A Simple MVC-Framework for Local Management of Online Course Material

Frode Eika Sandnes[1,2] and Evelyn Eika[1]

[1]Faculty of Technology, Art and Design, Oslo and Akershus University College of Applied Sciences, Oslo, Norway
[2]Westerdals Oslo School of Art, Communication and Technology, Oslo, Norway
{Frode-Eika.Sandnes, Evelyn.Eika}@hioa.no

Abstract. Managing online materials for large classes can be time-consuming and error prone. In particular, it can be challenging to manage long lists of students, lecture progress, and auditorium schedules as these often change on a daily basis. We therefore introduce a simple Model-View-Controller (MVC) framework implemented in Excel that can help teachers handle daily tasks more efficiently. Examples include how to generate lecture plans, student presentation schedules, and peer-review plans for students. The authors have successfully used the system for more than five years in several courses. The framework simplifies the task of reusing material from one teaching semester to another. Teachers only need to focus on the content and not the visual appearance.

Keywords: model-view-controller, course management, students, higher education, content management

1 Introduction

When teaching a course at university level, the teacher faces several challenges. First, courses often comprise many students. Some students enroll late, some students drop off in the middle of the course, and others become ill or need extensions or certain treatment for various reasons. Consequently, the list of students changes frequently.

Second, classrooms and lecture theatres are often a scarce resource. Limited resources are even more a challenge when different classes follow different teaching paradigms. For example, semester programs require the same rooms on a weekly basis throughout the semester, while intensive courses require rooms in concentrated intervals at certain time of the semester. Therefore, students and teachers must relate to classes scheduled at different weekdays, different times, and in different locations across different weeks.

Third, it is common practice to follow a teaching plan. Experienced teachers will deviate from the plan depending on the interaction and events in lectures and classes.

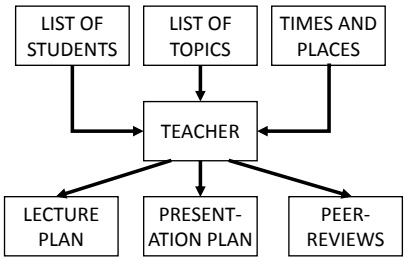


Fig. 1. The dynamic nature of micro managing a course.

If a class becomes engaged in a fruitful discussion, the teacher can postpone material to later lectures, while if there is little discussion, the teacher can cover more material than planned.

We assume that a learning management system (LMS) is the main channel for formal information and communication between the teacher and the students [1, 2, 3, 4]. Regular changes to the teaching plan, lists of students, and room-schedule take up much of teachers' time as they need to update the online material published on an institutional LMS. Often, these systems do not provide the functionality needed by teachers and students. For instance, if there are changes to the progress of lectures, the published teaching plan needs updating. A teaching plan usually comprises topics and dates. These are important as students use these to know which lectures to attend.

Moreover, some courses include compulsory student presentations, where each student, or a group of students, must present an assigned presentation at a designated time-slot. Several time-slots, perhaps across multiple weeks, are needed if there are many students. The teacher needs to organize and publish the presentation schedules and update these due to changes in the list of students or the assigned rooms.

A course may also utilize peer-review where students give feedback on each other's work. Such peer-review activities usually require careful planning and the task can be quite complicated and error-prone when done manually.

Fig. 1 shows how the teacher needs to act on changes in the list of students, in-class progress, and allocated auditoriums. This example includes lecture plans, student presentation plans, and peer-reviews; naturally other documents could also be included.

The examples above require focus and attention as mistakes can easily occur and may lead to misunderstandings. This work proposes a framework to assist teachers with managing updated material for publishing to students. The framework presented is created using the Microsoft Excel spreadsheet software as Microsoft Office is widely available. Moreover, one may substitute Microsoft Excel with any other spreadsheet application such as OpenOffice Calc as the same principles apply.

2 Spreadsheet MVC Framework

The following sections present the MVC framework.

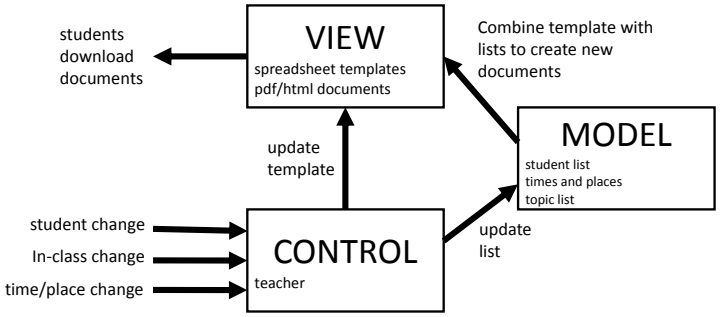


Fig. 2. MVC adopted for semi-automatic content micro management.

2.1 The MVC Pattern

The framework presented herein is inspired by the MVC pattern introduced by Reenskaug [5]. This pattern is widely used in all areas of user interfaces from native applications [6] to web-applications accessible via browsers and smartphones [7] and even LMSs [8]. The main advantage of the MVC pattern is to separate the visual appearance from the application logic, which again is separated from the data. It is then easier to make improvements in the various parts independently. As an example, a recent trend is that online resources should be universally accessible and satisfy the WCAG accessibility guidelines [9, 10]. Also, it is possible to experiment with the visual presentation of the published materials that enhance students' motivation and interest [11, 12]. The MVC pattern is the de facto standard for interactive systems.

Fig. 2 illustrates how course contents are changed dynamically, inspired by the MVC pattern. Unlike the traditional MVC pattern, the framework is semi-automatic as it relies on manual intervention from the teacher. For instance, changes to the list of students usually come in various forms such as emails from students or via oral messages in, or outside, class. Teachers follow the teaching progress and room allocations closely during the semester. Commonly teachers have to use a special room-booking system to find room changes.

Next, the list of students, the lecture topics, and the times and places represent the model. The views are static pdf or html-documents. Excel templates are used to generate these documents whenever the model is changed. Although document generation is automatic, the teacher must initiate the generation process. The following sections describe how to (a) use the spreadsheet to maintain lists and templates and (b) generate the static documents.

2.2 Mapping Topics to Times

Students often consult lecture plans to find out what contents teachers will cover in upcoming lectures. In our framework, a cell represents each lecture session with a textual descriptions. A column of lecture cells represents series of lecture sessions. The dates for the respective lectures are put into the cells of an adjoining column. Excel facilitates simple arithmetic on dates such as adding 7 to say that a cell is one week later, or adding 1 if it is the next day.

	A	B
1	21.08.2017	Introduction to the course
2	=A1+7	Basic theory
3	=A2+7	More theory…

Cells as they are input

	A	B
1	Mon. 21. Aug, 2017	Introduction to the course
2	Mon. 28. Aug, 2017	Basic theory
3	Mon. 4. Sep, 2017	More theory…

Cells as they are displayed

Fig. 3. Scheduling three weekly lectures.

	A	B
1	21.08.2017	Introduction to the course
2	=A1+2	Basic theory
3	=A1+7	More theory…
4	=A2+7	Some practice

Cells as they are input

	A	B
1	Mon. 21. Aug, 2017	Introduction to the course
2	Wed. 23. Aug, 2017	Basic theory
3	Mon. 28. Aug, 2017	More theory…
4	Wed. 30. Aug, 2017	Some practice

Cells as they are displayed

Fig. 4. Scheduling several lectures during a week with across-week references.

	A	B
1	21.08.2017	Introduction to the course
2	=A1+2	Basic theory
3	=A1+7	More theory…
4	=A3+2	Some practice

Cells as they are input

	A	B
1	Mon. 21. Aug, 2017	Introduction to the course
2	Wed. 23. Aug, 2017	Basic theory
3	Mon. 28. Aug, 2017	More theory…
4	Wed. 30. Aug, 2017	Some practice

Cells as they are displayed

Fig. 5. Scheduling several lectures during a week with within-week references.

The teacher only has to worry about the first date and then let Excel handle the day of the week, the month, number of days per month, etc. Fig. 3 illustrates how to schedule three weekly lectures.

The first lecture is scheduled at a specific date, in this instance, Monday August 21, 2017 (cell *A1*). The second lecture is August 28, 2017 (*A1*+1 in cell *A2*). The third lecture is scheduled for Monday, September 4, 2017.

Fig. 4 shows how to schedule courses on the Monday and Wednesday, recurring subsequent weeks. Here, both cells *A3* and *A4* refer back seven days. Fig. 5 shows an alternative where cell *A4* refers back two days within the same week instead. Note that the cell references are updated correspondingly when using copy and paste in the spreadsheet.

It is easy to schedule regular courses. For irregularly scheduled courses, each cell is given a specific date according to the room-booking schedule. If there are changes to either the times or the lecture progress, it is trivial to update the lists by moving cells around or modifying cells according to needs. When reusing the material for subsequent semesters, it may be sufficient simply to alter the start date to update the entire teaching plan with correct days of week and dates. The next section illustrates how to generate the documents.

2.3 Generating HTML-documents

The lists, including topic-time assignments and students, are stored in separate work-book sheets as lectures, students, etc. Each html document is constructed in a separate sheet. Once the sheet is constructed, all its contents are copied into a blank html-file with the extension html. The teacher can inspect the html-files using a web browser.

	A	B	C	D	E
1	=header				
2	=beginTable				
3	=beginLine	=lecture!a1	=separator	=lecture!b1	=endLine
4	=beginLine	=lecture!a2	=separator	=lecture!b2	=endLine
5	=beginLine	=lecture!a3	=separator	=lecture!b3	=endLine
6	=endTable				
7	=footer				

Fig. 6. Building html-markup.

	A	B
1	=header	<html><body>
2	=beginTable	<table>
3	=beginLine	=<tr><td>
4	=separator	=</tr></td><tr><td>
5	=endLine	=</tr></td>
6	=endTable	</table>
7	=footer	</body></html>

Fig. 7. html-definitions assigned to variables.

The html document in the html sheet is built using a mixture of html markup and content from the lists. Fig. 6 shows how to build an html document from the lecture example in Fig. 3. Cells in column A are references to spreadsheet variables. The teacher can define commonly-used markup in just one place, namely, a separate style sheet. Fig. 7 shows an example of a sheet with html-definitions. Changes to these definitions affect all the documents. Note that this example is stripped for certain markup-details and styling to simplify the presentation herein.

The labels in column *A* are provided for easy reference. To create an actual variable in excel, the cell needs to be given a name. Fig. 8 illustrates how to give cell *B4* the label *dateStart* (see the cell label field below the highlighted file menu). The content from the lecture lists is referred to in column *B* and *D*, respectively. Fig. 9 shows how the resulting document appears in the html sheet. These cells appear as ordinary html-markup when pasted into a text document.

2.4 Generating PDF-documents

We use Google Chrome to convert html-documents to pdf as Google chrome contains a convenient function for generating pdf-documents. Alternatively, one may use any other html-to-pdf converter.

2.5 Conditional Styling

Documents that are more complicated may require conditional styling of elements. For example, conditional styling is used if one wants certain elements to be marked as a deadline and no-lectures using consistent color and formatting.

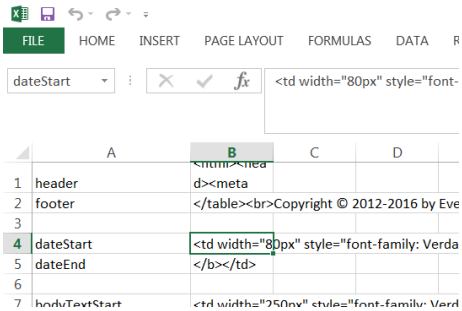


Fig. 8. Assigning variable names to cells in Excel.

	A	B	C	D	E
1	=<html><body>				
2	=<table>				
3	=<tr><td>	Mon. 21. Aug, 2017	=</tr></td><tr><td>	Introduction to the course	=</tr></td>
4	=<tr><td>	Mon. 28. Aug, 2017	=</tr></td><tr><td>	Basic theory	=</tr></td>
5	=<tr><td>	Mon. 4. Sep, 2017	=</tr></td><tr><td>	More theory..	=</tr></td>
6	=</table>				
7	=</body></html>				

Fig. 9. The appearance of the spreadsheet codes (from Figure 6).

A	B	C
21.08.2017		Introduction to the course
23.08.2017		Basic theory
28.08.2017	Holiday	Public holiday
30.08.2017	deadline	Some practice

Fig. 10. Controlling conditional markup.

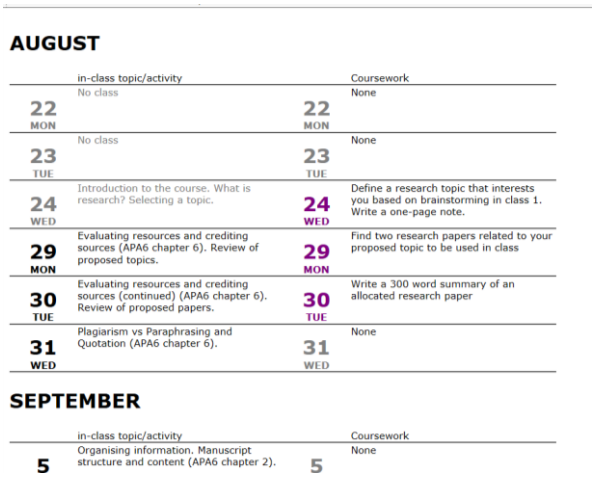
Fig. 10 shows an example of conditional markup of a lecture plan where column *A* contains the dates and column *C* the topics, while column *B* indicates the conditional markup. The corresponding html sheet may contain the following:

```
=IF(schedule!B4="holiday";holidayStart;IF(schedule!B4="
deadline";deadlineStart;normalStart))
```

This definition ensures the conditional formatting by checking if the corresponding cell (*B4*) in the schedule sheet contains the tag “holiday” or “deadline”, in which case the *holidayStart* markup or *deadlineStart* markup is included.

Fig. 11 illustrates how to generate a lecture schedule with the framework. This example uses purple colored texts for deadlines and grey colored texts for holidays. This example also contains decorated formatting of dates to assist students’ rapid comprehension. In addition, the framework automatically inserts month headings. The following definition achieves this.

```
=IF(tmp!F4<>tmp!F3;CONCATENATE(monthSeparatorBegin;UPPE
R(tmp!F4);monthSeparatorEnd);"")
```



AUGUST

	in-class topic/activity		Coursework
22 MON	No class	22 MON	None
23 TUE	No class	23 TUE	None
24 WED	Introduction to the course. What is research? Selecting a topic.	24 WED	Define a research topic that interests you based on brainstorming in class 1. Write a one-page note.
29 MON	Evaluating resources and crediting sources (APA6 chapter 6). Review of proposed topics.	29 MON	Find two research papers related to your proposed topic to be used in class
30 TUE	Evaluating resources and crediting sources (continued) (APA6 chapter 6). Review of proposed papers.	30 TUE	Write a 300 word summary of an allocated research paper
31 WED	Plagiarism vs Paraphrasing and Quotation (APA6 chapter 6).	31 WED	None

SEPTEMBER

	in-class topic/activity		Coursework
5	Organising information. Manuscript structure and content (APA6 chapter 2).	5	None

Fig. 11. Rendering of a lecture schedule with conditional formatting.

Group presentations

September 28, 2016	Erna Solberg and Jens Stoltenberg
September 29, 2016	Torbjørn jagland and Siv Jensen Kontra Bass and Electric Gitar
October 12, 2016	Ben Affleck and Emma Thomson Jonas Gahr Støre and Trine Skei Grande
October 13, 2016	Donald Duck and Mickey Mouse Erna Solberg and Jens Stoltenberg
October 19, 2016	Erna Solberg and Jens Stoltenberg Torbjørn jagland and Siv Jensen

Individual presentations

October 20, 2016	Bjarne Banan Erna Solberg Trond Giske
October 26, 2016	Kåre Willock Jens Stoltenberg Clemic Thommessen
October 27, 2016	Marit Nybakk Kennet Svendsen Svein Roald Hansen

Fig. 12. A fictitious presentation schedule with both individual and group presentations.

The definition refers to a tmp sheet with the month parts isolated from the dates. It is then easy to check if months of two neighboring cells are equal or different. The monthSperatorBegin and monthSeparatorEnd variables contain the styling, and the Excel UPPER function is used to convert the month name to uppercase for esthetical purposes.

	A	B	C	D	E	F	G	H
1	1							
2	=IF(LEN(C1)>0;A1+C1;A1)	=IF(LEN(C1)>0;schedule!A1;"")	=IF(ISNUMBER(SEARCH("group present";schedule!C1));IF(LEN(schedule!C1)<20;2*pairs;pairs);"")	=E5&F5&G5&H5	=IF($C2>E$1;INDIRECT("tmppair!a"&($A2+E$1))&delimiter;"")	=IF($C2>F$1;INDIRECT("tmppair!a"&($A2+F$1))&delimiter;"")		
3	=IF(LEN(C2)>0;A2+C2;A2)	=IF(LEN(C1)>0;schedule!A1;"")	=IF(ISNUMBER(SEARCH("group present";schedule!C2));IF(LEN(schedule!C2)<20;2*pairs;pairs);"")	=E5&F5&G5&H5	=IF($C3>E$1;INDIRECT("tmppair!a"&($A4+E$1))&delimiter;"")	=IF($C3>F$1;INDIRECT("tmppair!a"&($A3+F$1))&delimiter;"")		

Fig. 13. Structure for scheduling presentations.

2.6 Scheduling Presentations

Fig. 12 shows a document containing the presentation schedule for students enrolled onto a course. We rendered this document using both information in the lecture list and the student list.

In short, the method first identifies timeslots allocated for presentations. Presentation slots are numbered consecutively. These numbers are used to pull names from the student list using the INDIRECT excel function as the INDIRECT function parameterizes cell references. The process is only performed if the lecture cell is not empty.

This means that the presentation sheet contains more rows than what are actually used. Such empty rows pose no problem as html-browsers do not display blank space. An instance of the word “presentations” in the topic description of the schedule-sheet signals a presentation. If the “presentations” keyword is the only word, the entire session is allocated for (more) presentations; otherwise, half of the timeslot is allocated for the lecture and the rest is allocated for (fewer) presentations.

The first column of a student-sheet lists the students with one student per row. Similarly, a list of pairs can be set up where students are combined randomly or systematically.

Next, the main engine for generating the presentation schedule is a temporary *tmppres*-sheet. In this sheet, one column (in this case column *C*) is connected to the schedule-sheet. If the schedule sheet contains the keywords “group-presentation” or “individual presentation”, the corresponding cell is given a number indicating the number of students to present for the given date. This means that lecture slots without presentations will remain blank, or zero. For all cells in column-*C* which have a value greater than zero, the corresponding date in schedule is copied into the neighboring cell in column *B*. Column *A* is used to indicate the start index of the student to present from the list. It is updated by using the current start index with the number of students in cell *C* of the previous row added to it.

The current student is copied into a list in column *D*. This list is built by copying the names from the student list in the students-sheet to the remaining column of the tmppres-sheet. The start index in column A for a given row indicates the first student. The structure of the tmppres-sheet is provided in Fig. 13.

Finally, the presentation-output-sheet copies the cells in column *B* from the *tmppres*-sheet, namely, the date for the presentation, or nothing if there is no presentation. Html styling is added if the cell in column *B* for the given row is non-empty. For each row, the student is copied from cell *D* if the date field is non-empty.

	A	B	C	D	E	F
1	=RAND()	=RAND()	=RANK(A1;A:A)	=RANK(B1;B:B)	=INDIRECT(CONCATENATE("stduent!a";A1))	=INDIRECT(CONCATENATE("stduent!a";B1))
2	=RAND()	=RAND()	=RANK(A2;A:A)	=RANK(B2;B:B)	=INDIRECT(CONCATENATE("stduent!a";A2))	=INDIRECT(CONCATENATE("stduent!a";B2))
3	=RAND()	=RAND()	=RANK(A3;A:A)	=RANK(B3;B:B)	=INDIRECT(CONCATENATE("stduent!a";A3))	=INDIRECT(CONCATENATE("stduent!a";B3))

Fig. 14. Excel commands to generate a peer-review list.

	A	B	C	D	E	F
1	0,453	0,563	2	3	John	Beth
2	0,785	0,888	1	2	Lisa	John
3	0,123	0,931	3	1	Beth	Lisa

Fig. 15. The appearance of the peer-review Excel commands (depending on random seed).

Although the procedure may appear complicated, it only needs to be set up once. Given such a structure, the teacher can simply edit the progress plan, parameters controlling the number of students per lecture, and student lists. The framework performs the fitting of students into the lecture sessions. Certain fine-tuning may be needed as student numbers vary from year to year, while the amount of lectures usually remains fixed. However, with the framework, the teacher can easily adjust the parameters until the students fill up the slots.

2.7 Organizing Peer-Review

The last example illustrates how to generate peer review lists using Excel. For each student, the peer review list shows the other students who are to check the work. Clearly, students should not review their own work and no student should review the same work twice. Two reviewers are common, and the example presented herein assumes two reviewers.

Row *A* in the student-sheet contains the names of the students. For each student, a random number is generated for each reviewer. Given two reviewers, two random numbers are generated. To make the routine more generic, a random number is only generated if the corresponding cell in the student list is not empty. Next, the Excel RANK function is used to generate a rank of the random numbers in a new row for each row of the random numbers. This rank gives the index to the reviewing student in the student list. The student is thus accessed using the INDIRECT function based on the rank. The structure of the peer review sheet is shown in Fig. 14. Fig. 15 displays how this may appear (depending on the random number seed).

Note that only the reviewers' names are included here for simplicity. The reviewees' names are copied directly from the student list.

For each review, three simple checks are performed to ensure that the list is correct. First, we check if the two reviewers are identical. Second, we verify if the first reviewer is identical to the reviewee. Lastly, we inspect if the second reviewer is identical to reviewee. The results of all these checks are summed, where true is 1 and false is 0. If this sum is larger than zero, the sheet is refreshed giving it a new random seed. The process is repeated until there are no errors. It is usually sufficient to refresh the sheet around 3-5 times before the result is free of errors.

3 Conclusions

This paper presented a simple framework for managing course information. A spreadsheet was used to implement the framework, as spreadsheet software is commonly available. Although some aspects of the implementation are intricate, it only needs to be implemented once. The teacher can configure the framework to suit a specific teaching scenario without having to know programming of html. The framework ensures that information is only stored in one place. Changes, such as updated student lists and lecture schedule, propagate automatically to all affected documents. Furthermore, the framework separates the visual layout from the content, giving teachers freedom to tailor the visual appearance of teaching material without affecting the content and vice versa. The framework helps teachers reduce time and errors.

References

1. Wang, Q., Woo, H.L., Quek, C.L., Yang, Y., Liu, M.: Using the Facebook group as a learning management system: An exploratory study. British Journal of Educational Technology 43, 428-438 (2012)
2. McGill, T.J., Klobas, J.E.: A task–technology fit view of learning management system impact. Computers & Education 52, 496-508 (2009)
3. Beatty, B., Ulasewicz, C.: Faculty perspectives on moving from Blackboard to the Moodle learning management system. TechTrends 50, 36-45 (2006)
4. Weaver, D., Spratt, C., Nair, C.S.: Academic and student use of a learning management system: Implications for quality. Australasian journal of educational technology 24, 30-41 (2008)
5. Reenskaug, T.: Thing-model-view-editor—An example from a planningsystem. technical note, Xerox Parc (1979)
6. Krasner, G.E., Pope, S.T.: A description of the model-view-controller user interface paradigm in the smalltalk-80 system. Journal of object oriented programming, 1(3), 26-49 (1988)
7. Leff, A., Rayfield, J.T.: Web-application development using the model/view/controller design pattern. In Enterprise Distributed Object Computing Conference, 2001. EDOC'01. Proceedings. Fifth IEEE International (pp. 118-127). IEEE (2001)
8. Yu, Z.Q., Ran, S.Y., Li, S.: Design and Implementation of Teaching Material Management System Based on MVC Pattern. Computer Technology and Development 1, 58 (2006)
9. Eika, E., Sandnes, F.E.: Authoring WCAG2. 0-compliant texts for the web through text readability visualization. In: Proceedings of HCI International 2016, Universal Access in Human-Computer Interaction. Methods, Techniques, and Best Practices (eds: Margherita Antona and Constantine Stephanidis), LNCS Vol. 9737, pp. 49-58, Springer (2016)
10. Eika, E., Sandnes, F.E.: Assessing the Reading Level of Web Texts for WCAG2. 0 Compliance—Can It Be Done Automatically?. In: Advances in Design for Inclusion (eds: Di Bucchianico, G, Kercher, P.), pp. 361-371, Springer (2016)
11. Jian, H.L., Sandnes, F.E., Law, K.M., Huang, Y.P., Huang, Y.M.: The role of electronic pocket dictionaries as an English learning tool among Chinese students. Journal of Computer Assisted Learning 25, 503-514 (2009)
12. Jian, H.L., Sandnes, F.E., Huang, Y.P., Cai, L., Law, K. M.: On students' strategy-preferences for managing difficult course work. IEEE Transactions on Education 51, 157-165 (2008)