so lange weitergegangen werden, bis ein Stellenwert größer als 0 auftritt und all diese Schritte rückwärts entbündelt werden.

Erweiterungstechnik

Diese Technik nutzt das mathematische Gesetz der Konstanz der Differenz. Dies bedeutet, dass die Differenz gleich bleibt, wenn zu Minuend und Subtrahend die selbe Zahl addiert wird $[(a - b) = (a + c) - (b + c)]$. Bei dieser Erweiterung ist jedoch noch ein weiterer Trick erforderlich. Beide Zahlen werden zwar um die selbe Zahl erweitert, jedoch anders gebündelt. Dies ist in Abbildung 3.5 gut ersichtlich. Zum Minuenden wird die Zahl 10 bei den Einern hinzugezählt, während im Subtrahenden die Zahl 10 bei den Zehnern addiert

wird [vgl. Kühnhold und Padberg, 1986; Padberg, 2005].

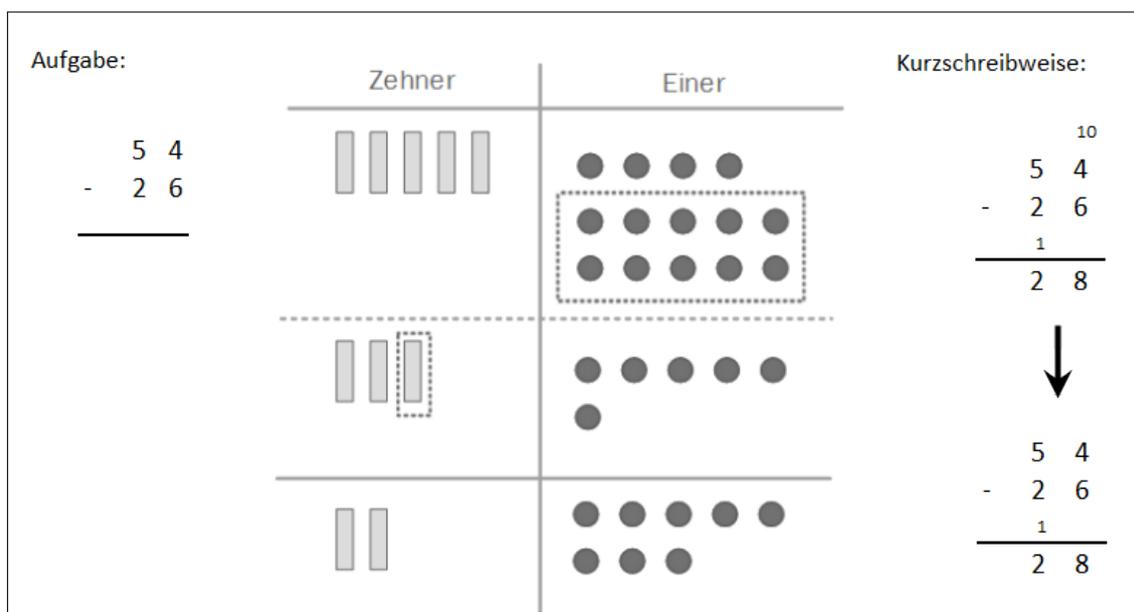


Abbildung 3.5: Erweiterungsverfahren [nach Padberg, 2005, S. 225]

Bei dieser Schreibweise werden bei der Einführung in die Subtraktion häufig beide Additionen angeschrieben (siehe Abbildung 3.5 - obere Kurzschreibweise). Sobald dies gefestigt ist, wird nur noch die 1 an die nächste Stelle übertragen. Diese muss in weiterer Folge zum Subtrahenden addiert bzw. gleich vom Minuenden subtrahiert werden.

Diese Technik kann sowohl mit dem Abziehen als auch mit dem Ergänzen kombiniert werden. Meistens wird sie jedoch mit der Differenzbildung des Ergänzens kombiniert [vgl. Padberg, 2005, 227 f].

Ein Vorteil dieser Technik liegt ebenfalls in der einfachen Veranschaulichung. Auch Aufgaben mit mehreren Nullen im Minuenden sind bei dieser Technik kein Problem und können ebenso wie Aufgaben mit mehreren Subtrahenden einfach gelöst werden.

Padberg [2005] sieht jedoch ein Problem beim Verständnis dieser Technik. Schülerinnen und Schüler müssen mit zwei Tricks operieren: Zuerst muss die Konstanz der Differenz klar sein und danach müssen auch noch die Additionen zu unterschiedlichen Stellenwerten durchgeführt werden. Ein weiterer Nachteil liegt in der Endschreibweise. Durch Vernachlässigen der 10 beim Stellenwert, wird die Idee des Erweiterns nicht mehr sichtbar und es besteht die Gefahr einer blinden Mechanisierung der Technik.

3.2.1.3 Zusammenfassung

Theoretisch ergeben sich aus den zwei Ansätzen zur Differenzbildung und den drei Techniken für die Überträge sechs Verfahren. Auffüllen und Abziehen lässt sich jedoch nicht kombinieren und so erhält man fünf verschiedene Verfahren (siehe Tabelle 3.1).

	Entbündeln	Erweitern	Auffüllen
Abziehen	①	②	-
Ergänzen	③	④	⑤

Tabelle 3.1: Verfahren für die schriftliche Subtraktion [nach Padberg, 2005]

Welches Verfahren an Schulen wirklich Verwendung findet, hängt in den meisten Fällen vom Lehrplan ab. In einigen Ländern (z.B. Deutschland) ist die Wahl des Subtraktionsverfahrens sogar dem Lehrer überlassen. Einige empirische Studien (z.B. J. Johnson [1938]) haben sich mit den Vor- und Nachteilen dieser Verfahren beschäftigt und jeweils für alle Methoden positive als auch negative Aspekte gefunden. Weiterführend kann dies in Padberg [2005, 231 ff] nachgelesen werden.

3.2.2 Schwierigkeiten und Fehler

Die schriftliche Subtraktion ist im Gegensatz zur Addition für Schülerinnen und Schüler wesentlich schwieriger. Nach einer Studie von Gerster [1982, S. 52] betrug die Fehlerquote bei der schriftlichen Subtraktion zwischen 5% und 53%. Charakteristika für Schwierigkeiten sind nach Padberg [2005] sowie Kühnhold und Padberg [1986] die Anzahl der notwendigen Überträge, die Anzahl der Stellen im Minuenden und Subtrahenden und die Anzahl der Nullen im Ergebnis oder in der Angabe.

Nach einer Studie von Kühnhold und Padberg [1986] machten 14% der gesamten Testpersonen mindestens einen systematischen Fehler. Als „systematischer Fehler“ wird jener Fehler bezeichnet, der mindestens bei der Hälfte der möglichen Aufgaben auftritt. Wenn Fehler relativ häufig in der Stichprobe auftreten, wird von „typischen Fehlern“ gesprochen. Insgesamt wurden 19 systematische Fehlertypen identifiziert. Die drei häufigsten davon sind:

- Spaltenweise Subtraktion der kleineren von der größeren Ziffer ohne anschließenden Übertrag (Beispiel a) in Abbildung 3.6)
- Kein Übertrag berücksichtigt (Beispiel b) in Abbildung 3.6)
- Kein Übertrag an eine leere Stelle (Beispiel c) in Abbildung 3.6)

Um einen Überblick über die Häufigkeit der Fehler zu erhalten, klassifizieren Kühnhold

und Padberg [1986] alle Fehler in Anlehnung an Gerster [1982] und teilen sie in sieben Gruppen. Berücksichtigt man zusätzlich die Anzahl der Aufgaben, bei denen die Fehlergruppen überhaupt auftreten können, erhält man als Ergebnis in absteigender Reihenfolge ihres Auftretens folgende Liste:

1. Übertragsfehler
2. Rechenrichtungsfehler
3. Perseverationsfehler
4. Fehler mit der Null
5. Fehler bei unterschiedlicher Stellenanzahl
6. Einsundeinsfehler
7. Anwendung der inversen Operation (Addition statt Subtraktion)

a) $\begin{array}{r} 547 \\ - 354 \\ \hline 213 \end{array}$	b) $\begin{array}{r} 667 \\ - 438 \\ \hline 239 \end{array}$	c) $\begin{array}{r} 763 \\ - 99 \\ \hline 764 \end{array}$
d) $\begin{array}{r} 773 \\ - 407 \\ \hline 376 \end{array}$	e) $\begin{array}{r} 178 \\ - 91 \\ \hline 187 \end{array}$	f) $\begin{array}{r} 300 \\ - 96 \\ \hline 214 \end{array}$
g) $\begin{array}{r} 853 \\ - 459 \\ \hline 404 \end{array}$	h) $\begin{array}{r} 604 \\ - 206 \\ \hline 498 \end{array}$	i) $\begin{array}{r} 232 \\ - 96 \\ \hline 236 \end{array}$

Abbildung 3.6: Beispiel einiger Fehlertypen bei der schriftlichen Subtraktion [in Anlehnung an Padberg, 2005]

Das Auftreten von Übertragsfehler liegt hierbei mit 49,9 % der gesamten Fehleranzahl sehr hoch. Im Vergleich dazu kommt die nächste Fehlergruppe „Rechenrichtungsfehler“ nur auf 16,5% [vgl. Kühnhold und Padberg, 1986]. Aus diesem Grund ist es sinnvoll, diese große Gruppe der Übertragsfehler genauer zu betrachten und noch einmal zu unterteilen. Gerster [1982] sowie Kühnhold und Padberg [1986] unterteilen diese Gruppe in drei weitere Untergruppen. Diese lauten:

- Übertrag nicht berücksichtigt
- Übertragsziffer zu viel

- Falsches Operieren mit der Übertragsziffer

Nach Kühnhold und Padberg [1986] ist in diesem Fall die erste Gruppe „Übertrag nicht berücksichtigt“ mit einem Anteil von 75% die häufigste Fehlergruppe. Gerster [1982] geht sogar noch einen Schritt weiter und zerteilt diese Gruppe in weitere Untergruppen. Bei ihm setzt sich die Gruppe „Übertrag nicht berücksichtigt“ zusammen aus:

Kein Übertrag...

- ... zu einer 0 (Beispiel d) in Abbildung 3.6)
- ... an eine leere Stelle (Beispiel e) in Abbildung 3.6)
- ... in Spalte mit gleichen Ziffern (Beispiel f) in Abbildung 3.6)
- ... nach Ergänzen bis 10 (Beispiel g) in Abbildung 3.6)
- ... nach Ergänzen ab $9 + 1$ (Beispiel h) in Abbildung 3.6)

Aus den Ergebnissen kann man sehr gut erkennen, dass die Schwierigkeit bei der schriftlichen Subtraktion hauptsächlich im Bereich des Übertrags liegt. Beispiele ohne Überträge werden wesentlich erfolgreicher gelöst als Beispiele mit Überträgen. Auffallend ist ebenfalls, dass sehr häufig der Übertrag nicht berücksichtigt wird. Viele Schülerinnen und Schüler können mit dem Übertrag zwar richtig operieren, verwenden bei kniffligen Aufgaben (z.B. Übertrag zur 0 oder 9) jedoch falsche Strategien [vgl. Padberg, 2005, 247 ff].

Durch die vielen möglichen Fehler und große Fehleranfälligkeit ist vor allem für die Subtraktion der Einsatz von diagnostischen Tests sehr interessant [vgl. Gerster, 1982, 52 ff]. Der Aufbau und die Vorteile dieser Tests werden im Abschnitt 3.3 (Diagnostische Tests) näher erläutert.

3.3 Diagnostische Tests

Schülerfehler beim schriftlichen Rechnen sind nicht, wie so oft geglaubt, Flüchtigkeitsfehler, sondern folgen in ca. 80% der Fälle einem bestimmten Muster. Für die vier Grundrechnungsarten gibt es eine Reihe von verschiedenen Fehlermustern. Bei solchen Mustern reagieren Schülerinnen und Schüler auf eine bestimmte Schwierigkeit mit einer bestimmten fehlerhaften Regelstruktur. Die Erkennung dieser Regelstrukturen ist jedoch im Unterricht selbst beinahe unmöglich. Speziell für diesen Fall wurden diagnostische Tests entwickelt [vgl. Gerster, 1982, 14 ff].

Um solche Tests gut und effektiv konstruieren zu können, wurden Klassenarbeiten, lernzielorientierte Tests und Hausübungen gesammelt und klassifiziert. Danach wurden die entstandenen Tests an verschiedenen Schulen erprobt. Daraus konnten bereits erste interessante Schlüsse gezogen werden. Zum einen hängt die Schwierigkeit eines Beispiels

nicht immer direkt von der angegebenen Schwierigkeitskomponente ab. Es kommt oftmals darauf an, welche Ziffernkombinationen in den einzelnen Teilaufgaben vorkommen (z.B. Auftreten einer Null, Übertrag zu 9, etc.). Diese speziellen Ziffernkombinationen kategorisiert man in sogenannte Schwierigkeitsmerkmale. Erst wenn ein solches Merkmal in mehreren Beispielen auftritt, zeigt es sich, ob Lernende für dieses Merkmal ein Fehlermuster entwickelt haben. Um den Umfang der Tests gering zu halten, konnte die Anzahl der Aufgaben auf eine pro Aufgabentyp (bei ausreichend vorkommenden Schwierigkeitsmerkmalen) reduziert werden. Gerster [1982] betont auch, dass diese Tests keinesfalls zur Benotung von Schülerinnen und Schülern dienen, sondern sie die Möglichkeit bieten, Fehlermuster bei Lernenden zu erkennen und diese in späterer Folge gezielt zu beheben [vgl. Gerster, 1982, 13 ff].

Durch den Einsatz und die Auswertung dieser Tests ist es für die Lehrperson möglich, typische und systematische Fehler bei Lernenden zu erkennen. Um das Potenzial der Tests auszuschöpfen, sollten einige Punkte beachtet werden. Zu Beginn des Tests muss den Schülerinnen und Schülern glaubhaft versichert werden, dass dieser Test nicht benotet wird. Es soll ihnen mitgeteilt werden, dass dieser Test etwaige Schwierigkeiten aufdecken kann, an denen in weiterer Folge gemeinsam mit der Lehrperson gearbeitet wird. Der Test sollte auf keinen Fall unter Zeitdruck durchgeführt werden. Als Lehrperson sollte man daher keine Zeitbegrenzung setzen, sondern warten, bis alle Schülerinnen und Schüler abgegeben haben. Um bei ungleichen Tempi keine Unruhe entstehen zu lassen, empfiehlt es sich, bereits vor dem Test Beschäftigungen für schnellere Schülerinnen und Schüler festzulegen. Des Weiteren sollte auch nicht mehr als ein Test pro Unterrichtseinheit geschrieben werden [vgl. Gerster, 1982, 13 ff].

Der Einsatz solcher Tests empfiehlt sich nach Gerster [1982] zu gewissen Zeitpunkten ganz besonders:

- **einige Wochen nach der Einführung eines neuen Verfahrens:**
Zu diesem Zeitpunkt tritt meistens bereits die Phase des Vergessens ein. Deswegen sollte auch keine Wiederholung des Verfahrens zu Beginn des Tests durchgeführt werden. Dadurch ist es möglich, die Fertigkeiten und die etwaigen Fehlermuster, welche die Lernenden gefestigt haben, zu erfahren.
- **kurz nach der Einführung der Subtraktion:**
Nach der Einführung der Subtraktion ist ein diagnostischer Test zur Addition oftmals sehr hilfreich, um Verwechslungen und Fehler zu erkennen, die auf Grund des neuen Verfahrens entstanden sind.
- **bei der Übernahme einer neuen Klasse:**
Mit der Hilfe von diagnostischen Tests kann sich die Lehrperson einen Überblick über den Wissensstand der Klasse verschaffen und erkennen wo Nachholbedarf besteht.

- **nach gezielter Förderung:**

Nach der gezielten Förderarbeit zur Behebung von Fehlermustern dient ein solcher Test zur Kontrolle und Verifizierung.

Bei der Auswertung diagnostischer Tests sollte immer bedacht werden, dass man nicht auf der Suche nach Flüchtigkeitsfehlern ist, sondern auf durchgehend verwendete Fehlerstrukturen. Daher empfiehlt es sich, mit Abkürzungen für jedes Fehlermuster zu arbeiten (für eine detaillierte Fehlerliste siehe Gerster [1982, 217 ff]). Nach einiger Einarbeitungszeit lässt sich eine Auswertung eines diagnostischen Tests für eine Klasse mit 25 Schülerinnen und Schülern in einer guten Stunde bewältigen [vgl. Gerster, 1982, 13 ff].

In den nächsten beiden Kapiteln wird jeweils ein diagnostischer Test für die Addition und Subtraktion vorgestellt und erläutert.

3.3.1 Addition

Wie bereits einige Male erwähnt, ist die schriftliche Addition für Schüler meist sehr einfach zu meistern. Jedoch gibt es auch bei dieser Operation des öfteren fehlerhafte Denk- und Handlungsmuster. Gerster [1982] sammelte über vier Jahre Daten von Schulen, um eine Liste von möglichen Fehlern bei der Addition zu erstellen.

In der Abbildung 3.7 ist ein solcher Diagnostetest zu sehen. Dieser Test ist auf Grund des behandelten Zahlenraums bereits ab der 3. Schulstufe einsetzbar und enthält viele verschiedene Schwierigkeitsmerkmale. Diese sind bei jedem Beispiel durch eine kleine Nummer angegeben [vgl. Gerster, 1982, 22 ff].

Interessant ist die Frage, weshalb die Beispiele 14 oder 24 auftauchen. Eigentlich sollten Kinder diese doch im Kopf lösen können. Normalerweise sollte dies so sein, jedoch rechnen viele Schüler, vor allem auf einem zusammenhängenden Übungszettel, formal und machen dabei typische Fehler. Diese beiden Fälle soll der Test ebenfalls untersuchen [vgl. Gerster, 1982, 22 ff].

	- Zahlenraum bis 1000 - zwei Summanden															
	ohne Null					Nullen in Summanden oder Ergebnis										
	ohne		mit			ohne		mit Stellenunterschied								
kein Übertrag	1	32 + 46		2	423 + 73			3	276 + 202		4	905 + 64		5	70 + 528	
				(9)(7)			(2)			(9)			(9)			
ein Übertrag	6	54 + 27		7	825 + 48			8	663 + 245		9	450 + 90		10	26 + 409	
				(1)(9)			(10)			(1)(7)			(4)			
	11	24 + 94		12	384 + 95			13	407 + 203		14	704 + 8		15	53 + 107	
			(2)(7)	(1)(9)			(4)(7)(10)			(1)			(4)(10)			
zwei Überträge	16	488 + 277		17	138 + 93			18	485 + 315		19	693 + 9		20	24 + 876	
			(8)	(8)(1)(5)			(8)(6)(10)			(6)(10)(1)			(9)(6)(10)			
	21	496 + 496		22	756 + 97			23	98 + 45		24	98 + 7		25	918 + 91	
			(9)(5)(7)	(1)(5)			(2)			(3)(6)(10)(1)			(3)(6)(1)(10)			

Zusätzliche Schwierigkeitsmerkmale:

- (1) Übertrag in leere Stelle obere Ziffer 0 oder 0 oder sonstige)
- (2) Übertrag in zusätzliche leere Stelle (Summanden stellengleich)
- (3) Übertrag in zusätzliche leere Stelle (Summanden stellenverschieden)
- (4) Übertrag zur Null
- (5) Übertrag zur 9
- (6) zweiter Übertrag resultiert aus erstem Übertrag
- (7) gleiche Ziffern übereinander
- (8) gleiche Ziffern übereinander nach Berücksichtigung des Übertrags
- (9) Summenziffer 9
- (10) Summenziffer 10

in Aufgabe Nr.

- 9, 12, 14, 17, 19, 22, 24, 25
 11, 23
 24, 25
 10, 13, 15
 17, 21, 22
 18, 19, 20, 24, 25
 2, 3, 9, 11, 13, 21
 16, 17, 18
 2, 4, 5, 12, 20, 21
 8, 13, 15, 18, 19, 20, 24, 25

Abbildung 3.7: Diagnostischer Test für die Addition [nach Gerster, 1982, S. 25]

3.3.2 Subtraktion

Die schriftliche Subtraktion ist für die meisten Schülerinnen und Schüler wesentlich komplexer und komplizierter als die Addition. Beim Erlernen der Subtraktion werden daher vermehrt falsche Rechenstrukturen angeeignet. Diagnostische Tests stellen hierbei eine gute Möglichkeit dar, verfestigte Fehlerstrukturen bei den Lernenden zu erkennen, um in weiterer Folge auf diese gezielt eingehen zu können. Der diagnostische Test in Abbildung 3.8 kann ebenfalls bereits in der 3. Klasse eingesetzt werden, da er nur den Zahlenraum bis 1000 behandelt. Die Schwierigkeitsmerkmale dieses Tests (gekennzeichnet durch die kleinen Zahlen am Rande der jeweiligen Rechnung) wurden auf Grund einer langjährigen Fehleranalyse festgelegt. Bei dieser Analyse wurden diese Merkmale als Auslöser für Fehler bestimmt [vgl. Gerster, 1982, 47 ff].

Die Aufgaben im diagnostischen Test in Abbildung 3.8 sind nach Schwierigkeitsmerkmalen angeordnet. Die Spalten- und Zeilenüberschriften beschreiben den jeweiligen Fehlertyp näher. Beispiel 12 etwa befindet sich in der dritten Reihe und vierten Spalte. Dies bedeutet, dass in diesem Beispiel ein Übertrag und keine Null in den gegebenen Zahlen vorkommt. Hinzu kommt noch die Schwierigkeit, dass nach Berücksichtigung des Übertrags zwei gleiche Ziffern beim höchsten Stellenwert übereinander stehen (siehe (8),(10) bei Beispiel 12).

	- Zahlenraum bis 1000 - Aufgaben ohne bzw. mit Stellenunterschied wechseln zeilenweise			
	keine Null	Null in der oberen Zeile	Null in der unteren Zeile	keine Null in den gegebenen Zahlen
kein Übertrag	1 45 - 32 ----- (7)	2 608 - 203 ----- (7)	3 958 - 104 ----- (7)	4 496 - 136 ----- (7)
	5 479 - 27 ----- (7)	6 309 - 4 ----- (7)	7 867 - 40 ----- (7)	8 687 - 82 ----- (7)
ein Übertrag	9 72 - 49 ----- (3)	10 704 - 262 ----- (3)	11 773 - 407 ----- (3)	12 365 - 258 ----- (8)(10)
	13 849 - 62 ----- (10)	14 690 - 23 ----- (10)	15 657 - 8 ----- (10)	16 295 - 87 ----- (10)
zwei Überträge	17 821 - 788 ----- (9)(4)	18 900 - 439 ----- (9)(4)	19 506 - 207 ----- (9)(4)	20 853 - 459 ----- (9)
	21 613 - 25 ----- (1)	22 604 - 8 ----- (1)(2)	23 703 - 97 ----- (1)(6)	24 994 - 98 ----- (1)(5)(9)
Sonderfälle	25 593 - 739 ----- (12)	26 1000 - 694 ----- (8)(6)	27 43 - 102 ----- (12)	28 951 - 67 ----- (1)(11)

Zusätzliche Schwierigkeitsmerkmale:**in Aufgabe Nr.**

- | | |
|---|----------------------------|
| (1) Übertrag an leere Stelle | 13, 15, 21, 22, 23, 24, 28 |
| (2) Übertrag an leere Stelle und Ergänzen bis 10 | 22 |
| (3) Übertrag zur Null | 11 |
| (4) Übertrag zur Null und Ergänzen bis 10 | 19 |
| (5) Übertrag zur 9 | 24 |
| (6) Übertrag zur 9 und Ergänzen bis 10 | 23, 26 |
| (7) gleiche Ziffern übereinander | 2, 4, 8 |
| (8) ... beim höchsten Stellenwert | 12, 17, 26 |
| (9) ... dazu Übertrag | 19, 20, 24 |
| (10) ... nach Berücksichtigung des Übertrags | 12, 16, 17 |
| (11) untere Ziffer minus Übertrag (gleich obere Ziffer) | 28 |
| (12) obere Zahl kleiner als untere Zahl | 25, 27 |

Abbildung 3.8: Diagnostischer Test für die Subtraktion [nach Gerster, 1982, S. 49]

3.4 Österreichischer Lehrplan

Im Lehrplan der Volksschule (Grundschule) des Bundesministerium für Unterricht, Kunst und Kultur (bmukk) ist zu Beginn folgender Leitsatz zu lesen:

Die Volksschule hat [...] die Aufgabe, an der Entwicklung der Anlagen der Jugend nach sittlichen, religiösen und sozialen Werten sowie nach den Werten des Wahren, Guten und Schönen durch einen ihrer Entwicklungsstufe und ihrem Bildungsweg entsprechenden Unterricht mitzuwirken. [bmukk, 2012]

Außerdem soll die Volksschule eine ausgewogene Bildung fördern und zur Entfaltung der Lernfreude, der Fähigkeiten und der Interessen der Kinder führen. Ebenfalls soll eine Vermittlung und Entwicklung von grundlegenden Kenntnissen und Fertigkeiten erreicht werden. Hierbei ist in bmukk [2012] explizit der „kindgerechte Umgang mit moderner Kommunikations- und Informationstechnologie“ erwähnt. Die individuelle Förderung jedes Kindes und eine kontinuierliche Lernentwicklung der ganzen Klasse sollten gewährleistet sein [vgl. bmukk, 2012, 9 ff].

Im dritten Teil des Lehrplans des bmukk [2012, 25 ff] zum Thema „Allgemeine didaktische Grundsätze“ ist in Bezug auf Individualisierung und Differenzierung folgendes zu lesen:

Individualisierung verlangt von der Lehrerin bzw. vom Lehrer, dass sie bzw. er trotz der vereinheitlichenden Tendenz jedes Klassenunterrichts die Verschiedenartigkeit der kindlichen Persönlichkeiten und ihrer Bedingtheiten ernst nimmt und ihnen zu entsprechen versucht.

In diesem Fall soll jeder Lehrer und jede Lehrerin die unterschiedlichen Entwicklungsstufen und Bildungsvoraussetzungen der Schülerinnen und Schüler berücksichtigen. Dies betrifft im Einzelnen die unterschiedlichen Lerntempi, die Lernbereitschaft und die Lernfähigkeit, sowie ihre Interessen und Vorerfahrungen. Diese Unterschiede soll der Lehrer bzw. die Lehrerin mit differenzierten und individualisierten Maßnahmen ausgleichen. Hierbei ist es sehr wichtig, dass diese Unterschiede erkannt und als Ausgangspunkt des differenzierten Lernangebots genommen werden. Dies hilft eine Über- und Unterforderung zu vermeiden [vgl. bmukk, 2012, 25 ff].

Die schriftliche Addition und Subtraktion sind im Lehrplan der Volksschule im Bereich der 3. Schulstufe zu finden. Unter dem Punkt „Durchführen der Rechenoperationen im Zahlenraum 1000“ und dem Unterpunkt „Schriftliches Rechnen im additiven und multiplikativen Bereich“ findet man folgenden kurzen Absatz zu den beiden schriftlichen Rechenverfahren:

Gewinnen der schriftlichen Rechenverfahren:

- Addieren und Subtrahieren (Ergänzungsverfahren) zwei- und dreistelliger Zahlen

[bmukk, 2012, S. 155]

In Österreich ist es durch den Lehrplan gesetzlich vorgeschrieben, dass Schülerinnen und Schüler der Volksschule die schriftliche Subtraktion mit Hilfe des Ergänzungsverfahrens erlernen. Dies ist insofern fragwürdig, da Forschungsergebnisse zeigen, dass dieses Verfahren von den Lernenden nicht verstanden und nur der Algorithmus gelernt wird [vgl. Brownell und Moser, 1949; Mosel-Göbel, 1988]. Im internationalen Vergleich wird hauptsächlich das Verfahren mit Hilfe der Entbündelungstechnik bevorzugt. Dieses Verfahren ist für die Lernenden einsichtiger und einfacher nachvollziehbar [vgl. Padberg, 1994].

Im nachfolgenden Absatz des Lehrplans sind die Forderungen nach dem Begründen und Verstehen der Rechenschritte und der Operationen, die ihnen zu Grunde liegen, aufgeführt. Ebenso wird das Durchführen von Rechenproben und das Abschätzen von Ergebnissen explizit gefordert. In der 4. Schulstufe wird der Zahlenraum bereits auf 100 000 erweitert und es stehen im additiven Bereich auch Rechnungen mit mehrstelligen Zahlen als Forderung im Lehrplan [vgl. bmukk, 2012, 155 ff].

4 Mathematische Lernsoftware

Im kommerziellen Bereich gibt es für den Mathematikunterricht eine schier grenzenlose Auswahl an verschiedenen Lernprogrammen. Dieses Angebot richtet sich jedoch in erster Linie an den finanzkräftigen Nachmittagsmarkt und ist für Schulen oftmals nicht leistbar. Durch unterschiedlichen Sprachgebrauch und unzureichender Anpassung an das Curriculum sind zudem viele dieser Programme für den täglichen Schulgebrauch ungeeignet. Nur wenige dieser Programme unterstützen einen adaptiven Ansatz und reagieren auf den aktuellen Lernstand der Lernenden. Feedback wird oftmals nur als „richtig“ oder „falsch“ an Schülerinnen und Schüler bzw. die Lehrperson zurückgegeben und die wichtige Komponente der Fehlerdiagnose ist meist nicht oder nur unzureichend vorhanden [vgl. Bender und Didaktik der Mathematik. Arbeitskreis Mathematikunterricht und Informatik, 2003, 16 ff].

4.1 Buggy

Da es sehr schwierig ist, fehlerhafte Denk- und Rechenmuster nur aus den Schülerantworten zu diagnostizieren, entwickelten J. S. Brown und Burton [1978] „BUGGY“. Das computerbasierte Spiel soll Lehrern dabei helfen, systematische Fehler von Lernenden zu erkennen und zu diagnostizieren. BUGGY nimmt die Rolle von Schülerinnen oder Schüler ein, die ein bestimmtes Fehlermuster besitzen. Anhand von Beispielen soll die Lehrperson dieses Fehlermuster erkennen und diagnostizieren. Zu Beginn erhält sie eine Anzahl von Beispielen, die nach diesem Fehlermuster gelöst worden sind. In weiterer Folge kann er dem System Aufgabenstellungen übermitteln, die das System genau nach diesem fehlerhaften Muster löst und an den Aufgabensteller zurückgibt. Glaubt die Lehrperson den Fehler gefunden zu haben, erwartet BUGGY eine kurze Beschreibung des Fehlers. Um wirklich sicher zu gehen, dass der Fehler erkannt wurde, muss die Lehrperson zusätzlich noch fünf gegebene Beispiele nach genau diesem Fehlermuster beantworten. Stimmen alle fünf Antworten mit dem Fehlermuster, überein erhält die Lehrperson einen neuen Fehler.

4.1.1 Diagnostische Modelle

Als Grundgerüst für die Konstruktion der Modelle wurde eine Technik mit dem Namen „procedural networks“ adaptiert und verwendet. Diese Technik wurde erstmals von Sacerdoti [1975] verwendet und folgendermaßen definiert:

It is a strongly connected network of nodes, each of which may contain both procedural and declarative information. The procedural information is used to represent the domain knowledge, as was done by the „procedural embedding“ school. But the plan knowledge is represented declaratively in the contents of the nodes and in the structure of the net itself. [Sacerdoti, 1975, S. 9]

Es handelt sich um ein Netzwerk indem die Knotenpunkte eng miteinander verbunden sind und jeder Knotenpunkt wieder Knotenpunkte und Informationen beinhalten kann. Dieses System wurde von Brown und Burton adaptiert und etwas vereinfacht, um eine Struktur zu erhalten, in der korrekte und inkorrekte Teilschritte einfach modelliert und auch vermischt werden können. Dadurch kann das Model sowohl die Teile erfassen, die korrekt als auch jene die nicht korrekt ausgeführt werden. Ein prozedurales Netzwerk Model für eine arithmetische Operation besteht aus einer Liste von Vorgängen die jeweils einen Knotenpunkt im Netzwerk entsprechen. Durch die Verbindungslinien werden die Beziehungen der Vorgänge zueinander definiert. Jeder dieser Vorgänge besteht aus zwei Teilen. Der konzeptuelle Teil beschreibt die Absicht des Vorgangs, während der operative Teil Methoden bereitstellt, um diese Absicht zu erreichen. Diese Methoden (auch Implementierungen genannt) sind Programme, die festlegen wie die Resultate der verknüpften Vorgänge zusammengefügt werden müssen um genau diese Absicht zu erreichen. Um für einen Vorgang mehrere Möglichkeiten der Abarbeitung bereitzustellen, kann dieser mehrere Implementierungen besitzen. Bei einem Fehler, oder wie J. S. Brown und Burton [1978] es nennen, einen „Bug“ wird mindestens eine dieser Implementierungen durch eine fehlerhafte ersetzt. Dadurch ist es dem System möglich, genau einen gewissen Fehlertyp, der auch aus mehreren fehlerhaften Mustern bestehen kann, zu simulieren. Um Schlüsse aus diesen Daten ziehen zu können, werden alle Fehler mit einem bestimmten Set von Beispielen simuliert und in weiterer Folge die Ergebnisse mit den Ergebnissen der Lernenden verglichen. Aus dieser Auswertung ist es bereits möglich, Schlüsse über bestimmte Fehlermuster zu ziehen. Oft ist es jedoch der Fall, dass Fehler nicht nur aus einem fehlerhaften Teilprozess stammen, sondern eine Kombination von mehreren falschen Teilvorgängen darstellen. Durch die gewählte Struktur ist es möglich, auch diese Fälle einfach zu simulieren. Allerdings sollte bedacht werden, dass es für einen Fehler oftmals mehrere Möglichkeiten gibt [vgl. J. S. Brown und Burton, 1978, 156 ff].

Um ein prozedurales Netzwerk für eine arithmetische Operation zu erstellen, muss diese in möglichst einfache Teiloperationen, so genannte Vorgänge, unterteilt werden. Die Unterteilung für die Addition ist in Abbildung 4.1 zu sehen. Diese Repräsentation unterstützt zwei Methoden. Die Methode „RIGHT-TO-LEFT“ steht für die Standardmethode. Hierbei werden die Spalten von rechts nach links addiert und die

Überträge einfach in die nächste Spalte angeschrieben. Die zweite Methode beginnt mit der linken Seite und geht in weiterer Folge spaltenweise nach rechts. Tritt hier ein Übertrag auf, muss dieser Übertrag unter das Ergebnis in die Spalte vor der aktuellen geschrieben werden. Danach muss eine weitere Addition durchgeführt werden. Bei Betrachten der Teilschritte fällt auf, dass diese Methoden Gemeinsamkeiten beim Summieren der Spalten und beim Schreiben einer Ziffer - jedoch Unterschiede beim Behandeln der Überträge aufweisen. Jeder Teilprozess, der von mehreren Überprozessen verwendet wird, soll explizit definiert werden [vgl. J. S. Brown und Burton, 1978, 156 ff].

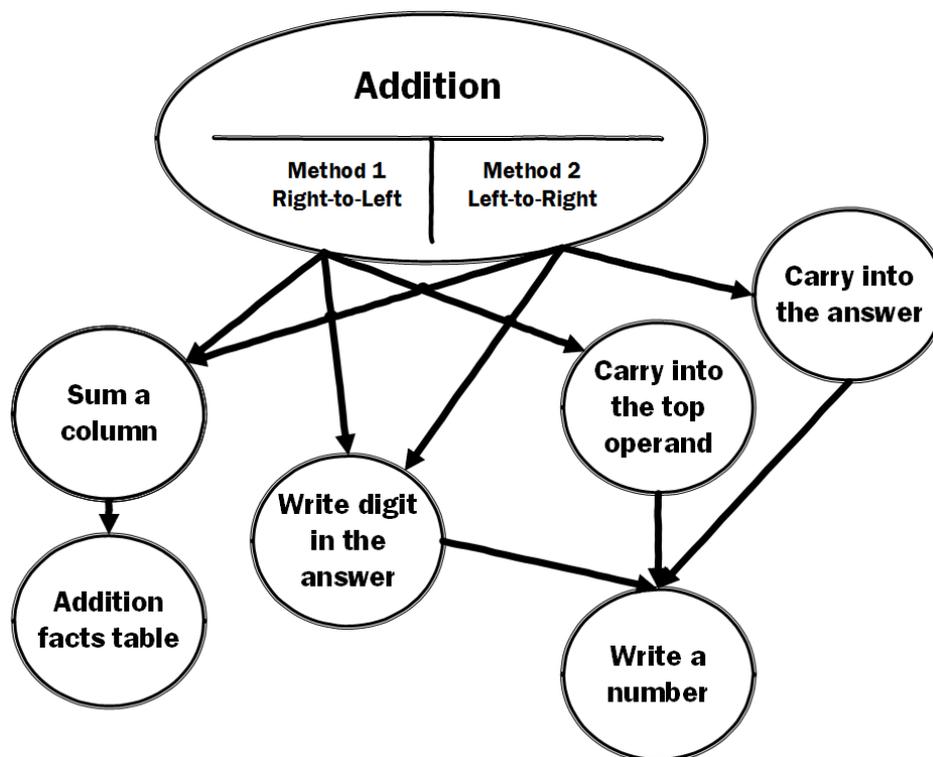


Abbildung 4.1: Zerlegung der Addition in ihre Teiloperationen [aus J. S. Brown und Burton, 1978, S. 160]

4.1.2 Erweiterungen von BUGGY

Da BUGGY nur für den Zweck konzipiert wurde, angehende Lehrer bei der diagnostischen Fehlersuche zu unterstützen und diese zu trainieren, wurde auf Basis von BUGGY eine Erweiterung entwickelt. Diese wurde DEBUGGY getauft und als reine offline Anwendung konzipiert. Zusätzlich wurde eine interaktive Version von DEBUGGY entwickelt - IDEBUGGY (I für interaktiv). Diese Version stellt den Lernenden konkrete Beispiele und führt aufgrund dieser Antworten eine Diagnose durch [vgl. Sleeman und J. Brown, 1982, 157 ff].

4.1.2.1 DEBUGGY

Wie bereits vorher erwähnt, wurde DEBUGGY als reines offline System implementiert. Das bedeutet, dass keine Daten über Kommunikationsschnittstellen eingelesen werden können. Die Daten müssen direkt im System gespeichert sein. Vereinfacht beschrieben hat DEBUGGY die Antworten der Lernenden im Speicher und vergleicht diese mit den Antworten, die auf Grund von Fehlermustern und Kombinationen dieser Fehlermuster berechnet werden. Nach diesem Vergleich wird eine Reihe von Fehlern vorliegen. Daraus wird mit einigen Methoden die beste Theorie gesucht und in weiterer Folge entschieden, ob sie gut genug für eine Diagnose ist [vgl. Sleeman und J. Brown, 1982, 167 ff].

DEBUGGY stellt zum Vergleich mit den Schülerantworten 110 einfache Fehler und 20 kombinierte Fehler, also 130 Fehler zur Verfügung. Durch den Abgleich mit den Benutzerantworten wird aus diesen Fehlern eine Teilmenge herausgefiltert, die mit den Antworten übereinstimmt. Alle diese möglichen Fehler werden zusammengefasst als „initial Hypothesis“. Für den Vergleich verwendet das System ein „full-problem evidence“-Schema. Das bedeutet, dass im Gegensatz zum „single-column evidence“-Schema, nur nach dem gesamten Ergebnis verglichen wird [vgl. Sleeman und J. Brown, 1982, 167 ff].

Alle diese möglichen Fehler eines Schülers werden nun zu neuen Hypothesen kombiniert. Da durch den Vergleich oftmals eine große Anzahl von Fehlern übernommen wird, versucht DEBUGGY diese Fehler noch einmal zu reduzieren bevor die neuen Hypothesen kombiniert werden [vgl. Sleeman und J. Brown, 1982, 167 ff].

Einige Fehler treten im Vergleich auf, da sie in bestimmten Fällen das gleiche Ergebnis liefern wie ein anderer Fehler. Der Fehler $700 - 5 = 705$ lässt auf das Fehlermuster „Addition statt Subtraktion“ schließen, beschreibt aber auch den Fehler „Kleiner von Größer“. Tritt der Fehler „Addition statt Subtraktion“ nur in diesem Beispiel auf und gibt es weitere Beweise für den Auftritt des anderen Fehlers, so wird dieses Fehlermuster in den Hypothesen gelöscht. Eine andere Möglichkeit besteht darin, dass ein Muster eine Generalisierung eines anderen Fehlers darstellt. Hier könnte man die Fehler „ $0 - n = n$ “ betrachten, der eine Generalisierung des „Kleiner von Größer“ Fehlermusters darstellt. Kommt dieser Fehler häufig vor, so wird der darunterliegende Fehler, in diesem Fall „ $0 - n = n$ “, entfernt [vgl. Sleeman und J. Brown, 1982, 167 ff].

Konkret geht die Reduzierung der Fehler nach einem exakten Schema vor. Dafür wurden drei Klassen definiert. Diese sind nach ihrer Wichtigkeit absteigend geordnet folgende:

- **Klasse 3:** Das Ergebnis der Berechnung mit der fehlerhaften Struktur stimmt mit der Benutzerantwort überein.
- **Klasse 2:** Das Ergebnis der Berechnung mit der fehlerhaften Struktur ergibt die korrekte Antwort und die Antwort des Schülers bzw. der Schülerin ist falsch.
- **Klasse 1:** Das Ergebnis der Berechnung mit der fehlerhaften Struktur ergibt ein falsches Ergebnis und stimmt nicht mit der Schülerantwort überein.

Für jeden Vergleich können die Fehler einer dieser drei Klassen zugeordnet werden. Ein Fehler A wird genau dann von den Hypothesen entfernt, falls es einen zweiten Fehler B gibt, der sich bei allen Beispielen in der gleichen oder höheren und mindestens für eines der Beispiele in einer höheren Klasse befindet [vgl. Sleeman und J. Brown, 1982, 168 f].

Nach dieser Reduzierung der Fehlerhypothesen wird versucht, die Fehler so zu kombinieren, um genau das Fehlverhalten einer Schülerin oder eines Schülers zu modellieren. Dabei wird zuerst jeder Fehler mit jedem kombiniert und getestet. Beschreibt das Ergebnis das Verhalten der Schülerin bzw. des Schülers besser als seine beiden erzeugenden Fehlerkomponenten, wird dieser neue Fehler zu den Hypothesen hinzugefügt und wird ebenfalls zur Kombination verwendet. Dies ist bis zu einer Kombination von vier Fehlern möglich. Alle Fehlerkombinationen, die dem Vergleich nicht standhalten, werden verworfen. Um die Performance zu erhöhen und unnötige Rechenleistung zu vermeiden, sind im Netzwerk zusätzlich Informationen gespeichert, welche Fehlermuster kombinierbar sind. Es würde zum Beispiel keinen Sinn machen, das Fehlermuster „Kleiner von Größer“ mit Übertragfehlern zu kombinieren, da bei diesem Fehlermuster nie ein Übertrag auftritt [vgl. Sleeman und J. Brown, 1982, S. 169].

In der nächsten Phase werden im ersten Schritt alle Fehler und Fehlerkombinationen, die noch vorhanden sind, nach einem bestimmten Kriterium gefiltert. Sleeman und J. Brown [1982, 169 ff] verwenden als Kriterium die prozentuale Übereinstimmung mit den Schülerantworten und setzen das Limit auf 40%. Alle Fehler und Fehlerkombinationen, die eine geringere Übereinstimmung besitzen, werden verworfen. Bei Antworten des Schülers die nicht mit den Fehlerantworten übereinstimmen wird in weiterer Folge versucht, mit kleinen Änderungen (coercions) am Verhalten des Fehlermusters genau die Antwort der Schülerin bzw. des Schülers zu erhalten. Dafür stellt das System drei unterschiedlichen Möglichkeiten zur Verfügung:

- **„fact-table errors“**: Dies beschreibt einen Fehler, der durch fehlerhaftes Wissen des kleinen Einsminuseins passiert. Um den Filter nicht zu offen zu gestalten, darf dieser Fehler nur einmal pro Rechnung auftreten und auch nur, wenn das Ergebnis der Spalte sich um maximal zwei vom korrekten Ergebnis unterscheidet ($9 - 3 = 3$ würde nicht mehr in dieses Schema fallen, da die Differenz zur korrekten Lösung drei beträgt).
- **„variants“**: Hierbei werden Probleme an den Rändern der Rechnung unter die Lupe genommen und dafür verschiedene Fehlverhalten simuliert. Darunter zählen zum Beispiel eine zusätzliche Behandlung der Zahl, die am linken Rand steht oder das jede Spalte ohne Zusammenhang behandelt wird.
- **„predicted inconsistencies“**: Diese Änderungen sind fehlerspezifisch und daher von Fehler zu Fehler unterschiedlich. Sie beschreiben Verhalten von Fehlermustern bei konkreten Zahlenkombinationen oder Problemstellungen. Gehen wir zum Beispiel von einem Fehlermuster aus, in dem eine Schülerin oder ein Schüler in jeder

Spalte einen Übertrag berücksichtigt und mit diesem rechnet, egal ob er nötig ist. In speziellen Fällen ist es dann möglich, dass die Differenz einer Spalte größer als neun ist. Manche Lernenden werden in diesem konkreten Problemfall vielleicht die Einerstelle anschreiben und andere werden einfach die gesamte Zahl als Lösung anschreiben. Andere wiederum versuchen den Übertrag in die Spalte, wo sie ihn geborgt haben, zurückzuschreiben und erhalten in diesem Fall sogar die richtige Antwort. Alle diese Möglichkeiten werden von den „predicted inconsistencies“ abgedeckt.

Im letzten Schritt werden die noch zur Auswahl stehenden Fehlermuster klassifiziert. Dies geschieht nach der Anzahl der Übereinstimmungen mit den Schülerantworten, dem Typ der falschen Vorhersage und der Anzahl der Änderungen (coercions), die notwendig waren um die Schülerantwort zu erreichen. Die Klassifizierung erfolgt in eine der vier Kategorien: ständiger Fehler; ständiger Fehler, aber mit zusätzlichen Symptomen; fehlerhaftes Verhalten, aber nicht ständig; unsystematisches Verhalten. Die Fehler, die in der höchsten nicht leeren Kategorie zu finden sind, werden ein letztes Mal verglichen und das Ergebnis als Diagnose angenommen [vgl. Sleeman und J. Brown, 1982, S. 171].

4.1.2.2 IDEBUGGY

DEBUGGY ist ein reines offline System und konnte nicht mit Schülerinnen oder Schülern interagieren. Um dies zu ermöglichen wurde IDEBUGGY als online System von DEBUGGY entwickelt. Durch die Möglichkeit, interaktive Analysis zu betreiben, können die Ergebnisse schneller und genauer ermittelt werden. Dies liegt daran, dass die Beispiele auf die Lernenden zugeschnitten werden können. Viele interne Routinen kann IDEBUGGY von DEBUGGY übernehmen. Zusätzlich muss die interaktive Version die Generierung von Beispielen unterstützen und die logischen Gleichheiten erkennen.

IDEBUGGY gibt den Lernenden eine Reihe von Beispielen aus und nutzt deren Antworten um Hypothesen aufzustellen. Durch die Generierung spezieller Beispiele, welche die aktuelle Hypothese widerlegen oder bestätigen, kann eine gezielte Diagnose durchgeführt werden. Sind genug Beweise für eine Hypothese vorhanden und gibt es keine andere Hypothese die plausibel erscheint, wird eine Diagnose gestellt.

4.2 Mathematik heute - Bruchrechnung

Dieses Lernprogramm wurde vom Schroedel Verlag zusammen mit der Universität Hildesheim im Jahr 2002 entwickelt und ist vor allem als begleitende Software für das Schulbuch „Mathematik heute“ der sechsten Schulstufe gedacht. Gedacht ist die Software, um den Umgang mit Brüchen und deren arithmetischen Operationen zu festigen (6-7 Schulstufe). Zusätzlich führt das Programm eine detaillierte Fehleranalyse und Protokollierung durch [vgl. Schumann, 2013, 5 ff].

Beim Programmstart ist es möglich, eine bereits vorhandene Protokolldatei zu laden oder eine neue anzulegen. Danach erhält man eine Übersicht der vorhandenen Möglichkeiten um sein Wissen zu testen und zu erweitern. Diese Übersicht enthält zum Beispiel die vier Grundrechnungsarten für Brüche oder auch das Erweitern und Kürzen von Brüchen. Anhand der verschiedenen Gruppen kann die Benutzerin bzw. der Benutzer gezielt Themenbereich und Schwierigkeitsgrad auswählen. Nach dem Klick auf einen Bereich gelangt er zur „Aufgabenseite“. Diese ist in Abbildung 4.2 zu sehen und besteht aus einem Bearbeitungsfenster auf der linken und einem Informationsfenster auf der rechten Seite. Die Eingabe der Zwischenschritte und Lösungen erfolgt über den Formeleditor im Bearbeitungsfenster [vgl. Schumann, 2013, 6 ff].

The screenshot shows a software window titled "Addiere die Brüche:". On the left is a form editor with a toolbar and an "OK?" button. The editor contains the following steps for adding $\frac{1}{11} + \frac{3}{5}$:

$$\frac{1}{11} + \frac{3}{5}$$

$$\neq \frac{1+3}{11+5}$$

$$= \frac{1}{18} \frac{4}{4}$$

$$= \frac{1}{4}$$

The second line is crossed out with a red diagonal line. On the right is an information window with a cartoon character and the following text:

Fehler in der 2. Zeile:
 Du verwendest eine falsche Regel bei der [Addition](#) der Brüche.
 Beachte, dass du die Zähler nur addieren darfst, wenn die Nenner [gleichnamig](#) sind.
 Korrigiere deine Rechnung.

At the top right of the window are buttons for "Hilfe", "Inhalt", and "Lexikon".

Abbildung 4.2: Aufgabenseite nach Validierung der Lösung [aus Schumann, 2013, S. 7]

Nach der Eingabe der Lösung und dem Betätigen des „OK“-Buttons gibt die Software eine detaillierte Rückmeldung zum Lösungsvorgang aus. Fehlerhafte Zeilen werden, wie in Abbildung 4.2 zu sehen, rot markiert und im Informationsfenster wird eine Beschreibung des Fehlers ausgegeben. Danach haben die Lernenden die Möglichkeit, ihre Ergebnisse mit Hilfe des Textes im Informationsfenster zu korrigieren. Sollte dies ebenfalls misslingen, steht die Möglichkeit bereit, die Musterlösung in einzelnen Teilschritten abzurufen. Zusätzlich steht im Programm ein Lexikon mit einer Reihe von wissenswerten Informationen zum Thema Bruchrechnung zur Verfügung. Im Informationsfenster werden Begriffe aus dem Lexikon immer mit einem Hyperlink auf deren Einträge dargestellt (ebenfalls in Abbildung 4.2 zu sehen) [vgl. Schumann, 2013, 6 ff].

Schritt 2

Wählen Sie nun die Schwierigkeitsfaktoren der Aufgaben aus, die Ihre Schülerinnen und Schüler speziell bearbeiten sollen.

In der oberen Liste können Sie festlegen, wie sich bei Aufgaben mit Brüchen oder gemischten Schreibweisen die **Nenner** zueinander verhalten sollen.

Die mittlere Auswahlliste ermöglicht Ihnen festzulegen, ob die Ihren Schülerinnen und Schüler gestellten Aufgaben im Ergebnis **kürzbar** sein sollen.

Wählen Sie in der unteren Auswahlliste aus, ob die Ergebnisse in natürliche Zahlen oder gemischte Schreibweisen **umgewandelt** werden müssen.

Schwierigkeitsfaktoren:

Verhältnis der Nenner:

Gemeinsame Nenner
 Hauptnennerbildung

Kürzbarkeit des Ergebnisses:

Nicht kürzbar
 Kürzbar

Umwandlung des Ergebnisses:

Nicht umwandelbar
 Umwandlung nötig

Detaillierter Modus

Abbildung 4.3: Konfiguration der Schwierigkeitsfaktoren [aus Schumann, 2013, S. 7]

Die Software umfasst eine Aufgabensammlung von 40 000 Aufgaben, die nach Art und Schwierigkeit sortiert sind. Der Lehrperson ist es möglich, mit Hilfe eines eigenen Programms die Schwierigkeitsniveaus selbst zu konfigurieren. Es können Schwierigkeitsniveaus angelegt, verändert und gelöscht werden. Niveaus bestehen aus der Definition des Aufgabentyps und der Zuordnung von Schwierigkeitsfaktoren. Bei Aufgabentypen stehen als Operationen jeweils die vier Grundrechnungsarten bereit und es kann zusätzlich gewählt werden, ob Brüche auch als gemischte Zahlen angegeben werden. Im zweiten Schritt können Schwierigkeitsfaktoren zu einem Niveau hinzugefügt werden. Dabei stehen das Verhältnis des Nenners, die Kürzbarkeit und die Umwandlung des Ergebnisses zur Auswahl (siehe Abbildung 4.3). Der Aufgabenpool, mit seiner beachtlichen Anzahl an Beispielen, kann leider nicht erweitert werden [vgl. Schumann, 2013, 6 ff].

4.2.1 Fehlerbehandlung

Die Software validiert die Eingaben der Benutzer und gibt ihnen ein ausführliches Feedback. Dies ist durch eine detaillierte Analyse, die häufige Fehlermuster aufspürt, möglich. Zu dem können durch diese Analyse sogar Fehlermuster erkannt werden, die eine Kombination mehrerer Fehler darstellen. Wird ein Fehler erkannt, reagiert das Programm mit einem „Therapieversuch“ indem es versucht, die nötigen Informationen für die Lösung des Beispiels im Informationsfenster auszugeben. Das Fehlerprotokoll und die Auswertungen der Fehler werden in Protokolldateien gespeichert und können von der Lehrerin oder dem Lehrer mit einem speziellen Programm angesehen werden (siehe Abbildung 4.4).

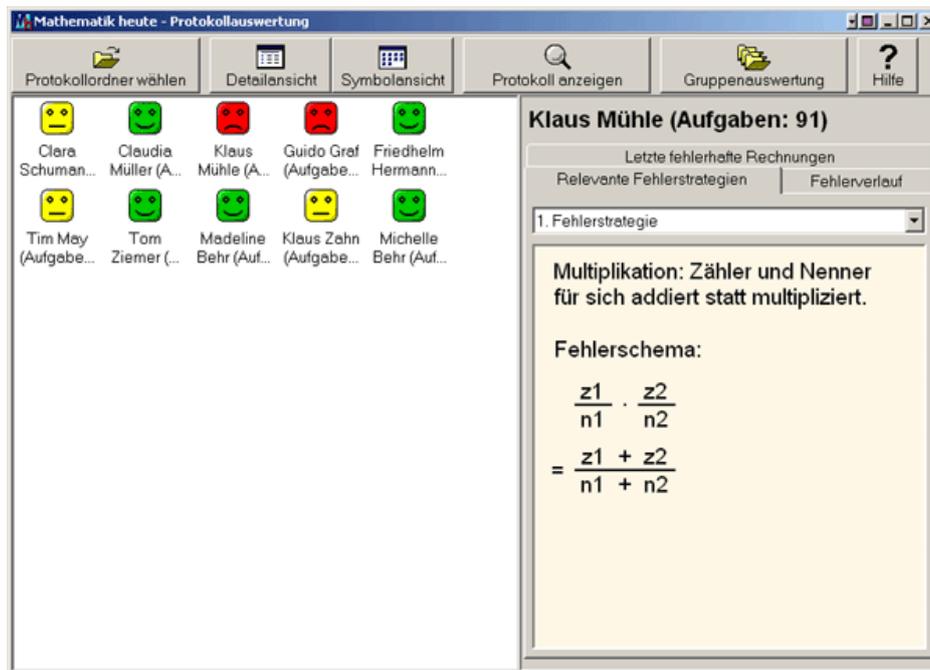


Abbildung 4.4: Programm für Lehrer um die Statistik der Schüler abzurufen [von Stiftung Universität Hildesheim, 2013]

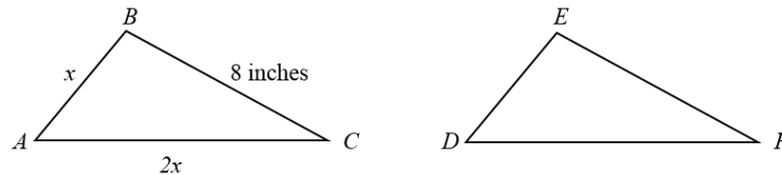
4.3 ASSISTments

Da Lehrpersonen vor allem in Mathematik oftmals wenig Zeit haben, um ihren Stoff an die Lernenden zu bringen, müssen sie sich sehr oft entscheiden, ob sie die Schülerinnen und Schüler unterstützen (assisting) oder ob sie die Leistungen und Kompetenzen einschätzen oder bewerten (assessing). Das web-basierende E-Learning System „ASSISTment“ versucht diese beiden Dinge miteinander zu kombinieren. Mit Hilfe von Lösungshilfen soll sollen die Lernenden gezielt unterstützt und anhand seiner Antworten bewertet werden [vgl. Feng et al., 2006, S. 307].

Im Schuljahr 2004/2005 wurde das System erstmals an Schulen eingesetzt. Über 600 Schülerinnen und Schüler nutzten das System mindestens jede zweite Woche. Dies war durch die Unterstützung von acht MathematiklehrerInnen aus zwei Schulen in Massachusetts möglich. Die Aufgaben, die das Programm den Schülern bereitstellt, stammen aus dem *Massachusetts Comprehensive Assessment System* (MCAS). Alle diese Aufgaben wurden überarbeitet und zu so genannten „ASSISTments“ erweitert indem die Tutoring Komponente hinzugefügt wurde. In diesen Entwicklungsprozess wurden die Lehrpersonen in einem sehr großen Maße involviert. Wenn eine Schülerinnen oder Schüler eine falsche Antwort an das System senden, erhalten sie dafür eine genaue interaktive Schritt für Schritt Erklärung. Durch die Verbreitung über das Internet ist keine Installation

oder Wartung für die Lehrpersonen notwendig [vgl. Feng et al., 2006; Razzaq et al., 2005].

Triangles ABC and DEF shown below are congruent.



The perimeter of $\triangle ABC$ is 23 inches. What is the length of side \overline{DF} in $\triangle DEF$?

Abbildung 4.5: Frage 19 aus dem 2003 Massachusetts Comprehensive Assessment System [aus Razzaq et al., 2005]

Jede Frage im System besteht aus einem Originalteil und einer Liste von Fragen, welche die Frage in Teilfragen zerlegen und dadurch versuchen auf das richtige Ergebnis bzw. den richtigen Lösungsweg hinzuführen. Diese Fragen werden nur für den Fall angezeigt, dass der Schüler eine falsche Antwort eingibt. In Abbildung 4.5 sehen wir den Originalteil der Frage 19 aus dem Fragenkatalog 2003. Gibt die Schülerin oder der Schüler, so wie in Abbildung 4.6, eine falsche Antwort ein, wird die erste Frage aus der Liste angezeigt. Eine Änderung der aktuellen Antwort ist zu diesem Zeitpunkt nicht mehr möglich. In den nächsten Schritten soll die Liste der Fragen beantwortet werden. Zusätzliche Hilfestellung kann durch Betätigen des „Hint“-Buttons angefordert werden. Wird eine Frage erfolgreich gelöst, wird die nächste angezeigt, so lange bis alle Fragen der Liste durchgearbeitet wurden und die Schülerin oder der Schüler nun das richtige Ergebnis erhalten hat. Ist eine Frage nicht lösbar, so kann die Antwort durch erneutes Klicken des „Hint“-Buttons angezeigt werden.

In Abbildung 4.6 sehen wir zusätzlich solche „Hints“ auf der rechten Seite angezeigt. Können die Lernenden die erste Frage nicht lösen, so wird ihnen nach der Betätigung des Buttons eine an das Beispiel angepasste Erklärung geliefert. Mit Hilfe dieser sollten sie das Beispiel nun lösen können. Geben Lernende eine falsche Antwort bei einer Frage, so erhalten sie spezifisches Feedback für diese Antwort (siehe Buggy message in Abbildung 4.6) [vgl. Razzaq et al., 2005].

The screenshot displays the ASSISTments interface for a math problem. The main question asks for the length of side DF in triangle DEF, with a text input field containing '23'. Below this, a multiple-choice question asks which side of triangle ABC has the same length as side DF, with radio buttons for AB, BC, and AC (selected). The next question asks for the perimeter of triangle ABC, with radio buttons for $\frac{1}{2} * 8x$, $2x + 8$, $2x + x + 8$ (selected), and $\frac{1}{2} * x(2x)$. A text input field contains '5'. The problem then provides the perimeter of triangle ABC as 23 inches and asks for the value of x, with a text input field containing '10'. The final question asks for the length of side AC, with a text input field containing '10'. The interface includes several help elements: a 'Messages' window showing a diagram of two congruent triangles with corresponding sides colored and labeled; a 'Buggy message' that appears if a student selects $\frac{1}{2} * 8x$, stating 'No. You might be thinking that the area is 1/2 base times height, but you are looking for the perimeter.'; and a 'Hint' window showing the calculation $AC = 2x$, $x = 5$, $AC = 2 * 5$, and $AC = 10$. Arrows indicate the flow from the original question to the hints and error messages.

Abbildung 4.6: Tutoring Beispiel für Frage 19 mit Hilfen und Fehlermeldungen [aus Razzaq et al., 2005]

Zusätzlich führt das System Auswertungen durch und stellt für die Lehrpersonen Reports zur Verfügung. Mit Hilfe dieser Reports können eine Reihe von Fragen beantwortet werden wie Razzaq et al. [2005] beschreibt:

Which items are my students finding difficult? Which items are my students doing worse on compared to the state average? Which students are 1) doing the best, 2) spending the most time, 3) asking for the most hints etc.? Which of the approximately 80 skills that we are tracking are students doing the best/worst on? What are the exact actions that a given student took?

Um diese Reports zu erstellen, müssen einige Daten gesammelt werden. Dies geschieht durch ein Java-basierendes System, welches jede Schüler Aktion aufzeichnet und als XML an einen Server schickt. Die Nachricht wird dort verarbeitet und in einer Datenbank gespeichert. Aus diesen Daten werden in weiterer Folge die Reports generiert. Der am meisten genutzte Bericht ist das „Grade Book“ - das Notenbuch. In diesem tabular aufgebauten Report wird jede Schülerin und jeder Schüler in einer Zeile dargestellt. Als Informationen werden die totale Nutzung von Assistment in Minuten, die Minuten, die heute hier verbracht worden sind, die Anzahl der Probleme und den Prozentsatz der korrekt gelösten Fragen angezeigt. Zusätzlich wird eine vom System vorhergesagte Punkteanzahl für den MCAS sowie ein Performancelevel angezeigt. Als Übersicht über die verschiedenen genutzten Hilfen werden zusätzlich die Anzahl der Teilfragen, die Performance der Lernenden bei diesen Teilfragen und die Anzahl der „Hints“ angezeigt. Weitere Reports, wie zum Beispiel der Klassenübersichts-Report oder der Klassen Fortschritts-Report werden ebenfalls vom System zur Verfügung gestellt.

4.4 Zusammenfassung

Jedes dieser vorgestellten Systeme weist sowohl Vor- als auch einige Nachteile auf. Das Konzept von BUGGY ist bereits sehr weit entwickelt und das diagnostische Model wurde sehr gut ausgearbeitet. Leider ist die Software (weder BUGGY, DEBUGGY noch IDEBUGGY) nicht zugänglich und kann daher von Lernenden nicht verwendet werden.

Mathematik heute besticht vor allem durch eine gute und detaillierte Fehleranalyse bei arithmetischen Operationen mit Brüchen und bietet zusätzlich einen virtuellen Tutor an. Leider ist diese Software kostenpflichtig und nicht als Webapplikation verfügbar. Zusätzlich wird keine automatische Adaption des Programms durchgeführt, was bedeutet, dass sich Mathematik heute nicht am Lernniveau der Benutzerin bzw. des Benutzers orientiert.

Als letzte Software wurde ASSISTment vorgestellt. Diese ist als Webapplikation frei zugänglich und verfügt ebenfalls über eine Tutoring-Komponente. Das Erstellen neuer eigener Assistenten ist für die Lehrperson allerdings aufwendig und zudem bietet das System keine detaillierte Fehlerdiagnose an.

Der Plusminus-Trainer versucht insbesondere die positiven Eigenschaften der oben angeführten Programme umzusetzen und die erwähnten Defizite zu beheben. Der Trainer wird für diese erste Version ohne eine Tutoring-Komponente entwickelt. Durch den modularen Aufbau und die Offenheit des Programms wäre dies in Zukunft jedoch möglich. Hauptaugenmerk soll auf das angemessene Design für Grundschülerinnen und -schüler und auf eine detaillierte Fehleranalyse gelegt werden. Hierbei soll die Erweiterung von Fehlertypen zu jeder Zeit ohne übermäßigen Aufwand möglich sein. Zudem wurde der Trainer als Webapplikation entwickelt und steht dadurch auf jedem internetfähigen Gerät zur Verfügung. In weiterer Folge werden auch mobile Applikationen entwickelt, welche auf diesen Trainer zugreifen sollen.

5 Technische Umsetzung

Dieses Kapitel beschäftigt sich mit der technischen Umsetzung des Prototypen. Der Name „Plusminus-Trainer“ wurde auf Grund seines Einsatzgebiets gewählt und soll den Benutzerinnen und Benutzern schnell einen kurzen Überblick über die Möglichkeiten bieten. Die Konzeption des Prototyps wurde nach einem Monat abgeschlossen und die Implementierungsphase gestartet. Diese dauerte zwei Monate und resultierte in einer stabilen Anwendung des Trainers in der Version 1.0. Obwohl auch jetzt noch immer wieder kleine Änderungen vorgenommen werden, deckt diese Version den gesamten erforderlichen Bereich ab und kann bereits verwendet werden.

Der Prototyp wurde als Webanwendung entwickelt und ist auf allen gängigen Browsern lauffähig. Dies gewährleistet eine hohe Verfügbarkeit und hat den Vorteil der zentralen Wartung. Der Plusminus-Trainer liegt, wie andere Lernapplikationen, auf dem Server der TU Graz und ist über die Url <http://plusminus.tugraz.at> erreichbar. Wie bei allen anderen Lernanwendungen, muss sich die Benutzerin bzw. der Benutzer vor der erstmaligen Verwendung beim zentralen Usermanagement registrieren. Dies ist notwendig, um eine exakte Zuordnung von Schülern und Lehrern zu gewährleisten. Die Nutzung dieses Angebots der TU Graz ist vollkommen kostenlos und Links zu den einzelnen Trainern können unter der Url <http://mathe.tugraz.at> gefunden werden.

In den weiteren Abschnitten werden die Konzeption sowie der technische Aufbau des Plusminus-Trainers näher erläutert. Dies umfasst die verwendeten Technologien bis hin zum Datenbankdesign und soll Aufschluss über die Arbeitsweise des Trainers geben.

5.1 Verwendete Technologien

Um eine problemlose Verbreitung und Plattformunabhängigkeit zu gewährleisten, wurde hauptsächlich auf Technologien zurückgegriffen, die unter einer OpenSource Lizenz stehen. Alle verwendeten Technologien sind frei zugänglich und können ohne Probleme bezogen werden. Die nächsten Abschnitten sollen einen kurzen Überblick über diese Technologien verschaffen.

5.1.1 PHP

Für die Programmierung des Prototypen wurde die Scriptsprache PHP¹ verwendet. PHP wurde hauptsächlich dafür entwickelt, dynamische Webseiten zu erstellen. Der Name selber ist ein rekursives Akronym und steht für **PHP Hyerptext Preprocessor**. Für die Entwicklung des Prototypen wurde PHP 5 in der Version 5.4 verwendet. PHP 5 hat im Gegensatz zu den vorherigen Versionen durch die Zend Engine II ein verbessertes Objektmodell. Dies ist für die Effizienz der Ausführung von objektorientierten Anwendungen, so wie dem Prototyp, ein enormer Vorteil.

Die Wahl für diese Implementierungssprache wurde aufgrund einiger Vorteile getroffen. PHP steht unter der PHP License v3.01², einer Opensource-Lizenz [vgl. The PHP Group, 2013] und garantiert daher eine problemlose Verbreitung von Projekten in PHP. Ein weiterer Vorteil liegt darin, dass PHP vollkommen plattformunabhängig arbeitet und auf jedem Server (bei korrekter Installation) verwendet werden kann. Um PHP-Files in HTML-Dateien umzuwandeln, wird lediglich ein PHP Parser benötigt. Für das Empfangen und das Versenden ist allerdings ebenfalls ein eigener Webserver (zum Beispiel: Apache³, IIS⁴,...) von Nöten [vgl. Lerdorf et al., 2006, S. 1].

PHP kann direkt in HTML eingebunden werden und wird bei jedem Zugriff auf diese Webseite geparkt. Der PHP Code wird direkt am Server interpretiert und generiert dann den erwarteten Output für die Benutzerin bzw. den Benutzer (zum Beispiel: HTML, PDF, et cetera). Im Fall unseres Prototypen werden nur HTML-Dateien generiert und an den Client übersendet [vgl. Welling und Thomson, 2003, S. 2].

5.1.2 Zend Framework

Zend Framework⁵ ist ein modulares und stabiles PHP Framework. Es benötigt mindestens PHP Version 5.3 und ist zur Zeit in der Version 2 verfügbar. Das Framework steht unter der *New BSD Licence*, einer Opensource-Lizenz und ist daher frei verwendbar [vgl. Zend Technologies Ltd., 2013a]. Für den Prototypen wird die Version 2.1.3 des gesamten Frameworks verwendet.

Ein großer Vorteil für die Entwicklung ist die effiziente und stabile MVC-Implementation von Zend [vgl. Zend Technologies Ltd., 2013b]. MVC steht für Model - View - Controller und ist ein Strukturmuster der Softwareentwicklung.

¹<http://php.net> [Zugriff am 09.07.2013]

²http://www.php.net/license/3_01.txt [Zugriff am 09.07.2013]

³<http://httpd.apache.org> [Zugriff am 09.07.2013]

⁴<http://www.iis.net> [Zugriff am 09.07.2013]

⁵<http://framework.zend.com> [Zugriff am 09.07.2013]

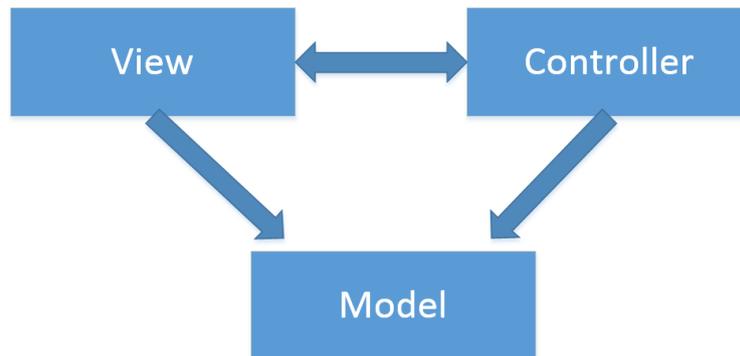


Abbildung 5.1: MVC Structure

Der Controller ist der zentrale Punkt der den Request von der View entgegen nimmt. Er verarbeitet ihn und interagiert mit dem Model. Das Model ist meist ein Set von Klassen, welche die Daten und die Business Roles für das Ändern, Hinzufügen und Löschen der Daten beschreiben [vgl. Galloway et al., 2011, S. 2]. Die View bekommt in weiterer Folge dieses Model vom Controller zurück und kann die Daten anzeigen. Durch diese Trennung von Business Logik (Model) und Darstellungslogik (View) ist es möglich, die Darstellung sogar während der Laufzeit auszutauschen und die Business Logik unabhängig vom Controller oder der View zu testen [vgl. Eilebrecht und Starke, 2010, S. 77-79].

5.1.3 MySql

MySql⁶ ist ein weitverbreitetes relationales Datenbankverwaltungssystem der Firma Oracle Corporation. Laut Oracle Corporation [2013a] selbst verzeichnet das System mehr als 65 000 Downloads am Tag und wird von vielen großen Unternehmen wie Facebook oder Google eingesetzt [vgl. Oracle Corporation, 2013b]. MySql steht als Opensource-Software zur Verfügung und kann direkt über die MySql-Webseite bezogen werden.

Der Prototyp verwendet MySql als Backend, um die gesammelten Daten zu speichern und abzurufen. Vor allem im Zusammenspiel mit PHP ist MySql ein sehr mächtiges Werkzeug. Um eine einfache Konfiguration und Wartbarkeit der Datenbank zu gewährleisten, gibt es das freie Softwaretool phpMyAdmin⁷. Die in PHP geschriebene Software bietet eine einfache grafische Oberfläche zur Verwaltung und Manipulation von Datenbanken, deren Tabellen und Daten.

⁶<http://www.mysql.com> [Zugriff am 09.07.2013]

⁷<http://www.phpmyadmin.net> [Zugriff am 09.07.2013]

5.1.4 Doctrine

Da MySQL (vgl. 5.1.3) ein relationales Datenbanksystem bereitstellt und der Prototyp mit Hilfe des Zend Framework (5.1.2) mit Objekten arbeitet, wird ein weiterer Zwischenlayer benötigt, der Objekte in relationale Daten umwandelt. Solche Frameworks oder Software nennt man Object Relational Mapper (ORM). Die Aufgabe eines solchen Frameworks ist es, Objekte, deren Beziehungen und Polymorphien performant auf eine Tabellenlandschaft abzubilden [vgl. Eilebrecht und Starke, 2010, S. 116]

Der Plusminus-Trainer verwendet als Zwischenlayer doctrine⁸ mit dem ORM und dem DBAL-Modul in der Version 2.3. Das DatabaseAbstractionLayer(DBAL)-Module stellt eine vereinheitlichte Schnittstelle zwischen der Datenbank und dem Programmcode da. Durch diese Schnittstelle ist es möglich, die Datenbank ohne Änderung des Programmcodes auszutauschen. Das ORM-Modul baut auf dem DBAL-Modul auf und verwendet dieses. Ein enormer Vorteil von doctrine ist, dass es Lazy Loading unterstützt.

Unter Lazy Loading versteht man, dass verknüpfte Objekte erst dann von der Datenbank geladen werden, wenn sie wirklich benötigt werden. Im Fall des Plusminus-Trainers ist dies sehr wesentlich, da sonst immer der ganze Objektbaum geladen werden müsste. Doctrine verwendet einen sogenannten Proxy, der bei Anfrage die benötigten Daten von der Datenbank nachlädt [vgl. Fowler, 2012, S. 179-180].

Außerdem besitzt doctrine mit seinem Migrations-Modul eine einfache und performante Lösung um Datenbankänderungen leicht vom Testsystem auf das Produktivsystem zu übertragen. Das Migration-Module ist zwar erst in der Version 2.0 (Alpha) verfügbar, macht aber trotz seines Alpha Status einen sehr guten Job. Mithilfe dieses Modules ist es möglich, die Datenbankzustände zu versionieren und so immer nur die aktuellsten Änderungen in das Produktivsystem einzuspielen. Daher muss nicht bei jedem Update die komplette Datenbank neu aufgesetzt werden.

5.1.5 CSS und HTML

Um eine kindgerechte und einfache Oberfläche im Front-End des Plusminus-Trainers zu erreichen, wurden Cascading Style Sheets⁹ (CSS) und die Hyertext Markup Language (HTML) verwendet. HTML ist eine textbasierte Auszeichnungssprache, um Webinhalte zu strukturieren. Dies geschieht meistens durch Überschriften, Paragraphen, Tabellen und andere Elemente. Mit Hilfe von CSS wird diese Darstellung verändert. CSS trennt damit den Inhalt einer Webseite von seiner visuellen Repräsentation [vgl. S. Mavrody und N. Mavrody, 2012, S. 10].

⁸<http://www.doctrine-project.org> [Zugriff am 09.07.2013]

⁹<http://www.w3.org/Style/CSS/> [Zugriff am 09.07.2013]

Das Design des Plusminus-Trainers wurde größtenteils vom Mathemulti-Trainer übernommen, um den Benutzerinnen und Benutzern eine einheitliche Darstellung zu gewährleisten. Sollten noch Änderungen im Design notwendig sein, so ist dies einerseits durch die Trennung von Business Logik und View durch das MVC Pattern und andererseits durch die Trennung von Inhalt und Design mit HTML und CSS sehr einfach zu bewerkstelligen.

5.1.6 jQuery

Um Elemente auf einer Website zu animieren oder den Fokus auf ein Eingabefeld zu legen, braucht man als Entwickler JavaScript (JS). Da solche Animationen in JavaScript sehr umständlich umzusetzen sind, entstanden bald eine Reihe von JS Frameworks, die den Programmierer bei seiner täglichen Arbeit unterstützen.

Im Plusminus-Trainer wird als JS Framework jQuery¹⁰ in der Version 1.9.1 verwendet. Es ist ein OpenSource-Framework [vgl. jQuery Foundation, 2013] und unterstützt den Programmierer beim Finden, Manipulieren und Animieren von Document Object Model (DOM) Elementen, beim Einfügen oder Verändern von Inhalten und beim Behandeln von Events [vgl. Bongers und Vollendorf, 2011, S. 17-19].

5.2 Konzept

Vor der Umsetzung des Plusminus-Trainers wurde ein ausführliches Konzept erstellt. Der Trainer soll für Schülerinnen und Schüler der dritten bis fünften Schulstufe entwickelt werden. Im österreichischen Kontext bedeutet dies die letzten beiden Klassen der Grundschule und die erste Klasse der Hauptschule. Um Kinder dieses Alters anzusprechen, muss auch beim grafischen Konzept auf diese Altersgruppe eingegangen werden. Der Trainer soll die schriftliche Addition sowie Subtraktion mit ein bis drei Stellen unterstützen.

Um eine stetige Motivation der Schülerinnen und Schüler zu gewährleisten, wurde ein Kompetenzsystem eingeführt. Mit Hilfe dieses Systems wird zu Beginn eine sehr leichte Rechnung aus dem ersten Kompetenzlevel präsentiert. Wenn diese erfolgreich bearbeitet wurde, wird die Schwierigkeit kontinuierlich erhöht. Um diesen Anstieg weder zu schnell noch zu langsam zu gestalten, wurden zusätzlich zwei Phasen definiert. Die Einstufungsphase wird nach dem erstmaligen Login gestartet. In dieser Phase führt jede richtige Antwort sofort zu einem Anstieg des Kompetenzlevel. Die Lernenden verbleiben so lange in dieser Phase, bis sie zum ersten Mal eine falsche Antwort eingeben. Diese Phase hat den Sinn, dass gute Schülerinnen und Schüler einen schnelleren Anstieg erreichen und nicht mit einfachen Beispielen demotiviert werden. Danach wird in die Trainingsphase

¹⁰<http://jquery.com> [Zugriff am 09.07.2013]

übergegangen. Hierbei erreicht der Schüler oder die Schülerin erst nach einer, vom Lehrer oder vom System festgelegten Anzahl, das nächste Kompetenzlevel. Diese Phase soll das Üben und Lernen unterstützen. Bei einer falschen Antwort wird sein Kompetenzlevel verringert, um ihn nicht konstant mit schweren Beispielen zu überfordern.

Um eine Monotonie in den Übungsphasen zu verhindern, wurden mehrere Vorkehrungen getroffen. Additionen und Subtraktionen werden in gemischter Reihenfolge präsentiert. Die Auftrittswahrscheinlichkeit der Operationen kann bei Schülern, die einer Klasse zugeordnet sind, vom Lehrer und bei anderen Benutzern selbst festgelegt werden. Zusätzlich werden in der Datenbank zwei Parameter definiert, die Wahrscheinlichkeiten angeben, um Beispiele aus einem höheren oder niedrigeren Kompetenzlevel zu erhalten.

Sollte eine Schülerin oder ein Schüler stetige Probleme bei einem Kompetenzlevel haben, so muss die Lehrperson eingreifen. Hierbei steht ihm eine detaillierte statistische Auswertung über die Rechnungen und das Fehlerverhalten des Schülers bzw. der Schülerin zur Verfügung. Dies kann sie nutzen, um Fehlermuster zu finden und den Schüler bzw. die Schülerin bestmöglich zu unterstützen und Hilfestellungen zu geben.

Die Aufteilung der Rechnungen in die einzelnen Kompetenzgruppen und Kompetenzlevel, die in den Kapiteln 5.2.1 und 5.2.2 beschrieben wird, ist vollkommen dynamisch und kann jederzeit geändert werden. Auch die Fehleranalyse (nähere Informationen in Kapitel 5.5.3) wurde dynamisch implementiert und bietet die Möglichkeit, zur Laufzeit weitere Fehlertypen hinzuzufügen.

5.2.1 Addition

Alle Additionen mit Summanden zwischen 1 und 999 wurden in Kompetenzlevel eingeteilt. Übergeordnet wurden noch Kompetenzgruppen zur besseren Übersicht und Kategorisierung angelegt. Alle Gruppen und Level sind in Abbildung 5.1 zu sehen. In vertikaler Richtung wurden die Kompetenzgruppen aufgetragen, und die horizontale Spalte gibt die Stellen bzw. die Additionsmuster an. Hierbei steht „n“ für eine natürliche Zahl. Die eingekreisten Zahlen in den einzelnen Zellen stellen das jeweilige Kompetenzlevel dar.

	$n + n$	$n + nn$	$nn + nn$	$n + nnn$	$nnn + nnn$	$nn + nnn$
Ohne Übertrag	①	②	③	④	⑤	⑥
Mit einfachem Übertrag	⑦	⑧	⑨	⑩	⑪	⑫
Mit Übertrag		⑬	⑭	⑮	⑯	⑰

Tabelle 5.1: Einteilung der Kompetenzlevel für die Addition.

Diese Reihung wurde aufgrund der Ergebnisse von Padberg [2005, S. 214] und Gerster [1982, S. 23-27] vorgenommen (siehe auch Abschnitt 3.1.2 „Schwierigkeiten und Fehler“). Die Kompetenzgruppen wurden anhand der Überträge folgendermaßen definiert:

- **Ohne Übertrag:** Bei Rechnungen dieser Gruppe finden keine Überträge statt. Dies sind jene Rechnungen die keine Teilsumme der schriftlichen Addition größer als neun aufweisen.
- **Mit einfachem Übertrag:** Diese Gruppe beinhaltet die Additionen, welche Überträge aufweisen. Allerdings keine Überträge zu Lücken (bei ungleicher Anzahl von Stellen) oder Stellen mit einer Null oder Neun. Diese sind nach Gerster [1982, S. 23-27] für Schüler schwieriger zu lösen.
- **Mit Übertrag:** In dieser Gruppe befinden sich Rechnungen, die nicht in die ersten beiden Kategorien fallen.

Insgesamt weist der Plusminus-Trainer 17 verschiedene Kompetenzlevel, aufgeteilt in drei Kompetenzgruppen, auf.

5.2.2 Subtraktion

Ebenso wie bei der Addition werden auch hier die einzelnen Rechnungen in Kompetenzlevel und Kompetenzgruppen unterteilt. Die Unterteilung der Kompetenzgruppen wurde anhand der Anzahl der Überträge und der Existenz von Nullen durchgeführt. Diese sind nach Padberg [2005, S. 243] sehr häufige Fehlerquellen (siehe auch Abschnitt 3.2.2 „Schwierigkeiten und Fehler“).

Die Einteilung ist in Abbildung 5.2 zu sehen. Hierbei geben die Kompetenzgruppen jeweils an, ob Überträge vorhanden sind und ob Nullen in irgendeiner Weise in der Subtraktion auftreten. In den Spalten sind wieder die unterschiedlichen Subtraktionsmuster zu finden.

	n - n	nn - n	nn - nn	nnn - n	nnn - nnn	nnn - nn
Ohne Übertrag, ohne Nullen	①	②	③	④	⑤	⑥
1 Übertrag, ohne Nullen		⑦	⑧	⑨	⑩	⑪
1 Übertrag, Nullen in Angabe		⑫	⑬	⑭	⑮	⑯
1 Übertrag, Nullen in Ergebnis				⑰	⑱	⑲
1 Übertrag, mit Nullen				⑳	㉑	㉒
min. 2 Überträge, mit Nullen				㉓	㉔	㉕

Tabelle 5.2: Einteilung der Kompetenzlevel für die Subtraktion.

5.3 Aufbau und Komponenten

In diesem Kapitel wird der allgemeine Aufbau des Plusminus-Trainers betrachtet. Wie bereits in Kapitel 5.1.2 beschrieben, verwendet der Trainer das Model-View-Controller (MVC) Softwaredesignpattern. Dieses Muster erlaubt vor allem die Trennung von Geschäftslogik und Design und erhöht die Wiederverwendbarkeit von Code sowie die Wartbarkeit. Für den Trainer wurden folgende fünf Controller und ein abstrakter Basis-Controller implementiert.

- **BaseController:**

Der BaseController wurde als abstrakte Klasse implementiert. Da im Zend Framework 2 Services via Dependency Injection geladen werden, stellt dieser Controller die Standardservices wie zum Beispiel Zugriff auf den Datenbanklayer oder auf das Authenticationservice über Member und Methoden zur Verfügung. Dies führt dazu, dass dieser SourceCode zentral in diesem Controller gewartet und verändert werden kann und nicht für alle anderen Controller kopiert werden muss. Alle übrigen Controller erben von diesem.

- **CalcController:**

Dieser Controller ist für das Auswählen der Rechnungen, die der Benutzer oder die Benutzerin erhält, zuständig. Nach einer erfolgreichen Auswahl werden die Daten an die View übermittelt und von dieser angezeigt. Durch das Bestätigen des Antworten Buttons in der View wird das Formular abgeschickt und dem Controller übermittelt. Die Daten werden empfangen und weiter verarbeitet.

- **SettingsController:**

Mit Hilfe des Settingscontrollers können benutzerspezifische Einstellungen vorgenommen werden. Dieser Controller wird benötigt, um Einstellungen wie Auftrittswahrscheinlichkeiten von Operationen, Übertragsposition und Anzahl der korrekten Antworten für den Levelaufstieg zu setzen. Bei Schülerinnen und Schülern die einer Klasse zugeordnet sind, kann dies durch die Lehrperson gesperrt werden, bei den verbleibenden Benutzern sind diese Einstellungen frei zugänglich.

- **StatisticController**

Dieser Controller hat die Aufgabe, das Abrufen der verschiedenen Statistiken zu koordinieren. Über ihn laufen die Statistiken für Klassen, die nur von Benutzern der Gruppe „Lehrer“ abgerufen werden können, sowie die Einzelstatistiken für Benutzer. Die Statistik ist hierarchisch aufgebaut. Die Grundstatistik stellt eine Unterteilung nur in Operationen zur Verfügung. Bei Bedarf kann der Benutzer diese Statistik bis hin zu den durchgeführten Rechnungen verfeinern.

- **UserController:**

Um einen Benutzer oder eine Benutzerin eindeutig zu identifizieren, wird dieser Controller verwendet. Durch Abschicken der Eingabemaske auf der Loginseite werden die Daten dem Controller übermittelt. Dieser verifiziert die Daten gegen die zentrale Userverwaltung und vergibt die Berechtigungen. Sollte ein Benutzer oder eine Benutzerin das erste Mal den Login des Trainers benutzen, werden zusätzlich Daten vom Usermanagement abgerufen und in die eigene Datenbank gespeichert. Dies erhöht die Performance für weitere nachfolgende Zugriffe.

Zu seinen weiteren Aufgaben zählen auch das Abmelden vom System, das Zurücksetzen der Benutzerstatistik sowie das Ändern der Benutzersprache.

- **WebserviceController:**

Um die Services des Plusminus-Trainers auch außerhalb der eigenen Applikation zur Verfügung zu stellen wurde ein Webservice implementiert. Diese Schnittstelle bedient sich des Simple Open Access Protocols¹¹ (SOAP) zur Übertragung der Daten. Nähere Informationen zum Webservice kann man in Kapitel 5.10 finden.

Um diese Funktionalität in den Controllern zu gewährleisten und um die Geschäftslogik bestmöglich davon zu trennen, wurde ein Servicelayer implementiert. Dieser besteht aus verschiedenen Services die per Dependency Injection (DI) zur Laufzeit in die Controller

¹¹<http://www.w3.org/TR/soap/> [Zugriff am 09.07.2013]

eingebunden werden. Durch die Verwendung von Interfaces für jedes Service und DI ist es einfach möglich, die Implementationen auszutauschen. In Abbildung 5.2 sieht man eine vereinfachte Darstellung dieses Zusammenspiels.

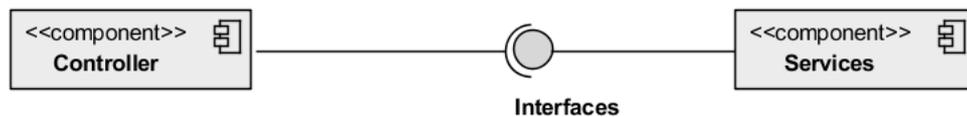


Abbildung 5.2: Vereinfachte Darstellung des Zusammenspiels von Controller und Servicelayer über Interfaces

Für die Funktionalität des Plusminus-Trainers wurden acht Services implementiert. Diese werden nachfolgend kurz erläutert:

- **LanguageService:**
Dieses Service stellt die grundlegenden Funktionen für die Mehrsprachigkeit des Programms zur Verfügung. Definiert werden hier die Standardsprache des Systems sowie die Sprache des jeweiligen Benutzers bzw. der Benutzerin. Durch das Speichern der Sprache in einer Session, bleibt diese dem Benutzer bzw. der Benutzerin auch nach dem Verlassen des Systems gespeichert.
- **MathAuthService:**
Das MathAuthService kapselt das in Zend Framework 2 standardmäßig enthaltene *AuthenticationService* und erweitert dieses um einige wichtige Funktionalitäten. Es ist zuständig für die Authentifizierung des Anwenders bzw. der Anwenderin und für das Abmelden und besitzt zudem noch Funktionen, um den aktuell eingeloggtten Benutzer bzw. die aktuell eingeloggte Benutzerin abzurufen.
- **ProblemService:**
Um Additionen und Subtraktionen in den passenden Schwierigkeiten erhalten zu können, gibt es dieses Service. Durch die Methode *getProblem(userlevel)* wird ein zufälliges Beispiel, passend zu dem übergebenen Userlevel, ausgewählt und zurückgegeben. Über eine zweite Methode verwendet das Service die Auftrittswahrscheinlichkeiten für die Operationen aus den Benutzereinstellungen und wählt aufgrund dieser zufällig eine Operation aus.
- **SettingsService:**
Dieses Service ist sehr einfach gehalten und besteht lediglich aus zwei Methoden. Die eine liest die Benutzereinstellungen anhand einer Benutzer Id aus der Datenbank und gibt diese zurück, während die zweite die globalen Einstellungen des Programms aus der Datenbank abrufen und weitergibt.

- **StatisticService:**

Das StatisticService ist bei weitem das umfangreichste Service. Es selektiert die richtigen Daten aus der Datenbank, führt Analysen durch und erstellt die Statistiken. Der Umfang des Services ergibt sich daraus, dass die statistischen Auswertungen zu einem großen Teil direkt über die SQL-Queries erstellt werden, was die Performance positiv beeinflusst. Diese Queries sind jedoch von Statistik zu Statistik unterschiedlich und müssen separat implementiert werden.

- **SynchronisationService:**

Da die Benutzer des Plusminus-Trainers zentral in einem Usermanagement verwaltet werden, müssen diese beim ersten Login in die Datenbank des Plusminus-Trainers übertragen werden. Diese Funktionalität wird mit Hilfe des SynchronisationService bereitgestellt. Alle für den Trainer wichtigen Daten, wie z.B.: Gruppenzugehörigkeiten, Klasse et cetera, werden durch dieses Service von der zentralen Stelle abgerufen und in die Datenbank übertragen. Zudem übernimmt es auch die Aufgabe, diese Daten stets aktuell zu halten und Änderungen aus der zentralen Userverwaltung zu synchronisieren.

- **UserLevelService:**

Dieses Service ist sehr vielseitig. Es dient hauptsächlich zur Verwaltung der Kompetenzlevel. Da die Kompetenzlevel und Gruppen dynamisch in der Datenbank gespeichert werden, kümmert sich dieses Service darum, Level in der richtigen Reihenfolge auszugeben und das nächste Kompetenzlevel für den Benutzer oder die Benutzerin zu bestimmen. Zur weiteren Funktionalität gehört das Initialisieren eines Benutzers bzw. einer Benutzerin beim ersten Login sowie das Zurücksetzen der Statistik.

- **UserSoapService:**

Da der Zugriff auf die zentrale Benutzerverwaltung über ein SOAP Webservice geschieht, kapselt dieses Service diese Schnittstelle. Alle Methoden für die Benutzerverwaltung sowie die wichtigen Daten für die Authentifizierung sind in diesem Service realisiert und gespeichert.

5.4 Beispielgenerierung

Der Plusminus-Trainer generiert die Beispiele nicht zur Laufzeit, sondern lädt diese aus einem Pool von Beispielen, die in einer Datenbank abgelegt sind. Für die Generierung wurde ein externes Programm entwickelt. Dieses wurde in der Programmiersprache C# und mit Hilfe der Windows Presentation Foundation (WPF) realisiert. Der Problemgenerator wurde sehr simpel gehalten, da er für eine einmalige Verwendung geschrieben wurde.

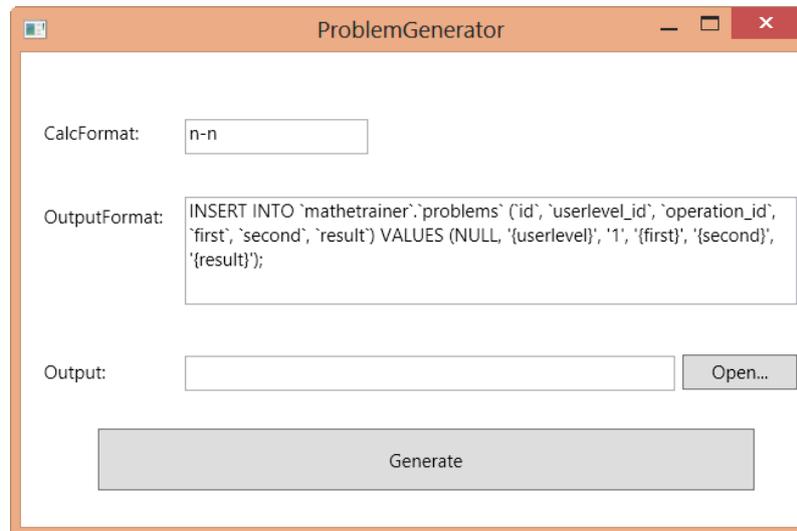


Abbildung 5.3: Grafische Oberfläche des Problem Generator

Wie in Abbildung 5.3 zu sehen ist, werden drei Inputparameter benötigt:

- **CalcFormat:** Dieser Parameter gibt an, wie viele Stellen die Probleme besitzen sollen und welche Operation (Addition oder Subtraktion) verwendet werden soll (z.B.: $n + nn$ et cetera).
- **OutputFormat:** Mit Hilfe dieses Parameters kann man bereits das Ausgabedesign steuern. Hierzu wird eine Zeichenkette mit sogenannten Platzhaltern eingegeben. Automatisch ersetzt werden `{first}`, `{second}` und `{result}`. `{userlevel}` kann nicht automatisch ersetzt werden, da sich die ID in der Datenbank von Installation zu Installation ändern kann. Dieses muss manuell vor dem Einfügen in die Datenbank ersetzt werden.
- **Output:** Hier wird angegeben, wo der Problemgenerator die generierten Dateien ablegen soll. Eine solche Ausgabe ist in Abbildung 5.4 zu sehen.

```

1 Type: n+n] Section NoCarry
2 INSERT INTO `mathetrainer`.`problems` (`id`, `userlevel_id`, `operation_id`, `first`, `second`, `result`) VALUES (NULL, '{userlevel}', '1', '1', '1', '2');
3 INSERT INTO `mathetrainer`.`problems` (`id`, `userlevel_id`, `operation_id`, `first`, `second`, `result`) VALUES (NULL, '{userlevel}', '1', '1', '2', '3');
4 INSERT INTO `mathetrainer`.`problems` (`id`, `userlevel_id`, `operation_id`, `first`, `second`, `result`) VALUES (NULL, '{userlevel}', '1', '1', '3', '4');
5 INSERT INTO `mathetrainer`.`problems` (`id`, `userlevel_id`, `operation_id`, `first`, `second`, `result`) VALUES (NULL, '{userlevel}', '1', '1', '4', '5');
6 INSERT INTO `mathetrainer`.`problems` (`id`, `userlevel_id`, `operation_id`, `first`, `second`, `result`) VALUES (NULL, '{userlevel}', '1', '1', '5', '6');
7 INSERT INTO `mathetrainer`.`problems` (`id`, `userlevel_id`, `operation_id`, `first`, `second`, `result`) VALUES (NULL, '{userlevel}', '1', '1', '6', '7');
8 INSERT INTO `mathetrainer`.`problems` (`id`, `userlevel_id`, `operation_id`, `first`, `second`, `result`) VALUES (NULL, '{userlevel}', '1', '1', '7', '8');
9 INSERT INTO `mathetrainer`.`problems` (`id`, `userlevel_id`, `operation_id`, `first`, `second`, `result`) VALUES (NULL, '{userlevel}', '1', '1', '8', '9');
10 INSERT INTO `mathetrainer`.`problems` (`id`, `userlevel_id`, `operation_id`, `first`, `second`, `result`) VALUES (NULL, '{userlevel}', '1', '2', '1', '3');
11 INSERT INTO `mathetrainer`.`problems` (`id`, `userlevel_id`, `operation_id`, `first`, `second`, `result`) VALUES (NULL, '{userlevel}', '1', '2', '2', '4');
12 INSERT INTO `mathetrainer`.`problems` (`id`, `userlevel_id`, `operation_id`, `first`, `second`, `result`) VALUES (NULL, '{userlevel}', '1', '2', '3', '5');
13 INSERT INTO `mathetrainer`.`problems` (`id`, `userlevel_id`, `operation_id`, `first`, `second`, `result`) VALUES (NULL, '{userlevel}', '1', '2', '4', '6');
14 INSERT INTO `mathetrainer`.`problems` (`id`, `userlevel_id`, `operation_id`, `first`, `second`, `result`) VALUES (NULL, '{userlevel}', '1', '2', '5', '7');
15 INSERT INTO `mathetrainer`.`problems` (`id`, `userlevel_id`, `operation_id`, `first`, `second`, `result`) VALUES (NULL, '{userlevel}', '1', '2', '6', '8');

```

Abbildung 5.4: Ausgabe des Problem Generators

5.4.1 Struktur und Aufbau

Der Problemgenerator ist sehr simpel aufgebaut. Die Komponente CalcGenerator übernimmt das Pattern, das der Benutzer oder die Benutzerin in das Feld „CalcFormat“ eingegeben hat und erhält so die Anzahl der Stellenwerte der beiden Zahlen und den Typ der Operation. Durch die Methode GenerateAllNumbers (siehe Abbildung 5.1), welcher das Pattern (z. B.: nn) übergeben wird, werden alle möglichen Zahlen, die diesem Pattern entsprechen, generiert. Dies geschieht, indem zuerst die Länge des Patterns überprüft wird (Zeile 4). Danach wird mit einer for-Schleife (Zeile 6) alle Zahlen von $10^{(count-1)}$ bis zu $10^{(count)} - 1$ durchlaufen und in eine Liste gespeichert (Zeile 8). Für den einfachsten Fall „n“ wären das Zahlen zwischen $10^0 = 1$ und $10^1 - 1 = 9$. Danach werden alle möglichen Kombinationen dieser Zahl als Addition bzw. als Subtraktion erzeugt.

Im nächsten Schritt werden diese generierten Rechnungen einem Filter übergeben. Dieser unterscheidet sich bei Addition und Subtraktion. Durch diesen Filter werden die Rechnungen dem richtigen Userlevel zugeordnet und in Listen gespeichert. Rechnungen, die keinem dieser Kriterien entsprechen, werden verworfen. Diese Userlevel werden an den Problemgenerator zurückgegeben und in weiterer Folge jedes Userlevel extra in eine Textdatei geschrieben.

```
1 private List<int> GenerateAllNumbers(string pattern)
2 {
3     var numbers = new List<int>();
4     int count = pattern.Length;
5
6     for (var i = (int)Math.Pow(10, count-1); i < Math.Pow(10, count); i++)
7     {
8         numbers.Add(i);
9     }
10
11     return numbers;
12 }
```

Quellcode 5.1: Die Methode zur Generierung aller Zahlen für ein Pattern

5.5 Programmablauf

In diesem Abschnitt werden der Ablauf des Plusminus-Trainers sowie seine Subprozesse näher erläutert. In Abbildung 5.5 sehen wir den allgemeinen Programmablauf im Übungsmodus. Nach dem Login wird zuerst überprüft, ob dieser Benutzer bzw. die Benutzerin bereits beim Plusminus-Trainer registriert ist. Ist er bereits im System vorhanden, werden seine Informationen zu den Kompetenzlevel und Kompetenzgruppen geladen. Ist dies sein erster Login, wird der Anwender oder die Anwenderin vom zentralen Usermanagement in die Datenbank des Plusminus-Trainers synchronisiert und seine

Kompetenzlevel auf null initialisiert. Danach wird der Subprozess der Beispielzuteilung gestartet. Dieser ist in Abschnitt 5.5.1 näher beschrieben und dient hauptsächlich dazu, die Operation und die Schwierigkeit des nächsten Beispiels zu bestimmen. Das Anzeigen des Beispiels durch das Rendern der richtigen Form (Addition oder Subtraktion) auf dem Bildschirm des Benutzers oder der Benutzerin findet ebenfalls noch in diesem Subprozess statt. Wenn die Formelemente ausgefüllt und abgesendet werden, wird der Prozess der Validierung aktiv (nähere Informationen in Abschnitt 5.5.2). Er prüft die Korrektheit der Antworten und schreibt alle wichtigen Daten in die Datenbank. Ist dieser Prozess abgeschlossen, kommt im nächsten Schritt die Fehleranalyse, genauer erläutert in Abschnitt 5.5.3, zum Zug. Diese untersucht das Beispiel und die Benutzerantwort auf typische Fehler und hält diese in der Datenbank fest.

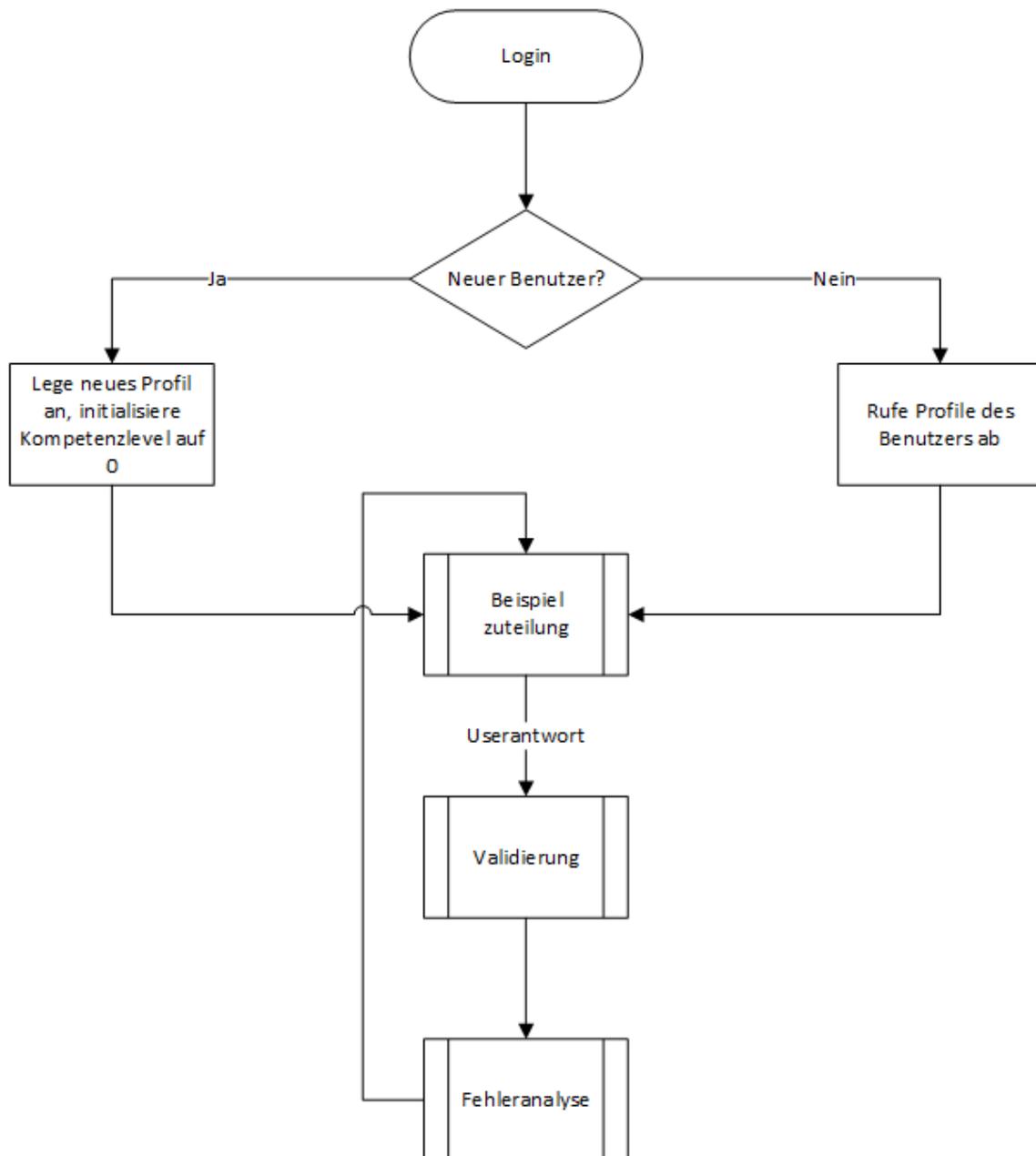


Abbildung 5.5: Programmablauf des Plusminus-Trainers im Übungsmodus

5.5.1 Beispielzuteilung

Da der Plusminus-Trainer einem adaptiven Ansatz folgt, ist die Beispielzuteilung ein zentraler Punkt. Sinn und Zweck des Trainers soll es sein, dass der Schüler oder die Schülerin durch die Beispiele weder über- noch unterfordert wird und sie etwa seinem

Lernniveau angepasst werden. Um dies zu erreichen, wurden, wie bereits in Kapitel 5.2 beschrieben, Kompetenzlevel und Kompetenzgruppen eingeführt. Diese spiegeln den aktuellen Wissensstand des Benutzers wider.

In Abbildung 5.6 sehen wir den vereinfachten Prozess der Beispieltzuteilung. Startet die Benutzerin oder der Benutzer den Übungsmodus, so wird vom Plusminus-Trainer das Profil mit den Userinstellungen geladen. Anhand der Werte für die Operationswahrscheinlichkeiten, die in den Benutzereinstellungen definiert werden können, lädt das ProblemService eine zufällige Operation. Diese Operation, Addition oder Subtraktion, und das Profil der aktuellen Anwenderin oder des aktuellen Anwenders, werden dem UserLevelService übergeben. Dieses Service liest sich anhand der übergebenen Operation das aktuelle Kompetenzlevel für genau diese Operation aus und gibt es zurück. Dadurch ist gewährleistet, dass das nächste Beispiel in etwa dem Wissensstand des Benutzers bzw. der Benutzerin entspricht. Über das ProblemService wird mit Hilfe dieses Kompetenzlevels ein dazugehöriges Beispiel ausgegeben. Der CalcController trägt danach Sorge, dass das die richtige Form (Addition oder Subtraktion) für den Benutzer bzw. die Benutzerin gerendert wird.

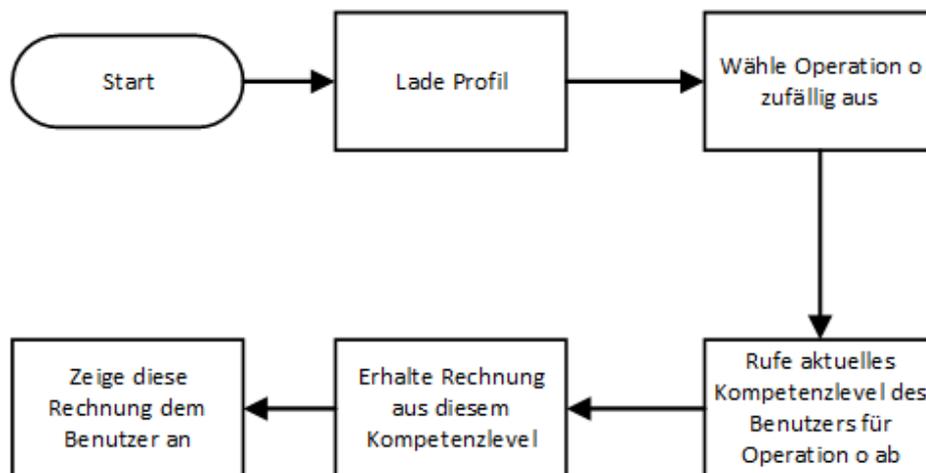


Abbildung 5.6: Vereinfachter Prozess der Beispieltzuteilung des Plusminus-Trainers

5.5.2 Validierung und Sicherung der Daten

In diesem Abschnitt geht es darum, wie die eingegebenen Daten validiert werden und welche Daten wie gespeichert werden. Vor allem die Speicherung der Daten ist ein sehr wichtiges Kriterium, da es auch möglich sein sollte, zu einem späteren Zeitpunkt noch Analysen anstellen zu können. Aus diesem Grund wurde darauf geachtet, dass keine bereits analysierten oder kombinierten Daten gespeichert werden, sondern wirklich die „Rohdaten“ der Benutzerinnen und Benutzer.

Wenn der Benutzer seine Webform absendet erhält der Controller, in diesem Fall der `CalcController`, seine Daten. Dieser gibt die erhaltenen Daten an einen eigenen Validator weiter. Die Grundlage für diesen Validator bildet der `AbstractValidator` des Zend Frameworks. Durch die Verwendung dieser Oberklasse muss in den Validatoren für die Addition und Subtraktion nur noch die Logik implementiert werden. Das Aufrufen und Kennzeichnen der fehlerhaften Formelemente übernimmt der `AbstractValidator`. Nachdem das Ergebnis validiert wurde, wird das `UserLevelService` verwendet, um das Kompetenzlevel anzupassen. Bei dieser Anpassung unterscheidet der Trainer zwischen zwei Phasen:

- **Einstufungsphase:** Diese Phase können die Benutzer bzw. die Benutzerinnen auf zwei Wegen erreichen. Zum einen werden sie dieser Phase zugeordnet, wenn sie sich das erste Mal beim Plusminus-Trainer einloggen, und zum anderen steigen sie in diese Phase ein, nachdem sie ihre Statistik zurückgesetzt haben. In dieser Phase erreichen sie das nächste Kompetenzlevel bereits nach einer richtigen Antwort. Sie soll der Einstufung der Lernenden dienen und soll sicherstellen, dass sie nicht mit zu einfachen Problemstellungen konfrontiert werden. Die Phase endet automatisch, wenn sie bei einer Rechnung einen Fehler machen oder bereits beim letzten Kompetenzlevel angelangt sind.
- **Trainingsphase:** In dieser Phase werden die Lernenden die meiste Zeit verbringen. Sie tritt genau dann auf, wenn die Einstufungsphase abgeschlossen ist. In dieser Phase steigt das Kompetenzlevel der Lernenden nicht mit jeder richtigen Antwort. Das nächste Level wird erst dann erreicht, wenn die Benutzer bzw. die Benutzerinnen eine gewisse Anzahl von Problemen in diesem Kompetenzlevel ohne Fehler gelöst haben. Die Anzahl für den Aufstieg ist in den Benutzereinstellungen zu definieren (für mehr Informationen siehe Abschnitt 5.7). Sobald bei den Benutzern bzw. den Benutzerinnen ein Fehler auf einem Kompetenzlevel auftritt, wird dieses Kompetenzlevel zurückgesetzt. Dies geschieht, indem der Zähler für die richtigen Beispiele in Folge auf 0 gesetzt wird und das aktuelle Kompetenzlevel auf das darunterliegende geändert wird.

Je nach Validierungsergebnis kümmert sich das `UserLevelService` um die korrekten Änderungen des Kompetenzlevels. Für die Validierung eines Ergebnisses kommen vier verschiedene Kombinationen in Frage.

Im besten Fall haben die Lernenden alle Ergebnis- und Übertragfelder korrekt ausgefüllt. Ist dies der Fall, bekommen sie ein Feedback wie in Abbildung 5.10 zu sehen ist. Die Ausgabe nach der Validierung eines falschen Ergebnisses wird in Abbildung 5.11 gezeigt.

Der Validator erkennt das Ergebnis auch als richtig, wenn die Übertragfelder nicht ausgefüllt werden, das Ergebnis jedoch stimmt. Welche Felder befüllt und welche leer gelassen worden sind, wird ebenfalls in der Datenbank gespeichert. In Abbildung 5.7

sehen wir ein Beispiel, wo die Überträge leer sind, das Ergebnis jedoch korrekt ist. In diesem Fall wird der Schüler bzw. die Schülerin das Anschreiben der Überträge nicht benötigen, da er oder sie sich diese im Kopf merkt. Die letzte Kombination ist jene, wo das Ergebnis zwar richtig ist, jedoch ein Fehler in einer Übertragungsspalte besteht. Diesen Fall wertet der Validator ebenfalls als Fehler, da irgendwo eine Fehlervorstellung vorliegen muss.



Abbildung 5.7: Anzeige nach Validierung eines richtigen Ergebnisses, jedoch ohne ausgefüllte Überträge

Sobald die Validierung abgeschlossen ist, werden im Controller alle Daten der Benutzereingabe gesammelt und in die Datenbank gespeichert. Dafür wird das Datenbank Entity „UserAnswer“ verwendet. Dies ist im Abschnitt 5.9 näher erläutert.

5.5.3 Fehleranalyse

Ein sehr wichtiger Prozess, wenn nicht der wichtigste, ist die Fehleranalyse. In diesem Prozess wird versucht, typische und systematische Fehler der Eingaben zu erkennen und diese zu speichern. Wichtig ist es auch, die Komponente für die Analyse so dynamisch und offen wie möglich zu gestalten, um nach späteren Erkenntnissen neue Fehlertypen leicht zum System hinzufügen zu können.

Aus diesem Grund wurde der `ErrorAnalyzer` als Komponente implementiert. Wie alle Services des Plusminus-Trainers kann auch diese Komponente leicht ausgetauscht werden, da sie ein Interface implementiert und über Dependency Injection eingebunden wird. Der `ErrorAnalyzer` verwendet eine Version des Chain of Responsibility Designpatterns [vgl. Schmidt, 2009, 339 ff]. Dieses Pattern bietet die nötige Flexibilität und Dynamik zum

Einbinden und Verwalten neuer Fehler.

Jeder Fehler, welcher zum System hinzugefügt wird, muss das `ErrorInterface` (siehe Quellcode 5.2) implementieren. Dieses Interface beinhaltet zwei Methoden. Der ersten Methode `isPossible` wird als Parameter nur das Problem übergeben. Diese Methode dient dazu, herauszufinden, ob bei dieser Rechnung dieser Fehler überhaupt auftreten kann. Dies wird benötigt, um ein aussagekräftiges Ergebnis in der Statistik zu bekommen. Sonst würden Rechnungen, die keinen Übertrag erfordern, trotzdem positiv zur prozentualen Auswertung eines Übertragfehlers gewertet werden. Die Methode gibt `true` zurück, wenn der Fehler auftreten kann und ansonsten `false`. Die zweite Methode `hasError` erhält zusätzlich zum aktuellen Problem die Benutzerantwort. Diese Methode überprüft, ob dieser Fehler vorhanden ist und retourniert `true`, falls der Fehler aufgetreten ist, und `false`, falls die Benutzerantwort diesen Fehler nicht aufweist.

```
1 interface ErrorInterface
2 {
3     /**
4     * @param Problem $problem
5     * @return bool
6     */
7     public function isPossible($problem);
8
9     /**
10    * @param Problem $problem
11    * @param UserAnswer $useranswer
12    * @return bool
13    */
14    public function hasError($problem, $useranswer);
15 }
```

Quellcode 5.2: `ErrorInterface`, das jeder Fehler implementieren muss

Die Reihenfolge der verschiedenen Fehler, die getestet werden, kann in der Datenbank über einen Parameter `order` gesetzt werden. Ebenfalls in der Datenbank existiert ein zusätzlicher Parameter `breakChain`, welcher 0 oder 1 als Wert annehmen kann. Ist dieser Parameter bei einem Fehler auf 1 und die Benutzerantwort enthält diesen Fehler, wird die Kette unterbrochen und die Fehleranalyse beendet. Dies wurde vor allem eingeführt, da es Fehler gibt, die zu Beginn bereits eindeutig identifiziert werden können. Ein Beispiel dafür ist der Eingabefehler. Wenn leere Daten dem `ErrorAnalyzer` übermittelt werden, kann es sich nur um einen Eingabefehler handeln, und die anderen Fehler müssen nicht mehr abgearbeitet werden.

Mit dem Stand dieser Arbeit wurden bereits 13 Fehler implementiert und getestet. Weitere werden im Laufe der Zeit folgen. Eine kurze Zusammenfassung der bereits implementierten Fehler ist in den Tabellen 5.3, 5.4 und 5.5 je nach Rechenoperation zu sehen.

Klassenname des Fehlers	Beschreibung
CalcWithZero	Schülerin bzw. Schüler glaubt $a + 0 = 0 = 0 + a$
CarryEver	Bei jeder Spalte wird ein Übertrag gerechnet, egal ob benötigt oder nicht.
DifferentDigitCount	Bei unterschiedlicher Stellenanzahl werden die Ergebnisstellen, in denen keine zwei Ziffern stehen, leer gelassen.
NoCarry	Es wird sonst richtig gerechnet, aber kein Übertrag berücksichtigt.
NoCarryReset	Die Schülerin bzw. der Schüler vergisst ihren bzw. seinen Übertraggähler auf 0 zurückzusetzen und zählt bei zwei nacheinander folgenden Überträgen 2 zur Summe hinzu.
NoCarryToZero	Die Benutzerin bzw. der Benutzer rechnet richtig, jedoch zählt sie bzw. er keinen Übertrag zur 0.
OnePlusOne	Dieser Fehler untersucht Einspluseins-Fehler, die ± 1 vom Ergebnis abweichen.
SubtractionInsteadOfAddition	Hier wird die falsche Rechenoperation durchgeführt.

Tabelle 5.3: Implementierte Fehler für die Addition

Klassenname des Fehlers	Beschreibung
AdditionInsteadOfSubtraction	Es wird die falsche Rechenoperation verwendet.
GreaterMinusSmaller	Es wird spaltenweise immer die kleinere von der größeren Zahl abgezogen.
NoCarry	Es wird zwar richtig erweitert, aber der Übertrag nicht berücksichtigt.
ToMuchCarry	Es wird ein Übertrag zu viel berechnet.

Tabelle 5.4: Implementierte Fehler für die Subtraktion

Klassenname des Fehlers	Beschreibung
InputError	Es wird kein Formularfeld befüllt. Dieser Fehler bricht die Analyseketten bei Eintreten sofort ab.

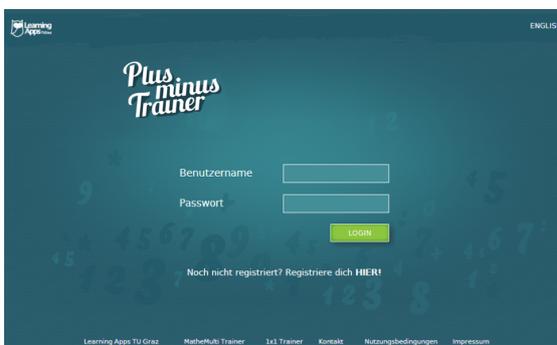
Tabelle 5.5: Gemeinsame Fehler

5.6 Layout

Um Kindern eine Motivation zu bieten, sich mit Mathematik auf freiwilliger Basis zu beschäftigen, wurde auch das Design sehr kindgerecht angepasst. Es wurde zu großen Teilen vom Mathemulti Trainer¹² übernommen, um den Benutzern eine einfache Eingewöhnung zu ermöglichen.

Aus technischer Sicht wird hier wieder sehr stark das MVC-Design-Pattern (näheres im Abschnitt 5.1.2) des Zend Frameworks verwendet. Dieses erlaubt, mit Layout Seiten zu arbeiten, welche eine einheitliche Gestaltung der ganzen Webanwendung ermöglichen. Eine Layoutseite dient als Grundgerüst für jede angezeigte Webseite. Die View-Elemente der unterschiedlichen Webseiten werden nur in dieses Grundgerüst eingebunden. Dies hat den enormen Vorteil, dass Designänderungen in der Layoutseite nur einmal durchgeführt werden müssen.

Wenn die Lernenden die Seite des Plusminus-Trainers über ihren Webbrowser betreten, werden sie aufgefordert, ihren Benutzernamen und ihr Passwort einzugeben oder sich über einen Link an der zentralen Benutzerverwaltung anzumelden (siehe Abbildung 5.8a). Nach einem erfolgreichen Einloggen bekommt sie noch die Überprüfungsabfrage, ob sie bereit sind zu starten, gestellt (siehe Abbildung 5.8b). Dies ist wichtig, da mit dem Betätigen des Buttons die Zeitmessung für das Beispiel gestartet wird.



5.8a: Loginformular für den Benutzer



5.8b: Überprüfungsabfrage vor dem Start

Abbildung 5.8: Formulare des Plusminus-Trainers

¹²<http://multitrainer.tugraz.at> [Zugriff am 09.07.2013]

Nach dem Klick auf die Schaltfläche „Los gehts“ wählt der Trainer eine Operation und eine dazugehörige Rechnung aus und zeigt sie auf dem Bildschirm an. Ein Beispiel für die Subtraktion ist in Abbildung 5.9 ersichtlich. Durch die Einbindung von JavaScript wird der Fokus des Cursors sofort in das richtige Feld gelegt. Im Fall der schriftlichen Addition und Subtraktion ist dies das letzte Feld (Einerstelle) des Ergebnisses.

Der Link mit dem Text „Switch“ ist nur bei der Subtraktion zu sehen und bewirkt einen Wechsel der Übertragzeile an den oberen Rand der Rechnung. Dies wurde implementiert, um alle Übertragstechniken der Subtraktion (siehe Abschnitt 3.2) zu unterstützen. Dieser Mechanismus wird genauer in Abschnitt 5.7 beschrieben.

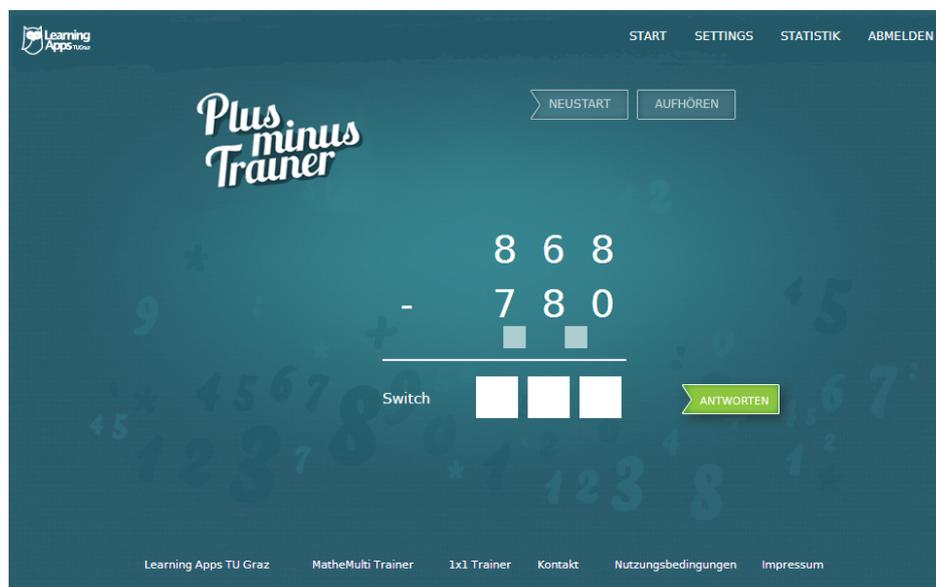


Abbildung 5.9: Die Eingabemaske für eine Subtraktion

In dieser Eingabemaske ist es möglich, mit der Tabulatortaste in das, nach der Schulmethode, nächste Element zu springen. Der Benutzer oder die Benutzerin füllt die Formelemente aus und schickt sie mit einem Klick auf den Button „ANTWORTEN“ ab. Sofort nach dem Betätigen des Buttons wird ein Feedback vom Trainer erhalten. Bei korrekter Eingabe der Lösung wird ein grüner Stempel mit der Aufschrift „RICHTIG“, wie in Abbildung 5.10, angezeigt. Bei einer inkorrekten Antwort werden die falschen Formelemente rot markiert und die richtige Lösung ausgegeben (siehe Abbildung 5.11). Eine weitere Bearbeitung der Rechnung ist nicht mehr möglich, da dies die statistische Auswertung verfälschen würde. Aus diesem Grund werden alle Formelemente auf read-only gesetzt. Der Benutzer oder die Benutzerin gelangt nach dem Klicken auf den Weiter-Button zur nächsten Aufgabe.

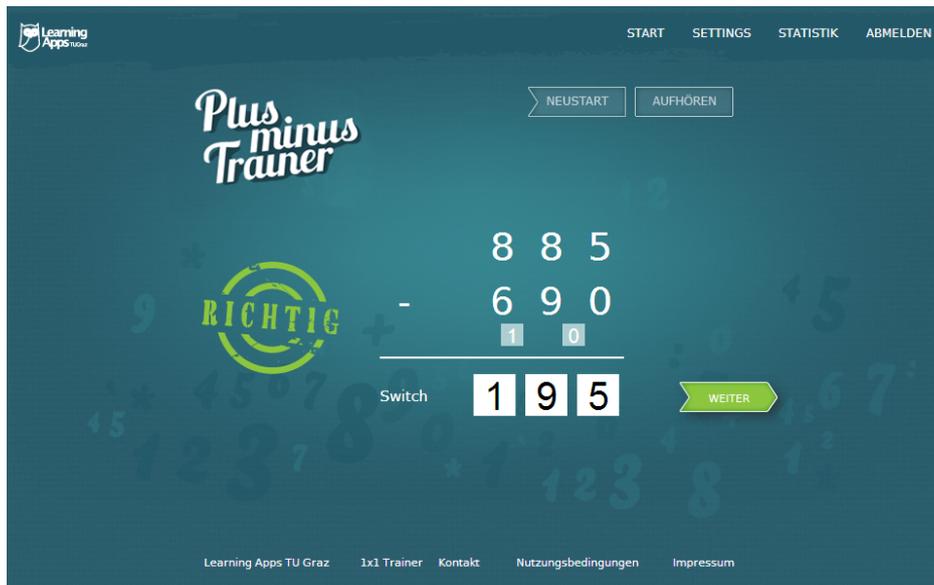


Abbildung 5.10: Anzeige nach Validierung eines richtigen Ergebnisses



Abbildung 5.11: Anzeige nach Validierung eines falschen Ergebnisses

Für die Nutzerstatistik wurde ein Tabellen ähnliches Layout entworfen. Durch den Einsatz der drei Ampelfarben, können der Schüler oder der Lehrer sofort erkennen, wo es Probleme gibt und wo keine Intervention erforderlich ist. Die Statistik ist hierarchisch aufgebaut und bis zu einer bestimmten Tiefe einsehbar. In Abbildung 5.12 sehen wir die Statistik der Addition aufgeklappt bis zur letzten Ebene der Kompetenzlevel. Die roten

Felder kennzeichnen Gruppen oder Level bei denen die Quote der richtigen Antworten unter 30% liegt. Ebenfalls schnell ersichtlich ist der häufigste Fehler in einer Gruppe oder einem Level. Dieser ist gleich rechts in der Spalte angeführt. Über den Link „Hier“ kann man eine andere Gruppe oder ein Kompetenzlevel im Detail betrachten. Im rechten oberen Bereich befinden sich zwei Registerkarten, mit Hilfe derer es möglich ist, zwischen der Statistikansicht und einer statistischen Auswertung der Fehler hin- und herzuschalten. Nähere Informationen zur Statistik finden sie im Abschnitt 5.8

Gruppe	Gesamt	Richtig	Falsch	Erfolgs Quote	Durchschnittliche Dauer (Sekunden)	Häufigster Fehler	Details
Kein Übertrag	9	9	0	100 %	00:00:04	Kein Fehler	>Hier<
Mit einfachem Übertrag	13	4	9	31 %	00:00:16	Kein Übertrag	>Hier<
Mit Übertrag	0	0	0	0 %	00:00:00	Kein Fehler	>Hier<

Gruppe	Gesamt	Richtig	Falsch	Abgeschlossen	Erfolgs Quote	Durchschnittliche Dauer (Sekunden)	Häufigster Fehler	Details
n + n	6	3	3	nein	50 %	00:00:25	Subtraktion statt Addition	>Hier<
n + nn nn + n	4	1	3	nein	25 %	00:00:05	Kein Übertrag	>Hier<
nn + nn	3	0	3	nein	0 %	00:00:13	Kein Übertrag	>Hier<
n + nnn nnn + n	0	0	0	nein	0 %	00:00:00	Kein Fehler	>Hier<
nnn + nnn	0	0	0	nein	0 %	00:00:00	Kein Fehler	>Hier<
nn + nnn nnn + nn	0	0	0	nein	0 %	00:00:00	Kein Fehler	>Hier<

Abbildung 5.12: Statistik eines Benutzers bis zur Ebene Kompetenzlevel aufgeklappt

5.7 Einstellungsmöglichkeiten

Da der Plusminus-Trainer so dynamisch wie möglich gestaltet werden sollte, gibt es eine Reihe von Einstellungsmöglichkeiten. Diese unterscheiden sich je nach Benutzergruppe. In der zentralen Benutzerverwaltung gibt es zum jetzigen Stand drei verschiedene Benutzergruppen - Schüler, Lehrer und Administratoren.

Als Schüler oder Schülerin kann man über den Link „Einstellungen“, der sich oben rechts befindet, zu den aktuellen Einstellungen gelangen. Über diesen Link gelangt man zur Einstellungsseite, wie in Abbildung 5.13 zu sehen. Die Einstellungen die in dieser Abbildung zu sehen sind, entsprechen den Standardeinstellungen, die vom Trainer beim ersten Login vorgenommen werden.

Die Verteilung der Rechenoperationen liegt auf jeweils 50% - also gleich wahrscheinlich. Dieser Parameter wurde eingeführt, um Schülern und Lehrern die Möglichkeit zu geben, nur eine Rechenoperation zu trainieren. Die nächste Einstellungsmöglichkeit bezieht sich auf die Anzeige des Übertrags. Wie in Abschnitt 3.2 beschrieben, gibt es bei der Subtraktion mehrere Verfahren. Die Übertragbildung mit der Bündelungstechnik schreibt den Übertrag über dem Minuenden an, während hingegen die Erweiterungstechnik den Übertrag unter dem Subtrahenden schreibt. Für diesen Fall wurde diese Einstellungsmöglichkeit eingeführt. Ihre Auswahlmöglichkeiten beschränken sich auf „Unten“ oder „Oben“. Als Standardwert ist die Übertragposition unter dem Subtrahenden. Die letzte Einstellungsmöglichkeit bezieht sich auf den Aufstieg des Kompetenzlevels. Hier kann eine Zahl $x \geq 1$ angegeben werden. Der Anstieg auf das nächste Kompetenzlevel erfolgt nach genau x richtigen Antworten in Folge. Dies ermöglicht längere Übungsphasen auf einem Kompetenzlevel oder einen rasanteren Anstieg zu Beginn. Die Standardeinstellung, die vom Trainer vorgegeben wird, liegt hier bei drei richtigen Antworten in Folge.

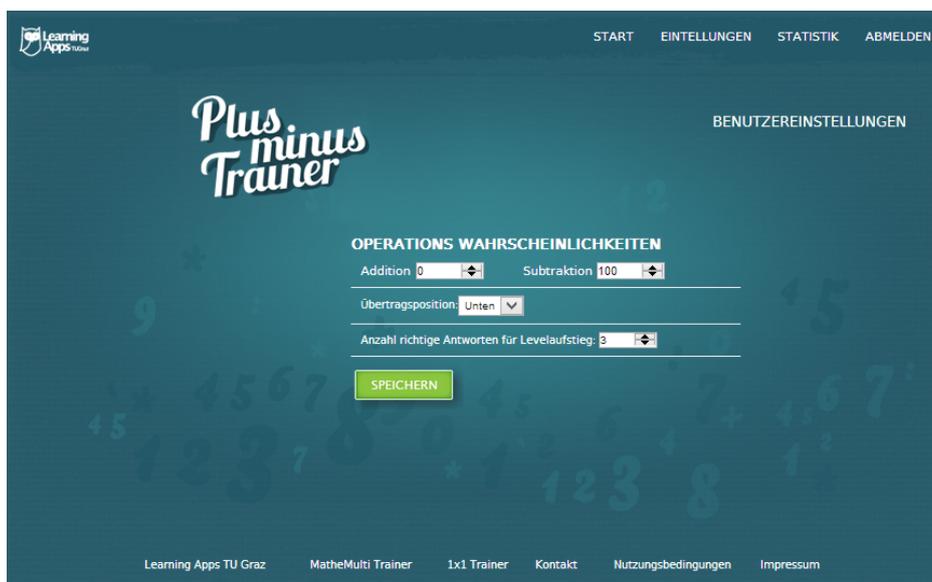


Abbildung 5.13: Benutzereinstellungen für den Plusminus-Trainer

Die Einstellungsseite für die Lehrperson unterscheidet sich nur minimal von der Einstellungsseite für Schüler. Die Lehrperson kann jedoch Einstellungen klassenweise treffen. Über den Link „EINSTELLUNGEN“ rechts oben, erhält sie zuerst eine Übersicht seiner Klassen. Wird eine Klasse ausgewählt, erhält die Lehrperson die Ausgabe, die in Abbildung 5.14 zu sehen ist. Der einzige Unterschied zu den Benutzereinstellungen ist eine kleine Checkbox am Ende der Liste. Mit Hilfe dieser Option kann die Lehrperson zusätzlich die Einstellungen für seine Schülerinnen und Schüler sperren. Durch betätigen des „Speichern“-Buttons werden diese Einstellungen an alle Schülerinnen und Schüler der

Klasse ausgerollt. Ist zusätzlich die Checkbox aktiv, ist es dem Schülerinnen und Schülern der Klasse nicht mehr möglich, Einstellungen zu ändern. Dadurch kann die Lehrperson die Einstellungen individuell für jede Klasse vornehmen.

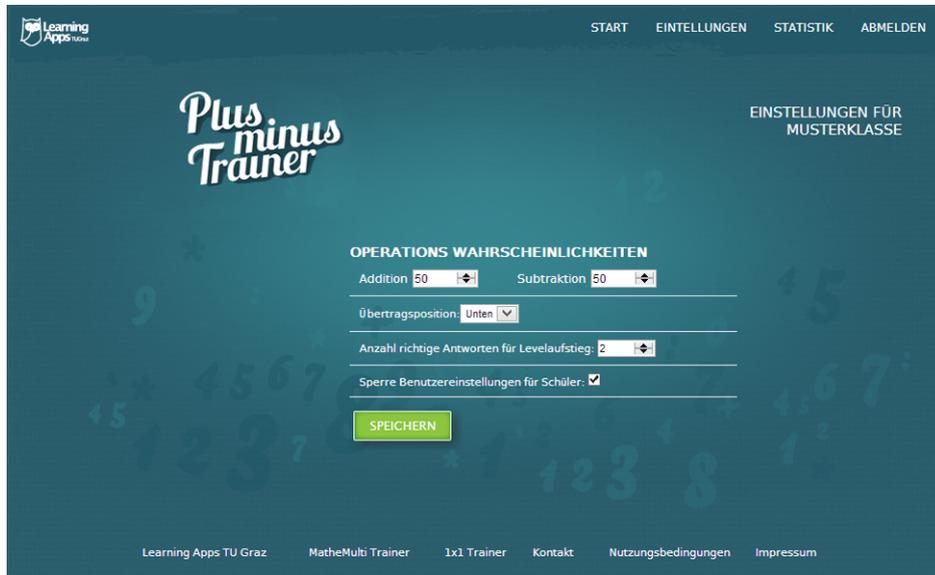


Abbildung 5.14: Einstellungen durch einen Lehrer für eine Klasse

Für Administratoren steht im Plusminus-Trainer noch keine grafische Oberfläche für Einstellungen zur Verfügung. Nichtsdestotrotz besteht die Möglichkeit, diese Einstellungen auf Datenbankebene vorzunehmen.

In den Tabellen `errors` und `userlevel` gibt es bei jedem Eintrag ein Feld `order`. Mit diesem Feld kann die Ordnung der Userlevel oder die Reihenfolge der Fehler geändert werden. Zusätzlich enthält die Datenbank eine Tabelle `ApplicationSettings` mit den Feldern `countInRowForLevelUp`, `probabilityForLowerLevel` und `probabilityForHigherLevel`. Das erste Feld gibt den Standardwert für die richtigen Antworten in Folge zum Levelaufstieg an. Dieser Wert wird bei jedem Benutzer und jeder Benutzerin gesetzt, der oder die sich zum ersten Mal einloggt. Die beiden anderen Felder sind Auftrittswahrscheinlichkeiten. `probabilityForLowerLevel` gibt die Wahrscheinlichkeit für das Auftreten einer Rechnung aus einem Userlevel unter dem aktuellen an und `probabilityForHigherLevel` bestimmt die Auftrittswahrscheinlichkeit einer Rechnung über dem aktuellen Userlevel.

5.8 Statistische Auswertung

Eine sehr wichtige Komponente für die Lehrperson ist die statistische Auswertung der Daten, die Schülerinnen und Schüler produzieren. Mit Hilfe dieser Komponente wird es ihm ermöglicht, einen aktuellen Einblick in die Lernstände der Lernenden zu erhalten. Individuelle Fördermaßnahmen können so getroffen und der Unterricht angepasst werden. Viele Auswertungen von Daten bieten keinen ausreichenden Überblick und führen dazu, dass Lehrende erst durch eigene Berechnungen Nutzen daraus ziehen können. Der Plusminus-Trainer versucht eine einfache und aussagekräftige Auswertung für die Lehrperson zu liefern.

Für einen schnellen Überblick über die gesamte Klasse stellt der Trainer eine grobe Übersicht aller Schüler zur Verfügung. Die Auswertung, die in Abbildung 5.15 zu sehen ist, bedient sich der bewährten Ampelfarben, um die Daten darzustellen. Durch den kleinen Pfeil am linken Ende jeder Zeile, kann zusätzlich eine Statistik über die einzelnen Kompetenzlevel eingeblendet werden. Die in grün dargestellten Level hat der Lernende bereits erfolgreich abgeschlossen. Orange markiert ein Kompetenzlevel, das der Schüler oder die Schülerin gerade bearbeitet oder wenn bereits Beispiele aus diesem Level richtig beantwortet wurden, ohne je einen Fehler auf diesem Kompetenzlevel gemacht zu haben. Dies ist in manchen Fällen möglich, da eine gewisse Wahrscheinlichkeit besteht, eine Rechnung aus einem Kompetenzlevel über dem aktuellen zu erhalten. Jene Felder, die rot markiert sind, weisen auf Fehler in diesem Level hin. Die Zahl in den Kästchen gibt immer die Gesamtanzahl der bearbeiteten Aufgaben an. Fährt man mit der Maus über ein Feld, wird direkt daneben der Name des Kompetenzlevels angezeigt.

Aus der Abbildung 5.15 kann die Lehrperson bereits einige Schlüsse ziehen. Beim Schüler *Felix Faul* ist sehr einfach zu erkennen, dass bei ihm dringend eine Intervention notwendig ist, da er kein einziges Beispiel im Trainer bearbeitet hat. Betrachtet man die Statistik von *Markus Fleißig*, kann ebenfalls eine leichte Tendenz festgestellt werden. Auffallend ist, dass sich die Anzahl der Beispiele zwischen dem sechsten und siebenten Kompetenzlevel häufen. Dies kann auf ein Problem hindeuten, da genau bei diesen beiden Kompetenzlevel der Übergang zwischen den Kompetenzgruppen „Kein Übertrag“ und „Einfacher Übertrag“ stattfindet. Ein weiterführender Link ist in der Spalte „Details“ zu finden. Auf diesem Weg ist es dem Lehrer möglich, weitere und detailliertere Statistiken der Schüler abzurufen.

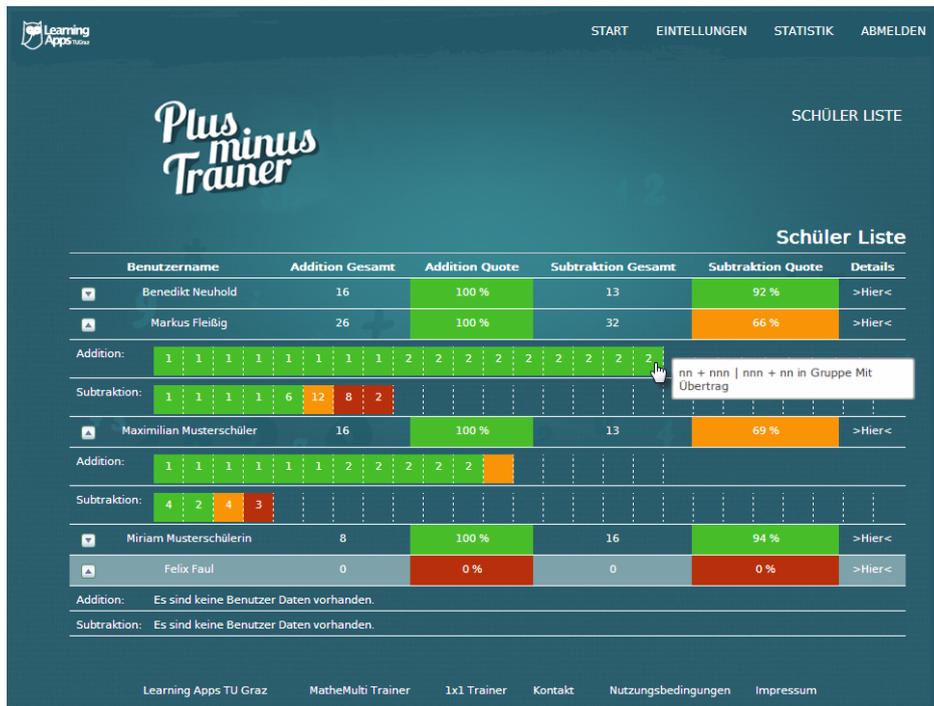


Abbildung 5.15: Statistische Übersicht für den Lehrer

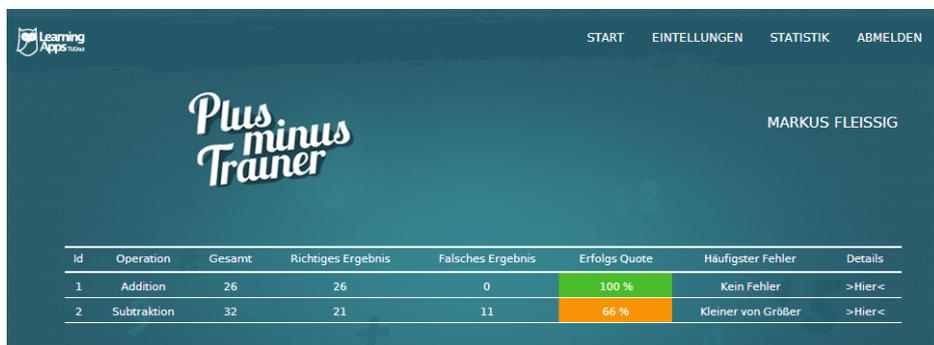


Abbildung 5.16: Zusammenfassende Ansicht über die Statistik der einzelnen Rechenoperationen

Eine solche detaillierte Statistik ist in Abbildung 5.16 zu sehen. Sie gibt einen groben Überblick über die beiden Rechenoperationen des Plusminus-Trainers und zeigt zusätzlich den am häufigsten erkannten Fehler an. Im Fall des Schülers *Markus Fleißig* in Abbildung 5.16 sind eine sehr gute Leistung in der Addition, jedoch einige Probleme in der Subtraktion zu erkennen. Der häufigste erkannte Fehler, der vom System ermittelt wurde, ist „Kleiner von Größer“. Dieser Fehler ist nicht als Aussage über die relative Häufigkeit des Auftretens zu sehen, sondern stellt nur das Maximum der absoluten Häufigkeiten dar. Um einen noch detaillierteren Einblick in die Lerndaten des Schülers zu bekommen, führt

der Link in der Spalte „Details“ weiter zur Benutzerstatistik, welche bereits in Abbildung 5.12 zu sehen war. Zusätzlich besteht die Möglichkeit, eine detaillierte Fehleranalyse, wie in Abbildung 5.17, durchführen zu lassen. In dieser Abbildung kann man sehr gut erkennen, dass der Schüler augenscheinlich ein Problem mit den Überträgen bei der schriftlichen Subtraktion hat. Alle möglichen Beispiele, in diesem Fall alle Beispiele mit Überträgen, wurden falsch gelöst. Durch diese Auswertung ist es dem Lehrer möglich, einen Überblick über die relative Häufigkeit der einzelnen Fehler zu gewinnen.



Abbildung 5.17: Statistische Auswertung der erkannten Fehler einer Rechenoperation

5.9 Datenbankmodell

Um die Daten der Schülerinnen und Schüler zu speichern und für weitere Analysen bereit zu stellen, setzt der Plusminus-Trainer, wie bereits in Abschnitt 5.1.3 beschrieben, auf das relationale Datenbanksystem MySQL. Um PHP-Objekte direkt in die Datenbank speichern zu können, wird die Object-relational mapping Software Doctrine verwendet (weitere Informationen in Abschnitt 5.1.4). Durch die hohe Dynamik des Trainers entstand ein komplexes Datenbankschema. Dieses ist in Abbildung 5.19 zu sehen und stellt alle Entities und ihre Beziehungen zueinander dar. Um einen groben Überblick über die Objekte in der Datenbank zu geben, werden in den nachfolgenden Abschnitten die wichtigsten Entities kurz erläutert.

Alle Entities erben von einer abstrakten Basisklasse `BaseEntity`, welche einige allgemeine Dinge definiert. Sie erstellt für jedes Entity einen Primärschlüssel mit dem Namen `id` und implementiert noch einige allgemeine Methoden.

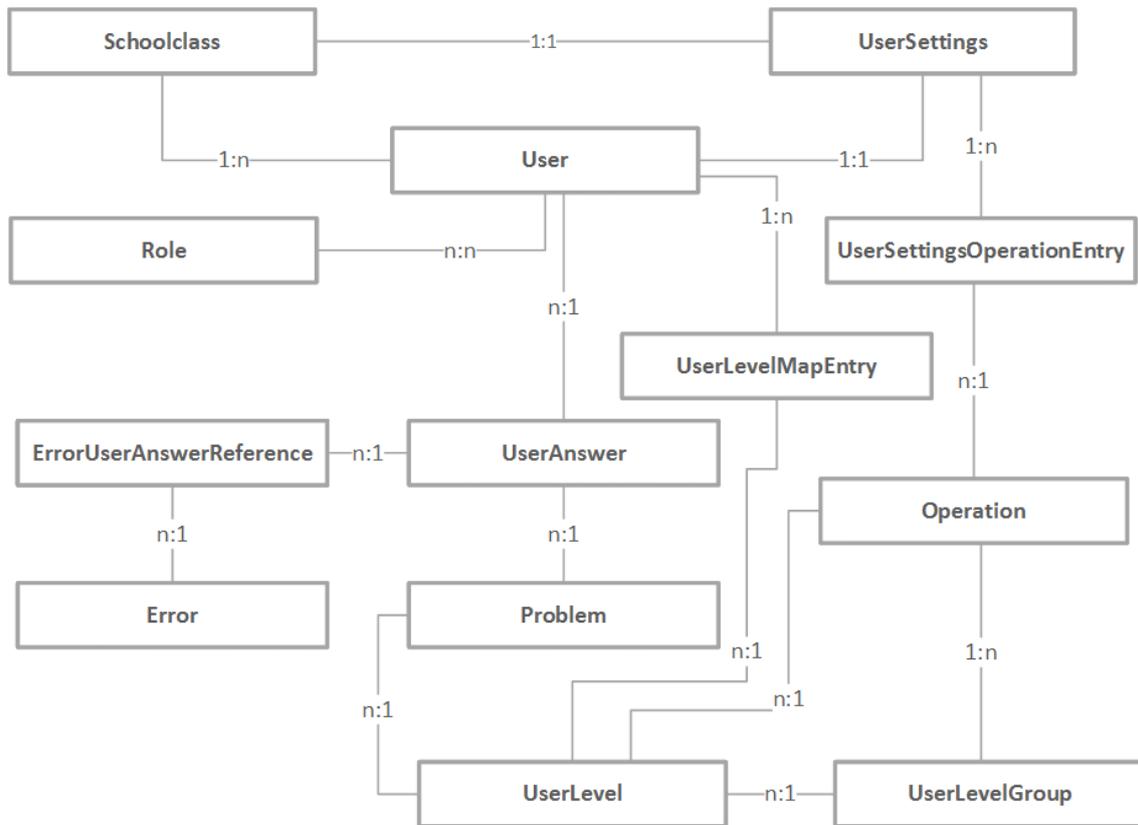


Abbildung 5.18: Entities im Plusminus-Trainer und ihre Beziehungen

5.9.1 Entity `UserAnswer`

Die Klasse `UserAnswer` enthält die Daten, die vom Benutzer oder der Benutzerin beim Lösen eines Beispiels gespeichert werden. Unabhängig von der Operation des Beispiels, werden in dieser Klasse sowohl das Resultat (`result`) als auch das Übertragresultat (`carryResult`) gespeichert. Zusätzlich werden der Zeitpunkt (`timestamp`) und die Lösungsdauer (`duration`), die für dieses Beispiel benötigt wurde, festgehalten. Um zusätzliche Datenbankabfragen zu verhindern, wird ebenfalls die Korrektheit der Eingabe in das Feld `correct` gesichert. Die Felder `resultEmptyPattern` und `carryEmptyPattern` beschreiben die Vollständigkeit der Eingabe. Für jede eingegebene Zahl steht im Pattern an dieser Stelle ein „n“. Sollte der Benutzer ein Formularelement leer gelassen haben, so erscheint an dieser Stelle im Pattern ein „e“. Damit ist es möglich, bei der Fehleranalyse leere Felder zu berücksichtigen. Die beiden Felder `user_id` und `problem_id` stellen jeweils Fremdschlüssel für die verknüpften Entities `Problem` und `User` dar (für eine Übersicht des Beziehungen siehe Abbildung 5.19).

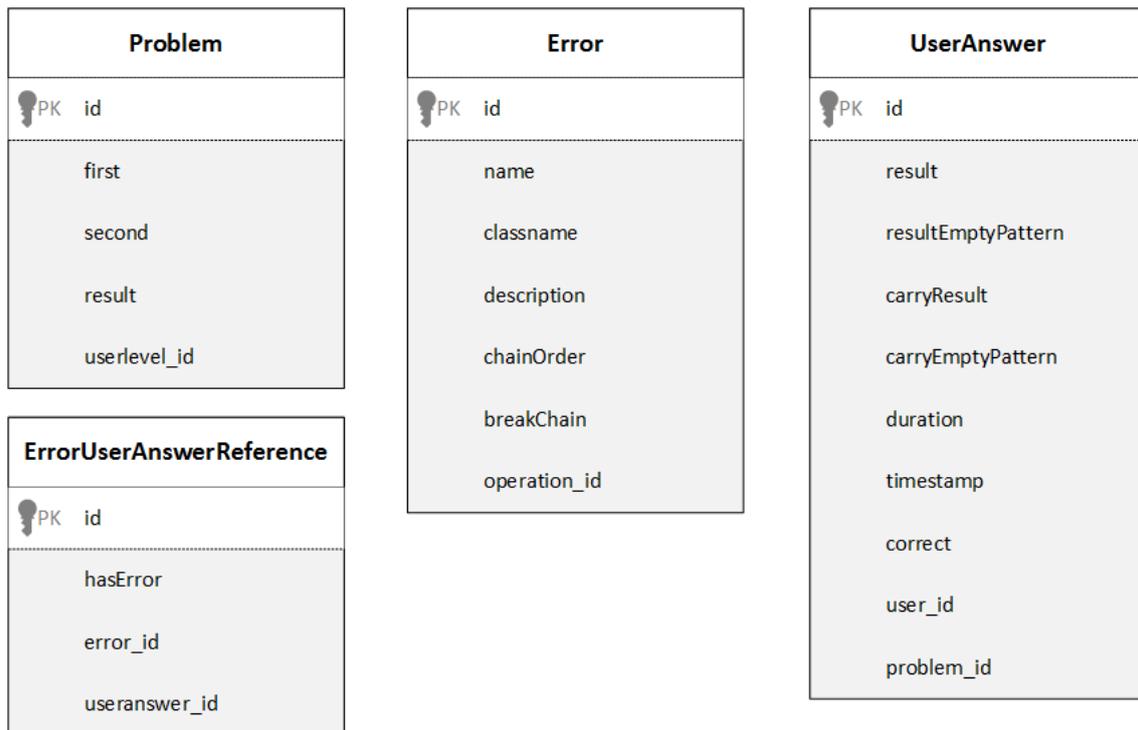


Abbildung 5.19: Die wichtigsten Entities des Plusminus-Trainers und ihre Attribute

5.9.2 Entities Error und ErrorUserAnswerReference

Die gesammelten Daten der Fehleranalyse werden mit Hilfe dieser beiden Entities in der Datenbank festgehalten. Das Entity ErrorUserAnswerReference dient dazu, um eine Verbindung zwischen einer Antwort des Lernenden und deren Fehleranalyse herzustellen. Die Tabelle dieses Entities enthält nur zwei Fremdschlüssel `error_id` und `useranswer_id`, welche Fehler und Benutzerantwort miteinander verknüpfen, und eine Variable `hasError` vom Typ `bool`, welche angibt, ob dieser Fehler bei der entsprechend verknüpften Benutzerantwort aufgetreten ist.

Die Klasse Error beschreibt einen Fehler näher und speichert Namen, Beschreibung, Reihenfolge der Analyse (`chainOrder`) und ob die Analyse beim Auftreten des Fehlers unterbrochen werden soll (`breakChain`). Das Feld `classname` gibt den gesamten Namespace der Klasse an, wo der Fehler implementiert wurde. Dieses Feld ist sehr wichtig, da die Fehleranalyse diese Klassen zur Laufzeit lädt. Der Fremdschlüssel `operation_id` verknüpft die einzelnen Fehler mit einer Rechenoperation, sodass nur die Fehler für die richtige Operation bei der Fehleranalyse berücksichtigt werden.

5.9.3 Entity Problem

Da die Beispiele nicht zur Laufzeit vom Trainer generiert werden, müssen sie in einer Datenbank gespeichert werden. Zuständig hierfür ist das `Entity Problem`. Da Addition sowie Subtraktion unterstützt werden sollen, wurde diese Klasse so einfach wie möglich gehalten. Die Felder `first` und `second` geben jeweils die Summanden bzw. Minuend und Subtrahend an und verwenden für die Speicherung einen Integer. Das `result` ist das korrekte Ergebnis des Beispiels in Abhängigkeit der Operation. Über den Fremdschlüssel `userlevel_id` wird jedes Beispiel mit einem Userlevel verknüpft und jedes Userlevel besitzt eine Verbindung zur entsprechenden Userlevelgruppe und zu seiner Operation.

5.10 Webservice

Um die Möglichkeiten des Plusminus-Trainers auch im mobilen Sektor einsetzen zu können, wurde ein Webservice entwickelt, das die wichtigsten Funktionen bereitstellt. Zu diesem Zeitpunkt sind einige mobile Applikationen in der Entwicklungsphase und greifen über diese Schnittstelle auf die Funktionen des Trainers zu. Durch die Verwendung des Webservices wird eine zentrale Datenverwaltung geschaffen und der Benutzer kann ortsunabhängig auf seine individuelle und adaptive Lernapplikation zugreifen. Das Webservice wird mit Hilfe des SoapServers des Zend Frameworks bereitgestellt und nutzt das Simple Object Access Protocol (SOAP¹³) in der Version 1.1 um die Daten zu übertragen. Für die Beschreibung der Schnittstelle wird die Web Services Description Language (WSDL¹⁴) ebenfalls in der Version 1.1 verwendet.

¹³<http://www.w3.org/TR/2000/NOTE-SOAP-20000508/> [Zugriff am 09.07.2013]

¹⁴<http://www.w3.org/TR/wsdl> [Zugriff am 09.07.2013]

6 Evaluation

In diesem Kapitel möchte ich auf die Vorbereitung, Durchführung und Nachbereitung der Evaluation eingehen. Die Evaluation einer Software ist ein wichtiger Bestandteil des Softwareentwicklungsprozesses und oft ausschlaggebend für Erfolg oder Misserfolg.

Die Evaluation des Plusminus-Trainers durfte ich an einer Volksschule in der Oststeiermark durchführen. Die Volksschule liegt in einer ländlichen Gegend und umfasst ca. 160 Schülerinnen und Schüler. Die Evaluation erfolgte am 13. Mai 2013 mit zwei vierten Klassen.

6.1 Vorbereitungen

Zu Beginn der Planung wurde Kontakt zur Schule aufgenommen und es gab sofort eine positive Resonanz. Dadurch konnte sehr schnell einen geeigneten Termin gefunden werden. Dieser fiel auf den 13. Mai 2013 von der ersten bis zur dritten Stunde.

Um eine solche Evaluation durchführen zu können, mussten einige Dinge bedacht werden. Zuerst war es nötig, die geeigneten Räumlichkeiten für den Versuch zu finden. Um die Evaluierung wirklich sinnvoll durchführen zu können wurde ein Computerraum mit siebzehn Einzelplätzen benötigt. Dieser wurde freundlicherweise von der Hauptschule des gleichen Ortes zur Verfügung gestellt. Zur Sicherung der Evaluation sollte die ganze Einheit auf Video festgehalten werden und am Ende der Einheit sollte qualitatives Feedback von den Schülerinnen und Schülern eingeholt werden. Dies sollte in Form eines Kreises, also im Plenum, geschehen und ebenfalls mit der Kamera festgehalten werden.

6.2 Durchführung

Die Evaluation wurde am 13. Mai 2013 mit zwei Klassen der vierten Schulstufe durchgeführt. Jeweils eine Schulstunde durfte mit jeder Klasse gearbeitet werden. Mit dem Gang in die Hauptschule und einer kurzen Einführung sowie einer Schlussrunde konnten die Kinder zwischen zwanzig und dreißig Minuten Zeit effektiv mit dem Plusminus-Trainer arbeiten.

Die Durchführung verlief so wie geplant. Die erste Klasse kam bereits in der Früh

in den Informatikraum der Hauptschule und verteilte sich auf die einzelnen Plätze. Es war sehr erfreulich, dass für jeden Schüler und jede Schülerin ein Platz zur Verfügung stand. So konnte jeder für sich mit dem Trainer arbeiten. Nach einer kurzen Vorstellung und Erklärung wurden die Zugangsdaten an die Schüler ausgeteilt. Um Zeit zu sparen, wurde jede Klasse bereits am Vortag im zentralen Usermanagement registriert. Ohne weitere Erklärungen begannen die Schülerinnen und Schüler mit dem Trainer zu arbeiten. Dies wurde bewusst so gewählt, um Feedback über die Einfachheit des Userinterfaces einzuholen. Nach ca. fünfundzwanzig-minütiger Arbeitszeit wurde die Übungseinheit beendet, und alle versammelten sich in einem Sesselkreis. Durch einige allgemeine Fragen wurde versucht, so viel Feedback wie möglich von den Schülerinnen und Schülern zu erhalten. Obwohl die Klassen etwas schüchtern waren, wurden einige Anregungen und Verbesserungsvorschläge eingebracht. Diese werden im Kapitel 6.3 (Ergebnisse) näher behandelt.

6.3 Ergebnisse

Bei der Evaluation wurde leider durch eine falsche Konfiguration in der Datenbank ein Kompetenzlevel bei der Subtraktion nicht berücksichtigt. Dieses entspricht dem Level 23 und enthält Subtraktionen vom Typ „ $nnn - n$ “ mit mindestens zwei Überträgen. Da jedoch nur eine Person dieses Level erreicht hätte, ist diese Auswirkung sehr gering. In den nachfolgenden statistischen Auswertungen wird dieses Kompetenzlevel nicht berücksichtigt und das maximale Level der Subtraktion wird von 25 auf 24 gesenkt.

In dieser Evaluierung nahmen 34 Schülerinnen und Schüler der vierten Klasse teil. Es wurden insgesamt 2678 Rechenaufgaben gelöst, was durchschnittlich ca. 79 Aufgaben pro Schüler oder Schülerin bedeutet. Die Erfolgsquote der Aufgaben lag bei 86,59%. Aus der Statistik von Abbildung 6.1 kann man ablesen, dass den Schülern die Subtraktion etwas schwerer fiel als die Addition. Die Verteilung der Wahrscheinlichkeiten der Operationen wurde in den Benutzereinstellungen für die Evaluation auf jeweils 50% gesetzt. Beim Test trat zu 49% eine Addition und zu 51% eine Subtraktion auf. Die durchschnittliche Zeit, die für eine Antwort benötigt wurde lag bei 16,77 Sekunden.

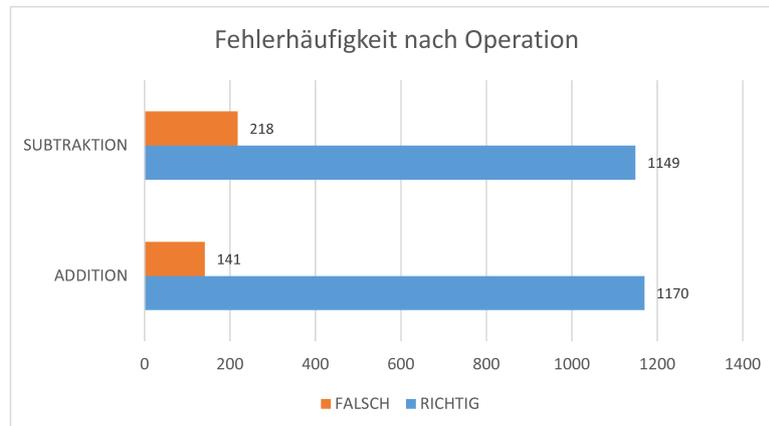


Abbildung 6.1: Grafische Darstellung der Antworten

Interessant ist der Verlauf des Levelanstiegs bei den beiden Rechenoperationen zu betrachten. Abbildung 6.2 zeigt den Verlauf des Kompetenzlevels aller Schüler über die Anzahl ihrer gelösten Aufgaben für die Addition. Das maximale Kompetenzlevel bei der Addition entspricht „ $nnn + nn$ oder $nn + nnn$ mit Übertrag“ und wird mit der Zahl 17 bezeichnet.

In Abbildung 6.3 ist dieselbe Auswertung für die Subtraktion zu sehen. In diesem Fall hat das schwierigste Kompetenzlevel die Zahl 24 und beinhaltet Subtraktionen der Form „ $nnn - nn$ “ mit mehr als einem Übertrag.

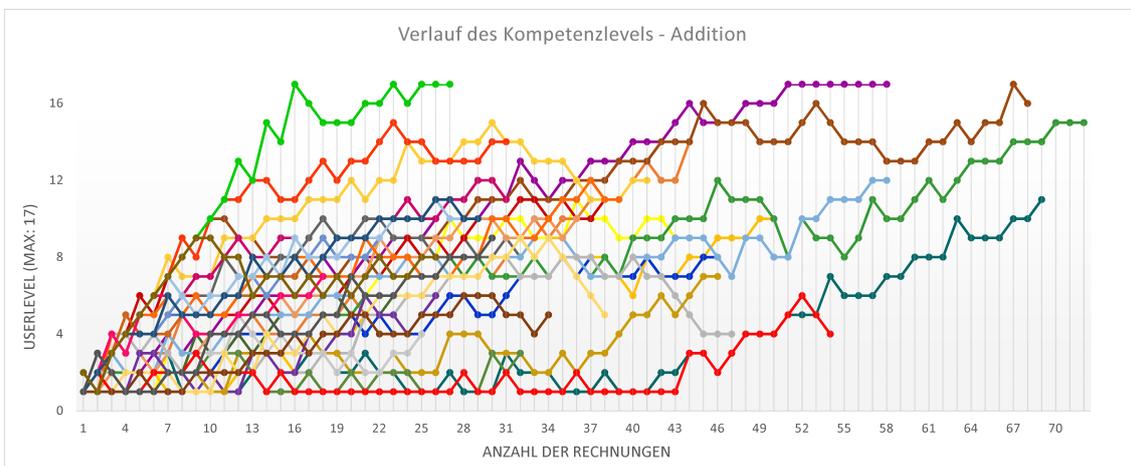


Abbildung 6.2: Anstieg des Userlevels bei der Addition

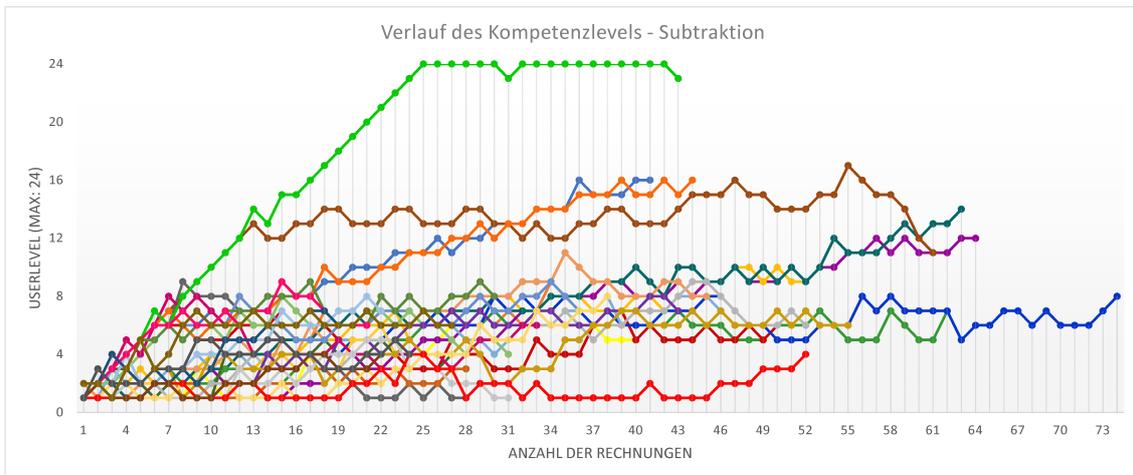


Abbildung 6.3: Anstieg des Userlevels bei der Subtraktion

Bei beiden Abbildungen sehen wir einen kontinuierlichen Anstieg des Kompetenzlevels. Betrachtet man die Abbildungen genauer, so ist zu erkennen, dass die Subtraktion für die meisten Schüler schwieriger ist als die Addition. Dies belegen auch die maximalen Kompetenzlevel beider Rechenoperationen, die in den Abbildungen 6.4 und 6.5 dargestellt sind. Das durchschnittliche Kompetenzlevel bei der Addition beträgt $\approx 9,2$, während hingegen bei der Subtraktion der durchschnittliche Wert nur bei ≈ 9 liegt. Es muss jedoch zusätzlich erwähnt werden, dass bei der Subtraktion höhere Kompetenzlevel möglich sind und daher der durchschnittliche Wert im Normalfall höher ausfallen sollte.

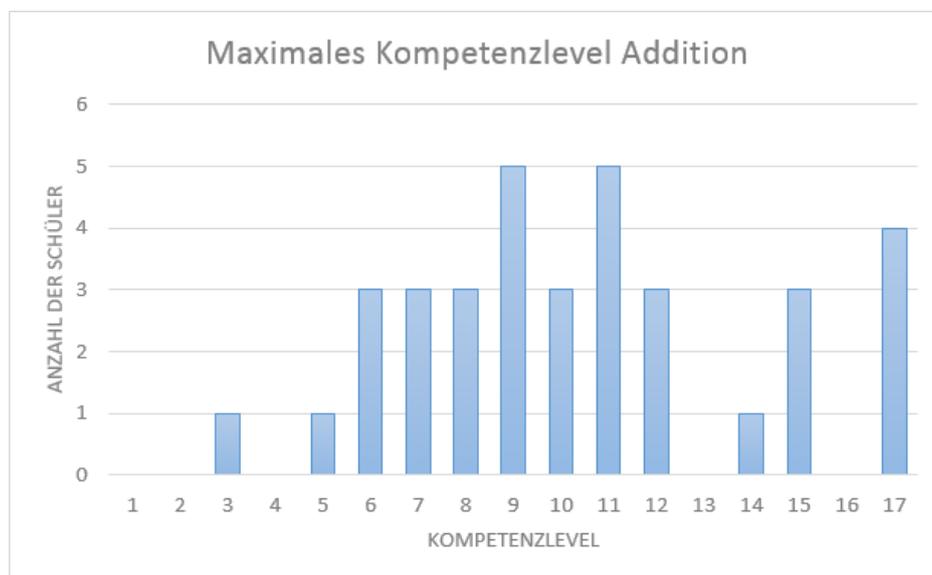


Abbildung 6.4: Maximal erreichtes Kompetenzlevel bei der Addition

Mehr als 50% der Schülerinnen und Schüler liegen bei der Subtraktion im Kompetenzbereich zwischen 7 und 9 (vgl. Abbildung 6.5). Dies ist insofern auffällig, da genau ab Kompetenzlevel 7 ein Übertrag zur Lösung der Rechnung benötigt wird. Eine genauere Aussage kann erst bei einer größeren Menge an Daten erhalten werden, jedoch gibt es schon eine leichte Tendenz, die auf die Schwierigkeit des Übertrags bei der Subtraktion hindeutet. Bei der Auswertung der Addition ist ein solches Phänomen nicht zu beobachten.

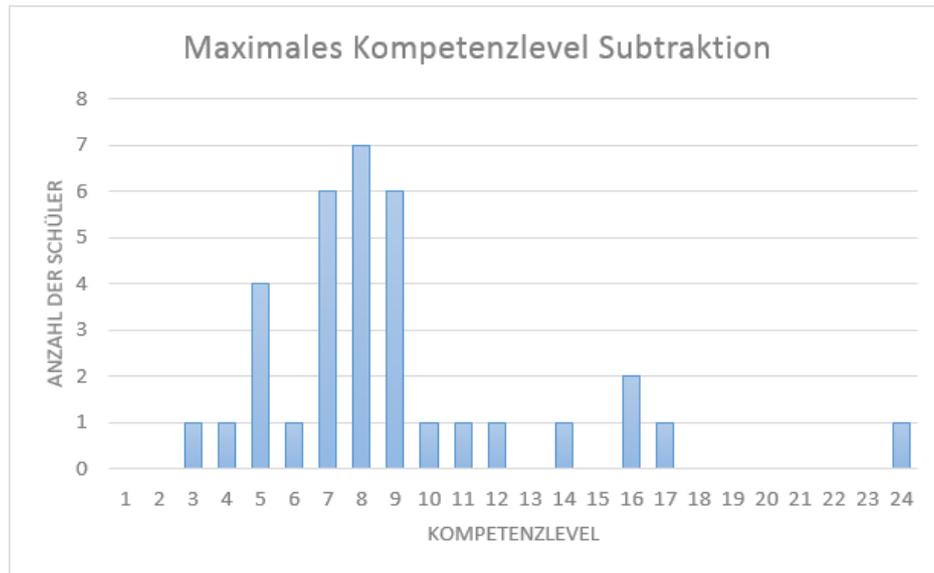


Abbildung 6.5: Maximal erreichtes Kompetenzlevel bei der Subtraktion

6.3.1 Unterschiedliche Lernertypen

Durch die Auswertung der gewonnenen Daten konnten unterschiedliche Lernertypen gefunden werden. Lernertypen bezeichnen in diesen Zusammenhang Lerner oder Lernerinnen, die gewisse charakteristische Merkmale aufweisen. Auch M. Schön et al. [2012] konnten bei ihrem Einmaleins-Trainer solche unterschiedlichen Lernertypen charakterisieren. Ähnliche Entdeckungen konnten ebenfalls bei der Datenanalyse des Plusminus-Trainers gemacht werden.

6.3.1.1 Der schwächere, demotivierte Lernertyp

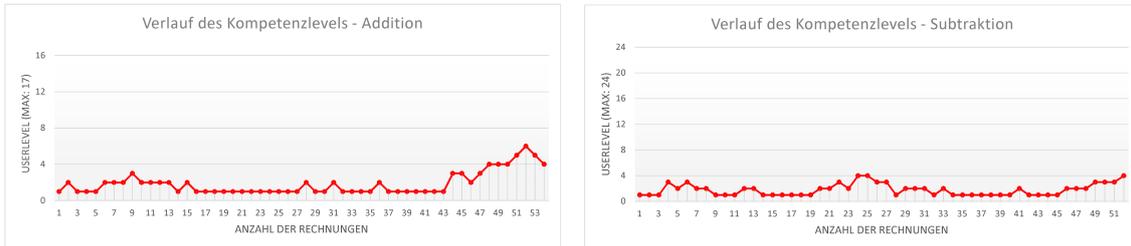


Abbildung 6.6: Kompetenzniveauverlauf eines schwächeren, demotivierten Lernertyps (ID: 93)

Dieser Lernertyp ist sehr auffallend, da er in Summe eine sehr große Anzahl an Beispielen gelöst hat (54 Additionen + 52 Subtraktionen = 108 Rechnungen in Summe), jedoch bereits bei diesen einfachen Rechnungen kaum einen Levelanstieg verzeichnen kann. Erst gegen Ende hin konnte er zumindest bei der Addition kurzzeitig das Kompetenzlevel 6 erreichen. Auffallend hierbei ist auch die kurze Antwortzeit. Im Durchschnitt wurden nur acht Sekunden für das Lösen einer Aufgabe verwendet. Daraus könnte man auf eine gewisse Demotivation des Lernertyps schließen. In diesem Fall ist es sehr wichtig, dass eine Intervention von Seiten der Lehrperson passiert.

6.3.1.2 Der starke Lernertyp

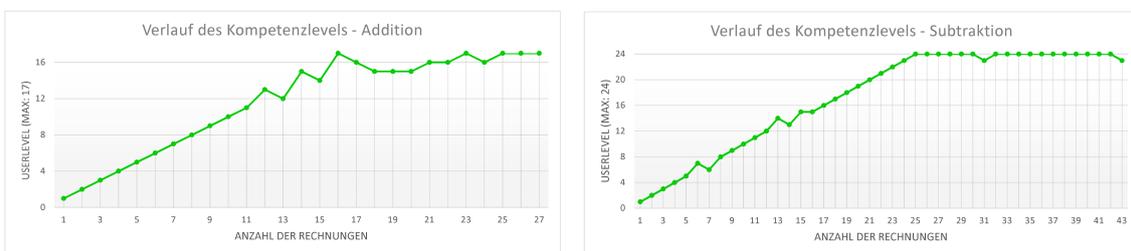


Abbildung 6.7: Verlauf der Kompetenzlevel eines starken Lernertyps (ID: 89)

Im Gegensatz zum schwachen Lernertyp hat dieser Typ eine wesentlich geringere Anzahl von Beispielen (27 Additionen + 43 Subtraktionen = 70 Rechnungen) gelöst. Jedoch kann man bereits anhand des Graphen erkennen, dass die Erfolgsquote wesentlich höher ausfällt als bei anderen Lernertypen. Je nur eine Rechnung bei der Addition und je eine bei der Subtraktion wurden mit einem falschen Ergebnis abgeschickt. Die anderen Sprünge im Graphen lassen sich auf die geringe Möglichkeit zurückführen, dass Rechnungen unter oder über dem aktuellen Kompetenzniveau ausgewählt werden. Dieser Lernertyp arbeitet auf einem sehr hohen Level und hat nach einer minimalen Zeitspanne das letzte Kompetenzniveau erreicht. In diesem Fall kann die Lehrperson davon ausgehen, dass die schriftliche Subtraktion sowie die schriftliche Addition von diesem Lernertyp beherrscht

werden. Mit diesem Wissen kann die Lehrperson gezielt neue Thematiken individuell mit diesem Lernertyp erarbeiten.

6.3.1.3 Der durchschnittliche Lernertyp

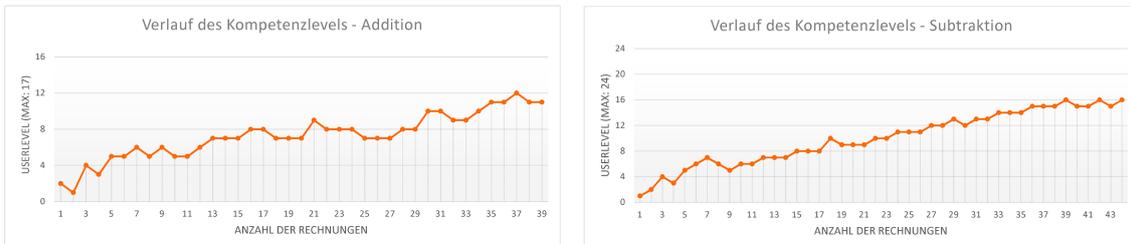


Abbildung 6.8: Verlauf der Kompetenzlevel eines durchschnittlichen Lernertyps (ID: 86)

Bei diesem Lernertyp ist in seiner Abarbeitung der Beispiele immer ein gewisser Anstieg zu erkennen. Die Lernkurve besteht immer wieder aus Höhen und Tiefen. Die Addition sowie die Subtraktion werden zwar grundsätzlich beherrscht und verstanden, jedoch schleichen sich immer wieder kleine Fehler ein. Insgesamt hat dieser Lernertyp 83 Rechnungen (39 Additionen + 44 Subtraktionen) gelöst und dabei jeweils 3 inkorrekte Antworten je Operation abgegeben. In diesem Fall ist nicht zwingend eine Intervention der Lehrperson nötig, da immer wieder ein gewisser Anstieg zu erkennen ist. Diesem Lernertyp wird es nur an der Übung und an Training fehlen.

6.3.1.4 Der festgefahrene Lernertyp

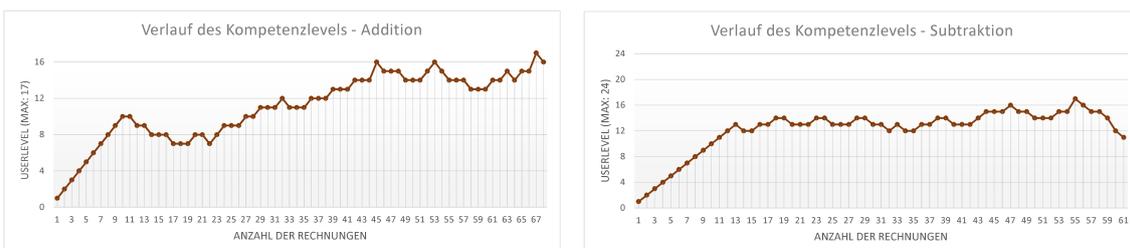


Abbildung 6.9: Verlauf der Kompetenzlevel eines festgefahrenen Lernertyps (ID: 71)

Dieser Lernertyp kann die Beispiele bis zu einem gewissen Kompetenzlevel ohne Probleme lösen, hat dann jedoch Probleme über dieses Level hinaus zu kommen. Dies ist besonders in der Subtraktion in Abbildung 6.9 zu erkennen. Die ersten 13 Subtraktionen, als auch die ersten 10 Additionen wurden ohne Fehler gemeistert, wodurch die Lernkurve zu Beginn kontinuierlich steigt. Besonders auffallend bei der Subtraktion ist, dass ab diesem Zeitpunkt kein wirklicher Anstieg des Kompetenzlevels mehr zu erkennen ist. Dies lässt den Schluss zu, dass dieser Lernertyp bei einer gewissen Schwierigkeit ein

Problem hat. In diesem Fall könnte ein möglicher Grund das Auftreten von Nullen in der Angabe sowie im Ergebnis im Zusammenspiel mit einem Übertrag sein. Diese Nullen mit Übertrag treten das erste Mal bei Kompetenzlevel 11 auf und reichen bis Kompetenzlevel 16 zurück. Dieses Kompetenzlevel wird von diesem Lernertyp nicht überschritten. In diesem Fall ist es sehr wichtig, dass die Lehrperson eingreift und gezielt dem Problem auf den Grund geht.

6.3.2 Fehlermuster

Durch die Daten und deren Analyse des Plusminus-Trainers konnten einige interessante Ergebnisse gewonnen werden. Diese werden in diesem Abschnitt kurz erläutert.

Insgesamt wurden 359 von 2678 Aufgaben fehlerhaft gelöst. Nicht jeder fehlerhaften Aufgabe kann zweifelsohne in fehlerhaftes Denkmuster zugeordnet werden. Über die gesamten Aufgaben verteilt wurden die Fehler in Tabelle 6.1a bei der Addition sowie die Fehler in Tabelle 6.1b bei der Subtraktion am häufigsten festgestellt.

Beide Operationen enthalten die falsche Rechenoperation und Eingabefehler in den Top 3 der Fehlerliste. Meiner Meinung nach kann man die Häufigkeit der falschen Rechenoperationen sowie die Eingabefehler auf die Einführungsphase zurückführen. Man muss bedenken, dass es für die Benutzung des Trainers keine speziellen Instruktionen gab und die Schüler selbst versuchen mussten damit umzugehen. Die Auswertung des relativen Fehlers pro Schüler stützen diese Annahme weiter. Diese ergab, dass der relative Fehler pro Schüler für den Fehler der falschen Rechenoperation nie über 21% liegt. Daher kann vielleicht von einem typischen jedoch sicher nicht von einem systematischen Fehler gesprochen werden.

Fehler	Häufigkeit	Fehler	Häufigkeit
Subtraktion anstatt Addition	26	Addition anstatt Subtraktion	51
Eingabefehler	25	Eingabefehler	28
Kein Übertrag	10	Kleiner von Größer	27

(a) Häufige Fehler bei der Addition

(b) Häufige Fehler bei der Subtraktion

Tabelle 6.1: Häufige Fehler in den Rechenoperationen

Der letzte Fehler der Addition „Kein Übertrag“ ist auf Grund seines geringen absoluten und noch geringeren relativen Auftretens pro Schüler eher als Unaufmerksamkeitsfehler zu verbuchen und stellt keine grobe Fehlerstruktur dar. Ebenso könnte eine falsche Eingabe durch Vertippen diesen Fehler hervorrufen.

Der dritte Fehler der Subtraktion tritt wesentlich öfter auf als jener der Addition. Dieser Fehler und seine relativen Häufigkeiten wurden konkret unter die Lupe genommen, und

so konnten zwei Schüler gefunden werden, die einen Ansatz eines typischen oder sogar eines systematischen Fehlers aufwiesen. Aus datenschutzrechtlichen Gründen wurden die Namen der Schüler durch fiktive Namen ersetzt.

In Abbildung 6.10 sehen wir die Klassenstatistik von Norbert. Man kann bereits erkennen, dass bei der Addition kein Problem besteht, sich jedoch die Aufgaben beim Kompetenzlevel 6 der Subtraktion häufen. Wichtig zu wissen ist, dass ab dem 7. Kompetenzlevel Beispiele mit einem Übertrag behandelt werden. Dies ist bereits ein Indikator, dass Norbert ein Problem mit Subtraktionen, die einen Übertrag erfordern, hat. Daher sehen wir uns die detaillierte Fehlerstatistik von Norbert an (siehe Abbildung 6.11). Hier ist schon sehr gut zu erkennen, dass es sich bei diesem Fehler wirklich um ein fehlerhaftes Denkmuster zu handeln scheint. Von neun möglichen Fällen wurde neun mal diese Rechenstrategie verwendet, um die Aufgabe zu lösen.

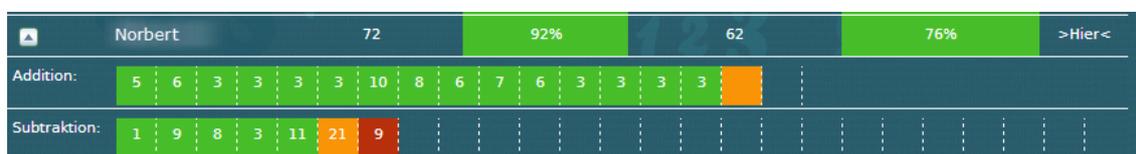


Abbildung 6.10: Klassenstatistik von Norbert

Subtraktion				
Häufige Fehler				
Fehler	Mögliche	Fehler	Fehlerquote	Details
Kleiner von Größer	9	9	100%	>Hier<
Addition statt Subtraktion	62	1	2%	>Hier<
Eingabefehler	62	1	2%	>Hier<
Kein Übertrag	9	1	11%	>Hier<

Abbildung 6.11: Fehlerstatistik von Norbert

Bei einer Schülerin, nennen wir sie Jana, führt uns ebenfalls der gleiche Indikator auf die Spur. In Abbildung 6.12 sehen wir die Überblickstatistik des Lehrers. Diese lässt die gleiche Häufung von Beispielen zwischen dem 6. und 7. Kompetenzlevel erkennen. Bei solchen Anzeichen ist es sehr sinnvoll, die Fehlerstatistik aufzurufen. Diese gibt einen Überblick über die relative Häufigkeit der einzelnen Fehlertypen. Diese Statistik ist in Abbildung 6.13 zu sehen. Um eine konkrete Aussage in diesem Fall treffen zu können, müssten mehr Daten vorhanden sein. Es zeigt sich jedoch auch hier eine klare Tendenz in Richtung fehlerhaftes Rechenmuster.

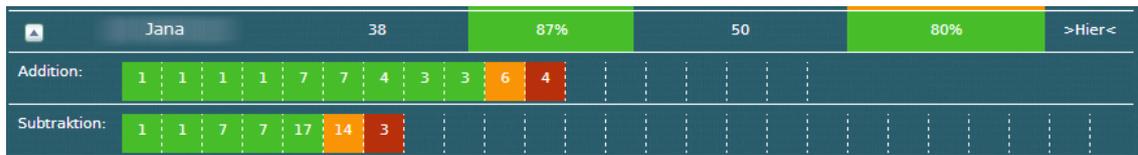


Abbildung 6.12: Klassenstatistik von Jana

Häufige Fehler					Subtraktion
Fehler	Mögliche	Fehler	Fehlerquote	Details	
Addition statt Subtraktion	50	3	6%	>Hier<	
Kleiner von Größer	3	2	67%	>Hier<	
Eingabefehler	50	1	2%	>Hier<	
Kein Übertrag	3	1	33%	>Hier<	

Abbildung 6.13: Fehlerstatistik von Jana

Um wirklich mehr Schlüsse ziehen zu können, müssten mehr Daten vorhanden sein. Es sind jedoch oftmals schon klare Tendenzen zu erkennen, denen der Lehrer oder die Lehrerin nachgehen kann.

6.3.3 Feedback

In diesem Abschnitt wird kurz das Feedback aus den beiden Schlussrunden zusammengefasst. Hierbei wird zwischen dem direkten Feedback der Schüler und dem Feedback aus persönlichen Beobachtungen unterschieden.

Die Grundtendenz war sehr positiv und es hat allen sehr gut gefallen. Konkret wurden trotzdem einige Verbesserungsvorschläge geäußert:

- **Druck Layout:** Dieser Vorschlag kam von einer Lehrperson, die sich gerne die Statistik ausdrucken wollte. Dies ist leider zur Zeit nur mit einigem Aufwand möglich (Copy & Paste in Word und Formatierung). Hier könnte man einen CSS Style einbauen, der die Statistik für das Drucken formatiert.
- **Sounds:** Einige Schüler kamen auf die Idee, bei richtiger Antwort einen Sound abzuspielen.
- **Fehlerhafte Eingaben nicht zulassen:** Bei einigen Schülern war zu erkennen, dass sie anstatt einer Null zuerst ein großes O verwendet haben. Dies führte zu einem Fehler. Hier meinten die Schüler, dass es besser sei, diese Lösung nicht zu akzeptieren und darauf hinzuweisen, dass nur Zahlen erlaubt sind.
- **Textbox anstatt Kästchen:** Einige Schüler würden gerne bei einfachen Rechnungen die Zahl von vorne nach hinten in einem durchschreiben. Dies ist jedoch mit

der jetzigen Implementierung nicht möglich, da jeder Stellenwert eine eigene Textbox besitzt. Da dies, meiner Meinung nach, auch nicht dem schriftlichen Rechenverfahren entspricht, glaube ich, dass hier keine Änderungen zu erwarten sind.

Durch die Evaluierung selbst und durch die Durchsicht der Aufnahmen konnten noch einige Punkte entdeckt werden:

- **Verhalten bei falscher Antwort:** Es war oft zu bemerken, dass Schülerinnen und Schüler versucht haben, bei falscher Antwort diese auszubessern und nicht auf den Button „Weiter“ zu klicken. Ein Verändern der Antwort ist jedoch nicht möglich, da sonst die Statistik durch das Abschreiben der richtigen Antwort verfälscht würde. Dies könnte man meiner Meinung nach mit zwei Ansätzen verbessern. Der erste Ansatz wäre, dass man die Meldung und den Button der richtigen Lösung größer und auffälliger gestaltet. Der zweite Weg (ging aus einem Input der Lehrperson hervor) wäre, dass man bei falscher Lösung zwei Buttons anzeigt. Der Button „Lösung anzeigen“ zeigt die Lösung an und der zweite Button „Noch einmal versuchen“ gibt dem Benutzer das selbe Beispiel ein weiteres Mal zur Bearbeitung.
- **Leere Eingabe:** Der Fehler „Eingabefehler“ war sehr häufig zu sehen. Dieser Fehler gibt an, dass die Benutzerin bzw. der Benutzer alle Eingabefelder leer gelassen hat. Dies kann zum Beispiel durch das versehentliche Drücken der Enter-Taste passieren. Hier könnte man eine zusätzliche Abfrage einbauen, ob die Benutzerin bzw. der Benutzer diese Daten wirklich abschicken möchte.
- **Einige Anzeigefehler:** Durch die Evaluierung entstand eine große Menge an Daten. Erst durch diese Vielfalt an Daten war es möglich, einige Anzeigefehler zu erkennen und diese zu beseitigen.

7 Diskussion

Das folgende Kapitel wird auf die Vorteile und etwaige Defizite der entwickelten Software eingegangen.

Nach Romero und Ventura [2010, S. 611-612] stehen bereits Tools und Programme für Lehrpersonen zur Auswertung von Daten zur Verfügung, jedoch sind diese für den normalen Verwender zu komplex aufgebaut und brauchen zusätzliches Wissen im Gebiet von Learning Analytics. Beim Aufbau des Plusminus-Trainers wurde daher darauf geachtet, dass er eine einfache, intuitive und verständliche Oberfläche bietet und kein weiteres Knowhow des Benutzers oder der Benutzerin benötigt wird. Durch den hierarchischen Aufbau der Statistiken und der Arbeit mit dem bekannten Ampelsystem wurde ein einfach zu verstehendes System geschaffen. Ebenso wurde auf wichtige Punkte, die bereits im Entwicklungsprozess von eLAT bedacht wurden (siehe Abschnitt 2.1.1.2), eingegangen. Es wurde großer Wert auf die Erweiterbarkeit des gesamten Systems gelegt und durch den Einsatz diverser Technologien eine Wiederverwendbarkeit gewährleistet. Zudem wird durch die sofortige Fehleranalyse nach Eingabe der Lösung eine Echtzeitverarbeitung und eine immer aktuelle statistische Auswertung ermöglicht.

Der Plusminus-Trainer wurde in erster Linie als Übungs- und Diagnosewerkzeug für Schüler, Eltern und Lehrer entwickelt und kann nicht als „Intelligent Tutoring System“ (ITS) gesehen werden. Der Trainer wurde als Hilfestellung für den Lehrer implementiert, um den Schüler beim Lernen zu unterstützen und nicht um den Lehrer zu ersetzen. Dieser Ansatz folgt dem Learning Analytics Cycle welcher in Abschnitt 2.3 beschrieben wird. Schüler sind darauf angewiesen, dass bei unüberwindbaren Problemen eine Intervention des Lehrers stattfindet. Wann diese Intervention statt zu finden hat, musste bis jetzt der Lehrer basierend auf persönlichen Eindrücken, Schularbeiten oder Lernzielkontrollen entscheiden. Mit dem Plusminus-Trainer besteht jetzt die Möglichkeit, dass diese gesammelten und aufbereiteten Lerndaten diese Entscheidung erleichtern. Zusätzlich konnten aus den Daten solcher Systeme interessante Informationen zum Lernen gewonnen werden. Martin Ebner und M. Schön [2013] beschreiben zwei ähnliche Systeme und konnten über einen langen Zeitraum feststellen, dass in irgendeiner Weise lernen passieren muss. Ein Schüler der lange auf einem Kompetenzniveau arbeitet steigt von einem auf den anderen Moment auf. Wo das Lernen passiert, ob dies durch die Intervention des Lehrers oder durch Nachfragen bei Eltern oder Freunden, kann nicht gesagt werden.

Einen weiteren nennenswerten Vorteil der implementieren Software bringt die Entwicklung als Webapplikation. Wie Martin Ebner, M. Schön und Nagler [2012] beschreiben, besitzen der Großteil der Schülerinnen und Schüler bereits ein Smartphone oder Tablet und verfügen zuhause über einen Zugriff auf das Internet. Der Trend geht tendenziell in die Richtung, dass Internet auch auf den mobilen Devices genutzt wird. Durch die Kompatibilität des Trainers mit allen gängigen Browsern ist es nahezu von überall aus möglich diesen zu benutzen. Da die Anwendung auf den mobilen Geräten mit geringer Displaygröße leider einige Anzeigeprobleme aufweist, wurde ein Webservice bereitgestellt. Zwei Projekte befinden sich derzeit in der Entwicklungsphase und sollen die Funktionalität des Trainers als App für die Plattformen iOS und Android in Zukunft zu Verfügung stellen.

Betrachten wir die entwickelte Software, so kann sie grob in drei Kernkomponenten geteilt werden. Als erste Komponente können wir die Rechnerkomponente sehen. Diese gibt verschiedene Rechnungen aus und gibt, je nach Benutzerantwort, richtig oder falsch zurück. Solche Trainingsprogramme sind im Internet zur Genüge zu finden. Die statistische Auswertung kann als zweite Komponente gesehen werden und gibt Eltern und Lehrern eine Überblick wie viel und wie erfolgreich ein Schüler gearbeitet hat. Diese Komponente ist bei freien online Übungsprogrammen für Mathematik nur selten zu finden und stellt daher einen besonderen Pluspunkt des Plusminus-Trainers dar. Die wichtigste Komponente stellt jedoch die dritte da - die Fehleranalyse. Dem Autor ist keine frei zugängliche Mathematiksoftware für die Addition und Subtraktion bekannt, die eine Echtzeit-Fehleranalyse unterstützt. Dadurch können Lehrer Zeit sparen, da keine diagnostischen Tests mehr zusammengestellt und korrigiert werden müssen und die Schüler empfinden in den meisten Fällen eine größere intrinsische Motivation wenn sie mit dem Computer arbeiten dürfen. Für die Eltern oder die Lehrpersonen wird es zudem einfacher, die Fehler der Kinder zu beheben. Im Gegensatz zur Aussage „Der Schüler ist schlecht bei der Addition“ ist die Erkenntnis „Der Schüler glaubt $0 + a = 0 = a + 0$ “ wesentlich detaillierter und die Fehlerstruktur in weiterer Folge einfacher zu beheben.

Die Anzahl der implementierten Fehler ist zur Zeit noch gering. Vom System werden 13 unterschiedliche Fehler der Addition und Subtraktion erkannt. Dies ist in Anbetracht der möglichen Fehler, die in Kapitel 3 behandelt werden, eine sehr geringe Zahl. Die Evaluation in Kapitel 6 hat jedoch gezeigt, dass bereits bei dieser geringen Anzahl der Daten und der Fehler, systematische Strukturen erkannt werden. Zusätzlich muss bedacht werden, dass die Software erst am Beginn steht und Fehler durch den einfachen modularen Aufbau mit wenig Aufwand zum Programm hinzugefügt werden können. In der aktuellen Version 1.0 der Software fehlt noch ein automatisierte Testung der Fehler durch Unit-Tests. Dies wird in Zukunft anzustreben sein, um die Fehler wirklich gut erweitern und implementieren zu können. Ebenso sollte in naher Zukunft eine Möglichkeit geschaffen werden, kombinierte Fehlermuster, ähnlich wie BUGGY (siehe Abschnitt 4.1), zu erkennen und zu diagnostizieren.

Um den Trainer auch über die Grenzen Österreichs hinaus verwenden zu können, wurden bereits bei der Entwicklung einige Dinge bedacht. Der Trainer steht zur Zeit in den Sprachen Deutsch und Englisch zur Verfügung und kann mit geringem Aufwand durch neue Übersetzungsfiles um andere Sprachen erweitert werden. Zusätzlich wurde die Möglichkeit bedacht, dass in anderen Ländern andere Rechenverfahren der Subtraktion eingesetzt werden (siehe auch Abschnitt 3.2.1). Durch die Möglichkeit die Überträge am oberen oder am unteren Rand der Rechnung anzuzeigen, kann das Beispiel mit Hilfe beider Verfahren gelöst werden. Diese Einstellung der Platzierung der Überträge kann pro Benutzer oder pro Klasse getroffen werden (näheres siehe Abschnitt 5.7).

Innere Differenzierung und individuelles Eingehen auf Schülerinnen und Schüler sind in der heutigen Zeit wichtige Techniken, die Lehrpersonen auf alle Fälle verwenden sollten. Der Trainer unterstützt das Lehrpersonal bei dieser Aufgabe und führt, durch den adaptiven Aufbau, bereits während des Arbeitens eine innere Differenzierung durch. Dies geschieht durch die angepasste Auswahl der Beispiele für jeden Schüler und jede Schülerin. Zusätzlich kann die Lehrperson durch die detaillierte Auswertung der Ergebnisse individueller auf einzelne Schüler bzw. auf die einzelnen Schülerinnen und deren Problemgebiete eingehen und diese gezielt fördern.

8 Zusammenfassung und Ausblick

Um eine Software wie den Plusminus-Trainer zu schreiben, ist es notwendig sich zuerst über die verschiedenen Möglichkeiten der Datensammlung und Analyse zu informieren. In dieser Arbeit wurden im ersten Kapitel der Forschungszeit von Learning Analytics näher betrachtet und wichtige Kernpunkte, die für den Trainer von Bedeutung sind, beleuchtet. Betrachtet man die bisher entwickelten Learning Analytics Tools, so ist aus der Sicht von Lehrerinnen und Lehrern zu sagen, dass sie in den meisten Fällen zu komplexe Einstellungen benötigen und daher schwer im Unterricht zu verwenden sind. Da bereits an fast jeder Schule Lernmanagement Systeme (LMS) eingesetzt werden, wurde ein großer Fokus auf Learning Analytics Tools gelegt, die Lehrpersonen bei der Beurteilung und Auswertung der Daten aus den LMS, unterstützen. Dabei wurden wichtige Schritte im Entwicklungsprozess sowie ihre Funktionsweisen erläutert und betrachtet. Hier konnte man vor allem die Wichtigkeit der einfachen Handhabung der Software sowie einer gut aufbereiteten und einfach zu lesenden Statistik erkennen. Um einen kurzen Überblick über die Möglichkeiten solcher Software zu geben, wurden ebenfalls Tutoring-Systeme kurz vorgestellt. Mit dem Learning-Analytics-Cycle wurde noch einmal die Wichtigkeit der Intervention von Seiten des Lehrers betont und versucht diesen mit etablierten Lerntheorien zu verknüpfen.

Schriftliche Rechenverfahren wurden auf Schwierigkeiten und Fehler untersucht, um diese in späterer Folge im Trainer zu implementieren. Durch das Aufzeigen der unterschiedlichen Rechenverfahren bei der schriftlichen Subtraktion konnte diese Problematik bereits im Entwicklungskonzept mitbedacht und gelöst werden. Systematische Fehlerstrukturen zu erkennen, war den Lehrpersonen bis zu diesem Zeitpunkt nur durch den Einsatz von diagnostischen Tests möglich. Der Aufbau sowie die Vor- und Nachteile wurden diskutiert und konkrete Beispiele für die Addition und Subtraktion gegeben. Zum Abschluss dieses Kapitels wurde durch Betrachten des Volksschullehrplans ein Bezug zum österreichischen Curriculum hergestellt.

Die Evaluation der mathematischen Lernsoftware konnte aufzeigen, dass so gut wie keine freie Software mit den Ansätzen des Plusminus-Trainers zur Verfügung steht. Obwohl bereits sehr früh Anstrengungen in diese Richtung (BUGGY) unternommen wurden, konnten kaum äquivalente Programme gefunden werden. Der Trend scheint in diesem Fall eher in Richtung Intelligente Tutoring Systeme zu gehen.

Um einen Einblick in die Arbeitsweise des Trainers zu geben werden die Ideen und

Überlegungen der Konzeptionsphase kurz vorgestellt. Die modulare Implementierung und die dadurch resultierende einfache Erweiterbarkeit stellt das Herzstück des Trainers da. Durch eine Reihe von Einstellungsmöglichkeiten ist es Lehrpersonen möglich, das Verhalten des Trainers auf ihre Wünsche hin anzupassen. Die Software wurde für den Einsatz an der Grundschule entwickelt und soll eine Unterstützung für die Lehrerinnen und Lehrer darstellen. Um dies zu gewährleisten konnte eine Evaluierung an einer Volksschule durchgeführt werden. Durch diese Evaluierung konnten wichtige Erkenntnisse für weitere Verbesserungen des Trainers gewonnen werden. Zudem konnte das System mit einer größeren Anzahl von Daten getestet und erste Fehlerstrukturen diagnostiziert werden.

Um eine gute Aussage über den Entwicklungsstand der Applikation zu treffen werden mehr Daten benötigt. Mit der Freischaltung des Plusminus-Trainers für die Öffentlichkeit wird erhofft, dass diese Daten gesammelt und analysiert werden. Die gewonnenen Erkenntnisse der Evaluation werden auf alle Fälle diskutiert und, sofern möglich, in Zukunft implementiert werden. Um eine noch bessere Erkennung von Fehlerstrukturen und Fehlermuster zu gewährleisten muss die Analyse in Zukunft erweitert und verfeinert werden. Der Trainer stellt jedoch bereits jetzt eine gute Unterstützungsmöglichkeit für den Diagnoseprozess von Fehlern bei der Addition und Subtraktion da und der Einsatz an Grundschulen würde einen großen Mehrwert für Lehrer und Schüler mit sich bringen.

Literatur

- 1st International Conference Learning Analytics & Knowledge (16. Mai 2013). *Call for Papers — 1st International Conference on Learning Analytics and Knowledge 2011*. URL: <https://tekri.athabascau.ca/analytics/call-papers> (besucht am 16.05.2013) (siehe S. 7).
- Bader-Natal, Ari und Thomas Lotze (2011). »Evolving a learning analytics platform«. In: *Proceedings of the 1st International Conference on Learning Analytics and Knowledge*. LAK '11. Banff, Alberta, Canada: ACM, S. 180–185. ISBN: 978-1-4503-0944-8 (siehe S. 14, 15).
- Baker, Ryan S. J. d., Erik Duval, John Stamper, David Wiley und Simon Buckingham Shum (2012). »Educational data mining meets learning analytics«. In: *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*. LAK '12. Vancouver, British Columbia, Canada: ACM, S. 20–20. ISBN: 978-1-4503-1111-3 (siehe S. 4, 7).
- Bender, P. und Gesellschaft für Didaktik der Mathematik. Arbeitskreis Mathematikunterricht und Informatik (2003). *Lehr- und Lernprogramme für den Mathematikunterricht: Bericht über die 20. Arbeitstagung des Arbeitskreises "Mathematikunterricht und Informatik der Gesellschaft für Didaktik der Mathematik e.V. vom 27. bis 29. September 2002 in Soest*. Bericht über die ... Arbeitstagung des Arbeitskreises "Mathematikunterricht und Informatik der Gesellschaft für Didaktik der Mathematik e.V. Franzbecker. ISBN: 9783881203517 (siehe S. 41).
- Bloom, Benjamin S. (Juni 1984). »The 2 Sigma Problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring«. In: *Educational Researcher* 13.6, S. 4–16 (siehe S. 17).
- bmukk (2012). *Lehrplan der Volksschule — BGBl. Nr. 134/1963 in der Fassung BGBl. II Nr. 303/2012 vom 13. September 2012*. URL: http://www.bmukk.gv.at/medienpool/14055/lp_vs_gesamt.pdf (besucht am 28.05.2013) (siehe S. 39, 40).
- Bongers, F. und M. Vollendorf (2011). *jQuery: Das Praxisbuch*. Galileo Computing. Galileo Press GmbH. ISBN: 9783836218108 (siehe S. 57).
- Brown, John Seely und Richard R. Burton (1978). »Diagnostic Models for Procedural Bugs in Basic Mathematical Skills*«. In: *Cognitive Science* 2.2, S. 155–192. ISSN: 1551-6709 (siehe S. 41–43).
- Brownell, William A. und Harold E. Moser (1949). *Meaningful Vs. Mechanical Learning: A Study in Grade III Subtraction*. Duke University research studies in education. Duke University Press (siehe S. 40).

- Brynjolfsson, Erik, Lorin M. Hitt und Heekyung H. Kim (24. Apr. 2011). »Strength in Numbers: How Does Data-Driven Decisionmaking Affect Firm Performance?« In: *Social Science Research Network Working Paper Series* (siehe S. 3).
- Campell, John P. und Diana G. Oblinger (2007). »Academic Analytics«. In: *EDUCAUSE Quarterly* Oktober 2007 (siehe S. 18).
- Clow, Doug (2012). »The learning analytics cycle: closing the loop effectively«. In: *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*. LAK '12. Vancouver, British Columbia, Canada: ACM, S. 134–138. ISBN: 978-1-4503-1111-3 (siehe S. 19–21).
- Dillenbourg, Pierre, Sanna Järvelä und Frank Fischer (2009). »The Evolution of Research on Computer-Supported Collaborative Learning«. In: *Technology-Enhanced Learning*. Hrsg. von Nicolas Balacheff, Sten Ludvigsen, Ton Jong, Ard Lazonder und Sally Barnes. Springer Netherlands, S. 3–19. ISBN: 978-1-4020-9826-0 (siehe S. 8).
- Dimopoulos, Ioannis, Ourania Petropoulou und Symeon Retalis (2013). »Assessing students' performance using the learning analytics enriched rubrics«. In: *Proceedings of the Third International Conference on Learning Analytics and Knowledge*. LAK '13. Leuven, Belgium: ACM, S. 195–199. ISBN: 978-1-4503-1785-6 (siehe S. 8–11).
- Dimopoulos, John (18. Mai 2013). *Learning Analytics Enriched Rubric - MoodleDocs*. URL: http://docs.moodle.org/23/en/Learning_Analytics_Enriched_Rubric (besucht am 18.05.2013) (siehe S. 10).
- Dröge, Rotraut, Astrid Ebeling und Wilhelm Schipper (1999). *Handbuch für den Mathematikunterricht. 3. Schuljahr*. Braunschweig: Schroedel Verlag GmbH. ISBN: 978-3-507-34052-7 (siehe S. 27).
- Dyckhoff, A. L., V. Lukarov, A. Muslim, M. A. Chatti und U. Schroeder (2013). »Supporting action research with learning analytics«. In: *Proceedings of the Third International Conference on Learning Analytics and Knowledge*. LAK '13. Leuven, Belgium: ACM, S. 220–229. ISBN: 978-1-4503-1785-6 (siehe S. 9).
- Dyckhoff, Anna Lea, Dennis Zielke, Mareike Bültmann, Mohamed Amine Chatti und Ulrik Schroeder (2012). »Design and Implementation of a Learning Analytics Toolkit for Teachers«. In: *Educational Technology & Society* 15.3, S. 58–76 (siehe S. 9, 11–14).
- Ebner, M. und S. Schön (2012). *Die Zukunft von Lern- und Lehrmaterialien: Entwicklungen, Initiativen, Vorhersagen*. Beiträge zu offenen Bildungsressourcen. Books on Demand. ISBN: 9783842382459 (siehe S. 3).
- Ebner, Martin und Martin Schön (Apr. 2013). »Why Learning Analytics for Primary Education Matters!« In: *Bulletin of the IEEE Technical Committee on Learning Technology* 15.2, S. 14–17. ISSN: 2306-0212 (siehe S. 96).
- Ebner, Martin, Martin Schön und Walther Nagler (2012). »Have They Changed? Five Years of Survey on Academic Net-Generation«. In: *In Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2012*, S. 343–353 (siehe S. 97).

- Eilebrecht, Karl und Gernot Starke (2010). *Patterns kompakt. Entwurfsmuster für effektive Software-Entwicklung*. 3nd. Spektrum Akademischer Verlag Heidelberg. ISBN: 978-3-8274-2525-6 (siehe S. 55, 56).
- Feng, Mingyu, Neil T. Heffernan und Kenneth R. Koedinger (2006). »Addressing the testing challenge with a web-based e-assessment system that tutors as it assesses«. In: *Proceedings of the 15th international conference on World Wide Web. WWW '06*. Edinburgh, Scotland: ACM, S. 307–316. ISBN: 1-59593-323-9 (siehe S. 49, 50).
- Few, S. (2006). *Information Dashboard Design: The Effective Visual Communication of Data*. O'Reilly Series. O'Reilly Media, Incorporated. ISBN: 9780596100162 (siehe S. 12).
- Fowler, M. (2012). *Patterns of Enterprise Application Architecture*. Addison-Wesley Signature Series (Fowler). Pearson Education. ISBN: 9780133065213 (siehe S. 56).
- Galloway, John, Phil Haack, Brad Wilson und K. Scott Allen (2011). *Professional asp.NET MVC 3*. John Wiley & Sons, Inc. ISBN: 9781118076583 (siehe S. 55).
- Gerster, H.D. (1982). *Schülerfehler bei schriftlichen Rechenverfahren: Diagnose und Therapie*. Studienbücher Mathematik Didaktik. Herder. ISBN: 9783451193156 (siehe S. 25–27, 31–38, 59).
- Holzinger, A. (2001). *Basiswissen Multimedia. 2. Lernen*. Basiswissen Multimedia. Vogel. ISBN: 9783802318573 (siehe S. 16).
- International Educational Data Mining Society (5. Mai 2013). *Home — International Educational Data Mining Society*. URL: <http://www.educationaldatamining.org/> (besucht am 05.05.2013) (siehe S. 4).
- Johnson, J.T. (1938). *The Relative Merits of Three Methods of Subtraction: An Experimental Comparison of the Decomposition Method of Subtraction with the Equal Additions Method and the Austrian Method*. Contributions to education. Bureau of Publications, Teachers College, Columbia University (siehe S. 31).
- Johnson, L., S. Adams Becker, M. Cummins, V. Estrada, A. Freeman und H. Ludgate (2013). *NMC Horizon Report: 2013 Higher Education*. Deutsche Ausgabe, Übersetzung: Helga Bechmann. Austin: Texas: The New Media Consortium (siehe S. 7).
- jQuery Foundation (6. Mai 2013). *License — jQuery Foundation*. URL: <https://jquery.org/license/> (besucht am 06.05.2013) (siehe S. 57).
- Kalz, Marco, Sandra Schön, Martin Linder, Detlev Roth und Peter Baumgartner (2011). In: Ebner, M. und S. Schön. *Lehrbuch für Lernen und Lehren mit Technologien (L3T)*. Books on Demand. Kap. Systeme im Einsatz. ISBN: 9783842340114 (siehe S. 8).
- Kamii, Constance und Ann Dominick (1997). »To teach or not to teach algorithms«. In: *The Journal of Mathematical Behavior* 16.1, S. 51–61. ISSN: 0732-3123 (siehe S. 22, 23).
- Kolb, D.A. (1984). *Experiential learning: experience as the source of learning and development*. Englewood Cliffs, NJ: Prentice Hall (siehe S. 20).
- Kühnhold, Karola und Friedhelm Padberg (1986). »Über typische Schülerfehler bei der schriftlichen Subtraktion natürlicher Zahlen«. German. In: *Der Mathematikunterricht* 32.3, S. 6–16. ISSN: 0025-5807 (siehe S. 28, 30–33).

- Laurillard, Diana (2002). *Rethinking university teaching: a conversational framework for the effective use of learning technologies*. Routledge/Falmer. ISBN: 9780415256797 (siehe S. 21).
- Lerdorf, Rasmus, Kevin Tatroe und Peter MacIntyre (2006). *Programming PHP*. 2nd. ISBN: 0596006810 (siehe S. 54).
- Lockyer, Lori und Shane Dawson (2011). »Learning designs and learning analytics«. In: *Proceedings of the 1st International Conference on Learning Analytics and Knowledge*. LAK '11. Banff, Alberta, Canada: ACM, S. 153–156. ISBN: 978-1-4503-0944-8 (siehe S. 8).
- Lorenz, J.H. und H. Radatz (1993). *Handbuch des Förderns im Mathematikunterricht*. Schroedel Schulbuchverlag. ISBN: 9783507340442 (siehe S. 27).
- Mavrody, S. und N. Mavrody (2012). *Sergey's Html5 & Css3: Quick Reference. PDF EBook (2nd Edition)*. Belisso. ISBN: 9780983386735 (siehe S. 56).
- McCarthy, Mary (2010). »Experiential Learning Theory: From Theory To Practice«. In: *Journal of Business & Economics Research - May 2010*, S. 131–140 (siehe S. 20).
- Mosel-Göbel, Doris (Dez 1988). »Algorithmusverständnis am Beispiel ausgewählter Verfahren der schriftlichen Subtraktion. Eine Fallstudienanalyse bei Grundschulern«. In: *Sachunterricht und Mathematik in der Primarstufe*, S. 554–559 (siehe S. 40).
- Niemann, Katja, Hans-Christian Schmitz, Maren Scheffel und Martin Wolpers (2011). »Usage contexts for object similarity: exploratory investigations«. In: *Proceedings of the 1st International Conference on Learning Analytics and Knowledge*. LAK '11. Banff, Alberta, Canada: ACM, S. 81–85. ISBN: 978-1-4503-0944-8 (siehe S. 8).
- Nwana, Hyacinth S. (1990). »Intelligent tutoring systems: an overview«. English. In: *Artificial Intelligence Review* 4.4, S. 251–277. ISSN: 0269-2821 (siehe S. 17, 18).
- Oracle Corporation (6. Mai 2013a). *MySQL :: Marktanteil*. URL: <http://www.mysql.de/why-mysql/marketshare/> (besucht am 06.05.2013) (siehe S. 55).
- (6. Mai 2013b). *MySQL :: Warum MySQL?* URL: <http://www.mysql.de/why-mysql/> (besucht am 06.05.2013) (siehe S. 55).
- Padberg, Friedhelm (1994). »Schriftliche Subtraktion - Änderungen erforderlich!«. German. In: *Mathematische Unterrichtspraxis* 15.2, S. 24–34. ISSN: 0721-8419 (siehe S. 27, 40).
- (2005). *Didaktik der Arithmetik: für Lehrerbildung und Lehrerfortbildung*. Mathematik Primar- und Sekundarstufe. Elsevier, Spektrum, Akad. Verlag. ISBN: 9783827409935 (siehe S. 22, 23, 25–33, 59).
- Pardos, Zachary A., Ryan S. J. D. Baker, Maria O. C. Z. San Pedro, Sujith M. Gowda und Supreeth M. Gowda (2013). »Affective states and state tests: investigating how affect throughout the school year predicts end of year learning outcomes«. In: *Proceedings of the Third International Conference on Learning Analytics and Knowledge*. LAK '13. Leuven, Belgium: ACM, S. 117–124. ISBN: 978-1-4503-1785-6 (siehe S. 18).
- Pask, G. (1976). *Conversation Theory: Applications in Education and Epistemology*. Elsevier Science Limited. ISBN: 9780444414243 (siehe S. 21).
- Quellmalz, Edys S. und Robert Kozma (2003). »Designing Assessments of Learning with Technology«. In: *Assessment in Education* (siehe S. 8).

- Razzaq, Leena et al. (2005). »Blending Assessment and Instructional Assisting«. In: *Proceedings of the 2005 conference on Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology*. Amsterdam, The Netherlands, The Netherlands: IOS Press, S. 555–562. ISBN: 1-58603-530-4 (siehe S. 50, 51).
- Romero, C. und S. Ventura (2010). »Educational Data Mining: A Review of the State of the Art«. In: *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 40.6, S. 601–618. ISSN: 1094-6977 (siehe S. 8, 96).
- Sacerdoti, Earl David (1975). »A structure for plans and behavior.« AAI7605794. Diss. Stanford, CA, USA (siehe S. 42).
- Schmidt, S. (2009). *PHP Design Patterns*. O'Reilly Vlg. GmbH & Company. ISBN: 9783897218642 (siehe S. 70).
- Schön, D.A. (1983). *The reflective practitioner: how professionals think in action*. Harp torchbooks ; TB5126. Basic Books Publ. ISBN: 9780465068784 (siehe S. 21).
- Schön, Martin, Martin Ebner und Georg Kothmeier (2012). »It's just about learning the multiplication table«. In: *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*. LAK '12. Vancouver, British Columbia, Canada: ACM, S. 73–81. ISBN: 978-1-4503-1111-3 (siehe S. 89).
- Schumann, Heinz (28. Mai 2013). *Medien im fachlichen und überfachlichen Unterricht. - Medienverwendung im Fach Mathematik. Bewertung mathematischer Lernprogramme*. URL: <http://thales.cs.upb.de:8080/mksu/mathe/Mathe2.pdf> (besucht am 28.05.2013) (siehe S. 46–48).
- Sicilia, Miguel-Angel, Xavier Ochoa, Giannis Stoitsis und Joris Klerkx (2013). »Learning object analytics for collections, repositories & federations«. In: *Proceedings of the Third International Conference on Learning Analytics and Knowledge*. LAK '13. Leuven, Belgium: ACM, S. 285–286. ISBN: 978-1-4503-1785-6 (siehe S. 3).
- Siemens, George (16. Mai 2013). *elearnspace - What are Learning Analytics?* URL: <http://www.elearnspace.org/blog/2010/08/25/what-are-learning-analytics/> (besucht am 16.05.2013) (siehe S. 5–7).
- Siemens, George und Ryan S. J. d. Baker (2012). »Learning analytics and educational data mining: towards communication and collaboration«. In: *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*. LAK '12. Vancouver, British Columbia, Canada: ACM, S. 252–254. ISBN: 978-1-4503-1111-3 (siehe S. 3–5).
- Siemens, George und Phil Long (2011). »Penetrating the Fog: Analytics in Learning and Education«. In: *EDUCAUSE Review* 46.5, S. 30–32 (siehe S. 3).
- Sleeman, D. und J.S. Brown (1982). *Intelligent tutoring systems*. Computers and people series. Academic Press. ISBN: 9780126486810 (siehe S. 43–46).
- Stiftung Universität Hildesheim (28. Mai 2013). *Mathematik heute - Bruchrechnung*. URL: <http://www.uni-hildesheim.de/index.php?id=1843> (besucht am 28.05.2013) (siehe S. 49).
- The PHP Group (5. Mai 2013). *PHP: Licence Information*. URL: <http://www.php.net/license/> (besucht am 05.05.2013) (siehe S. 54).

- Welling, L. und L. Thomson (2003). *Php and Mysql Web Development*. Developer's library. Sams Publishing. ISBN: 9780672325250 (siehe S. 54).
- Zend Technologies Ltd. (6. Mai 2013a). *Zend Framework*. URL: <http://framework.zend.com/license/> (besucht am 06.05.2013) (siehe S. 54).
- (5. Mai 2013b). *Zend Framework*. URL: <http://framework.zend.com/about/> (besucht am 05.05.2013) (siehe S. 54).
- Ziegenbalg, J., B. Ziegenbalg und O. Ziegenbalg (2007). *Algorithmen: von Hammurapi bis Gödel*. Deutsch. ISBN: 9783817118144 (siehe S. 22).