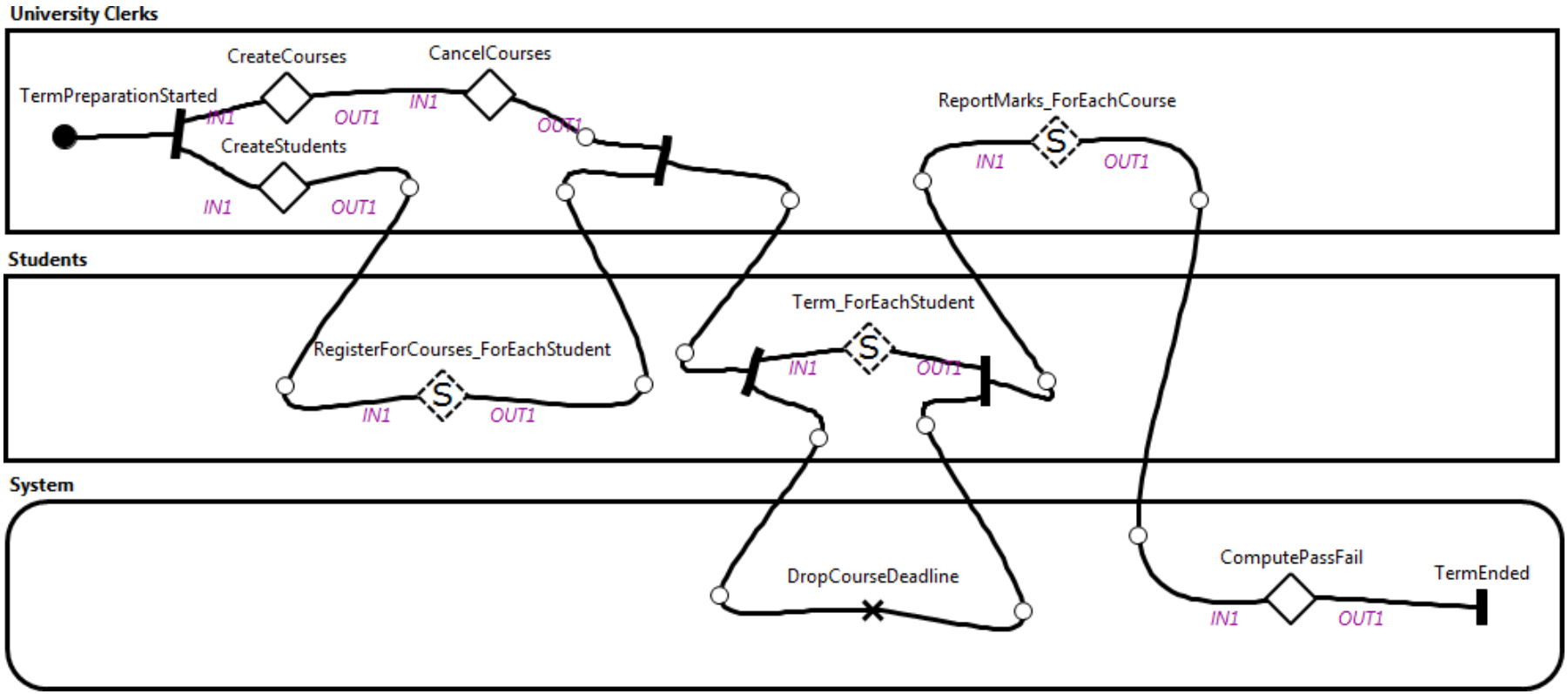
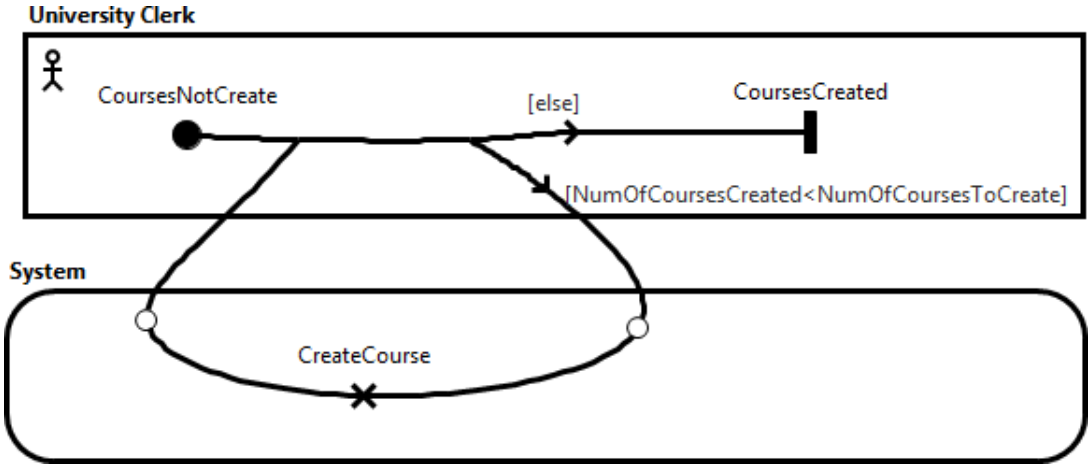
Table of Contents

Use Case Maps (UCMs) for the University System

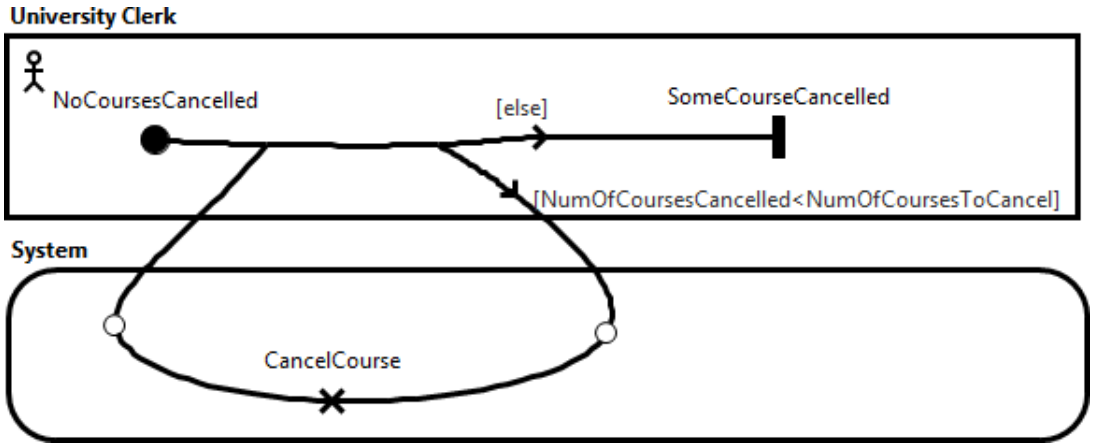
Main



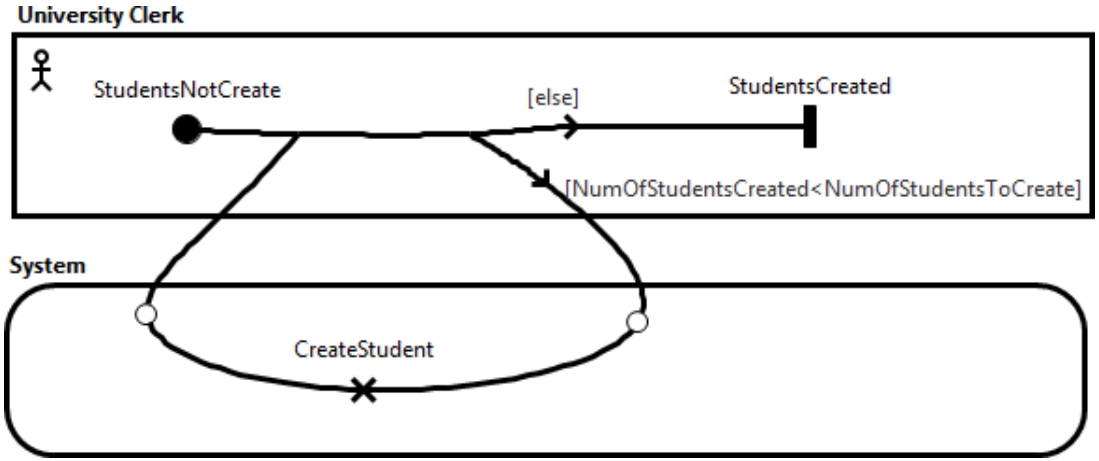
Create Courses



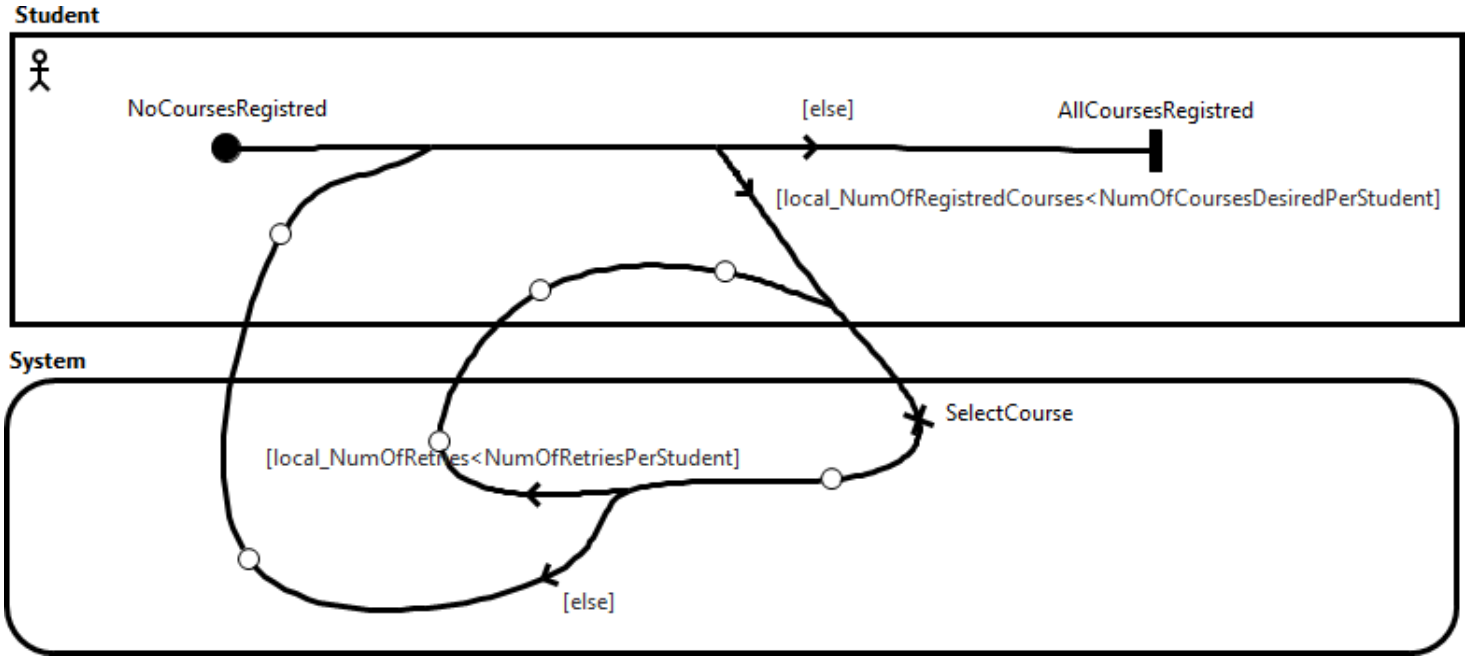
Cancel Courses



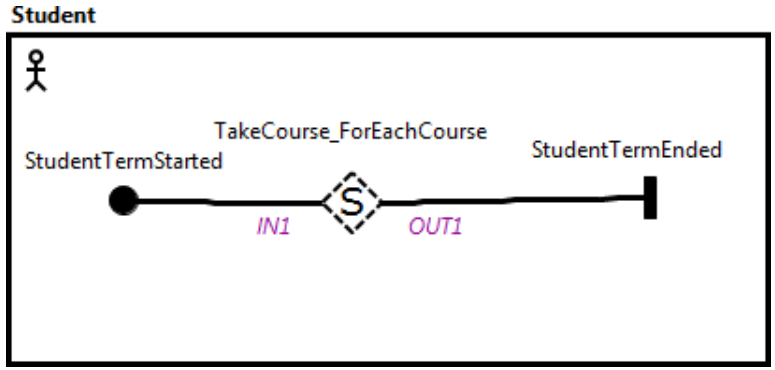
Create Students



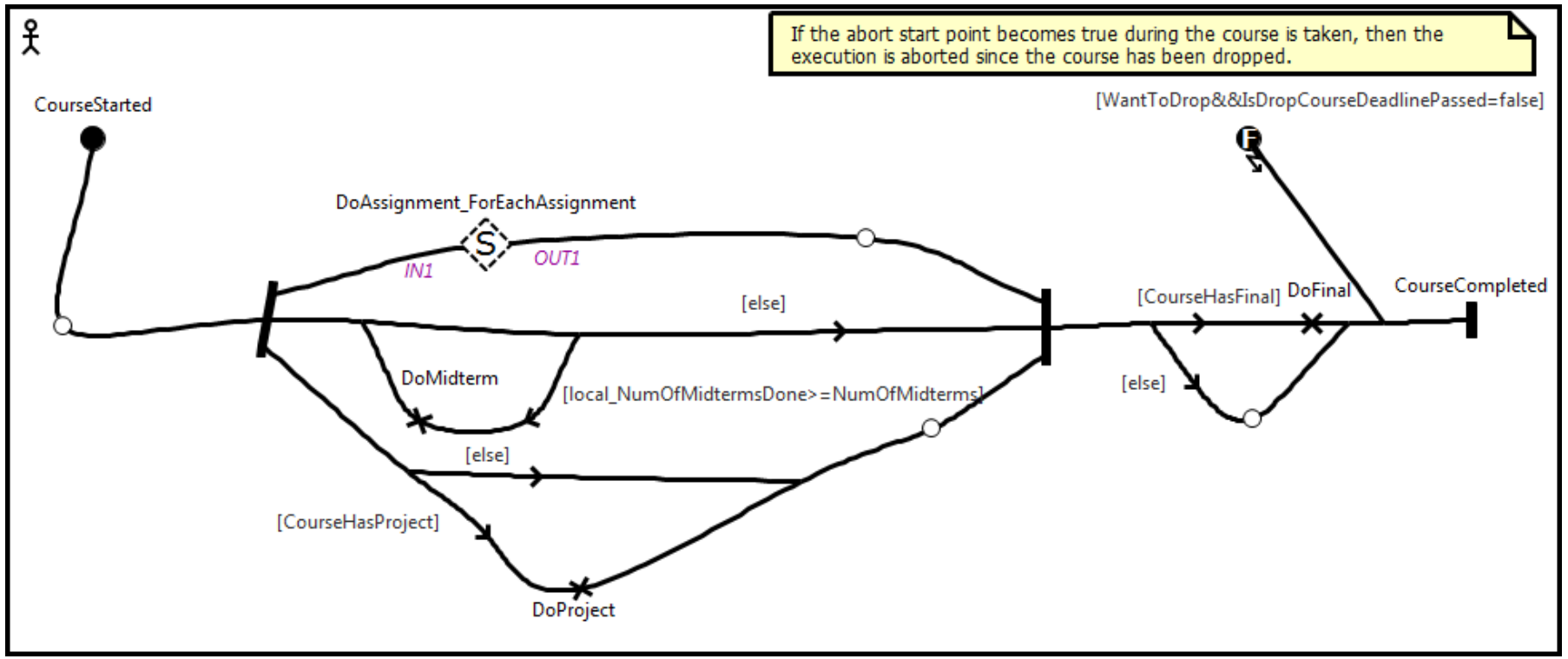
Register for Courses



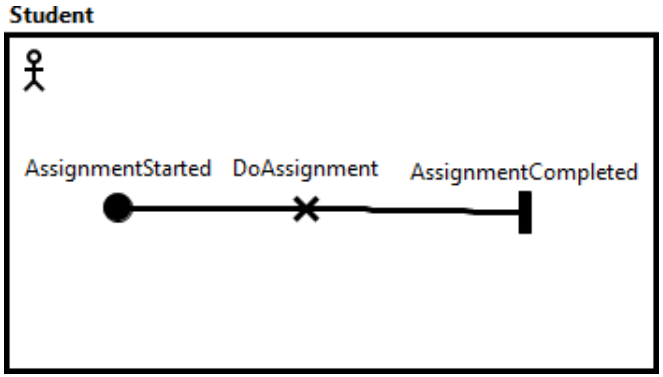
Term



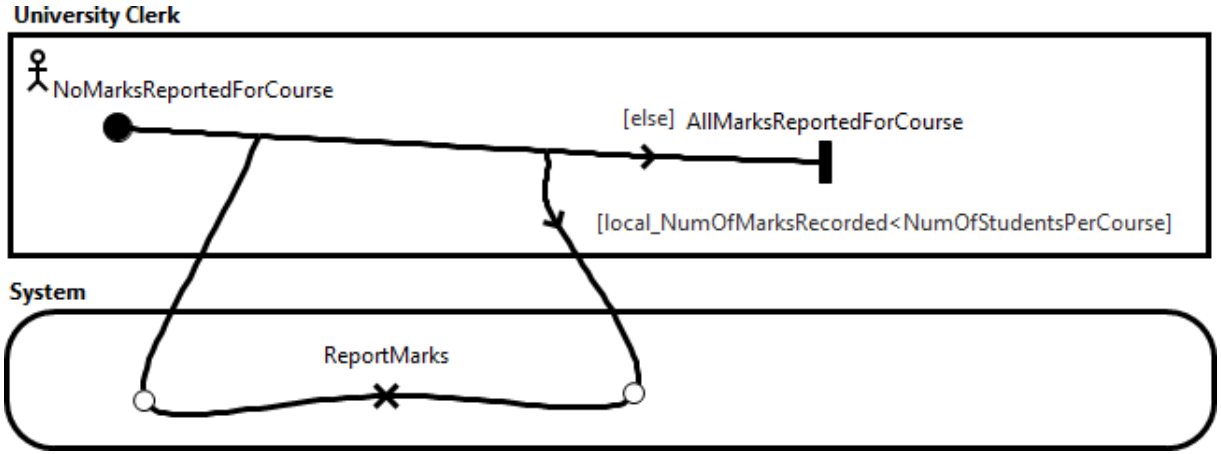
Take Course



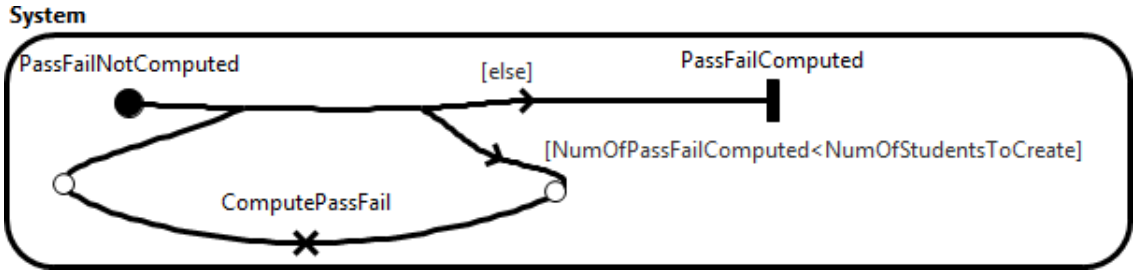
Do Assignment



Report Marks



Compute Pass/Fail



Test Paths for the University ACL Model

Note

Similarly to the approach presented in the SERP paper, the scope of the variables are limited to a scenario. That is, if the number of assignments is set to 2 for a scenario related to many courses, then all courses have two assignments. Moreover, as stated in the assignment "[We] need not worry about testing observabilities, responsibilities and invariants: the focus is exclusively on scenarios and path sensitization". For example, the SERP paper states that a PSV named numberOfFailingMarks is required for the scenario ReportMarks. Since the choice whether a mark passes or fails is defined in a responsibility, we ignored the operational variable named numberOfFailingMarks in our assignment even if it is used in the example presented in the SERP paper.

For School Interaction, we assumed that the referenced scenarios were contained by the relation. That is, a relation contains all test paths of the referenced scenarios and may contain new test paths that result in the interaction between the referenced scenarios. Our approach for selecting test paths was trying to find the minimal number of combinations of value for the operation variables that would lead to a maximum number of "significantly different" paths in the ACL scenarios/relations. We did not include any expected results in our analysis, since checking the expected result is the responsibility of the ACL scenarios, preconditions, postconditions and invariants.

In the SERP paper, the minimum concurrency is set to 0 when no instances are overlapping. If 0 means no overlapping, then what does 1 mean? To avoid this ambiguity, we set the minimum concurrency to 1 when there were no overlapping, to 2 when two instances overlapped and to 4 when 4 instances overlapped. Moreover, the concept of minimum concurrency could be applied to the independence symbol (||) from the interaction grammar and to the disjunction symbol (|) from the scenario grammar since they create parallelism. However, we assumed that these symbols means "any order" and not parallelism. Thus, we did not create any operational variables to control the minimum concurrency in such cases.

Finally, DC means don't care.

Course ReportMarks

Operational Variables

Variable Name	Motivation	Interesting Values
NumOfStudents	Loop coverage	Min=0, In-between=8, Max=256

Test Paths

Variant	Operational Variables
	NumOfStudents
1	0
2	8
3	256

Description

Omitted since the test paths are self-descriptive.

Student_RegisterForCourses

Operational Variables

Variable Name	Motivation	Interesting Values
IsFullTime	Branch coverage	True|False
NumOfCurrentCourses	Loop coverage	Min=0, In-between=1, MaxFullTime=4, MaxPartTime=2
NumOfAttemptsToRegisterBeforeCourseIsNotFull	Loop coverage	Min=0, In-between=1, Max=4

Test Paths

	Operational Variables		
Variant	IsFullTime	NumOfCurrent Courses	NumOfAttemptsToRegisterBeforeCourseIsNotFull
1	DC	0	DC
2	DC	1	0
3	True	4	1
4	False	2	4

Description

Variant	Description
1	No Courses
2	One courses registered without retrying
3	A full-time student registers to 4 courses, each of them registered successfully after 1 retry

4	A part-time student registers to 2 courses, each of them registered successfully after 4 retry

Student_TakeCourses

Operational Variables

Variable Name	Motivation	Interesting Values
NumOfCurrentCourses	Parallel instance coverage	Min=0, In-between=1, Max=4
NumOfConcurentCurrentCourses	Parallel concurrency coverage	Min=1, In-between=2, Max=4
NumOfCoursesToDrop	Branch coverage within parallel	Min=0, In-between=1, Max=4
NumOfCurrentAssignments	Parallel instance coverage	Min=0, In-between=1, Max=4
NumOfConcurentCurrentAssignments	Parallel concurrency coverage	Min=1, Max=2
NumOfMidterms	Loop coverage	Min=0, In-between=1, Max=2
CourseHasProject	Branch coverage	True|False

Test Paths

Variant	Operational Variables						
	NumOfCurrent Courses	NumOfConcurent CurrentCourses	NumOfCourses ToDrop	NumOfCurrent Assignments	NumOfConcurent CurrentAssignments	NumOfMidterms	CourseHasProject
1	0	DC	DC	DC	DC	DC	DC
2	1	DC	0	0	DC	0	False
3	4	1	0	1	DC	1	True
4	4	2	0	4	1	2	DC
5	4	4	1	4	2	DC	DC
6	4	DC	4	DC	DC	DC	DC

Description

Variant	Description
1	No Courses
2	One courses completed without any assignments, midterms or project
3	Four courses completed sequentially with 1 assignment, 1 midterm and a project
4	Four courses completed where only 2 courses can overlap at the same time, each courses has 4 assignments completed sequentially and 2 midterms
5	Four courses that can overlap where one course is dropped, the remaining courses has four assignments where only 2 assignments can overlap at the same time
6	Four courses dropped

University - CreateCourses

Operational Variables

Variable Name	Motivation	Interesting Values
NumOfCoursesToCreate	Loop coverage	Min=0, In-between=8, Max=256

Test Paths

Variant	Operational Variables
	NumOfCoursesToCreate
1	0
2	8
3	256

Description

Omitted since the test paths are self-descriptive.

University - CreateStudents

Operational Variables

Variable Name	Motivation	Interesting Values
NumOfStudentsToCreate	Loop coverage	Min=1, In-between=8, Max=65536

Test Paths

Variant	Operational Variables
	NumOfStudentsToCreate
1	1
2	8
3	65536

Description

Omitted since the test paths are self-descriptive.

University - Terms

Operational Variables

Variable Name	Motivation	Interesting Values
NumOfTermsToComplete	Loop coverage	Min=1, In-between=8, Max=256
NumOfCoursesToCreate	Loop coverage	Min=0, In-between=8, Max=256

Test Paths

Variant	Operational Variables	
	NumOfTermsToComplete	NumOfCoursesToCreate
1	1	256
2	8	8
3	256	1

Description

Omitted since the test paths are self-descriptive.

School - Creation

Operational Variables

Variable Name	Motivation	Interesting Values
NumOfStudentsToCreate	Loop coverage	Min=1, In-between=8, Max=65536
NumOfCoursesToCreate	Loop coverage	Min=0, In-between=8, Max=256

Test Paths

Variant	Operational Variables	
	NumOfStudentsToCreate	NumOfCoursesToCreate
1	1	256
2	8	8
3	65536	1

Description

Variant	Description
1	CreateStudents-Variant1, CreateCourses-Variant3
2	CreateStudents-Variant2, CreateCourses-Variant2
3	CreateStudents-Variant3, CreateCourses-Variant1

School - Cancelling

Operational Variables

Variable Name	Motivation	Interesting Values
IsCourseCancelled	Branch coverage	True|False

Test Paths

Variant	Operational Variables
	IsCourseCancelled
1	True
2	False

Description

Omitted since the test paths are self-descriptive.

School - Students

Operational Variables

Variable Name	Motivation	Interesting Values
NumOfTermsToComplete	Loop coverage	Min=0, In-between=1, Max=8
IsFullTime	Branch coverage	True|False
NumOfRegistredCourses	Loop coverage	Min=0, MaxFullTime=4, MaxPartTime=2
NumOfAttemptsToRegisterBeforeCourseIsNotFull	Loop coverage	Min=0, Max=4
NumOfCurrentCourses	Parallel instance coverage	Min=0, In-between=1, Max=4
NumOfConcurentCurrentCourses	Parallel concurrency coverage	Min=1, In-between=2, Max=4
NumOfCoursesToDrop	Branch coverage within parallel	Min=0, In-between=1, Max=4
NumOfCurrentAssignments	Parallel instance coverage	Min=0, In-between=1, Max=4
NumOfConcurentCurrentAssignments	Parallel concurrency coverage	Min=1, Max=2
NumOfMidterms	Loop coverage	Min=0, In-between=1, Max=2
CourseHasProject	Branch coverage	True|False

Test Paths

Variant	Operational Variables									
	NumOf TermsTo Complete	IsFullTime	NumOf Attempts ToRegister Before Course IsNotFull	NumOf Current Courses	NumOf Concurent CurrentCourses	NumOf Courses ToDrop	NumOf Current Assignments	NumOf Concurent CurrentAssignments	NumOf Midterms	Course HasProject
1	1	DC	DC	0	DC	DC	DC	DC	DC	DC
2	8	DC	0	1	DC	0	0	DC	0	False
3	DC	True	1	4	1	0	1	DC	1	True
4	DC	DC	DC	4	2	0	4	1	2	DC
5	DC	DC	DC	4	4	1	4	2	DC	DC
6	DC	DC	DC	4	DC	4	DC	DC	DC	DC
7	DC	False	4	2	DC	DC	DC	DC	DC	DC
8	0	DC	DC	DC	DC	DC	DC	DC	DC	DC

Test Path Description

Variant	Description
1	RegisterForCourses-Variant1, TakeCourses-Variant1
2	RegisterForCourses-Variant2, TakeCourses-Variant2
3	RegisterForCourses-Variant3, TakeCourses-Variant3
4	TakeCourses-Variant4
5	TakeCourses-Variant5
6	TakeCourses-Variant6
7	RegisterForCourses-Variant4
8	No Terms

Discussion of Completeness and Accuracy of the ACL Model

The only available description of the university system was the university example available on the ACL/VF website. Therefore, the use case maps are highly influenced by the ACL model at hand. For example, the ACL model does not state that students can register to another course if one of the course that they register for is cancelled. Since no other description of the university system is available, the scenario is ignored in the UCMs. Therefore, the use case maps are of a higher level of abstraction then the ACL model, but essentially they are more or less as complete and accurate than the ACL model.

The SERP paper describes an approach to create operational variables from the ACL model. However, it may be difficult to understand the impact of these operational variables on the ACL scenarios/relations. Indeed, the interaction between ACL scenarios/relations is defined by a mechanism of events (fire/observe) that makes the interaction between ACL scenarios/relations difficult to follow. For example, the relation Students observe the event CourseComplete but the author of the ACL model forgot that the event was never fired. On the other hand, jUCMNav allows for tracing the relationship between the UCMs by starting at the Main UCM which provides a high-level view of the system and drilling down to all others UCMs. Also, the operational variables are defined explicitly in the UCMs. Setting values for operational variables in order to define test paths is fully supported by the UCMs. However, from our understanding, the UCMs only allow for defining global variables where the ACL models allows for defining variables at the contract level. This is a major limitation of the UCMs since it prevents from executing several UCMs that access the same global variables in parallel.

System-level testing approaches such as TOTEM can be applied to both UCMs and ACL model in order to obtain test paths. The main downside of the TOTEM approach is that the number of combinations to test grows exponentially as the number of operational variables grows. However, the TOTEM approach can be automated and yield a comprehensive set of test paths. Regardless if the UCM model or the ACL model is used, the code required to execute the test paths generated by TOTEM against the IUT will have to be written manually. Indeed, the UCM model is not connected at all to the IUT and the ACL model does not contains the data required in order to generate test drivers. However, the ACL model has a major advantage: it can be use as an oracle to monitor the proper execution of the scenarios/relations.

Both ACL models and UCMs can be use to formalize paths on use cases describe in natural language. Once the paths on use cases has been formally defined, SCENT requires us to create state machine for each scenario manually. Therefore, both the ACL models and the UCMs can be used in conjunction with the SCENT approach.