

Лабораторная работа № 4

УПРАВЛЕНИЕ ТРЕБОВАНИЯМИ К ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ С ПОМОЩЬЮ IBM RATIONAL REQUISITE PRO. СОЗДАНИЕ ТЕСТОВЫХ СЦЕНАРИЕВ

Цель работы – получения навыков создания тестовых сценариев на основе методики управления требованиями.

Задачи работы – освоение принципов применения программного продукта IBM Rational Requisite Pro для управления требованиями к программному обеспечению.

Теоретическая часть

Процесс создания тестовых сценариев включает в себя четыре шага:

1. Определение переменных для каждого шага сценариев использования.
2. Определение существенно разных вариантов для каждой переменной.
3. Комбинирование вариантов для тестирования в тестовые сценарии.
4. Определение значений переменных.

Первым делом нужно определить все входные переменные во всех шагах в представленном сценарии (алгоритме). Например, если на некотором шаге пользователь вводит свой идентификатор и пароль, это две переменных. Первая переменная – Идентификатор, вторая – Пароль. Переменной также может быть выбор, который пользователь должен сделать (такой как выбор рейса из списка).

Количество переменных может зависеть от введенных на предыдущем шаге значений.

На следующем шаге следует определить существенно различные варианты для каждой переменной. Варианты считаются «существенно разными», если они могут вызывать разное поведение системы.

Например, если выбрать Идентификатор, который предполагается быть длиной до 10-ти символов, следующие приведенные значения будут существенно разными:

Alex – слишком короткий и мы ожидаем появления сообщения об ошибке.

Alexandra – верный Идентификатор.

Alexandrena – слишком длинный, и мы ожидаем, что система не позволит нам вводить слишком длинный Идентификатор.

Однако, Alexandria или JohnGordon различны не существенно, т.к. оба являются верными идентификаторами, вызывающими одно поведение системы.

Вариант может считаться существенно другим, если:

– Он вызывает другой ход процесса (обычно альтернативный поток).

Пример: Ввод неверного пароля вызовет Альтернативный Поток 2.

Он вызывает другое сообщение об ошибке.

Пример: Если адрес электронной почты слишком длинный, сообщение должно быть: «E-mail должен быть не более 50-ти символов».

Пример: Если адрес электронной почты не содержит символа «@», сообщение должно быть: «Неверный E-mail адрес».

– Он вызывает другой вид пользовательского интерфейса.

Пример: Если кредитная карта была выбрана в качестве способа оплаты, система должна отображать экран с полями для ввода номера кредитной карты, даты окончания срока действия и название организации, обслуживающей карту.

– Он вызывает различный набор значений списков.

Пример: Экран регистрации пользователя должен содержать список Страна и Штат/Провинция. Список Штат/Провинция будет содержать значения на основе выбранной страны: для США он должен содержать все штаты, для Канады – все провинции, для других стран он должен быть недоступен.

– Он является условием ограничения.

Пример: Пароль должен быть не менее 6-ти символов.

В этом случае мы должны протестировать следующее:

Пароль с пятью символами.

Пароль с шестью символами

– Что-то должно быть изменено вместо использования значения по умолчанию.

Пример: На экране оплаты кредитной картой название организации, обслуживающей кредитную карту, должно быть заполнено именем лица, размещающего заказ. Пользователь должен иметь возможность хранить это значение по умолчанию или ввести новое.

– Это создает два различных варианта:

Хранить установленное по умолчанию название организации, обслуживающей кредитную карту.

Изменить установленное по умолчанию название организации на другое.

– Формат ввода точно не определен и может быть интерпретирован разными способами.

Пример: поле ввода номера телефона должно принимать текст в свободной форме.

Номера телефонов разными лицами пишутся по-разному:

Использование скобок: (973) 123-4567

Использование тире: 973-123-4567

Использование пробелов: 973 123 4567

Без пробелов: 9731234567

Все доступные варианты должны быть протестированы.

– Стандартные варианты могут быть разными для разных стран.

Формат даты срока окончания действия кредитной карты может быть разным для США и Европы.

На следующем шаге нужно скомбинировать их в последовательность шагов тестового сценария. Один из способов это сделать – создать Матрицу Распределения Тестовых Сценариев (Test Case Allocation Matrix). Строки этой матрицы содержат все переменные для всех шагов, требующие ввода данных от пользователя. Первая колонка содержит номер шага, вторая – название переменной, а остальные – тестовые сценарии. Их можно назвать T1, T2, и т.д. Нужно оценить, насколько много тестовых сценариев нужно для охвата данного сценария (алгоритма). Жестким вариантом оценки будет максимальное количество существенно различных вариантов, определенное для переменной. Не проблема, если при оценке будет допущена ошибка, т.к. можно добавить или удалить колонку при заполнении матрицы. Обычно типичный сценарий охватывают от пяти до семи тестовых сценариев. Тем не менее, иногда в особых случаях требуется больше тестовых сценариев.

На четвертом шаге следует заменить неопределенные варианты, такие как «очень длинная фамилия» или «длинный номер телефона с ext. (дополнительным номером)» на действительные значения, например «Georgiamistopolis» и «011-48 (242) 425-3456 ext. 1234» соответственно. На этом шаге также разделяют все тестовые сценарии из Матрицы Распределения Тестовых Сценариев, создавая отдельную таблицу для каждого тестового сценария.

Метод извлечения функциональных тестовых сценариев (test cases) из сценариев использования (use cases) имеет несколько преимуществ:

– Тестовые сценарии получаются с использованием более автоматического подхода.

– Уход от дублирования тестовых сценариев.

– Достигается больший охват тестового пространства.

– Легкость мониторинга тестового процесса.

– Легкость распределения работы между тестерами.

– Легкость регрессионного тестирования.

– Раннее обнаружение пропущенных требований.

Созданные тестовые сценарии могут быть использованы для ручного тестирования, также как и для автоматического тестирования с использованием таких инструментов, как IMB Rational Robot или IBM Rational Functional Tester.

Описание задачи

На основе сценариев использования предыдущей лабораторной работы построить тестовые сценарии.

Методика выполнения работы

А. Определение переменных для каждого шага сценариев использования

1. Переменные для основного потока сценария использования:

Переписать основной поток сценария «Бронирование билетов» с указанием используемых переменных (с префиксом V):

V1. Турист вводит URL сайта.

V2. Система отображает домашнюю страницу сайта.

V3. Турист вводит информацию о полете:

V3.1. Дата вылета

V3.2. Время вылета

V3.3. Дата прибытия

V3.4. Время прибытия

V3.5. Число путешествующих взрослых

V3.6. Число путешествующих детей.

V3.7. Наличие путешествующих животных.

Турист выбирает «Поиск рейсов».

V4. Система отображает рейсы вылета, отсортированные по цене.

V5. Турист выбирает рейс.

V5.1. Рейс вылета

V6. Система отображает рейс прибытия.

V7. Турист выбирает рейс прибытия.

V7.1. Рейс прибытия.

V8. Система отображает детали рейса.

V9. Турист подтверждает рейс.

V10. Пользователь предоставляет Идентификатор и Пароль для покупки билета.

V10.1. Идентификатор

V10.2. Пароль

V11. Турист предоставляет информацию пассажира.

V11.1. Фамилия.

V11.2. Имя.

V11.3. Отчество.

V11.4. Пол.

V11.5. Дата рождения.

V11.6. Номер паспорта.

V11.7. Серия паспорта.

V12. Система отображает свободные места.

V13. Турист выбирает места.

V13.1. Место.

V14. Система отображает доступное меню.

V15. Турист выбирает меню.

V15.1. Тип меню.

V15.2. Тип напитков.

V16. Турист предоставляет информацию по кредитной карте и расчетный адрес.

V16.1. Тип кредитной карты

- V14.2. Номер кредитной карты
- V14.3. Дата окончания срока действия
- V14.4. Название карты
- V14.5. Адрес

B15. Система предоставляет номер подтверждения.

2. Аналогичным образом ввести перечень переменных для каждого альтернативного потока.

Б. Определение различных вариантов для каждой переменной

1. Определить значения переменных для тестирования:

V3.1. Дата вылета

- Верная дата в будущем, установленная вручную
- Верная дата в будущем, установленная из календаря
- Дата в прошлом
- Сегодняшняя дата
- Февраль 30 или 31 число
- Пустое поле

V3.2. Время вылета:

- Верное время для будущей даты, установленное вручную
- Верное время для текущей даты, установленное вручную
- Неверный формат даты
- Пустое поле

V3.5. Число путешествующих взрослых

- 0
- 1
- 2
- Максимально допустимое

V10.1. Идентификатор

- Верный идентификатор пользователя
- Идентификатор, содержащие недопустимые символы
- Несуществующий идентификатор пользователя
- Пустое поле

V10.2. Пароль

- Правильный пароль пользователя (с правильным идентификатором)
- Неправильный пароль (с правильным идентификатором)
- Верный пароль (с неверным идентификатором)
- Пароль, содержащий недопустимые символы
- Пустое поле

По аналогии определить возможные значения для остальных переменных (не менее трех значений на переменную).

В. Создание тестовых сценариев

1. Построить матрицу распределения тестовых сценариев:

Шаг	Переменная	T1	T2	T3	T4	T5	T6
B3	Время вылета						
B3	Дата прибытия						
B3	Время прибытия						
B3	Число путешествующих взрослых						

2. Для каждой строки ввести все необходимые для тестирования варианты:

Шаг	Переменная	T1	T2	T3	T4	T5	T6
B3	Время вылета	Верная дата в будущем, установленная вручную	Верная дата в будущем, установленная из календаря	Дата в прошлом	Сегодняшняя дата	Февраль 30 или 31 число	Пустое поле
				Верная дата в будущем, установленная из календаря		Верная дата в будущем, установленная из календаря	Верная дата в будущем, установленная вручную
B10	Пароль	Правильный пароль пользователя (с правильным идентификатором)	Неправильный пароль (с правильным идентификатором)	Верный пароль (с неверным идентификатором)	Пароль, содержащий недопустимые символы (с неверным идентификатором)	Пароль, содержащий недопустимые символы (с верным идентификатором)	Пустое поле
			Правильный пароль пользователя (с правильным идентификатором)	Правильный пароль пользователя (с правильным идентификатором)	Правильный пароль пользователя (с правильным идентификатором)	Правильный пароль пользователя (с правильным идентификатором)	Правильный пароль пользователя (с правильным идентификатором)

3. Определить значения переменных:

3.1. Для первого тестового сценария (T1) определить конкретные значения переменных:

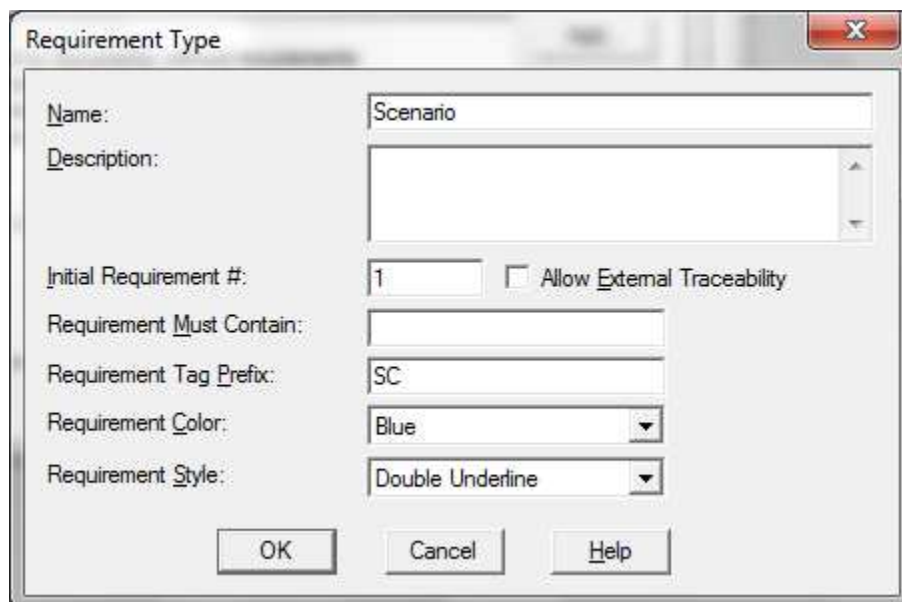
Шаг	Переменная	T1	Значение	Ожидаемый результат	Фактический результат	Удачный / неудачный	Комментарии
B3	Время вылета	Верная дата в будущем, установленная вручную	16.04.2015	Принято			
B10	Пароль	Правильный пароль пользователя (с правильным идентификатором)	hcvbbnsj	Принято			

3.2. Заполнить аналогичные таблицы по остальным тестовым сценариям.

Г. Создание тестовых сценариев (Test Case) в RequisitePro

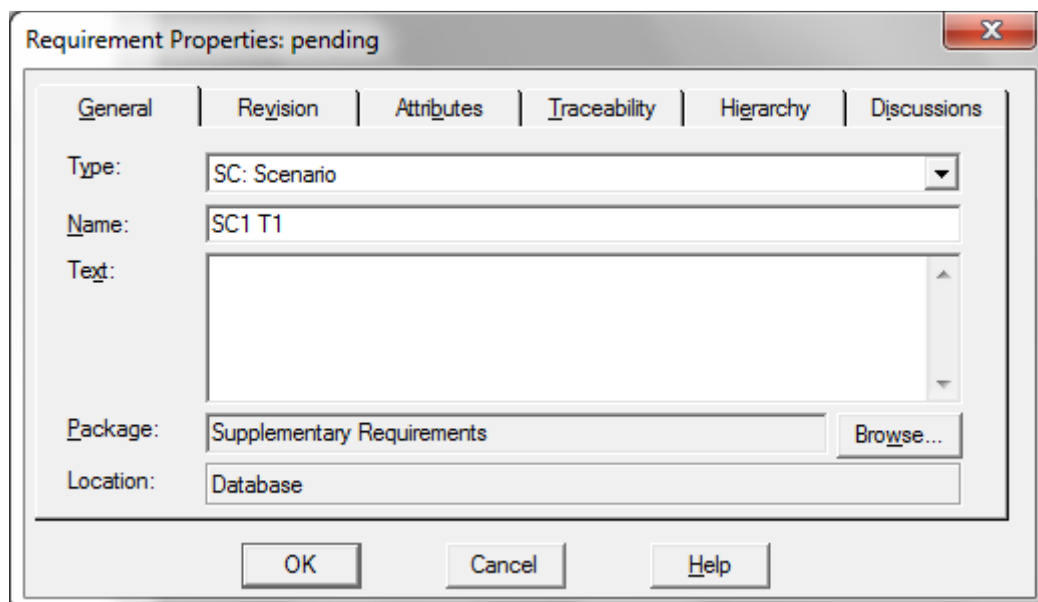
1. Создать новый тип требований

конт. меню SpaceTravel → Properties → ф. Project Properties | вкл. Requirements Type | кн. Add → ф. Requirement Type | Name ← Scenario, Requirement Tag Prefix ← SC, остальные параметры – по умолчанию, кн. Ok → ф. Project Properties | кн. Ok

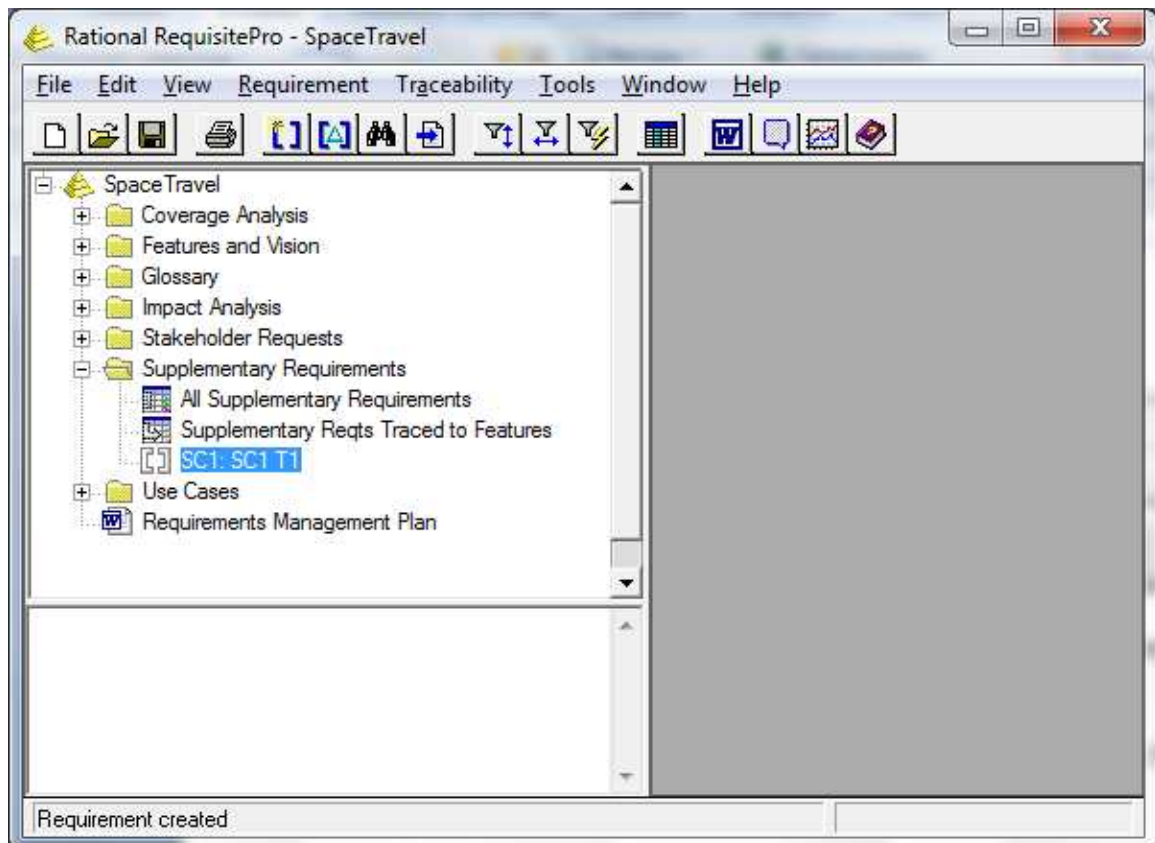


2. Создать новое требование:

SpaceTravel → Supplementary Requirements → конт. меню → New → Requirement... → ф. Requirement Properties: pending | Type ← SC: Scenario, Name ← SC1 T1 (номер сценария использования + номер тестового сценария), кн. Ok



Проверить наличие нового требования в окне проекта.

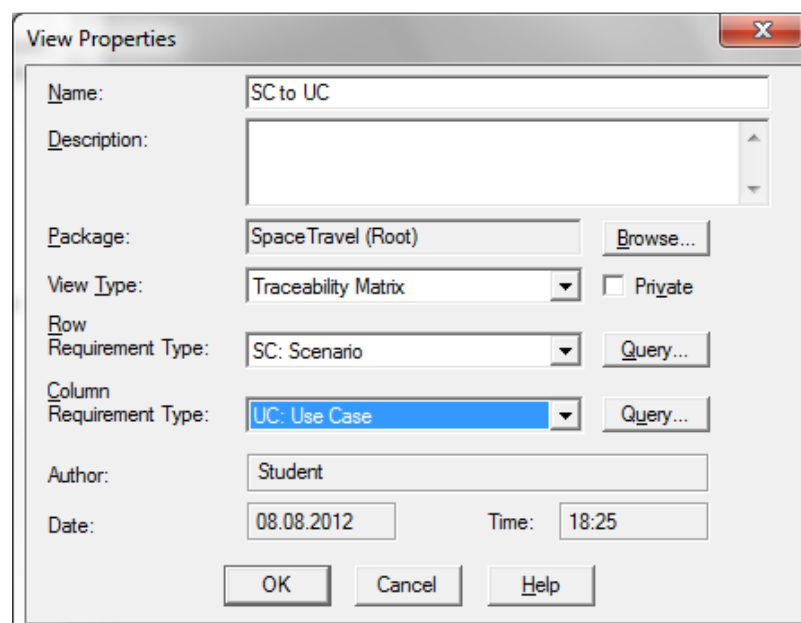


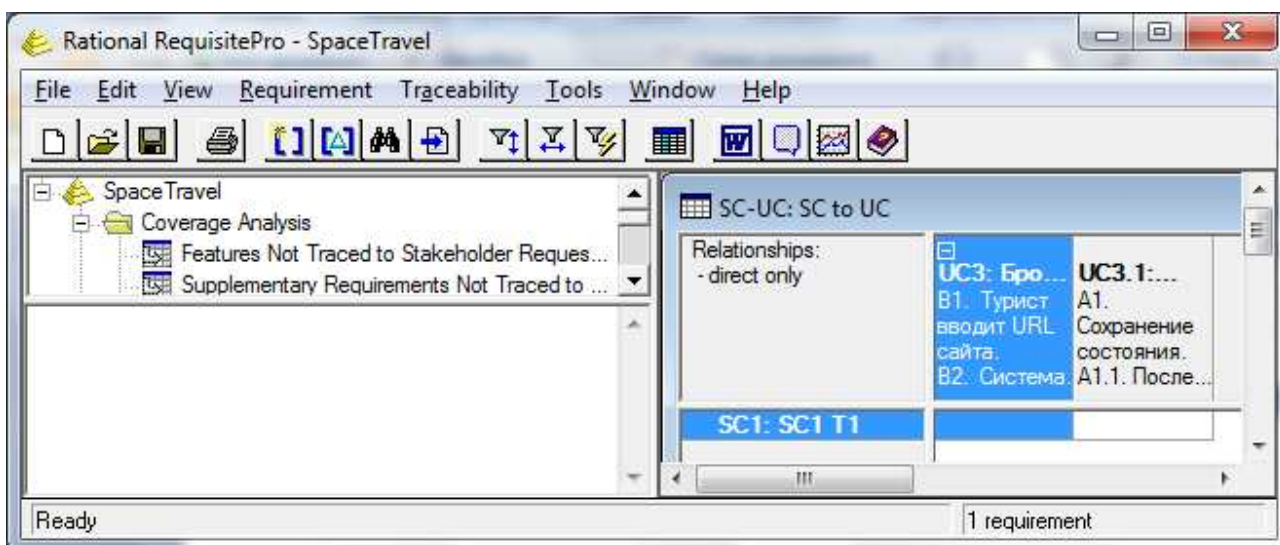
Аналогичным образом определить требования типа Scenario для остальных сценариев использования и тестовых сценариев.

3. Создать матрицу трассировки для связи требований Scenario с требованиями других типов:

3.1. Создать матрицу трассировки

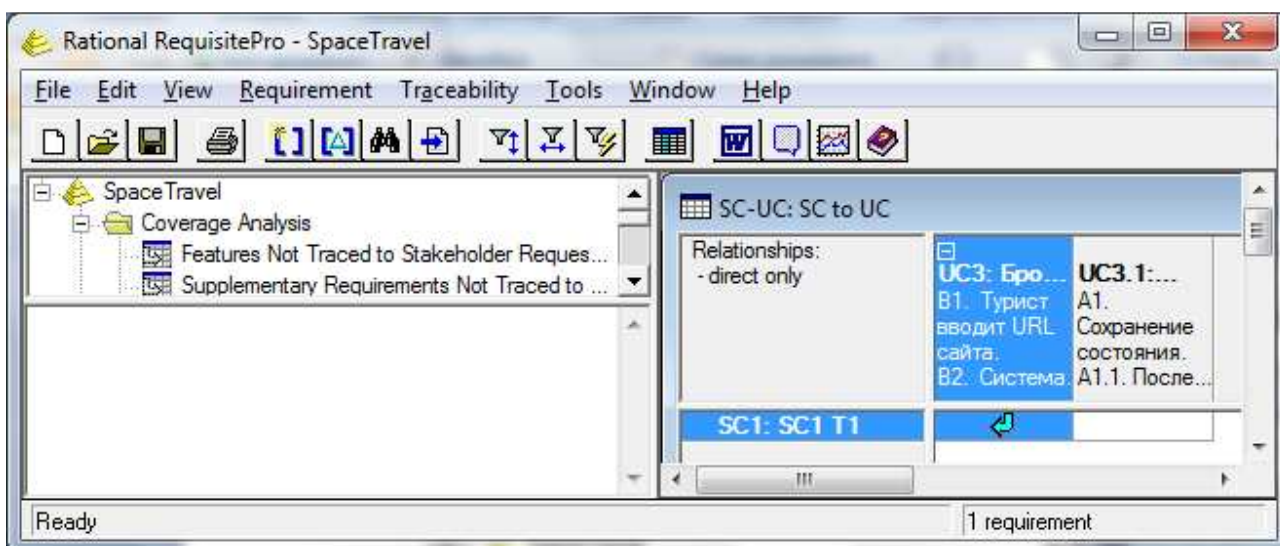
ф. Rational RequisitePro – SpaceTravel → конт. меню → New → View... → ф. View Properties | Name ← SC to UC, View Type ← Traceability Matrix, Row Requirement Type ← SC: Scenario, Column Requirement Type ← UC: Use Case, кн. Ok → матрица трассировки на экране





3.2. Установить связи между требованиями типа SC и UC

Сценарий использования (Use Case) и основанный на нем тестовый сценарий (Scenario) → установить курсор на пересечении строки, соответствующей тестовому сценарию, и столбца, соответствующего сценарию использования → контекстное меню → Trace From



3.3. Аналогичным образом установить связи для всех тестовых сценариев (один сценарий использования может привести к нескольким тестовым сценариям)

3.4. Закрывать матрицу трассировки.

4. Создать дерево трассировки

ф. Rational RequisitePro – SpaceTravel → конт. меню → New → View... → ф. View Properties | Name ← Traceability Tree for SC, View Type ← Traceability Tree (Traced into), Row Requirement Type ← SC: Scenario, кн. Ok → результат на экране

View Properties

Name: Traceability Tree for SC

Description:

Package: SpaceTravel (Root) Browse...

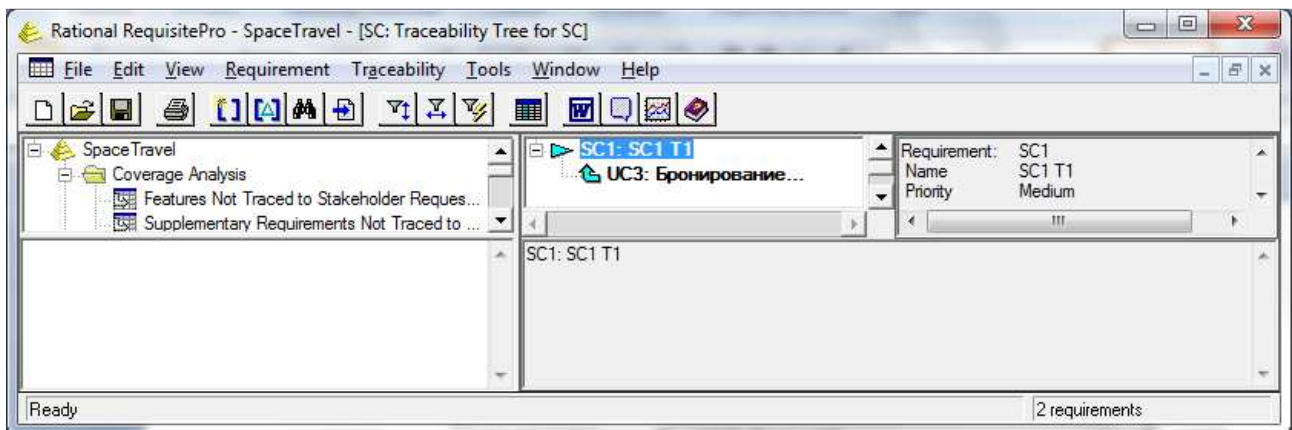
View Type: Traceability Tree (Traced into) Private

Row Requirement Type: SC: Scenario Query...

Author: Student

Date: 08.08.2012 Time: 18:36

OK Cancel Help



Требования к отчету

Отчет должен содержать:

- название, цель и задачи лабораторной работы;
- краткую теоретическую часть;
- переменные для основного потока сценария использования;
- значения переменных для тестирования;
- матрицу распределения тестовых сценариев;
- распределение значений переменных по тестовым сценариям;
- экранные формы работы с RequisitePro;
- выводы по результатам работы.