Use Cases: *Course Management*

David Bialik, Julian Fleischer, Hagen Mahnke,
Konrad Reiche, André Zoufahl

March 4, 2011

	Program manager creates a new Course
Actors	Program Manager
Scope	GUI
Precondition	The user is logged in.
Postcondition	
Postcondition on Success	A new Course is created
Basic Course of Events	**1.** The user navigates to the admin list of all courses. **2.** The user clicks a "create new" button at the top of the list. **3.** The user sees a form to enter details about the course: name, Courses that are required for this Course (from a list of existing courses) with a button to add more, required features (from a list of existing features) and their quantity with a button to add more, CourseAttributes (from a list of existing CourseAttributes) and their value with a button to add more, a year. **4.** The user submits the form. **5.** The Course is created. The user sees an empty "new Course" form.
Alternative Paths	**1a.** The user does not have permission to edit courses and sees a message "Permission denied" **5a.** The Course is not created due to invalid values and the form is presented again, containing the data that was entered and information about which values are defective. **5b.** A database exception occurs. The same form including the data is presented again plus a message stating the problem.
Open Questions	-
Solved issues	-
Implementation Notes	• Values for CourseAttributes are always Strings.
Implementation Status	• not implemented

	Program manager edits a course
Actors	Program Manager, GUI, Database
Scope	
Precondition	The system is running and the user is logged in with the rights of a program manager.
Postcondition	The system is still running, the user is still logged in and previous course data is not lost due to exceptions.
Postcondition on Success	The course information is updated into the database.
Basic Course of Events	**1.** The user navigates to the administrator list of all courses. **2.** The user clicks an edit button beside the course the user wishes to edit. **3.** The GUI displays a form with details about the course, e.g. name, course elements, course instances, courses that are required for this course with a button to add more, required features and their quantity with a button to add more, course attributes abd their value with a button to add more and a year. **4.** The user edits the information. **5.** The user submits the form. **6.** The GUI updates the altered course information to the databasse. The user sees the admin list of all courses.
Alternative Paths	**1a.** The user does not have permission to edit courses and the GUI displays the message "Permission denied". **6a.** The Course information is not altered due to invalid values and the form is presented again, containing the data that was entered and information about which values are defective **6b.** A database exception occurs. The same form including the data is presented again plus a message stating the problem
Open Questions	-
Solved issues	-
Implementation Notes	• Values for CourseAttributes are always Strings.
Implementation Status	• not implemented

	Program manager views the admin list of Courses
Actors	Program Manager
Scope	
Precondition	The user is logged in.
Postcondition	
Postcondition on Success	The user sees the admin list of Courses
Basic Course of Events	**1.** The user navigates to the admin list of all Courses. **2.** The user navigates to the admin list of all Courses for a certain program. The page contains a button to create a new course at the top and a list of single letters for navigation to Courses that start with this letter. Below that are the courses (ordered lexicographically). Beside each Course are buttons to delete and edit
Alternative Paths	**1a.** The user does not have permission to see a list of Courses and sees a message "Permission denied"
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	• not implemented

	Program manager deletes a Course
Actors	Program Manager
Scope	
Precondition	The user is logged in.
Postcondition	
Postcondition on Success	The Course is deleted.
Basic Course of Events	**1.** The user navigates to the admin list of all Courses. **2.** The user clicks the delete button beside the Course he wishes to delete. **3.** The user sees a prompt to choose what shall happen to associated CourseInstances, CourseElements and CourseElementInstances. The choices are "delete all", "keep all" and "delete selected" as radio buttons. When selecting "delete marked", a list with CI, CE, CEI and a check-box for each becomes available. The prompt can be terminated by pressing "ok" or "abort", and "abort" is the default selection. **4.** The user makes his decision and presses "ok". **5.** The user sees a prompt, to ensure he wants to delete the Course. The choices are "yes", "no" and "abort" The default choice is "no". **6.** The user chooses "yes". **7.** The Course (and CI,CE,CEI if desired) is tagged as deleted. The user sees the admin list of all Courses.
Alternative Paths	**1a.** The user does not have permission to edit courses and sees a message "Permission denied" **7a.** The Course is not tagged as deleted due to a database exception. The user sees a prompt stating the problem and a "retry" and a "abort" button. The default is "retry". Every subsequent failed attempt brings up the same prompt.
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	• not implemented

	Program Manager or Main Lecturer views the detail view of a Course
Actors	Program Manager or Main Lecturer
Scope	
Precondition	The user is logged in.
Postcondition	
Postcondition on Success	The user views the detail view of a Course.
Basic Course of Events	**1.** The user navigates to an admin list of Courses that contains the desired Course. **2.** The user clicks on the name of the Course. **3.** The user sees the detail view of that Course. It contains information: name, CourseElements, CourseInstances, Courses that are required for this course with a button to add more, required features and their quantity with a button to add more, CourseAttributes and their value with a button to add more, a year.
Alternative Paths	-
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	• not implemented

	Program manager creates a new CourseInstance
Actors	Program Manager
Scope	
Precondition	The user is logged in.
Postcondition	
Postcondition on Success	A new CourseInstance is created
Basic Course of Events	**1.** The user navigates to the admin list of all CourseInstances. **2.** The user clicks a "create new" button at the top of the list. **3.** The user sees a form to enter details about the course: Course, program (from a list of programs), start and end date, main-lecturer (from a list of lecturers), CourseElementInstance (from a list of CEIs) with a button to add more. The form also contains a smart-copy option. Here a Course and a Program of which the Course was part can be chosen from a list and a button to copy those values into the form is present. **4.** The user enters the information manually or enters them via smart-copy (maybe adjusting manually). **5.** The user submits the form. **6.** The CourseInstance is created. The user sees an empty "new Course" form.
Alternative Paths	**1a.** The user does not have permission to edit CourseInstances and sees the public list of all CourseInstances. **6a.** The CourseInstance is not created due to invalid values and the form is presented again, containing the data that was entered and information about which values are defective **6b.** The CourseInstance is not created due to a database exception. The same form including the data is presented again plus a message stating the problem.
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	• not implemented

	Main lecturer or Program Manager edits a CourseInstance
Actors	Main Lecturer or Program Manager
Scope	
Precondition	The user is logged in.
Postcondition	
Postcondition on Success	The information of a CourseInstance is altered.
Basic Course of Events	**1.** The user navigates to a (public or admin) list of all CourseInstances. **2.** The user clicks a CourseInstance. **3.** The user sees the detail view of that CourseInstance. **4.** The user clicks the the edit button. **5.** The user sees a form, containig the current values for that CourseInstance. **6.** The user edits the information and submits the form. **7.** The altered CourseInstance information is saved. The user sees the detail view of the CourseInstance.
Alternative Paths	**4a.** The user does not have permission to edit this CourseInstance and gets a view without the button. **5a.** The user does not have permission to edit this CourseInstance and sees a message stating that he lacks permission for that action. **7a.** The altered CourseInstance information is not saved due to invalid values and the form is presented again, containing the data that was entered and information about which values are defective **7b.** The altered CourseInstance information is not saved due to a database exception. The same form including the data is presented again plus a message stating the problem.
Open Questions	**Q:** Does our system support the restrictions stated above, that is especially a Main Lecturer may only edit his CourseInstances and a Program Manager may only edit CourseInstances of his program?
Solved issues	-
Implementation Notes	• A Program Manager may only edit CourseInstances from his Program. A Main Lecturer may only edit CourseInstances where he is assigned as Main Lecturer.
Implementation Status	• not implemented

	A Program Manager deletes a CourseInstance
Actors	Program Manager
Scope	
Precondition	The user is logged in.
Postcondition	
Postcondition on Success	The CourseInstance is tagged as deleted.
Basic Course of Events	**1.** The user navigates to the admin list of all CourseInstances. **2.** The user clicks the "delete" button right next to the CourseInstance she wishes to delete. **3.** The user sees a prompt asking whether associated CourseElementInstances shall be deleted as well. The Choices are "yes", "no" and "abort" and "yes" is the default. **4.** The user chooses "yes". **5.** The user sees a prompt asking to verify the deletion of the CourseInstance. The Choices are "yes", "no" and "abort" and "no" is the default. **6.** The user chooses "yes". **7.** The CourseInstance (and associated CEI if desired)is tagged as deleted. The user sees the admin list of all CourseInstances.
Alternative Paths	**3a.** The user does not have permission to edit that CourseInstance and sees a message stating that he lacks permission for that action. **7a.** The CourseInstance is not tagged as deleted due to a database exception. The user sees a prompt stating the problem and a "retry" and a "abort" button. The default is "retry". Every subsequent failed attempt brings up the same prompt.
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	• not implemented

	A user views a list of CourseInstances.
Actors	Any user
Scope	
Precondition	
Postcondition	
Postcondition on Success	The user sees the list of CourseInstances she desires.
Basic Course of Events	**1.** The user navigates to the public list of all CourseInstances. **2.** The user sees a list of all CourseInstances and (horizontal) lists for academic terms, departments and letters. Also a check-box for "My CourseInstances" is available. Beneath the lists is an "apply" button. **3.** The user makes his selection for a sub-list and presses "apply". **4.** The user sees the selected sub-list and the same selection possibilities as before, indicating the selection via color/bold font.
Alternative Paths	**1a.** The user is a Program Manager or Main Lecturer and sees the admin list of all CourseInstances. **2a.** >The user sees the public list and beside each CourseInstance are "edit" and "delete" buttons plus a "create new" button at the top of the page. **3a.** The user is satisfied with the list shown.
Open Questions	-
Solved issues	-
Implementation Notes	• Each entry in a list defines a subset of CourseInstances. For example applying the selection "Computer Science" will result in a list that contains all CourseInstances associated with computer science. Multiple selections are possible, selecting multiple values from one list will produce the union of the subsets. Selecting values from different lists will produce the intersection. Each list has "all" as an entry. All is the same as not selecting for that list. Selecting "My CourseInstances" will produce the subset of CourseInstances where the user is involved.
Implementation Status	• not implemented

	A user views the detail view of a CourseInstance
Actors	Any user
Scope	
Precondition	
Postcondition	
Postcondition on Success	The user sees the detail view of the CourseInstance he is interested in.
Basic Course of Events	**1.** The user navigates to a list of CourseInstances that includes the desired CourseInstance (see use case "A user views a list of CourseInstances") **2.** The user clicks on the name of the CourseInstance **3.** The user sees the detail view of that CourseInstance, which includes: Main lecturer, academic term, department(s),recommended for year, description and (if already allocated) time and place or the associated CourseInstanceElements.
Alternative Paths	**3a.** The user has permission to edit this CourseInstance and also sees buttons to delete and edit.
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	• not implemented

	Program Manager creates a CourseElement
Actors	Program Manager
Scope	
Precondition	The user is logged in.
Postcondition	
Postcondition on Success	A new CourseElement is created.
Basic Course of Events	**1.** The user navigates to the admin list of CEs. **2.** The user presses the "create new" button at the top of the list. **3.** The user sees a form to enter the information about the new CE. The fields include: The Course that this CE is associated with, CourseElementType (from a list of existing CETs), name, duration, whether the CE is mandatory, the features that are required and their quantity (with a button to add more). Of these fields only the name and the type can be null. **4.** The user submits the form **5.** The CourseElement is created. The user sees an empty "new CourseElement" form.
Alternative Paths	**5a.** The CourseElement is not created due to invalid values and the form is presented again, containing the data that was entered and information about which values are defective. **5b.** A database exception occurs. The same form including the data is presented again plus a message stating the problem.
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	-

	Program Manager edits a CourseElement
Actors	Program Manager
Scope	
Precondition	The user is logged in.
Postcondition	
Postcondition on Success	A CourseElement is altered.
Basic Course of Events	**1.** The user navigates to the admin list of CEs. **2.** The user presses the "edit" button beside the CE. **3.** The user sees a form containing the current information about the CE. The fields include: The Course that this CE is associated with, CourseElementType (from a list of existing CETs), name, duration, whether the CE is mandatory, the features that are required and their quantity (with a button to add more). Of these fields only the name and the type can be null. **4.** The user edits the values. **5.** The user submits the form. **6.** The CourseElement is altered. The user sees the detail view of the CE.
Alternative Paths	**6a.** The CourseElement is not altered due to invalid values and the form is presented again, containing the data that was entered and information about which values are defective. **6b.** A database exception occurs. The same form including the data is presented again plus a message stating the problem.
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	-

	Program Manager deletes a CourseElement
Actors	Program Manager
Scope	
Precondition	The user is logged in.
Postcondition	
Postcondition on Success	A CourseElement is deleted.
Basic Course of Events	**1.** The user navigates to the admin list of CEs. **2.** The user presses the "delete" button beside the CE. **3.** The user sees a prompt asking whether associated CourseElementInstances shall be deleted. The choices are "yes", "no" and "abort". Default is yes. **4.** The user chooses "yes". **5.** The user sees a prompt to confirm the deletion of the CE. The choices are "yes" and "no", "no" being default. **6.** The user chooses "yes". **7.** The CourseElement (and associated CEI if desired)is tagged as deleted. The user sees the admin list of all CourseElements.
Alternative Paths	**7a.** The CourseElement is not tagged as deleted due to a database exception. The user sees a prompt stating the problem and a "retry" and a "abort" button. The default is "retry". Every subsequent failed attempt brings up the same prompt.
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	-

	A Program Manager views an admin list of CourseElements.
Actors	Program Manager
Scope	
Precondition	
Postcondition	
Postcondition on Success	The user sees the admin list of CourseElements she wants.
Basic Course of Events	**1.** The user navigates to the admin list of all CourseElements. **2.** The user sees a list of all CourseElements and (horizontal) lists for departments and letters. Also a textfield to search for CEs belonging to a specific Course is available. Beneath the lists is an "apply" button. Beside each CourseElement are "edit" and "delete" buttons plus a "create new" button at the top of the page. **3.** The user makes his selection for a sub-list and presses "apply". **4.** The user sees the selected sub-list and the same selection possibilities as before, indicating the selection via color/bold font.
Alternative Paths	**3a.** The user is satisfied with the list shown.
Open Questions	-
Solved issues	-
Implementation Notes	• Each entry in a list defines a subset of CourseElements. For example applying the selection "Computer Science" will result in a list that contains all CourseElementss associated with computer science. Multiple selections are possible, selecting multiple values from one list will produce the union of the subsets. Selecting values from different lists will produce the intersection. Each list has "all" as an entry. All is the same as not selecting for that list. Entering a name of a Course in the textfield will produce the subset of CourseElements that are asscociated with that Course.
Implementation Status	• not implemented

	A Program Manager views the detail view of a CourseElement
Actors	Program Manager
Scope	
Precondition	The user is logged in.
Postcondition	
Postcondition on Success	The user views the detail view of a CourseElement.
Basic Course of Events	**1.** The user navigates to an admin list of CourseElements containing the desired CE. **2.** The user clicks the name of the CE. **3.** The user sees the detail view of the CE. Information about the Course that this CE is associated with, CourseElementType (from a list of existing CETs), name, duration, whether the CE is mandatory, the features that are required and their quantity (with a button to add more) is displayed. Buttons to "edit" or "delete" this CE are also available.
Alternative Paths	-
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	• not implemented

	Main lecturer creates a CourseElementInstance
Actors	Main lecturer
Scope	
Precondition	The user is logged in.
Postcondition	
Postcondition on Success	A new CourseElementInstance is created.
Basic Course of Events	**1.** The user navigates to the detail view of a Course. **2.** The user presses the "create new instance" button beside a CourseElement. **3.** The user sees a form to enter the information about the new CEI. The fields include: The CourseInstance that this CEI is associated with from a list of CIs (default is the most recent CourseInstance), lecturer, duration. Of these fields only the lecturer can be null. **4.** The user submits the form **5.** The CourseElementInstance is created. The user sees the detail view of the Course.
Alternative Paths	**5a.** The CourseElementInstance is not created due to invalid values and the form is presented again, containing the data that was entered and information about which values are defective. **5b.** A database exception occurs. The same form including the data is presented again plus a message stating the problem.
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	-

	Main Lecturer edits a CourseElementInstance
Actors	Main Lecturer
Scope	
Precondition	The user is logged in.
Postcondition	
Postcondition on Success	The CEI is altered.
Basic Course of Events	**1.** The user navigates to an admin list of CEIs. **2.** The user presses the "edit" button beside the CEI. **3.** The user sees a form containing the current information about the CEI. The fields include: The CourseInstance that this CEI is associated with, the CourseElement that this CEI is an instance of, lecturer, duration. Of these fields only the lecturer can be null. **4.** The user edits the values. **5.** The user submits the form. **6.** The CourseElementInstance is altered. The user sees an admin list of CEIs.
Alternative Paths	**6a.** The CourseElementInstance is not altered due to invalid values and the form is presented again, containing the data that was entered and information about which values are defective. **6b.** A database exception occurs. The same form including the data is presented again plus a message stating the problem.
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	-

	Main Lecturer deletes a CourseElementInstance
Actors	Main Lecturer
Scope	
Precondition	The user is logged in.
Postcondition	
Postcondition on Success	A CourseElementInstance is deleted.
Basic Course of Events	**1.** The user navigates to an admin list of CEIs containing the desired CEI. **2.** The user presses the "delete" button beside the CEI. **3.** The user sees a prompt to confirm the deletion of the CEI. The choices are "yes" and "no", "no" being default. **4.** The user chooses "yes". **5.** The CourseElement is tagged as deleted. The user sees the admin list of all CEI.
Alternative Paths	**5a.** The CEI is not tagged as deleted due to a database exception. The user sees a prompt stating the problem and a "retry" and a "abort" button. The default is "retry". Every subsequent failed attempt brings up the same prompt.
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	-

	A user views a list of CourseElementInstances.
Actors	Any user
Scope	
Precondition	
Postcondition	
Postcondition on Success	The user sees list of CEIs she wants.
Basic Course of Events	**1.** The user navigates to the list of all CEIs. **2.** The user sees a list of all CEIS and (horizontal) lists for academic terms, departments and letters. Also a textfield to search for CEIs belonging to a specific Course is available. Beneath the lists is an "apply" button. If the user has permission to edit CEIs there are "edit" and "delete" buttons beside each CEI plus a "create new" button at the top of the page. **3.** The user makes his selection for a sub-list and presses "apply". **4.** The user sees the selected sub-list and the same selection possibilities as before, indicating the selection via color/bold font.
Alternative Paths	**3a.** The user is satisfied with the list shown.
Open Questions	-
Solved issues	-
Implementation Notes	• Each entry in a list defines a subset of CEIs. For example applying the selection "Computer Science" will result in a list that contains all CEIs associated with computer science. Multiple selections are possible, selecting multiple values from one list will produce the union of the subsets. Selecting values from different lists will produce the intersection. Each list has "all" as an entry. All is the same as not selecting for that list. Entering a name of a Course in the textfield will produce the subset of CEIs that are asscociated with that Course.
Implementation Status	• not implemented

	A user views the detail view of a CourseElementInstance
Actors	Any user
Scope	
Precondition	
Postcondition	
Postcondition on Success	The user views the detail view of a CEI.
Basic Course of Events	**1.** The user navigates to a list of CEIs containing the desired CEI. **2.** The user clicks the name of the CEI. **3.** The user sees the detail view of the CEI. Information includes: The CourseInstance that this CEI is associated with, the CourseElement that this CEI is an instance of, duration. If the user has permisson to edit CEIs, buttons to "edit" or "delete" this CEI are also available.
Alternative Paths	-
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	• not implemented

	A course element is removed from a course
Actors	Program Manager, GUI, Database
Scope	GUI
Precondition	The system is running, the user is still logged in.
Postcondition	
Postcondition on Success	The course element is deleted from the database.
Basic Course of Events	**1.** User opens the course element panel. **2.** User selects a course elements. **3.** User clicks on submit. **4.** GUI deletes the course element from the datbase.
Alternative Paths	-
Open Questions	**Q:** What happens with already instantiated course elements (which we call course element instance)?
Solved issues	-
Implementation Notes	-
Implementation Status	-

	Create a Course Element Types
Actors	Program Manager, GUI, Database
Scope	GUI
Precondition	The system is running, the user is still logged in.
Postcondition	
Postcondition on Success	The new course element type is insirted into the database.
Basic Course of Events	**1.** User opens the course element type panel. **2.** User enters the name of the course element type. **3.** User clicks on submit **4.** GUI inserts the new course element type into the database.
Alternative Paths	**4a.** GUI tries to insert the new course element type into the database. **4b.** Database throws a conflict on the insertion, because a course element type of this name already exists. **4c.** GUI displays the exception.
Open Questions	-
Solved issues	**Q:** What's the use for course element types? Do we need or want them at all? **A:** Course Element types have no functional use, for instance for the scheduler or similiar. Instead it is just an information about the type of the course element.
Implementation Notes	-
Implementation Status	-

	Program Manager creates a new program
Actors	Program Manager, Database, GUI
Scope	GUI
Precondition	The System is running and the user is logged in.
Postcondition	The System is still running and the user is still logged in.
Postcondition on Success	A new program is inserted into the database.
Basic Course of Events	**1.** User opens the program panel. **2.** User chooses to create a new program. **3.** User selects the department of the new program. **4.** User selects the academic term of the new program. **5.** User clicks on submit. **6.** GUI inserts the new program.
Alternative Paths	-
Open Questions	-
Solved issues	**Q:** What is the difference between the creation of a program and an academic term? Should a program automatically be generated per department when an Academic Term is created? **A:** Yes, they should. It might be an unnecessary step.
Implementation Notes	• The program creation was implemented very differently. We have come to realize the program creation is a starting point for creating alot of entities which are dependent to the program, for instance course element instances. Therefore the program creation is now a guided process of chained forms.
Implementation Status	-

	Admin creates a new Academic Term
Actors	Program Manager, Database, GUI
Scope	GUI
Precondition	The System is running and the user is logged in.
Postcondition	The System is still running and the user is still logged in.
Postcondition on Success	A new academic term is inserted into the database.
Basic Course of Events	**1.** User opens the academic term panel. **2.** User chooses to create a new academic term. **3.** User the name of the new academic term. **4.** User clicks on submit. **5.** GUI inserts the new academic term into the database.
Alternative Paths	-
Open Questions	**Q:** How exactly does smarty copy work? How are starting/ending dates resolved? How Course Element Instances and main-lecturers? **Q:** Should this use case be splitted up into two use cases: "create AT", "smarty create AT"?
Solved issues	-
Implementation Notes	• Smarty copy.
Implementation Status	-

	Edit an Academic Term
Actors	Program Manager, Database, GUI
Scope	GUI
Precondition	The System is running and the user is logged in.
Postcondition	The System is still running and the user is still logged in.
Postcondition on Success	The academic is altered.
Basic Course of Events	**1.** User opens the academic term panel. **2.** User chooses to edit an academic term. **3.** User changes the name of the new academic term. **4.** User clicks on submit. **5.** GUI updates the new academic term into the database..
Alternative Paths	-
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	-