Use Cases: *Scheduling (both collaborative and automatic)*

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

February 14, 2011

	Program admin starts scheduling
Actors	Program administrator, GUI, Scheduler, Database
Scope	GUI
Precondition	The system is running and the user is logged in with the rights of a program administrator.
Postcondition	The system is still running, the user is still logged in and if there was a previous proposed scheduling it is not lost due to exceptions.
Postcondition on Success	The proposed schedule is inserted or updated into the database.
Basic Course of Events	**1.** User opens schedule panel. **2.** User defines a program to schedule. **3.** User clicks on start. **4.** Scheduler launches the scheduling of the defined program. **5.** Scheduler changes its status to running. **6.** User waits until the scheduling is finished by itself. **7.** Scheduler inserts the proposed schedule into the database. **8.** Scheduler changes its status to ready.
Alternative Paths	**6a.** User clicks on stop. **6b.** Scheduler changes its status to stopping. **7a.** Scheduler updates the proposed schedule in the database.
Open Questions	**Q:** Should a program administrator only be enabled to schedule his own program? **Q:** Should an administrator also be enabled to schedule a program?
Solved issues	**Q:** Should the program administrator get a notifaction on his next login? **A:** Yes. **Q:** Will there be scheduling proposals for conctlict resolutions? **A:** No, but the user will be informed about the reasons of the conflicts
Implementation Notes	-
Implementation Status	• Program administrator notification is missing.

	Pause the scheduling process
Actors	Program administrator, GUI, Scheduler, Database
Scope	GUI
Precondition	The system is running, the scheduling process has been started, it is running and the user is still logged in with the rights of a program administrator.
Postcondition	The system is still running, the user is still logged in and if the pausing did not take action yet the scheduling status is set to stopping.
Postcondition on Success	The scheduling process is paused.
Basic Course of Events	**1.** User opens schedule panel. **2.** User clicks on stop. **3.** Scheduler changes its status to stopping. **4.** Schedulers inserts current proposed schedule into the database. **5.** Scheduler changes its status to ready.
Alternative Paths	**4a.** Scheduler updates current proposed schedule in the database.
Open Questions	-
Solved issues	-
Implementation Notes	• The Java thread interruption policy is used for implementation.
Implementation Status	-

	Resume a scheduling process
Actors	Program administrator, GUI, Scheduler, Database
Scope	GUI
Precondition	The system is running, the scheduling process is ready and the user is still logged in with the rights of a program administrator.
Postcondition	The system is still running, the user is still logged in and if there was no proposed schedule the scheduling process starts as a new scheduling.
Postcondition on Success	The resumed scheduling starts with the previous number of resolved constraints.
Basic Course of Events	**1.** User opens schedule panel. **2.** User clicks on resume. **3.** Scheduler launches the scheduling of the defined program. **4.** Scheduler changes its status to running.
Alternative Paths	-
Open Questions	-
Solved issues	-
Implementation Notes	• Resume is implemented by using the current proposed schedule of each course to reallocate their position in the scheduler internal presentation of rooms, time slots and courses.
Implementation Status	-

	A conflict occurs
Actors	Program administrator, GUI, Scheduler, Database
Scope	GUI
Precondition	The system is running, the user is logged in with the rights of a program administrator and the scheduling was started.
Postcondition	The system is still running, the user is still logged in and if there was a previous proposed scheduling it is not lost due to the conflict.
Postcondition on Success	The reasons for the conflicts are displayed.
Basic Course of Events	**1.** Scheduler spots a non-solveable conflict while scheduling. **2.** Scheduler notfies the systems about the conflict. **3.** Scheduler terminates the scheduling process. **4.** User opens schedule panel. **5.** GUI displays the reason of the conflict.
Alternative Paths	-
Open Questions	**Q:** The scheduler uses exceptions to inform the system about conflicts. Will the exceptions be saved to display them until the user opens the schedule panel again?
Solved issues	**Q:** Should the current proposed schedule of the scheduling process be inserted or updated into the database? **A:** No, as the current proposed schedule is incomplete.
Implementation Notes	-
Implementation Status	-

	Change the proposed schedule of a Course Element Instance
Actors	Program administrator, Main lecturer, GUI, Scheduler, Database
Scope	GUI
Precondition	The system is running and user is logged in with the rights of a program administrator or main lecturer.
Postcondition	The system is still running, the user is still logged in and if there was a previous proposed schedule it is not lost due to exceptions.
Postcondition on Success	The proposed schedule is inserted or updated into the database.
Basic Course of Events	**1.** User opens the timetable of a room. **2.** User moves the Course Element Instance to its new time slot. **3.** User clicks on submit. **4.** Scheduler calculates the score and potential conflicts. **5.** GUI displays potential conflicts. **6.** User clicks on accept. **7.** Scheduler inserts the new proposed schedule into the database.
Alternative Paths	**3a.** User clicks on discard. **3b.** GUI reverses the changes and displays the original state. **6a.** User clicks on discard. **6b.** GUI reverses the changes and displays the original state. **7a.** Scheduler updates the existing proposed schedule in the database.
Open Questions	-
Solved issues	**Q:** Should the program administrator get a notification on his next login? **A:** Yes. **Q:** Will there be scheduling proposals for conflict resolutions? **A:** No, but the user will be informed about the reasons of the conflicts.
Implementation Notes	-
Implementation Status	-

	Automatically reschedule a subset of courses
Actors	Program administrator, GUI, Scheduler, Database
Scope	GUI
Precondition	The system is running, the user is logged in with the rights of a program administrator and there is no running scheduling process.
Postcondition	The system is still running, the user is still logged in and if there was a previous proposed schedule it is not lost due to exceptions.
Postcondition on Success	The proposed schedule is inserted or updated into the database.
Basic Course of Events	**1.** User opens the course panel. **2.** User defines a program **3.** User selects the course element instances which belong to the subset of courses. **4.** User clicks on schedule. **5.** System updates the course element instances attribute scheduleable lesson of those course element instances which were not selected but belong to the same program to false. **6.** Scheduler launches the scheduling of the defined program. **7.** Scheduler changes its status to running. **8.** User waits until the scheduling is finished by itself. **9.** Scheduler inserts the proposed schedule into the database. **10.** Scheduler changes its status to ready.
Alternative Paths	**8a.** User clicks on stop. **8b.** Scheduler changes its status to stopping. **9a.** Scheduler updates the proposed schedule in the database.
Open Questions	**Q:** Should the subset scheduling of courses be offered as a method of the scheduler?
Solved issues	-
Implementation Notes	-
Implementation Status	-

	Define a program
Actors	Program administrator, GUI, Scheduler, Database
Scope	GUI
Precondition	The system is running and the user is logged in with the rights of a program administrator.
Postcondition	The system is still running and the user is still logged in.
Postcondition on Success	The system finds a program fitting for the specified academic term and department.
Basic Course of Events	**1.** User opens the schedule panel. **2.** User selects an academic term from the available academic terms. **3.** User selects a department from the available departments. **4.** User clicks on start. **5.** System queries the database for a program with the given academic term and the given department. **6.** Database returns the required program.
Alternative Paths	**4a.** User clicks on resume.
Open Questions	-
Solved issues	-
Implementation Notes	-
Implementation Status	-