Lessons Learned 1, 1&2 (2021)
Submitted: 31.05.2021
Accepted: 14.06.2021
DOI: https://doi.org/10.25369/ll.v1i1/2.21
ISSN: 2749-1293 (Print); 2749-1307 (Online)

Möglichkeiten von OpalExam zur Verbesserung von Klausurbewertung und –einsicht

M. Fiedler, M. Kästner

Professur für Numerische und Experimentelle Festkörpermechanik, Institut für Festkörpermechanik, Fakultät Maschinenwesen, TU Dresden

Abstract

Durch die Umstellung von Präsenz- auf Onlineklausuren stehen Lehrende vor neuen Herausforderungen. Dabei geht es nicht alleine nur um die technische Herausforderung, sondern auch um den eigenen Anspruch, Klausuren online so zu modellieren, dass Betrugsversuche entweder verhindert werden können oder dem Betrügenden keine Vor- sondern Nachteile bieten. Lehrenden stehen dabei während der Klausurmodellierung in OpalExam verschiedene Möglichkeiten zur Verfügung: Definition von Zufallswerten durch Verwendung von Variablen in der Aufgabenstellung, Erstellung mehrerer Varianten einer Aufgabe, die den Studierenden zufällig zugewiesen werden oder auch Änderung der Aufgabenreihenfolge für die Studierenden und Definition eines linearen Durchlaufs durch die Klausur. Die einzelnen Optionen können dabei natürlich auch kombiniert werden.
Wird ein nichtlinearer Verlauf (Studierende können zwischen den Aufgaben beliebig hin- und herspringen) durch die Klausur angestrebt, so werden die Klausuren durch die Wahl der ersten beiden Optionen komplexer. Für Studierende wird dadurch zwar die Zusammenarbeit erschwert, die Bewertung der Antworten des Studierenden durch den Prüfenden allerdings auch. Innerhalb dieses Aufsatzes werden deshalb Möglichkeiten aufgezeigt, wie der Bewertungsaufwand gesenkt und dabei gleichzeitig die Klausureinsicht erleichtert werden kann.

The switch from offline to online exams presents teachers with new challenges. Not all challenges are technically, a challenge is also the own demand to model online exams so that cheating attempts can either be prevented or do not offer advantages but disadvantages to the cheater. Teachers have various options at their disposal when modelling exams in OpalExam: Definition of random values by using variables in the assignment, creation of several variants of an assignment that are randomly assigned to the students or also change of the assignment order for the students and definition of a linear run through the exam. Of course the individual options can also be combined.
If a non-linear progression (students can jump back and forth between tasks at will) through the exam is the goal, the exams become more complex by choosing the first two options. While this makes it more difficult for students to collaborate, it also makes it more difficult for the examiner to evaluate the student's answers. In this paper, it will be shown how the assessment effort can be reduced while at the same time making it easier for the students to review the exam results.

*Corresponding author: melanie.fiedler@tu-dresden.de

1. Einleitung

Die Umstellung von Präsenz- auf Onlineklausuren stellt Lehrende vor neue Probleme und Herausforderungen. Neben der Umstrukturierung von Aufgaben und auftretenden technischen Problemen und Herausforderungen, siehe z.B. [1, 2], stellen Betrugsversuche ein ernstzunehmendes Problem dar, vgl. [3, 4, 5]. Um Betrug durch Zusammenarbeit ohne Online-Proctoring zu verhindern, können die Aufgaben individuell auf die Studierenden zugeschnitten werden, was jedoch den Aufwand bei Aufgabenerstellung, -bewertung und -einsicht deutlich erhöhen kann. Onyx in OpalExam enthält dabei einige Funktionen, die Klausurbewertung und -einsicht vereinfachen können. Sektion 2 gibt zunächst eine Übersicht über aktuelle Möglichkeiten zur Vermeidung von Betrugsversuchen während der Klausurerstellung in OpalExam. Sektion 3 diskutiert die Vor- und Nachteile der dadurch entstehenden, komplexeren Modellierung. Die Dokumentation der erreichten Punkte pro Antwort für Aufgaben mit mehreren Aufgabenfeldern ist in Opal bisher nicht als Einstellung verfügbar. Wie dies trotzdem realisiert werden kann, wird in Sektion 4 beschrieben. Eine Vergabe von Teilpunkten auf Antworten, die nicht der richtigen Lösung entsprechen, weil z.B. ein Term in der Berechnung vergessen wurde, ist in Opal nur bei der Verwendung von Lückentexten mit dem Lückentyp Formel möglich. Eine Dokumentation der notwendigen Programmierung ist in Sektion 5 gegeben. Die Berücksichtigung von Folgefehlern in längeren Rechenaufgaben ist ebenfalls in Lückentexten mit dem Lückentyp Formel möglich. Wie genau wird in Sektion 6 beschrieben. In Sektion 7 folgt schließlich eine Zusammenfassung der hier beschriebenen Möglichkeiten.

2. Möglichkeiten zur Vermeidung von Betrugsversuchen

Bei der Nutzung der Testfunktion in Opal bzw. OpalExam können Aufgaben und Tests in Onyx modelliert und den Studierenden online zur Verfügung gestellt werden. Werden Klausuren online durchgeführt, so kann eine Zusammenarbeit von Studierenden nicht ausgeschlossen werden. Sei es in der Form von mehreren Studierenden, die theoretisch mit ihren Laptops nebeneinandersitzen oder auch per Videochat miteinander verbunden sein könnten. Eine Kontrolle durch den Lehrenden ist gerade was das letztere Szenario angeht, schwierig bis nicht umsetzbar.

Es bleibt nur, Klausuren so zu modellieren, dass das Resultat einer Zusammenarbeit zwischen Studierenden zu großen Zeiteinbußen führt und so für Studierende weniger attraktiv wird.

Dafür gibt es verschiedene Möglichkeiten:

- Die Verwendung von Zufallswerten in der Aufgabenstellung durch Definition von Variablen,
- die Erstellung mehrerer Aufgabenvarianten,
- die Änderung der Aufgabenreihenfolge und Einstellung eines linearen Verlaufs durch die Klausur.

Wird nur die Aufgabenreihenfolge einer Klausur geändert, so können Studierende zwar nicht per Videochat die Aufgabe zusammen bearbeiten, sind aber dennoch in der Lage, die Ergebnisse untereinander auszutauschen, wenn alle die gleichen Zahlenwerte, Fragen oder Bilder haben. Ein einfacher Screenshot kann sehr leicht untereinander ausgetauscht werden. Außerdem ist ein großer Unterschied zur Präsenz dadurch gegeben, dass Studierende durch die Einstellung des linearen Verlaufs nicht mehr beliebig zwischen einzelnen Aufgaben hin und her wechseln können. Wird stattdessen ein nichtlinearer Verlauf eingestellt, so dass die Studierenden beliebig zwischen den Aufgaben wechseln können, ist die Wirkung der Änderung der Aufgabenreihenfolge fast schon kosmetischer Natur.

Eine Verbesserung kann durch die Verwendung von Zufallswerten in der Aufgabenstellung erreicht werden. Dafür werden die gegebenen Größen, z.B. eine Länge a mit einer Zufallszahl belegt. Dies geschieht durch die Einführung von Variablen, die bei Aufrufen der Aufgabe durch die Studierenden automatisch belegt und in den Aufgabentext eingebunden werden, siehe Abb. 1. Diese Variablen werden

im Weiteren als Vor-Abgabe-Variablen bezeichnet, da ihre Werte bereits vor Abgabe der Aufgabe durch die Studierenden festgelegt werden und nicht mehr geändert werden können.

Für diese Vor-Abgabe-Variablen können im Reiter „Variablen“ der Aufgabe z.B. Zahlenintervalle mit zu definierender Schrittweite festgelegt werden, Abb. 2.

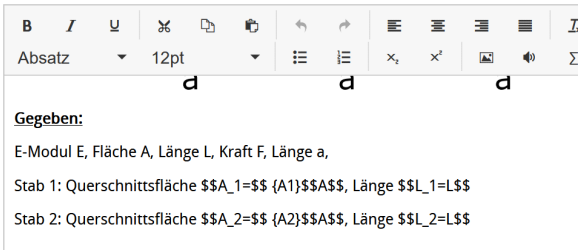

Abb. 1: Einbindung von Vor-Abgabe-Variablen in den Aufgabentext

Den Studierenden werden dann bei Aufruf der Aufgabe automatisch Zahlenwerte zugewiesen, die sich von denen ihrer Kommilitonen unterscheiden. Dadurch kann zumindest das Abschreiben von Zahlenwerten vermieden werden.

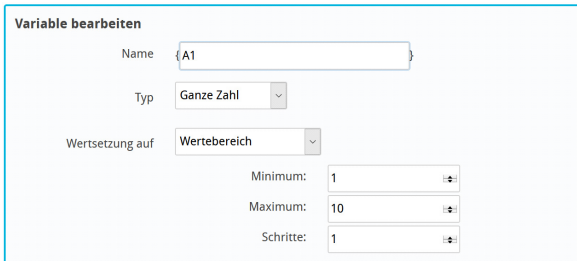

Abb. 2: Definition eines Intervalls an Zufallswerten für eine Vor-Abgabe-Variable

Was natürlich immer noch möglich ist, ist das Teilen des Lösungswegs unter mehreren Studierenden. Um dies zu unterbinden, können verschiedene Versionen einer Aufgabe entwickelt werden, siehe Abb. 3. Diese Option ist vergleichbar mit der Ausgabe von A und B Klausurversionen in der Präsenz, wenn eine Einhaltung der notwendigen Sitzabstände in der Vergangenheit nicht eingehalten werden konnte.

Durch kleinere Änderungen in der Aufgabenstellung ist eine 1:1 Übernahme des Lösungsweges der Kommilitonen so nicht mehr so einfach möglich. Eine Orientierung an einer anderen Musterlösung kann stattdessen zu Zeit- und Punkteverlust führen, wenn der Studierende die Unterschiede finden muss bzw. diese übersieht.

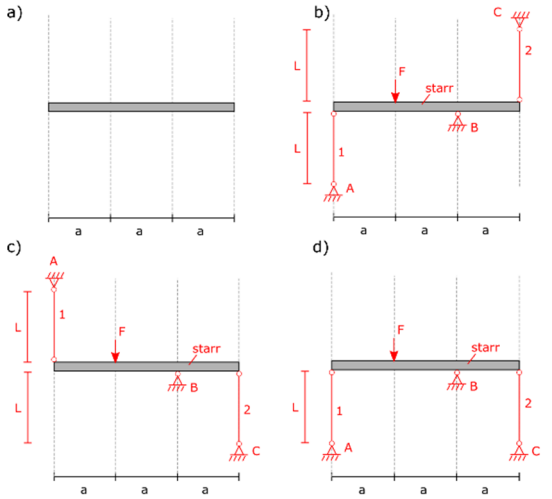

Abb. 3: Darstellung verschiedener Varianten für eine Aufgabe: a) Klausurvorlage für die Studierenden, b)-d) Varianten der Aufgabe

3. Vor- und Nachteile der komplexen Modellierung

Eine Kombination aus Zufallswerten und Aufgabenvarianten macht ein Abschreiben und Zusammenarbeiten so für die Studierenden unattraktiver. Für den Modellierer der Klausur wird dadurch allerdings auch der Modellierungsaufwand und so die Komplexität deutlich erhöht. Außerdem kann es zu Zeitunterschieden bei der Klausurbewertung kommen, wenn die automatische Auswertung überprüft wird. Anders als in einer Präsenzklausur sind Korrigierende dann nicht mehr in der Lage, auf den ersten Blick den Fehler in der Lösung alleine anhand des Ergebnisses zu erkennen, da jeder Studierende eigene Zahlenwerte erhält und es zudem noch verschiedene Aufgabenvarianten gibt.

Modellierer haben also zwei Möglichkeiten: Entweder die Klausuraufgaben und deren Ergebnisabfragen so engmaschig zu modellieren, dass eine Nachbewertung bzw. Vergabe von Teilpunkten durch Korrigierende nicht notwendig wird. Allerdings besteht bei einer

engmaschigen Ergebnisabfrage schnell die Gefahr, bereits einen Lösungsweg vorgeben zu müssen, was den Schwierigkeitsgrad der Klausur senkt.

Oder bereits während der Modellierung der Klausur die Klausurbewertung im Blick zu haben und die Aufgaben so zu erstellen, dass eine Vergabe von Teilpunkten, eine Berücksichtigung von Folgefehlern und eine Dokumentation von erreichten Punkten pro Antwort und nicht nur pro Frage, automatisiert möglich ist. Die zweite Variante führt dabei zu einer schnelleren, weil übersichtlicheren Nachbewertung sowie weniger Fragen von Studierenden bei der nachfolgenden Klausureinsicht.

4. Anzeige von erreichten Punkten pro Antwort bei Aufgaben mit mehreren Antwortfeldern

Innerhalb der Klausurmodellierung können Aufgaben erstellt werden, in der Studierende mehr als eine Antwort geben müssen, z.B. eine Lückentextaufgabe mit mehreren Lücken oder eine Auswahlaufgabe mit mehreren Auswahlfeldern. Die Opal-Auswertung zeigt dabei jedoch nur die Gesamtpunktzahl der Aufgabe sowie die Information richtig/falsch für jede Antwort an. Die Information, wie viele Punkte pro Lücke erzielt wurden, fehlt.

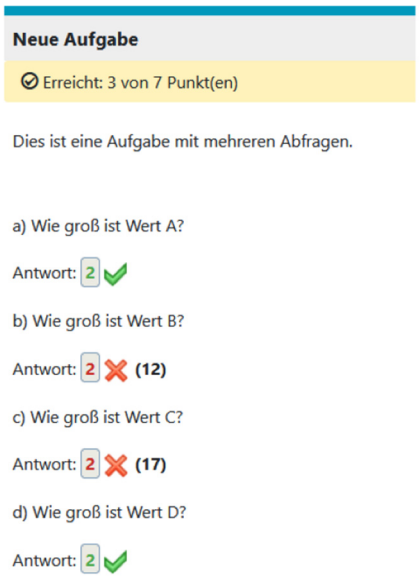

Abb. 4: Anzeige der Auswertung einer Aufgabe mit mehreren Lücken. Lediglich die Gesamtpunktzahl wird oben angezeigt, [6]

Haben alle Antworten den selben Schwierigkeitsgrad und jeder Lücke wird die selbe Punktzahl zugewiesen, so ist dies bei der Auswertung kein Problem. Sollen die einzelnen Antworten hingegen unterschiedlich gewichtet werden, so fehlt die Angabe der erreichten Punkte für jede Antwort. Dies führt zu unnötigen Schwierigkeiten bei der Nachkorrektur der automatischen Bewertung und der Klausureinsicht, siehe Abb. 4. Eine Anzeige der erreichten Punktzahl hinter jeder Antwort wäre deshalb sinnvoll.

Zur Anzeige von erreichten Punkten pro Antwort können Nach-Abgabe-Variablen verwendet werden, vgl. [6]. Diese werden genauso wie Vor-Abgabe-Variablen in den Aufgabentext eingebunden, jedoch erst nach Abgabe der Aufgabe mit Werten belegt. Dadurch erscheinen Sie für Studierende innerhalb der Klausur nicht und werden erst bei der Klausurkorrektur sichtbar. Neben der Verwendung zur Anzeige von erreichten Punkten pro Antwort können Nach-Abgabe-Variablen so auch zur Anzeige von expliziten Hinweisen für Bewertende oder auch für die Klausureinsicht genutzt werden.

Die Definition der Nach-Abgabe-Variable erfolgt dabei im Reiter „Variablen“ unterhalb der Vor-Abgabe-Variablen. Nach-Abgabe-Variablen können dabei ebenso mit Text, festen Werten oder Variablenwerten belegt werden. Anders als Vor-Abgabe-Variablen können Nach-Abgabe-Variablen dabei auch auf die programminternen Variablen SCORE, MAXSCORE, MINSCORE, LEARNERRESPONSE, etc. zugreifen.

Wird z.B. die Nach-Abgabe-Variable „{1Score}“ definiert und gleich der programminternen Variable „SCORE_GAP_1“ gesetzt, so enthält diese Variable die erreichte Punktzahl des Studierenden nach Beantwortung der Lücke 1. Wird diese Variable in den Aufgabentext eingebunden, so erscheint nach Abgabe des Studierenden seine erreichte Punktzahl im Aufgabentext, siehe Abb. 5.

In [6] ist eine Schritt-für-Schritt-Anleitung zur Nutzung der Nach-Abgabe-Variablen zur Anzeige von erreichten Punkten pro Lücke dokumentiert.

Abb. 5: Erweiterung der Aufgabenstellung durch Nach-Abgabe-Variablen zur Anzeige der erreichten Punkte pro Lücke, [6]

5. Vergabe von Teilpunkten

Eine Erweiterung der Punktevergabe für richtige und falsche Antworten durch Teilpunkte kann ebenfalls während der Klausurmodellierung realisiert werden. Basis hierfür ist der Aufgabentyp Lückentext mit dem Lückentyp „Formel“, vgl. [7]. Die Höhe der Maximalpunktzahl der Lücke wird dabei weiterhin über die Einstellungsoption „Punkte“ der Lücke definiert. Die korrekte Antwort wird ebenfalls weiterhin über das vorgesehene Feld „Lösung“ definiert.

Um Teilpunkte zu vergeben, muss der Lückentyp „Formel“ im Expertenmodus verwendet werden, was die Möglichkeit zur Programmierung einer if-else-Bedingung ermöglicht. Als Bewertungskriterium muss dabei von „Richtig/Falsch“ auf „Punkte“ gewechselt werden, siehe Abb. 6.

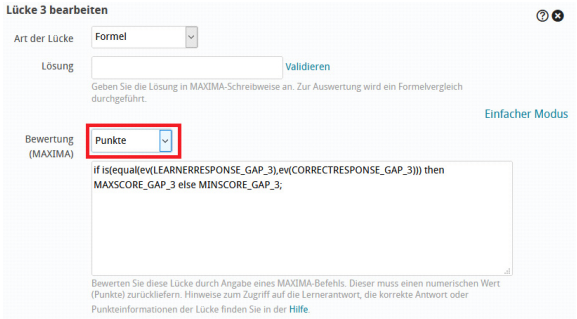

Abb. 6: Bildunterschriften in Open Sans, 9pt, kursiv, linksbündig. 4pt Abstand vor dem Bildunterschriftstext, [7]

Voreingestellt ist dabei eine if-else-Bedingung die überprüft, ob die Studierendenantwort (LEARNERRESPONSE) der korrekten Antwort (CORRECTRESPONSE) entspricht. Ist dies der Fall, so erhält der Studierende die Maximalpunktzahl (MAXSCORE), andernfalls die Minimalpunktzahl (MINSCORE) also 0.

Um neben der Maximal- und Minimalpunktzahl auch Teilpunkte vergeben zu können, muss die if-else-Bedingung durch eine zusätzliche Bedingung erweitert werden. Dazu wird eine weitere if-else-Bedingung modelliert, die überprüft, ob der Studierende z.B. eine Abweichung von 1 von der korrekten Antwort hat. Ist dies der Fall, so erhält der Studierende noch 1/3 der Maximalpunktzahl, siehe Abbildung 7.

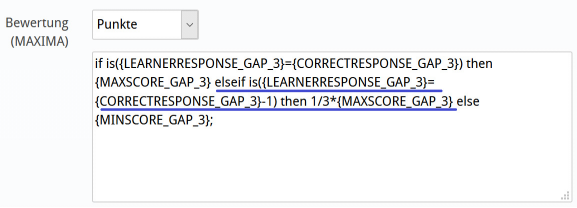

Abb. 7: Erweiterung der Programmierung durch eine weitere if-Bedingung(blau markiert)

Wird nicht mehr die volle Punktzahl der Aufgabe an den Studierenden vergeben, so zeigt Opal dies in der Auswertung durch die Vergabe eines gelben Hakens an, siehe Abbildung 8.

Abb. 8: Darstellung der Bewertung von richtig, falsch und Teilpunkten, [7]

Seit dem 30.03.2021 werden die grünen und gelben Haken sowie die roten Kreuze auch in der assessment-PDF beim Export über die Datenarchivierung angezeigt, [8]. Die Anzeige kann so auch für die Klausurbewertung und -einsicht genutzt werden.

6. Berücksichtigung von Folgefehlern

Bei der Bewertung von Präsenzklausuren macht die Bewertung von Folgefehlern einen großen Anteil der Klausurbewertungszeit aus. Um diesen Anteil zu schmälern, kann die automatische Klausurauswertung genutzt werden. Dies macht vor allem bei Klausuren mit einer hohen Teilnehmerzahl Sinn.

Sollen innerhalb einer Aufgabe Folgefehler gewertet werden können, so gibt es verschiedene Möglichkeiten, wie dies realisiert werden kann. Eine Diskussion dazu ist in [9] enthalten.

Für eine nichtlineare Klausurmodellierung wird hier der Aufgabenbaustein „Lückentext“ mit dem Lückentyp „Formel“ verwendet. Innerhalb des Expertenmodus wird weiterhin die Option „Punkte“ gewählt.

Die in Sektion 5 definierte, erweiterte if-else-Bedingung wird zur Berücksichtigung eines Folgefehlers aus Lücke 2 um eine weitere if-else-Bedingung erweitert. Diese fragt bei einer falschen Lösung ab, ob die falsche Lösung als Folgefehler entstanden ist. Falls ja, so können darauf basierend ebenfalls Punkte gegeben werden, falls nein, so erhält der Studierende 0 Punkte.

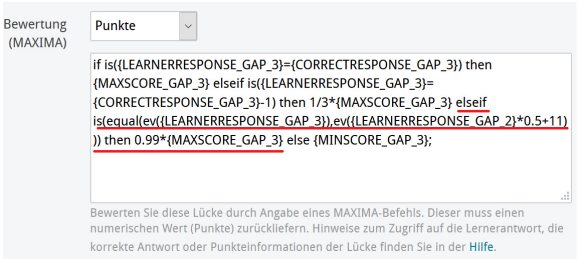

Abb. 9: Ergänzung der Maxima-Programmierung um eine weitere if-Bedingung (rot unterstrichen)

In Abb. 10 wurde für einen Folgefehler 1% der erreichbaren Punktzahl der Lücke abgezogen. Durch den geringen Abzug wurde die Lücke als Teilpunkte gewertet, vgl. Abschnitt 5, und für den Korrigierenden kenntlich gemacht. Wie groß der Abzug sein soll, kann bei der Aufgabenmodellierung frei gewählt werden.

Abb. 10: Automatische Auswertung mit Berücksichtigung von Folgefehlern

Soll noch deutlicher sichtbar werden, dass hier ein Folgefehler gewertet wurde, kann eine Nach-Abgabe-Variable {3FF} definiert werden, die im Folgefehler-Fall den Textwert „Folgefehler!“ und ansonsten den Wert „“ annimmt, Abb. 11.

Abb. 11: Automatische Auswertung mit Bewertung und gesondertem Hinweis auf Folgefehler

Dafür muss auch für die Nach-Abgabe-Variable eine if-Bedingung erstellt werden, siehe Abb. 12. Anschließend kann diese dann genauso wie die anderen Nach-Abgabe-Variablen in den Aufgabentext eingebunden werden.

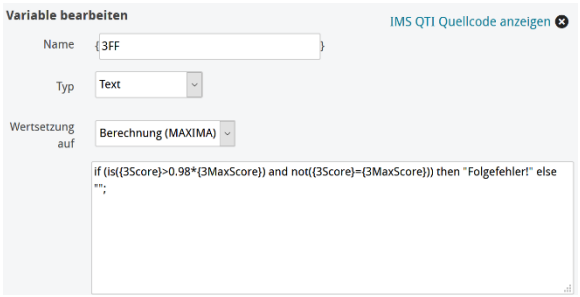

Abb. 12: Erstellung der Folgefehler-Textvariablen für Lücke 3

7. Zusammenfassung

Die Umsetzung von Präsenz- auf Onlineklausuren hält einige Hürden bereit. Um Betrugsversuche zu minimieren, müssen Abschreiben und Zusammenarbeiten unattraktiv gemacht werden. Dies geht mit einer Zunahme an Komplexität der Aufgabenmodellierung einher. Um dabei nicht auch noch eine Zunahme des Bewertungsaufwandes zu verursachen, können bereits während der Aufgabenmodellierung gezielt Nach-Abgabe-Variablen eingesetzt oder Formellücken programmiert werden.

Nicht verschwiegen werden darf dabei, dass dies natürlich die Zeit der Aufgabenmodellierung sowie der -überprüfung zunächst deutlich erhöht. Ist der Einsatz von digitalen Klausuren jedoch auch weiterhin geplant, z.B. in Kombination mit einer Präsenzklausur [9], so kann sich der Modellierungsaufwand lohnen, wenn in der nachfolgenden Prüfungsperiode ähnliche Aufgaben durch kleinere Anpassungen aus den bereits vorhandenen erzeugt werden können. Auf Dauer kann so ein Aufgabenpool entstehen, der die Klausurbewertungszeit gezielt und dauerhaft minimiert.

Literatur

[1] Schlecht, B.; Rosenlöcher, T.: Möglichkeiten und Grenzen der digitalen Lehre und Prüfung im Grundlagenfach Maschinenelemente; Book of Abstracts der 1. Lessons Learned Konferenz; 14.+15.10.2020 in Dresden; TU Dresden; 2020

[2] Dohmen, Eike.; Lange, Adrian; Odenbach, Stefan: Online-Prüfung mit OPAL und ONYX – Wieviel besser sind 8 Monate?; Book of Abstracts der 2. Lessons Learned Konferenz; 18.+19.03.2021 in Dresden; TU Dresden; 2021

[3] Schmidt, Th.: Morgen ist das heute von gestern, oder: Wie laufend alles anders kam; Book of Abstracts der 1. Lessons Learned Konferenz; 14.+15.10.2020 in Dresden; TU Dresden; 2020

[4] Bilen, Eren; Matros, Alexander.: Online cheating amid Covid-19; Journal of Economic Behavior and Organization; Vol. 182; p. 196-211; 2021

[5] Abdelrahim, Yousif; How Covid-19 quarantine influenced online exam cheating: A case of Bangladesh university students; Journal of Southwest Jiaotong University; Vol. 56; 2021

[6] Fiedler, Melanie: Verwendung von Nach-Abgabe-Variablen zur Anzeige der Punktzahlen jeder Lücke in Lückentextaufgaben; https://tud.link/2m1r; Stand: 31.05.2021;

[7] Fiedler, Melanie: Vergabe von Teilpunkten in Lückentextaufgaben; https://tud.link/sbuw; Stand: 31.05.2021

[8] BPS Bildungsportal Sachsen GmbH: BPS Release Notes; https://www.bps-system.de/help/display/OR/ONYX+Testsuite+9.9, Stand: 30.05.2021

[9] Fiedler, Melanie; Kästner, Markus: Rechnerische Klausuren in OpalExam unter Berücksichtigung von Folgefehlern modellieren – Vorteile, Nachteile und Zukunftsaussichten; Book of Abstracts der 2. Lessons Learned Konferenz; 18.+19.03.2021 in Dresden; TU Dresden; 2021