User Requirements Notation Models

for

RaiderNAV

Table of Contents

1. Goal Requirements Language

The GRL models shown in this document demonstrate the high-level goals, as well as the rationales associated with achieving those goals, for the two main actors connected to the *RaiderNAV* project: Team Getana and Texas Tech students. Diagrams illustrating this information are provided both individually by actor, as well as in an integrated view.

1.1 Strategies

Four strategies were evaluated in an effort to determine the best possible framework to satisfy the goals of both Team Getana and Texas Tech students. Team Getana's primary goals are to get an 'A' in the Software Engineering II course and to ensure the *RaiderNAV* app is popular. The Texas Tech students' top goals are to find their class locations more easily while still maintaining a high level of privacy regarding their personal information.

Analysis of Team Getana's goals reveals them to be mutually exclusive to a certain degree. For example, devoting more time to *RaiderNAV* development is beneficial to the app's popularity, but detrimental to the group's grades if the resources necessary are diverted from other important class activities. The case for Texas Tech students is somewhat similar. Utilizing *RaiderNAV* makes it much easier to locate class locations but does so at the cost of certain private information that must be provided to Google Maps.

Since this is the case, the four strategies evaluated each represent a unique combination of the goals predominantly pursued by the actors.

1.1.1 Combined Strategy #1: Team Getana Deemphasizes Project and Students Avoid RaiderNAV

Strategy one represents the case where Team Getana prioritizes tests and homework over the *RaiderNAV* project and a Texas Tech student elects not to utilize *RaiderNAV* in order to maintain information privacy. This results in a high satisfaction value for Team Getana's goal of achieving an 'A'. However, it also leads to a poorly executed app which is not popular. Meanwhile, the choice to not use the *RaiderNAV* app effectively preserves the student's privacy but does not improve their ability to locate classes more easily.

Team Getana model for Strategy #1 is depicted in figure 1.1.1.1

Texas Tech Student model for Strategy #1 is depicted in figure 1.1.1.2

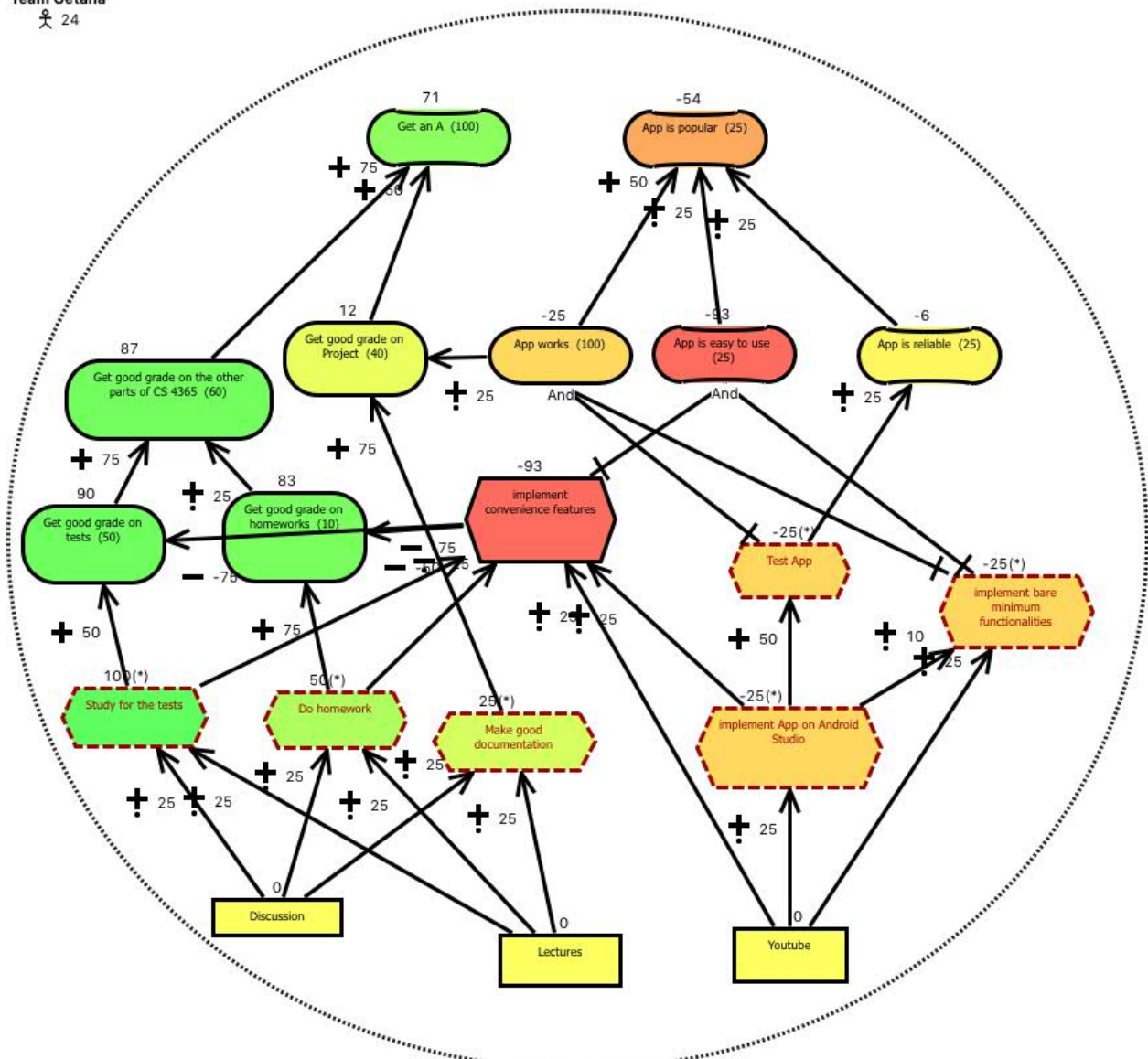Combined model for Team Getana & Texas Tech Students for Strategy #1 is depicted in figure 1.1.1.3

Figure 1.1.1.1 Team Getana GRL model under strategy 1
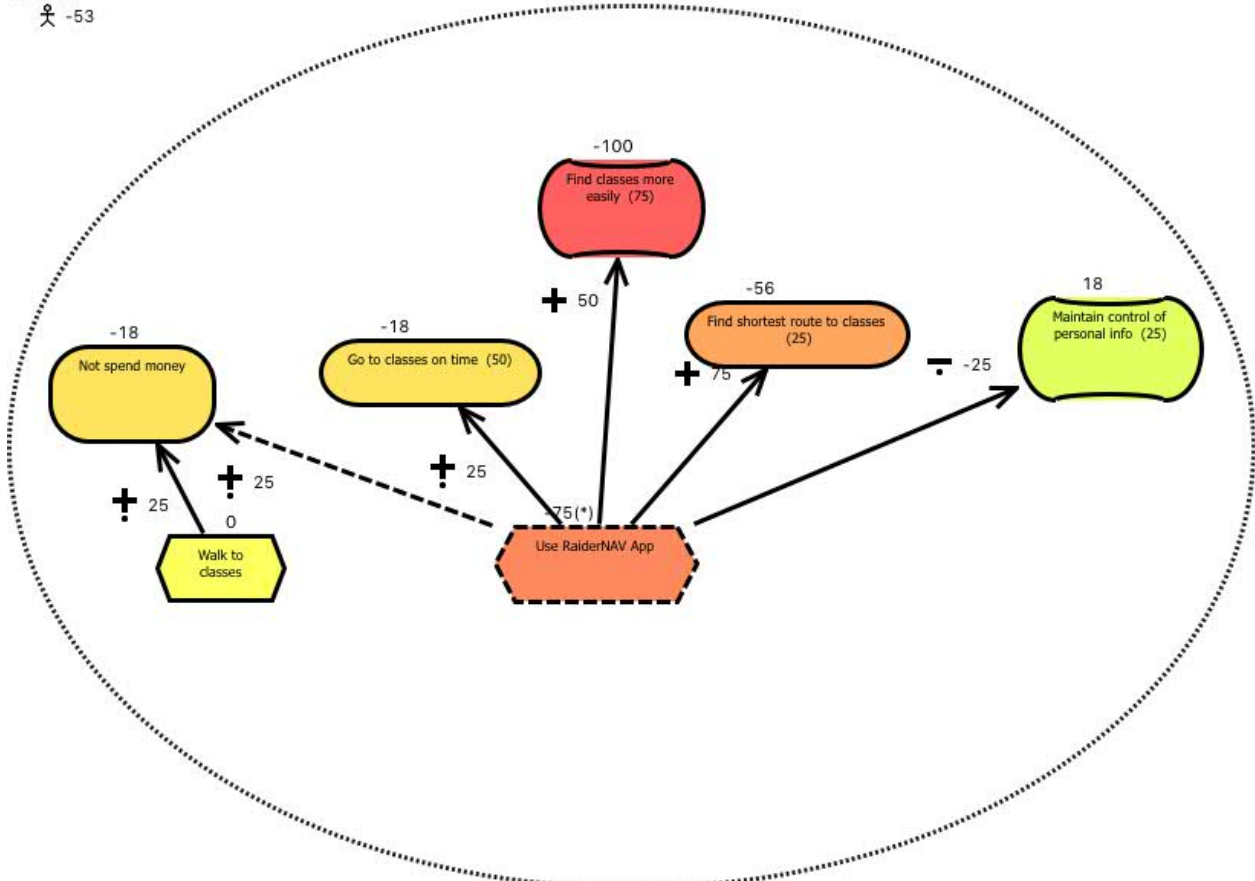
Figure 1.1.1.2 Student GRL model under strategy 1

Team Getana
👤 24

Student
👤 -53

Find classes more easily (-100)

Go to classes on time (50) (-18)

Find shortest route to classes (25) (-56)

Maintain control of personal info (100)

Use RaiderNAV App

Get an A (100) (71)

App is popular (25) (-54)

Get good grade on the other parts of CS 4365 (60) (87)

App works (100) (-25)

App is easy to use (25) (93)

Study for the tests 100(*)

implement convenience features (-93)

Implement App on Android Studio -25(*)

Do homework 50(*)

And

Figure 1.1.1.3 Combined GRL model under strategy 1

7

1.1.2 Combined Strategy #2: Team Getana Emphasizes Project and Students Avoid RaiderNAV

Strategy #2, unlike above, reflects a scenario wherein Team Getana focuses their efforts primarily on the *RaiderNAV* project, even to the point of neglecting other classwork while a student still chooses not to utilize *RaiderNAV*. The increased effort put toward the app is reflected in a more well-designed final product that is more popular among users in general, but adversely affects Team Getana's overall grade in the Software Engineering course. The popularity of the app also makes it slightly easier for an individual student to locate a course even if they choose not to utilize the app, because other students are more readily able to assist, but not meaningfully so in other situations.

Team Getana model for Strategy #2 is depicted in figure 1.1.2.1

Texas Tech Student model for Strategy #2 is depicted in figure 1.1.2.2

Combined model for Team Getana & Texas Tech Students for Strategy #2 is depicted in figure 1.1.2.3

-15
Get an A
(100)

35
App is
popular (25)

+ 75
+ 50

+ 50

+ 25

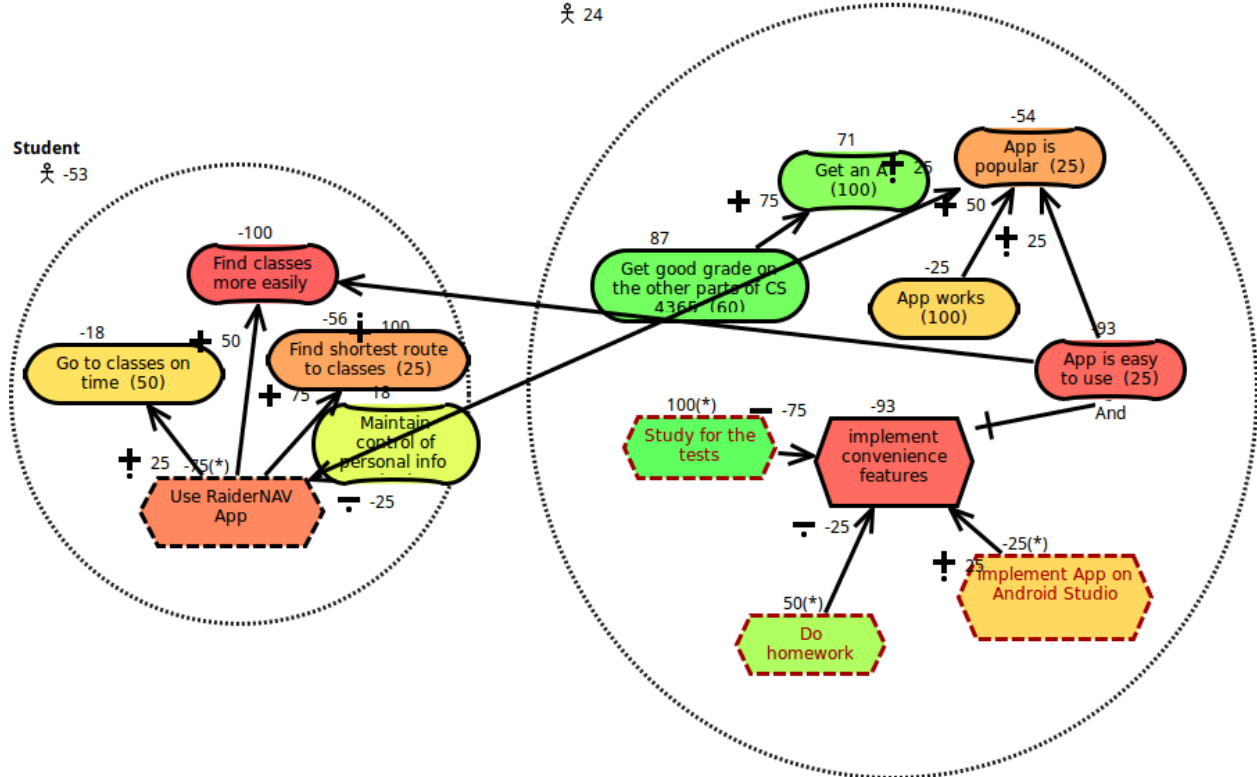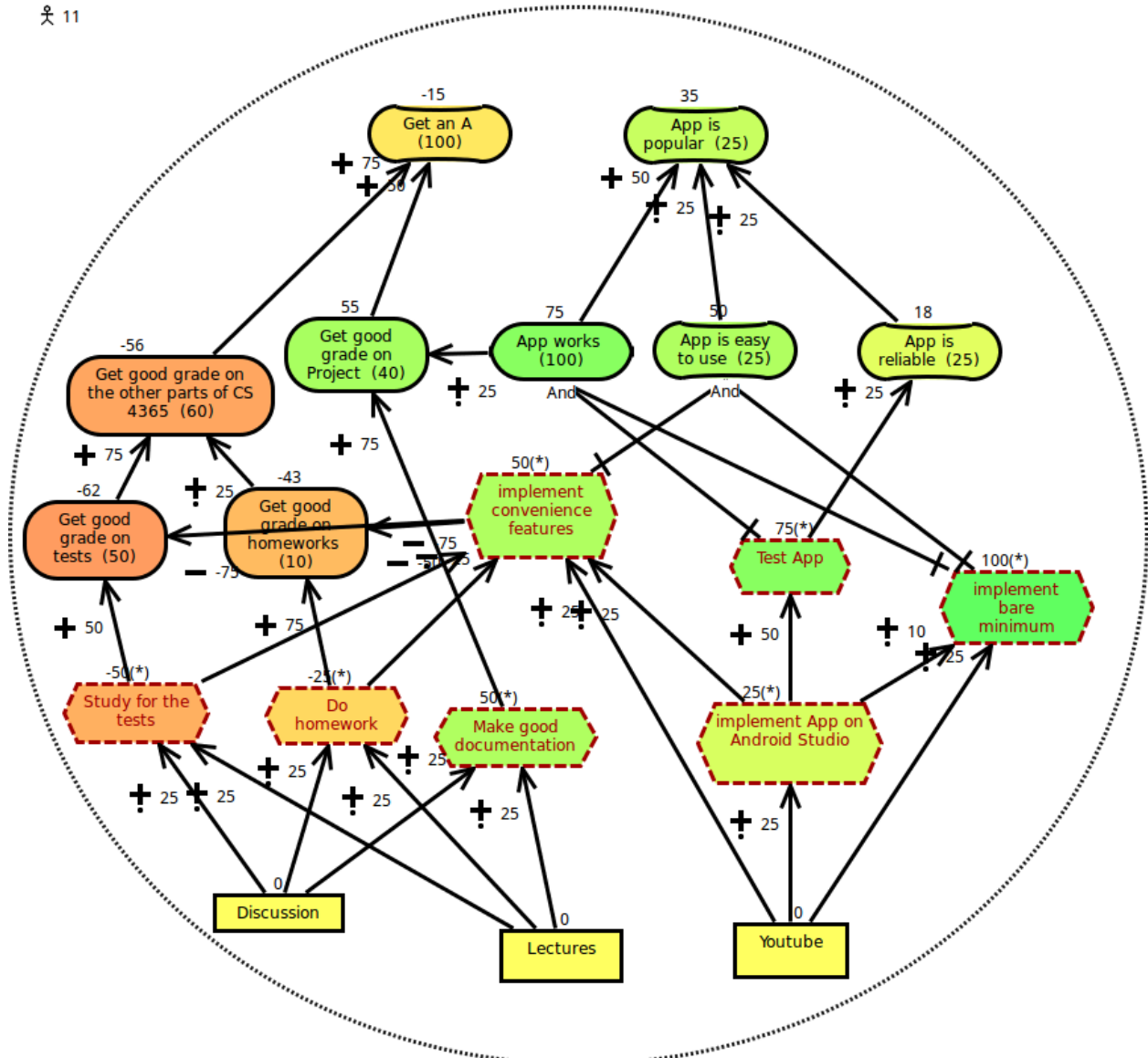+ 25

-56
Get good grade on
the other parts of CS
4365 (60)

55
Get good
grade on
Project (40)

75
App works
(100)

50
App is easy
to use (25)

18
App is
reliable (25)

+ 75

+ 25

And

And

+ 25

+ 25

-62
Get good
grade on
tests (50)

-43
Get good
grade on
homeworks
(10)

50(*)
implement
convenience
features

75(*)
Test App

100(*)
implement
bare
minimum

+ 75

− 75

− 75

− 75

+ 50

+ 10

+ 25

+ 50

+ 75

+ 25 + 25

+ 50

+ 25

-50(*)
Study for the
tests

-25(*)
Do
homework

50(*)
Make good
documentation

25(*)
implement App on
Android Studio

+ 25 + 25

+ 25

+ 25

+ 25

+ 25

+ 25

0

Discussion

0

Lectures

0

Youtube

Figure 1.1.2.1 Team Getana GRL model under strategy 2

9

Student

👤 -5

13
Find classes more easily (75)

-18
Not spend money

-18
Go to classes on time (50)

-56
Find shortest route to classes (25)

18
Maintain control of personal info (25)

➕ 50

➕ 75

➖ -25

➕ 25

➕ 25

➕ 25

0

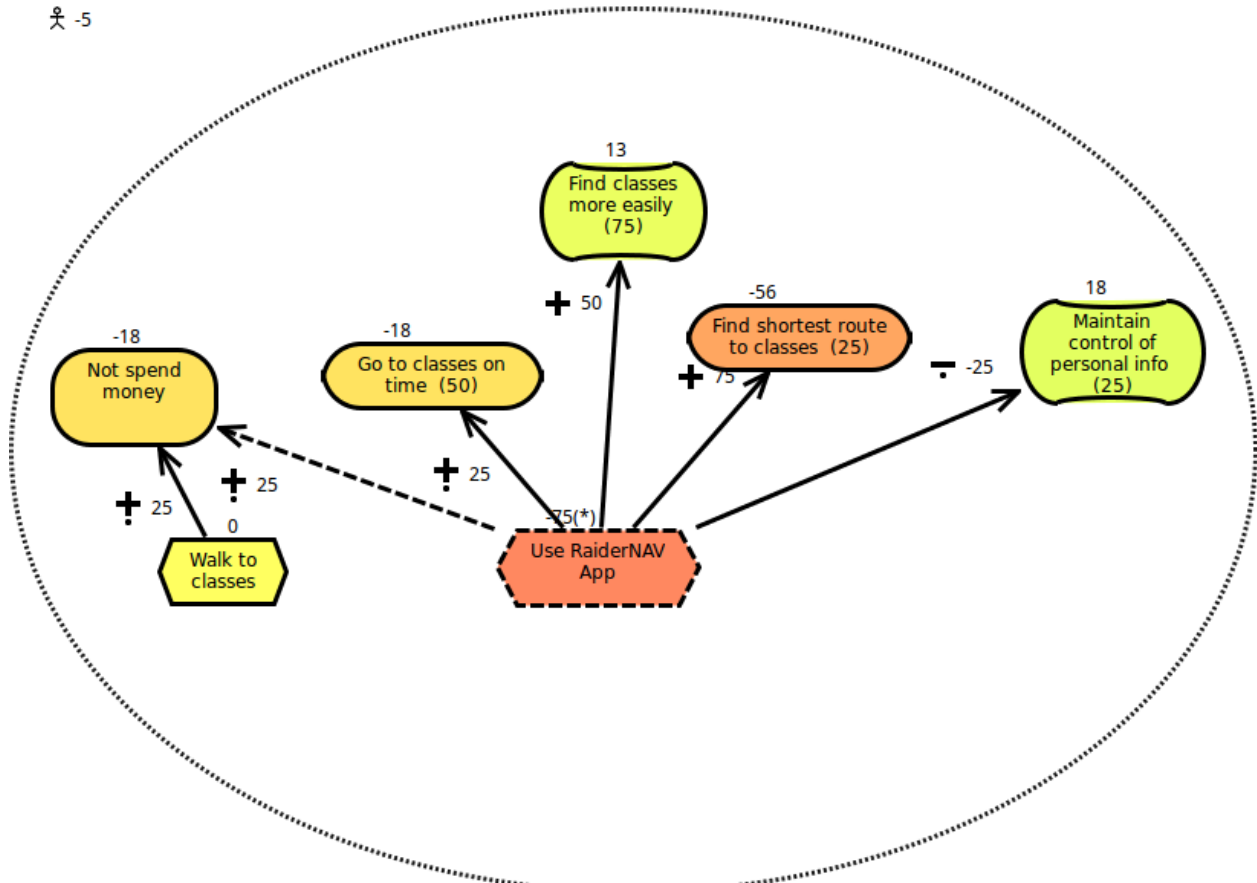**Walk to classes**

-75(*)
Use RaiderNAV App

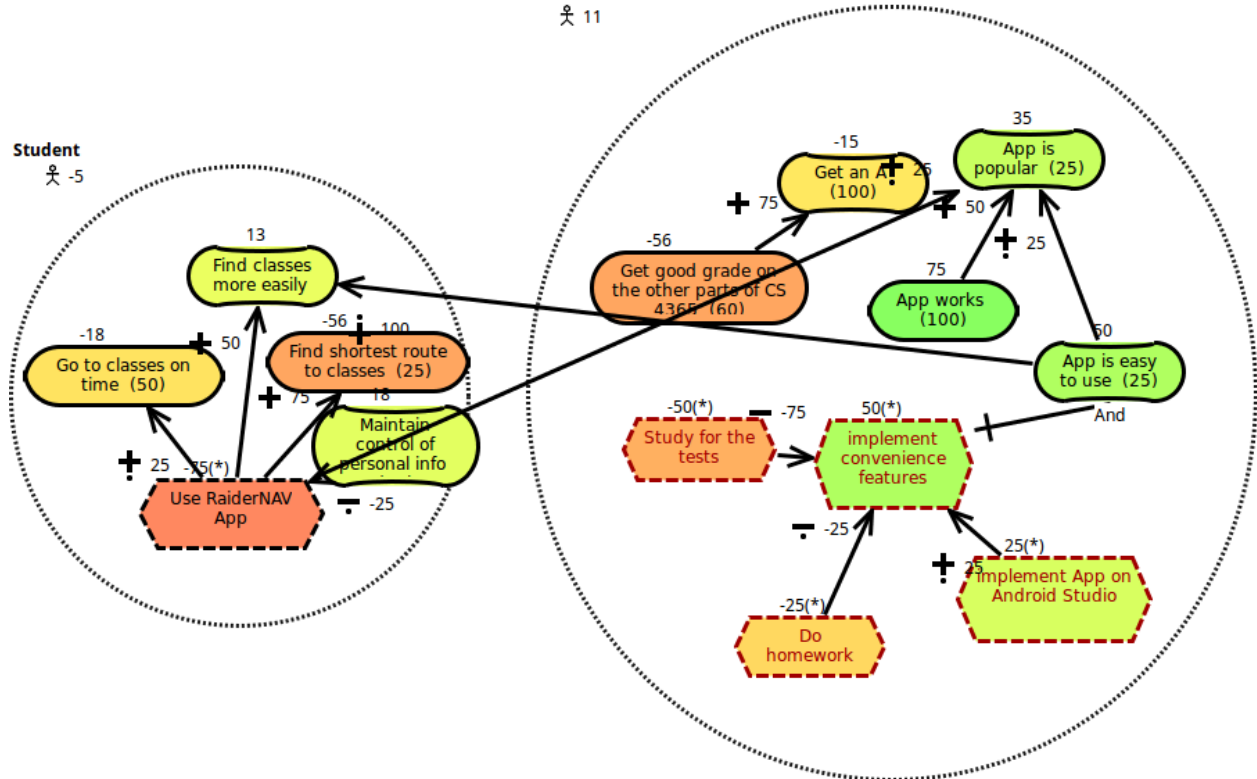Figure 1.1.2.2 Student GRL model under strategy 2

Figure 1.1.2.3 Combined GRL model under strategy 2

1.1.3 Combined Strategy #3: Team Getana Deemphasizes Project and Students Use RaiderNAV

Strategy #3 represents the inverse of strategy #2. In this situation, Team Getana again focuses on homework and tests to improve their grades for Software Engineering II, but the student in question does use *RaiderNAV* to locate his/her classes. Because Team Getana dedicates more of their resources to other assignments that constitute a larger portion of their overall grade, their chance of getting an 'A' significantly improves. Unfortunately, this also results in a decrease in the quality of the *RaiderNAV* app, which is not as intuitive for the user. Consequently, the student's ability to more easily find their classes is still improved, although not as much as if the app was better designed.

Team Getana model for Strategy #3 is depicted in figure 1.1.3.1

Texas Tech Student model for Strategy #3 is depicted in figure 1.1.3.2

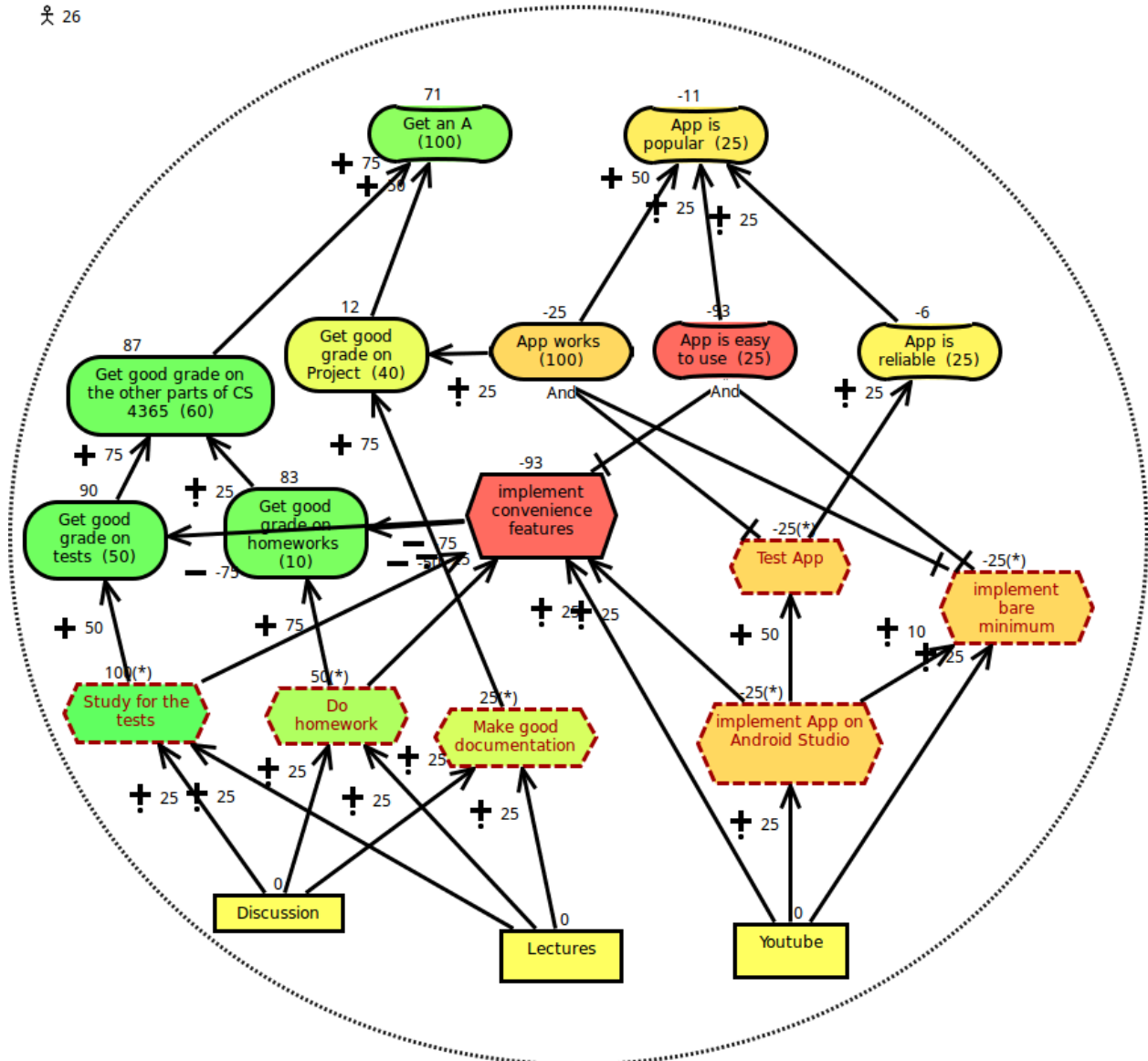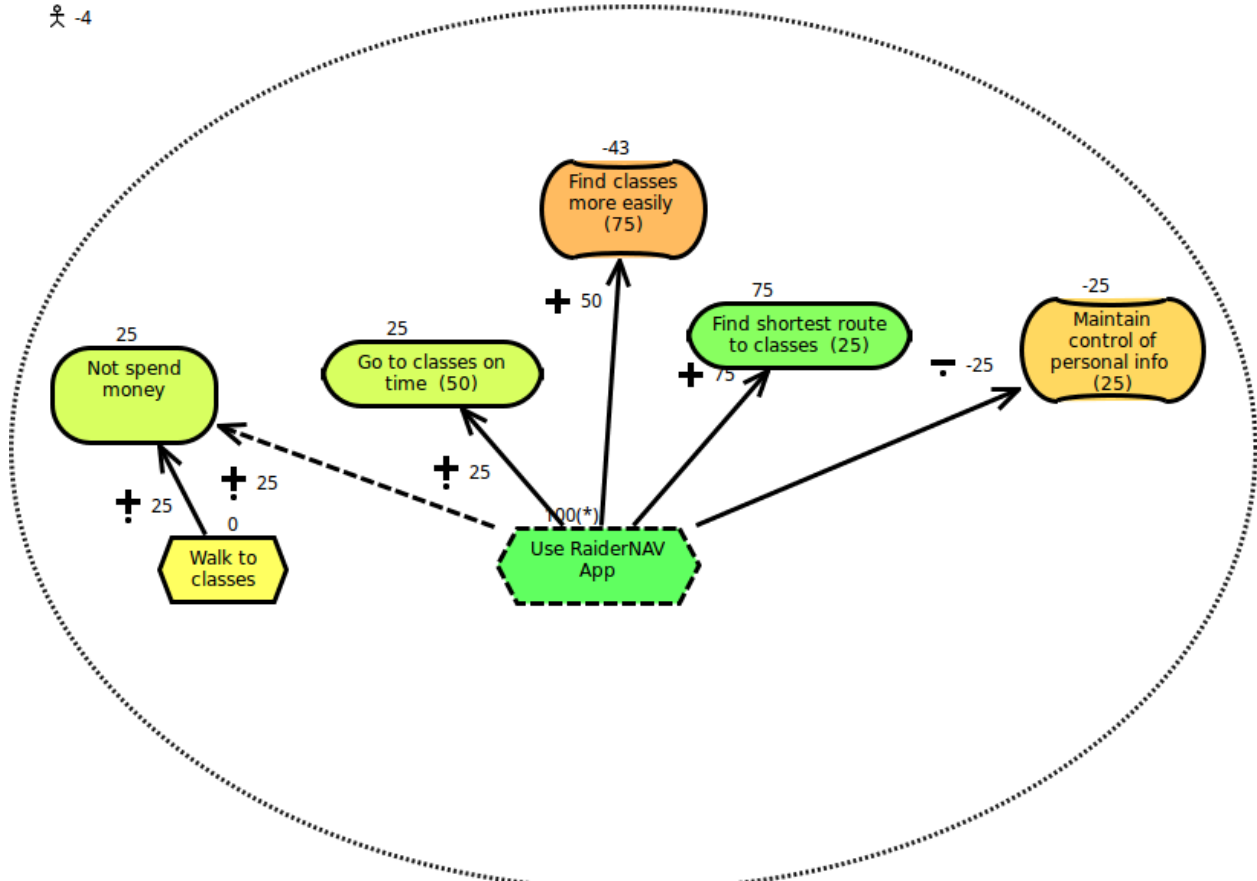Combined model for Team Getana & Texas Tech Students for Strategy #3 is depicted in figure 1.1.3.3

Team Getana

👤 26

71
Get an A (100)

-11
App is popular (25)

+ 75
+ 50

+ 50

+ 25
+ 25

12
Get good grade on Project (40)

87
Get good grade on the other parts of CS 4365 (60)

-25
App works (100)

-93
App is easy to use (25)

-6
App is reliable (25)

And

And

+ 25

+ 25

+ 75

-93
implement convenience features

90
Get good grade on tests (50)

83
Get good grade on homeworks (10)

+ 75

+ 25

- 75

- 75

- 75

-25(*)
Test App

-25(*)
implement bare minimum

+ 50

- 50

+ 25

+ 25

+ 25

+ 25

+ 10

+ 25

100(*)
Study for the tests

50(*)
Do homework

25(*)
Make good documentation

+ 75

+ 50

-25(*)
implement App on Android Studio

+ 25

+ 25

+ 25

+ 25

+ 25

+ 25

0
Discussion

0
Lectures

0
Youtube

Figure 1.1.3.1 Team Getana GRL model under strategy 3

13

Student
⚇ -4

-43
Find classes
more easily
(75)

+ 50

25
Not spend
money

25
Go to classes on
time (50)

75
Find shortest route
to classes (25)

-25
Maintain
control of
personal info
(25)

+ 75

÷ -25

+ 25

+ 25

00(*)

0

Walk to
classes

Use RaiderNAV
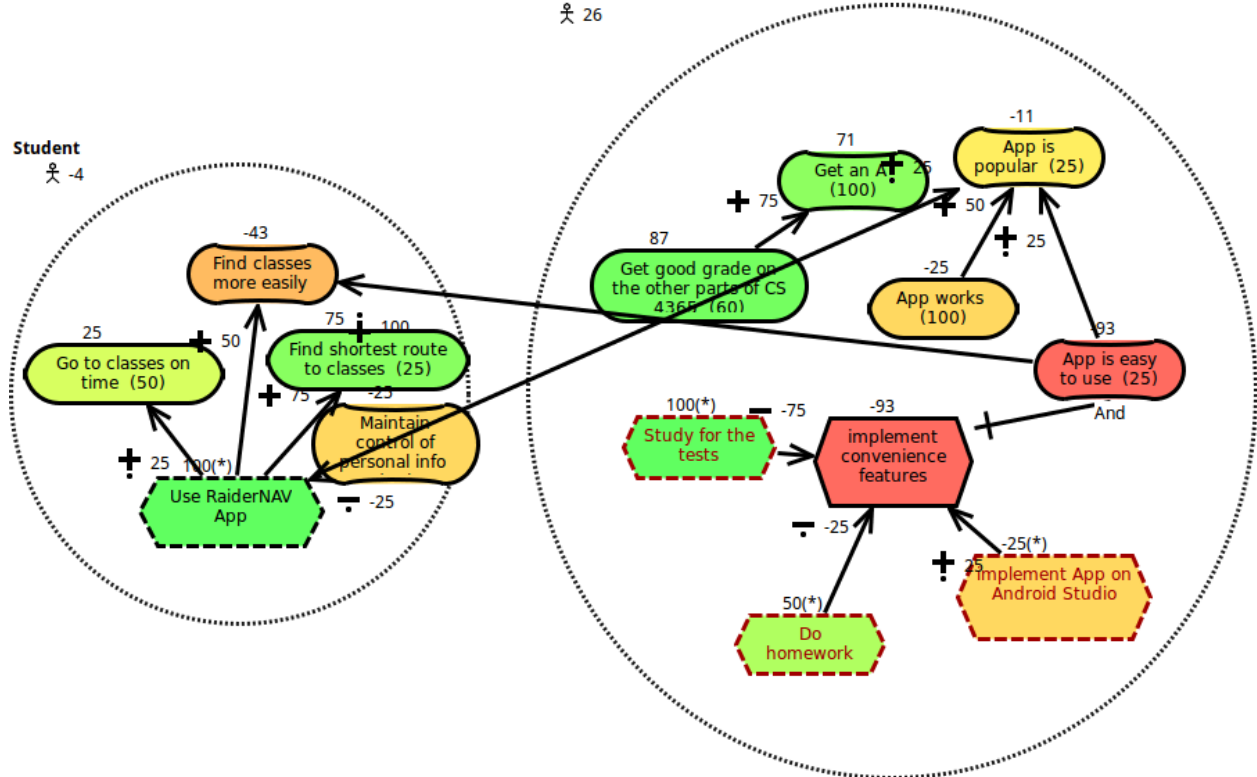App

Figure 1.1.3.2 Student GRL model under strategy 3

Figure 1.1.3.3 Combined GRL model under strategy 3

1.1.4 Combined Strategy #4: Team Getana Emphasizes Project and Students Use RaiderNAV

The final strategy evaluated, strategy #4, explores the effect when Team Getana prioritizes the development of the *RaiderNAV* app and a student utilizes the resulting product. As expected the result is a well-produced app that is both easy to use and popular, but at the cost of Team Getana's overall grade in the Software Engineering II course.

Team Getana model for Strategy #4 is depicted in figure 1.1.4.1

Texas Tech Student model for Strategy #4 is depicted in figure 1.1.4.2

Combined model for Team Getana & Texas Tech Students for Strategy #4 is depicted in figure 1.1.4.3
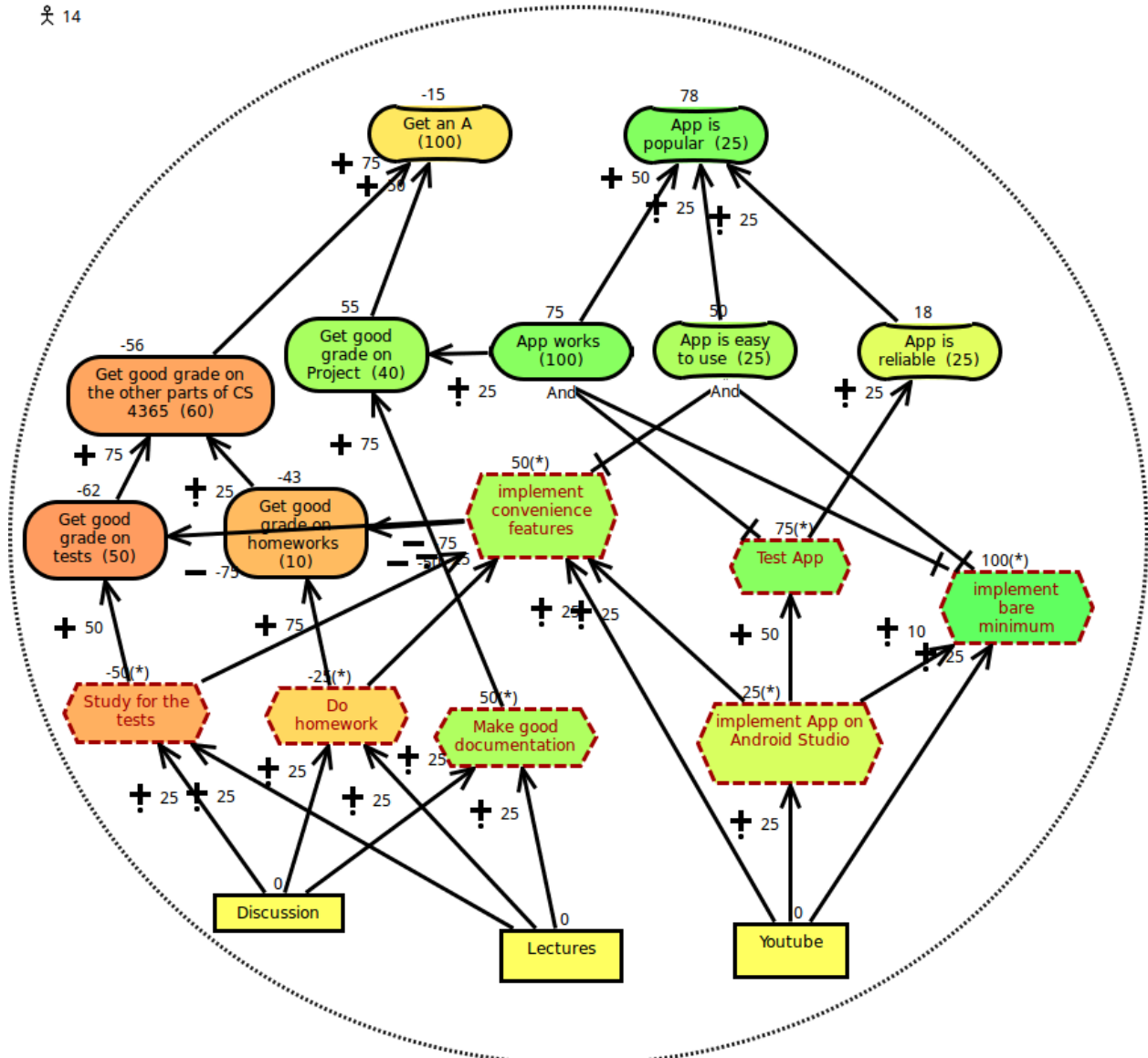
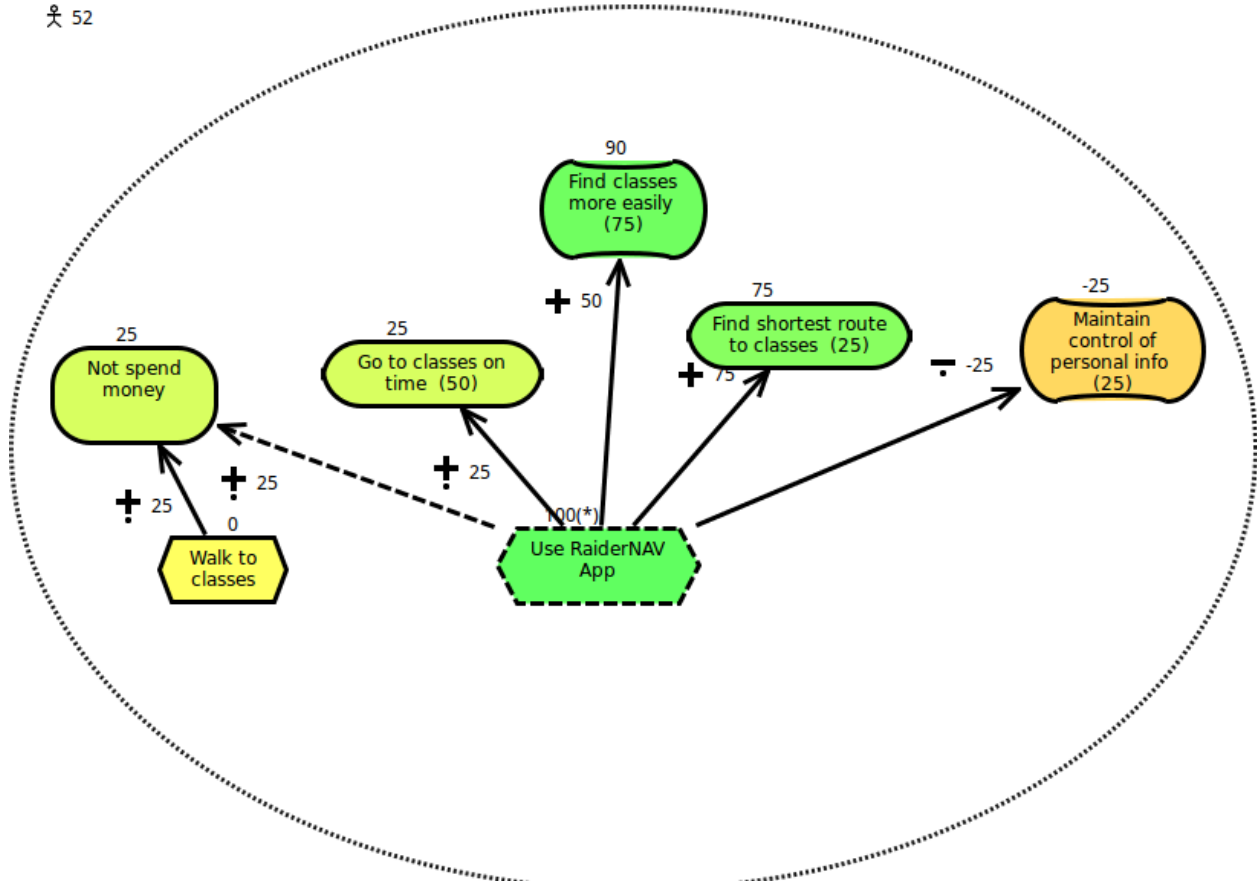Figure 1.1.4.1 Team Getana GRL model under strategy 4

Student
人 52

90
Find classes
more easily
(75)

25
Not spend
money

25
Go to classes on
time (50)

75
Find shortest route
to classes (25)

-25
Maintain
control of
personal info
(25)

+ 50

+ 25

+ 25

+ 25

0

+ 75

- -25

100(*)

Walk to
classes

Use RaiderNAV
App

Figure 1.1.4.2 Student GRL model under strategy 4

18
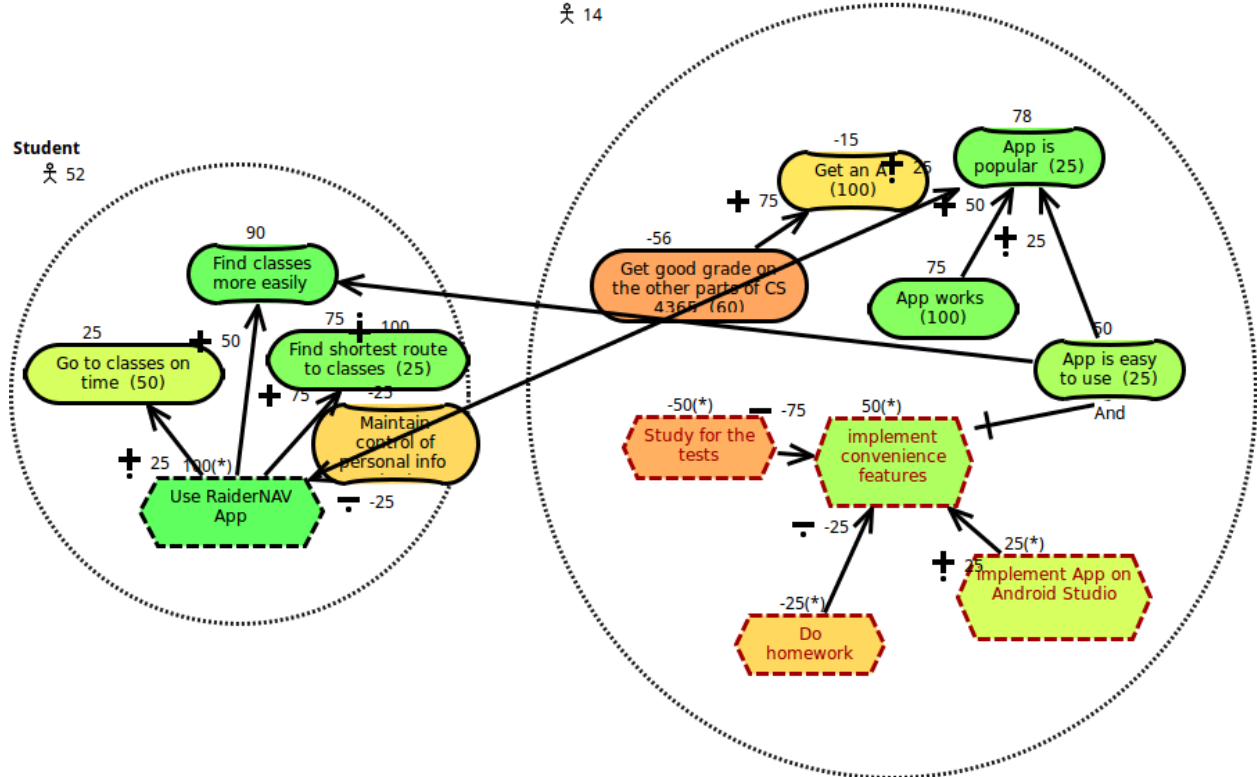
Figure 1.1.4.3 Combined GRL model under strategy 4

1.2 Analysis of Combined Strategies

Any analysis of GRL strategies must pay careful consideration to the concerns of each stakeholder or actor. The goals of each party may vary significantly and are often in fact at odds with one another. As such it is beneficial to address each strategy with regard to the actors involved, as well in a cumulative manner.

First, we examine the case of an individual student. The two high-level goals established earlier for a student are to find their class locations more easily and to maintain the privacy of their data. However, these goals are not weighted evenly. In fact, the ability to find class locations easily is shown to be far more important than information privacy. As such, we can expect the strategies that satisfy this goal to produce much higher satisfaction ratings for the student actor. This proves to be the case, and the highest overall satisfaction for the student actor is achieved from strategy 4. Here the student utilizes the *RaiderNAV* app to improve their navigation skills around campus, and the app itself is well-designed as a result of Team Getana's heightened efforts towards its development. Conversely, the student's satisfaction rating is lowest in strategy 1 when he/she does not utilize the *RaiderNAV* app and Team Getana focuses on other coursework.

The "best" strategy for Team Getana, without regard for Texas Tech students, is markedly different. Again, we see that while Team Getana had two primary high-level goals, to make an 'A' in Software Engineering II, and to produce a popular app, one weighs far more heavily than the other. In this case, the goal of making an 'A' in the course is more important than producing a popular app. Naturally, strategies 1 and 3, which favor this goal, are better options for Team Getana. Regrettably, these are also the strategies for which the goals of a Texas Tech student are least satisfied.

Decisions regarding the best strategy to implement should strive to satisfy as many of the stakeholders as possible. Achieving such a goal can be expected to involve some degree of compromise and may even leave some stakeholders completely unsatisfied for the collective good. Examining the implementable strategies shown above for the *RaiderNAV* project demonstrates that only strategy 4 satisfies both concerned actors, if only weakly. Although strategies 1 and 3 produce better results for Team Getana, they do not address the concerns of Texas Tech students. As such, strategy 4 is the best choice for implementation.

2. Use Case Maps

This section contains use case maps and scenarios which reflect the features currently implemented in the *RaiderNAV* app. Section 2.1 presents a top-down overview of all available app options beginning with the main sequence of events common to each interaction. Subsequent use case maps represent exchanges that may or may not take place during a given execution of the *RaiderNAV* app. Section 2.2 then demonstrates two of the most common scenarios that are likely to take place during typical use: First, a user opening the app and navigating to an unscheduled location. Second, a user opening the app for the first time and editing an individual course in a saved (imported) schedule

.

2.1 Main Sequence Actions

The first time *RaiderNAV* is launched after installation, the system will display the app's privacy policy to the user. Each following run will skip this notification. After the system has either displayed the privacy policy, or skipped over it, the user will be presented with the app's main interface. From here, the user chooses to perform one of two actions: perform an unscheduled navigation or perform an action on a schedule. Future releases may also incorporate other actions from the main interface.
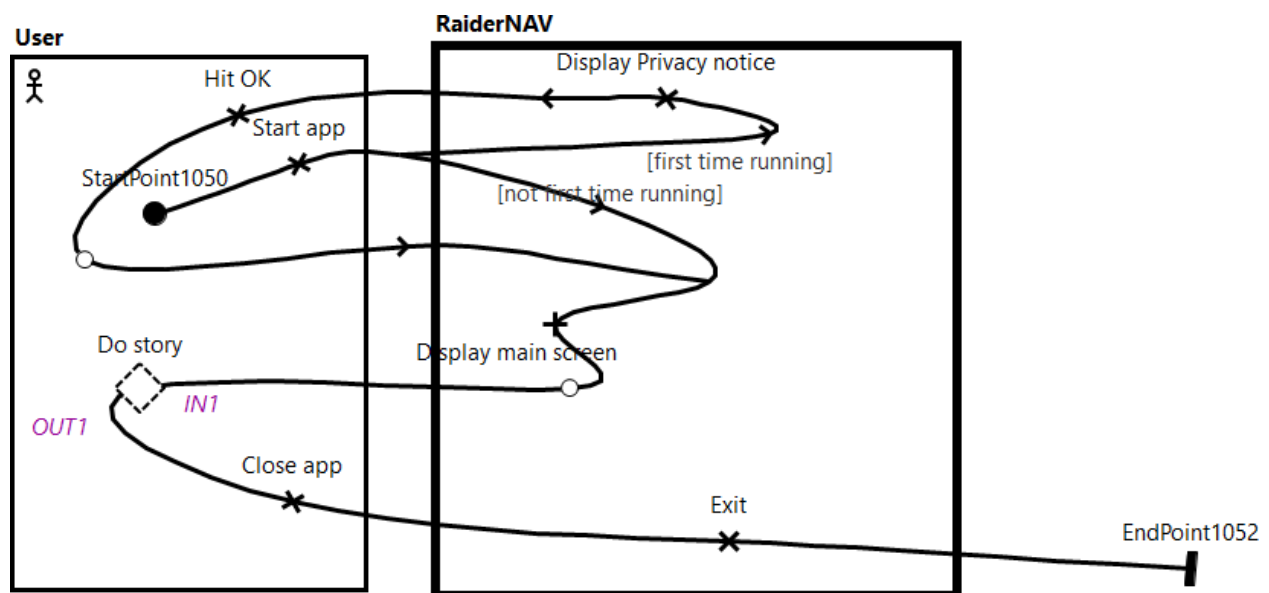

Figure 2.1 Main Sequence UCM

2.1.1 Unscheduled Navigation

The user may choose to perform an unscheduled navigation by pressing the unscheduled navigation icon. In this case, he/she is first prompted to choose a destination from a drop-down menu. At this point the user may choose to either cancel the current operation and return to the main menu by selecting "Cancel" or choose an option and select "OK" to proceed. After selecting a valid option, a map showing the user's current location and route to selected destination is displayed. The user may linger here for as long as desired and may manipulate the map. After the user is finished, the app again moves to the main screen.
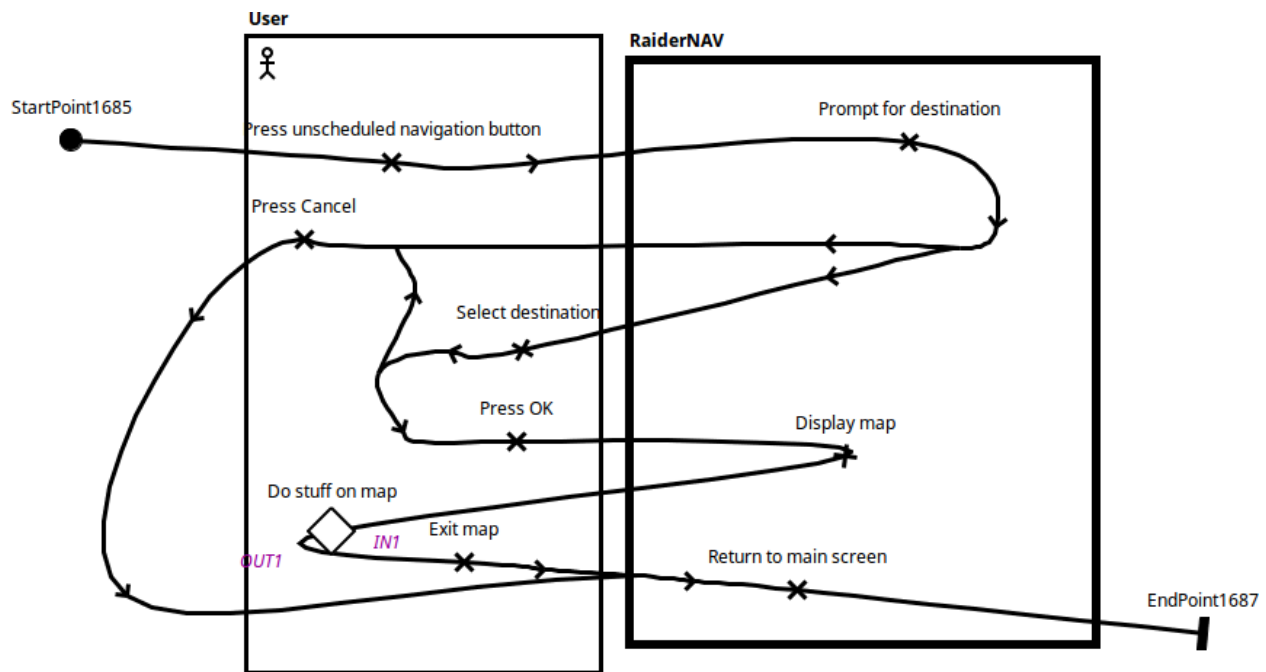
Figure 2.1.1 Unscheduled Navigation UCM

2.1.1.1 Show Shortest Route between Two Locations

While interacting with an existing map the user may also choose to dynamically generate the shortest route between two selected points. In order to do so the user need only long-press a location on the map, which *RaiderNAV* will designate as the source location and display an appropriate marker, followed by another long-press to set a destination. Once these steps have been completed *RaiderNAV* will display the shortest route between the two given points. The user may then utilize the map as desired and press the "Back" button to return to the main screen.
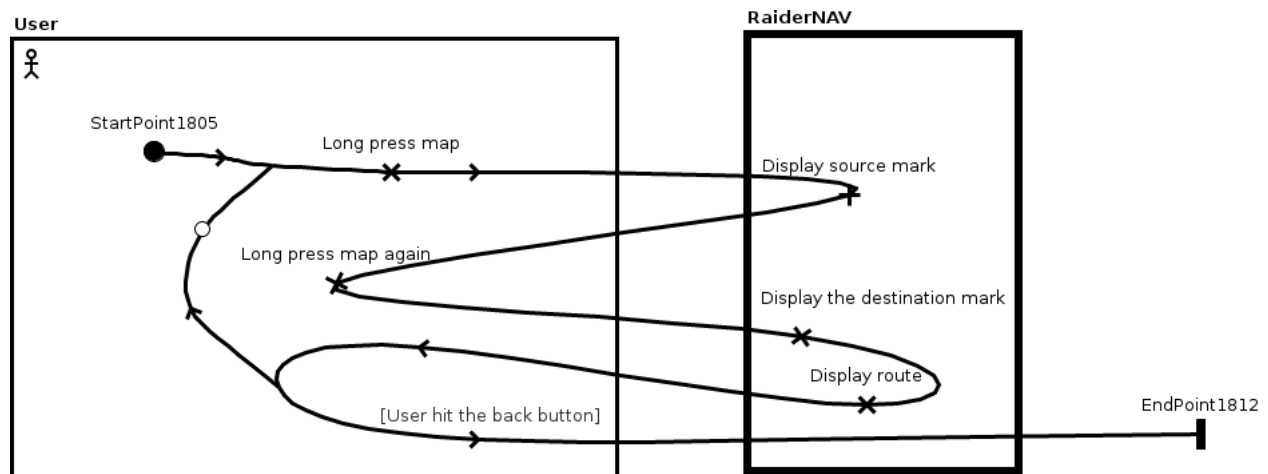
Figure 2.1.1.1 Show the shortest route between two locations

2.1.2 Schedule Action

Alternatively, the user may also select the "My Schedules" option on the main interface to perform a schedule action. If the user has done so, he/she will be redirected to the schedule selection screen. From here, the user may choose to either: create a new schedule by selecting the "Create Schedule" option or edit a particular saved schedule by selecting it.
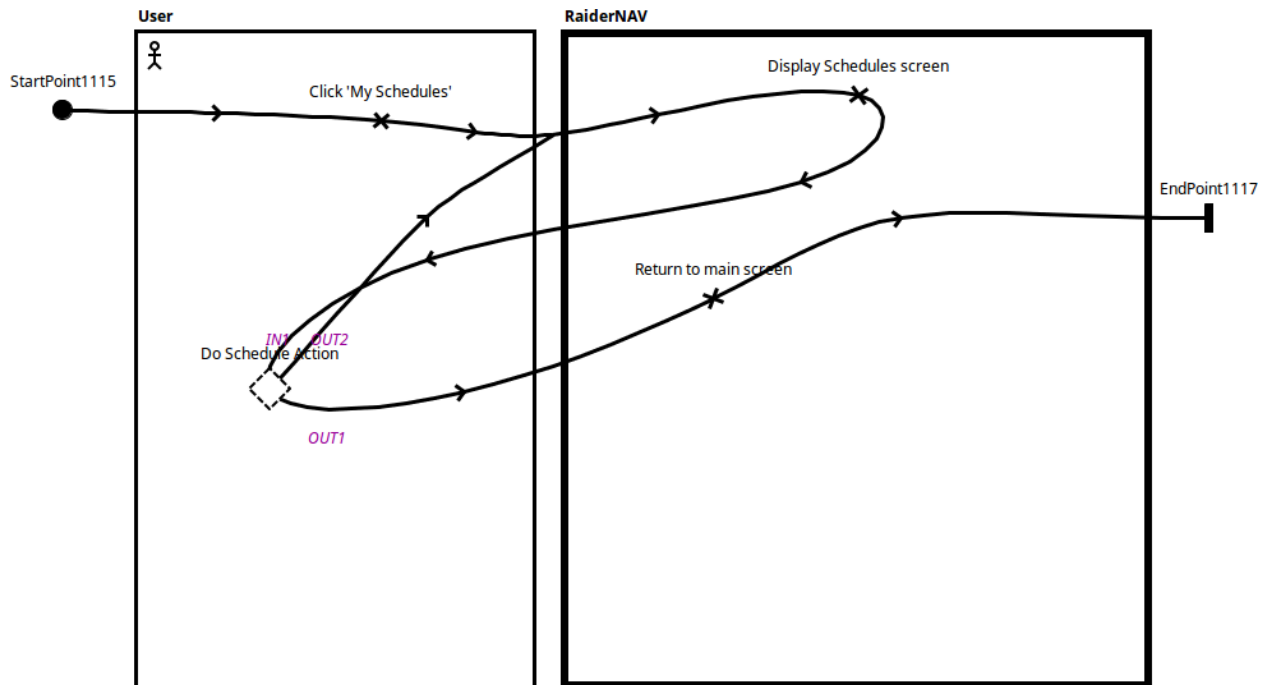

Figure 2.1.2 Schedule Main Sequence UCM

2.1.2.1 Create New Schedule

If the user chooses to create a new schedule, he/she is first prompted to enter a schedule name using the Android keyboard. The user may do so and select "OK" to add the newly named schedule to the schedule list or select "Cancel" if a new schedule is no longer desired. The user is then returned to schedule list screen regardless.
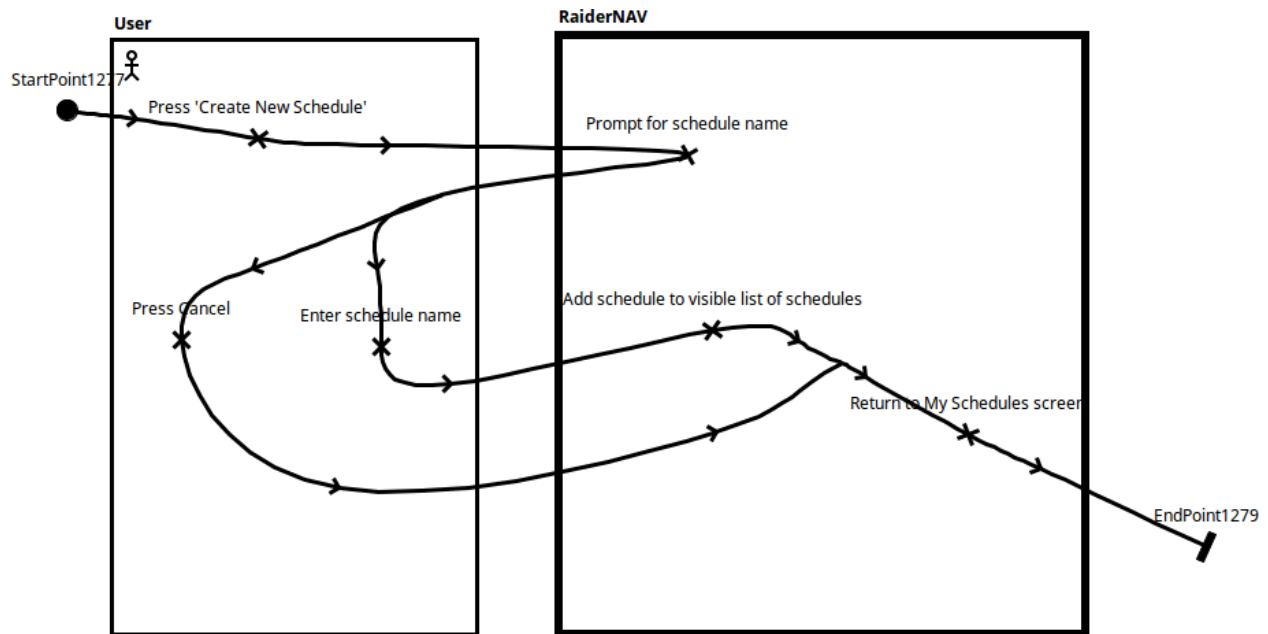

Figure 2.1.2.1 Create New Schedule UCM

2.1.2.2 Edit Existing Schedule

Selection of an existing schedule will present the user will details of the courses associated with the selected schedule. The user may then choose to add a new course to the existing schedule with the "Create Course" button, delete the schedule entirely by selecting "Delete Schedule," rename the schedule with the "Rename Schedule" option, or, finally, edit an existing course on the schedule by selecting the course.
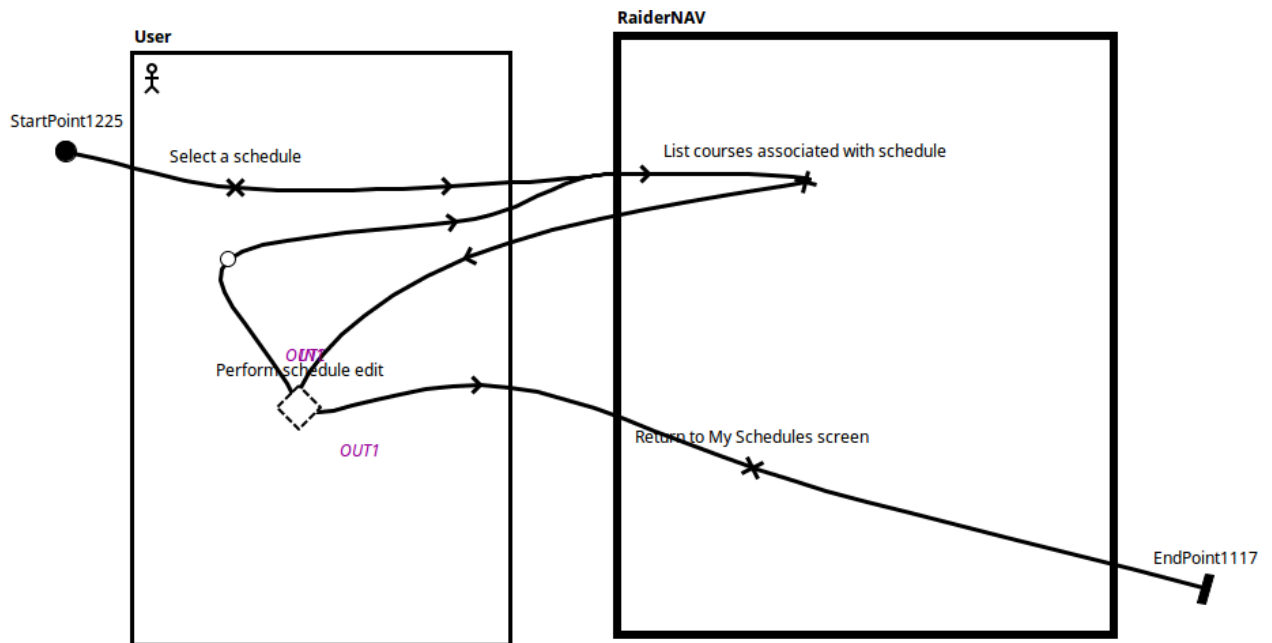

Figure 2.1.2.2 Edit Schedule Main Sequence UCM

2.1.2.2.1 Creating a New Course

If the user elects to add a new course to an existing schedule, they simply fill in the appropriate values for the course name, building, start time, end time, and days of the week in the fields provided by the app. The user then selects "Save" and is returned to the schedule detail page.
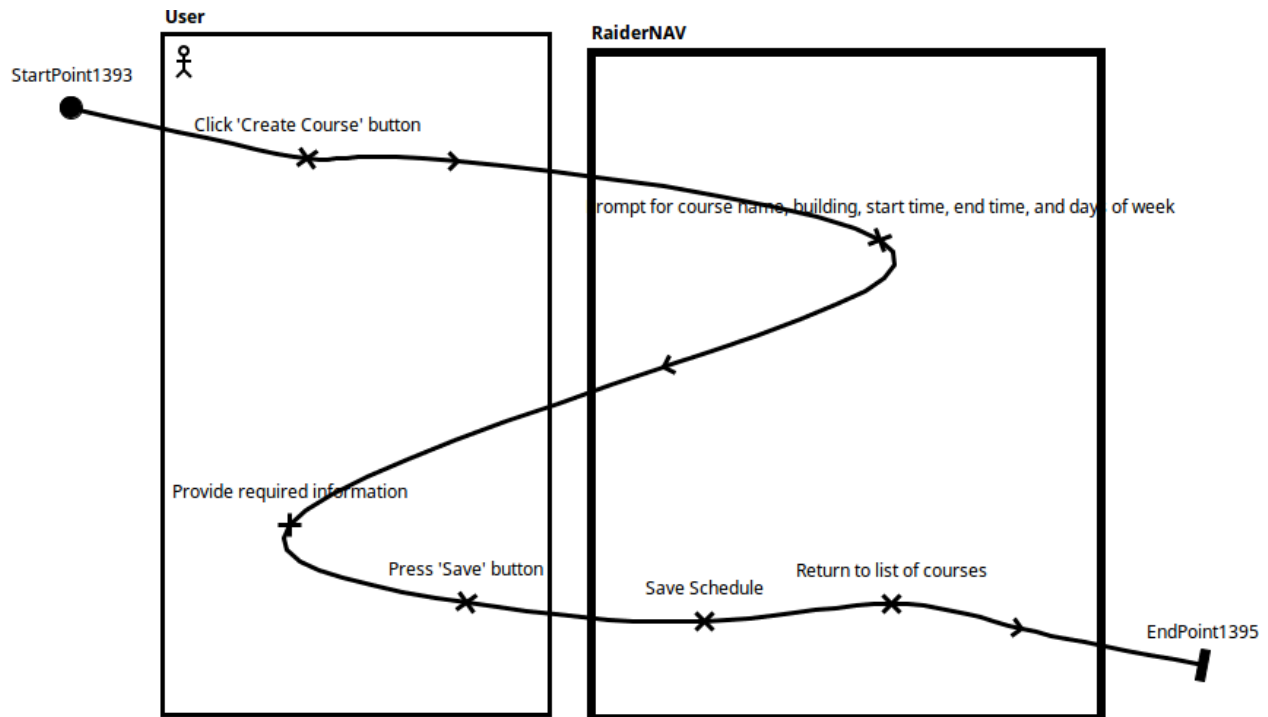
Figure 2.1.2.2.1 Create Course UCM

2.1.2.2.2 Deleting a Schedule

When deleting a schedule from the list of saved schedules, the user is prompted by *RaiderNAV* for confirmation. If the user affirms the decision by selecting "Confirm," the schedule is removed from *RaiderNAV* and the user is returned to the list of schedules. Alternatively, if the user presses "Cancel" the list of schedules remains unaltered and the user returned to the current schedule detail.
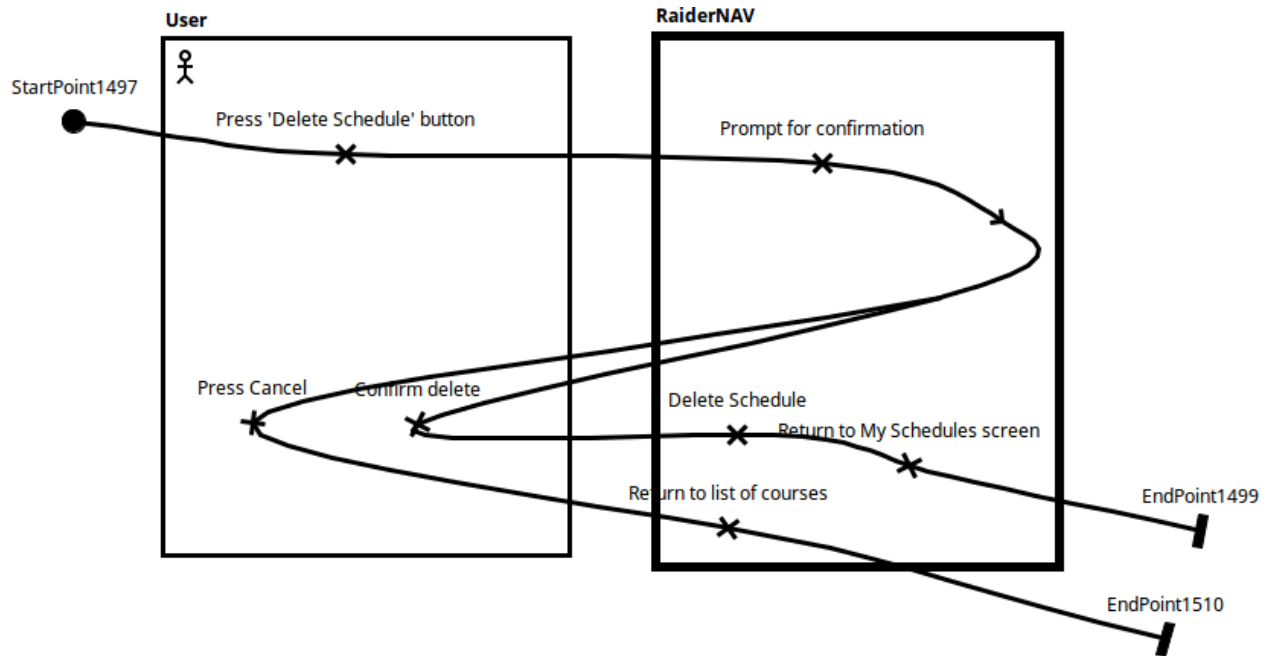

Figure 2.1.2.2.2 Delete Schedule UCM

2.1.2.2.3 Renaming a Schedule

Selection of the "Rename Schedule" option will cause the system to prompt the user for a new schedule name. The user may then provide a new schedule name using the Android keyboard and select "Save" to change the schedule name or select "Cancel" to keep the current schedule name. If the user changes the schedule's name, he/she is taken to the list of schedules. Otherwise the user is returned to the detail of the current schedule.
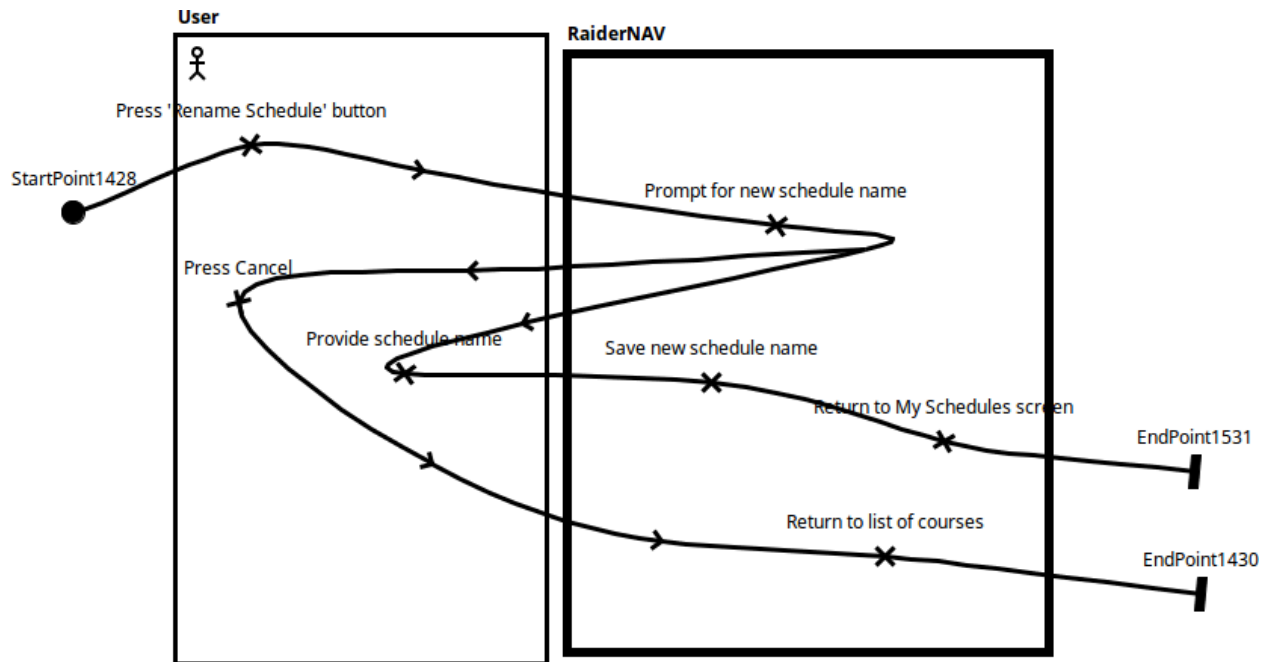
Figure 2.1.2.2.3 Rename Schedule UCM

2.1.2.2.4 Editing a Course

Finally, should the user choose to edit the details of an existing course by selecting it, the user will be directed to a screen showing each field of the course details in an editable format. From here the user may edit the desired fields and save the edited course by pressing "Save Course" or delete the course by selecting "Delete Course" and confirming their decision. The user also has the option to leave the course in its current state by utilizing the "Back" button. After the user has finished editing and/or deleting a course from a schedule, he/she has the option to modify another course or select "Save Schedule" to make the changes permanent and return to list of stored schedules. If the user fails to confirm deletion of a single course, he/she is returned to the course detail screen.
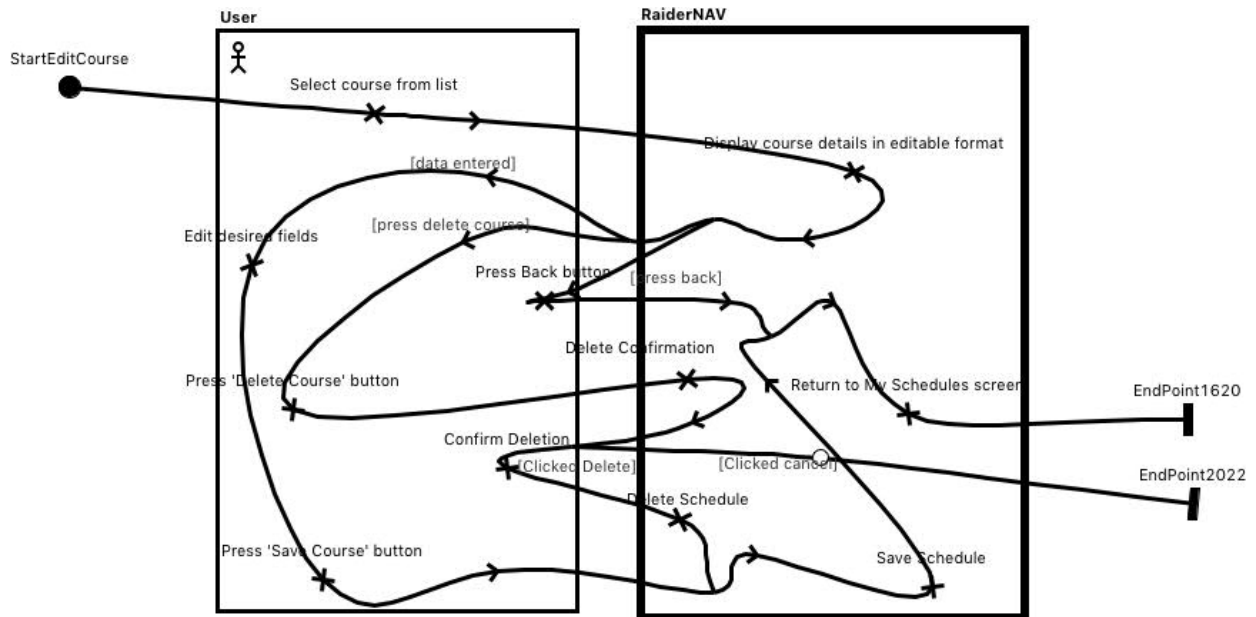

Figure 2.1.2.2.4 Edit Course UCM

2.2 Scenarios

2.2.1 User performs an unscheduled navigation

This scenario details in figures 2.2.1.1 & 2.2.1.2 the process by which a user may open the app and navigate to an unscheduled location. After launching the app, the user is directed to the main interface bypassing the privacy policy notification since the user has utilized the app before. The user then selects the unscheduled navigation button and is prompted to select a destination from the drop-down menu by *RaiderNAV*. The user chooses a valid option and selects "OK" to begin navigation. *RaiderNAV* then provides the user with a route from his/her current location to the appropriate destination on the map screen. After the user has finished interacting with the map screen he/she is returned to the main screen and closes the app. *RaiderNAV* then ceases execution and exits.
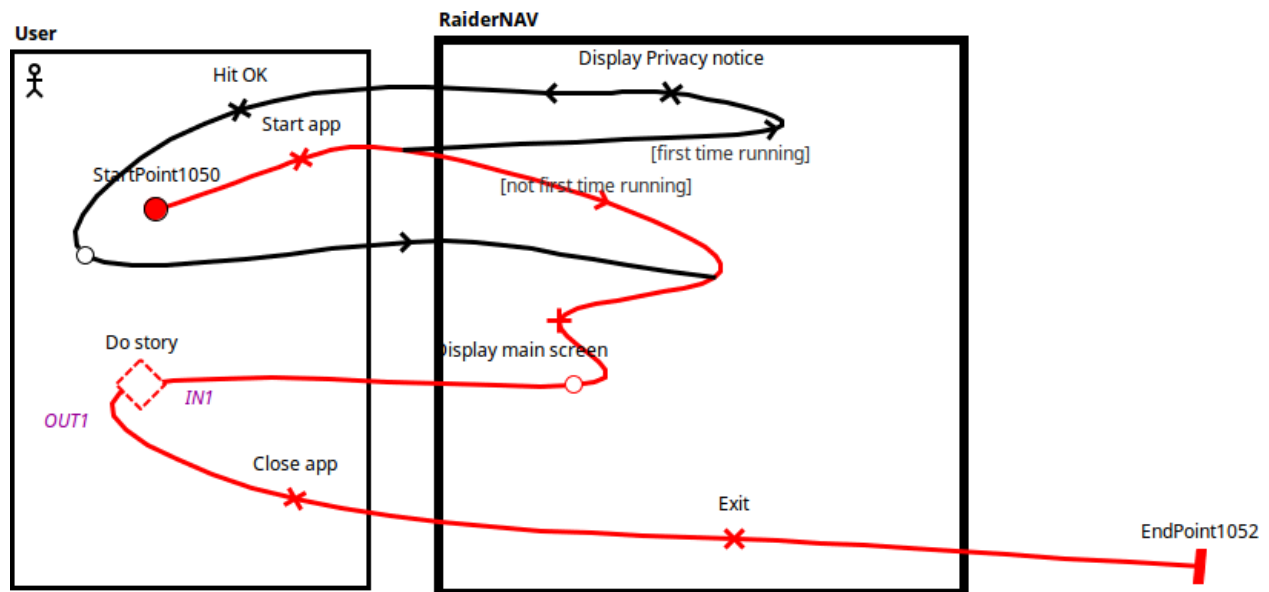

Figure 2.2.1.1 Top Level Sequence

Figure 2.2.1.2 Performing Unscheduled Navigation

StartPoint1685

User

Press unscheduled navigation button

Press Cancel

[!ok_after_destination]

Select destination

[ok_after_destination]

Press OK

Do stuff on map

OUT1   IN1   Exit map

RaiderNAV

Prompt for destination

[!destination_selected]

[destination_selected]

Display map

Return to main screen

EndPoint1687

2.2.2 User edits a course in a schedule

The second scenario shows the chain of events wherein a user launches the app for the first time and edits the course details on a saved (imported) schedule. This chain is detailed through figures 2.2.2.1, 2.2.2.2, 2.2.2.3, and 2.2.2.4. After launching the app, the user is first greeted by *RaiderNAV*'s privacy notification. The user selects "OK" to close the notification and is then directed to the main interface. At this point the user selects "My Schedules," and *RaiderNAV* displays the list of schedules to the user. From here, the user selects the schedule they wish to edit. *RaiderNAV* then displays the course details for the selected schedule from which the user then selects an individual course to edit. *RaiderNAV* now displays the courses details to the user in an editable format. The user makes the desired changes and presses the "Save Course" button to commit the changes. *RaiderNAV* responds by saving the updated course and respective schedule and then returns the user to the schedule selection screen. The user then returns to the main screen before electing to close the app. *RaiderNAV* responds by terminating execution and exiting.
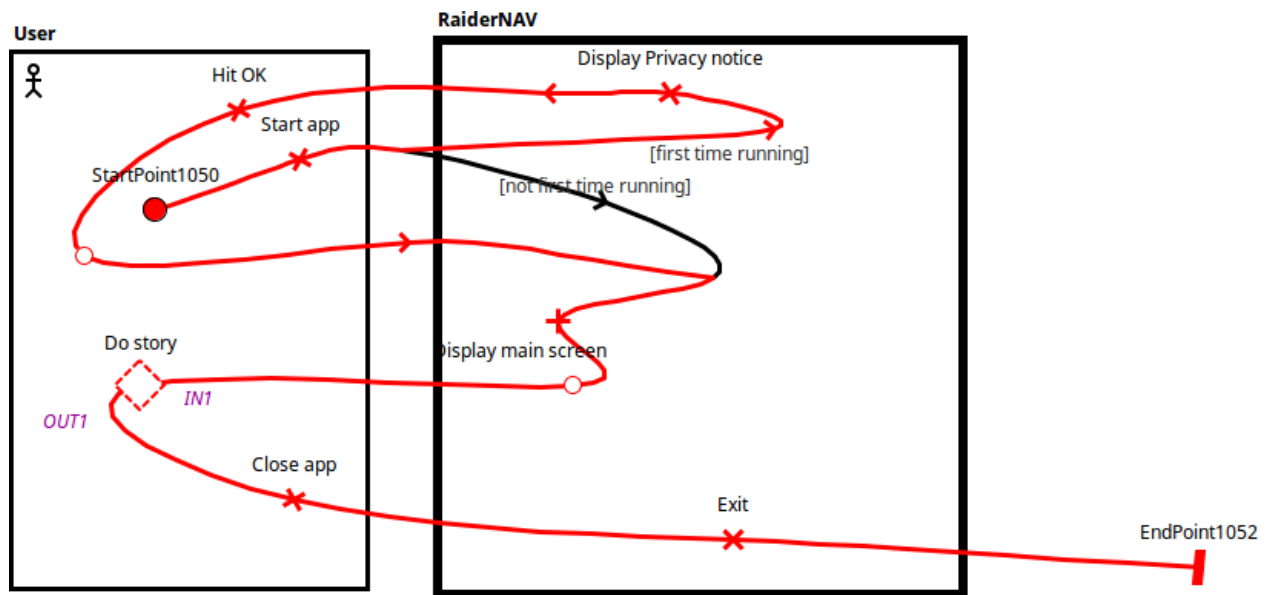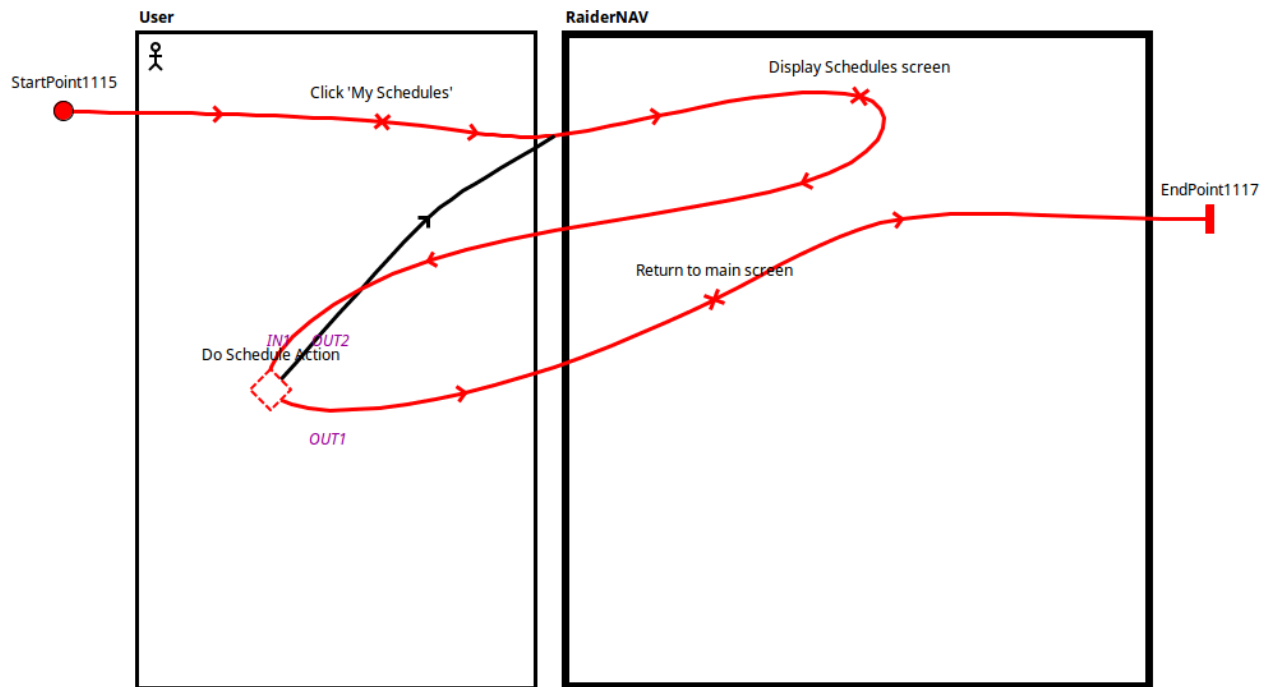

Figure 2.2.2.1 Top Level Sequence

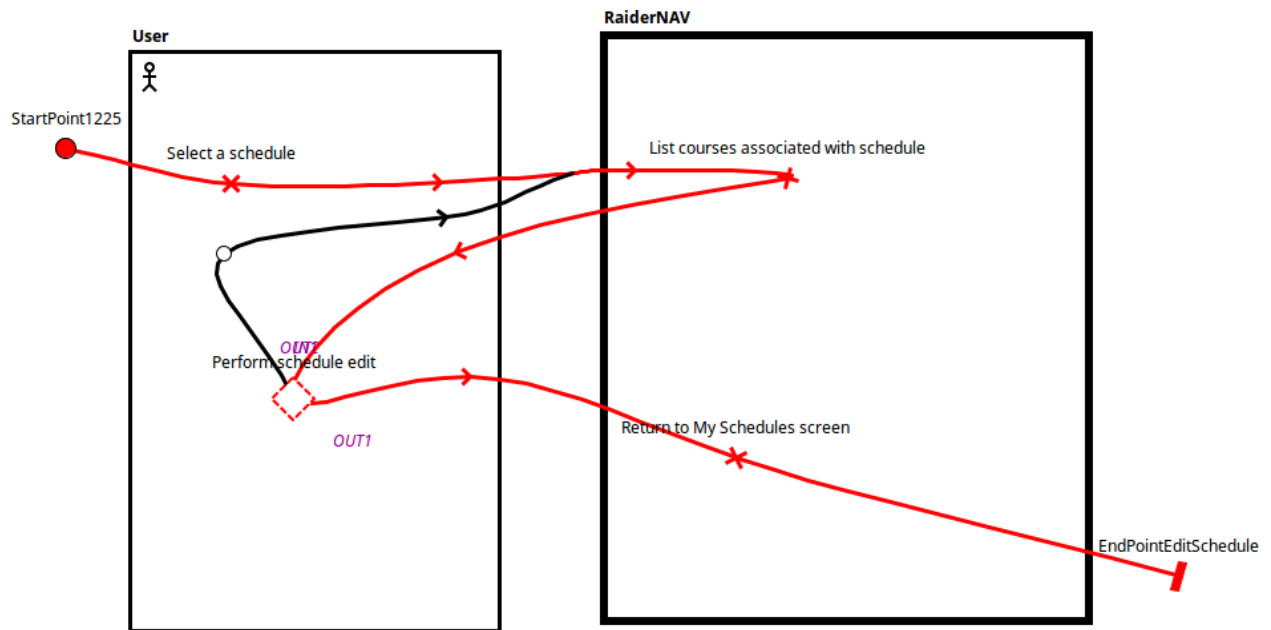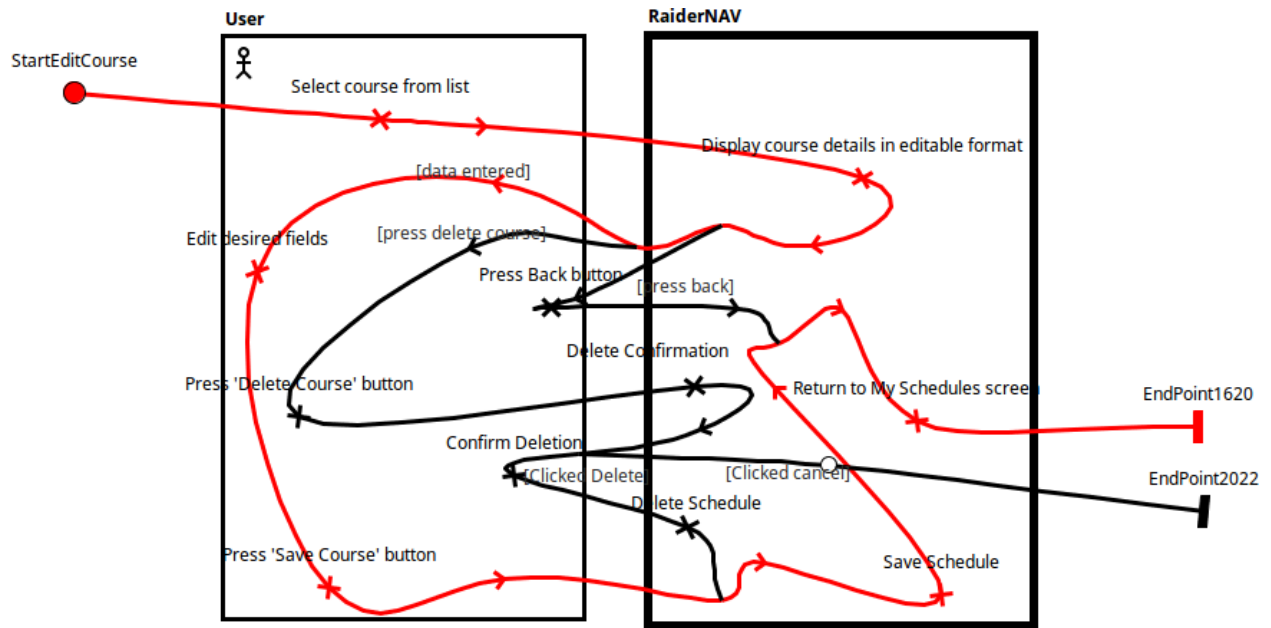Figure 2.2.2.2 User Enters Schedules View

Figure 2.2.2.3 User Selects a Schedule

Figure 2.2.2.4 User Edits a Course