EPAM Systems, RD Dep.

Конспект и раздаточный материал

CTECH.01 Основы UML

REVISION HISTORY					
Ver.	**Description of Change**	**Author**	**Date**	**Approved**	
				Name	**Effective Date**
<1.0>	Первая версия	Евгений Пешкур Святослав Куликов	<21.09.2011>		

Содержание

1. Общие сведения о UML

UML (Unified Modeling Language, унифицированный язык моделирования) – язык графического описания для объектного моделирования в области разработки программного обеспечения.

UML – открытый стандарт, использующий графические обозначения для создания абстрактной модели системы.

UML создан для визуализации проектирования и документирования.

UML не является языком программирования, но на его основе возможна кодогенерация.

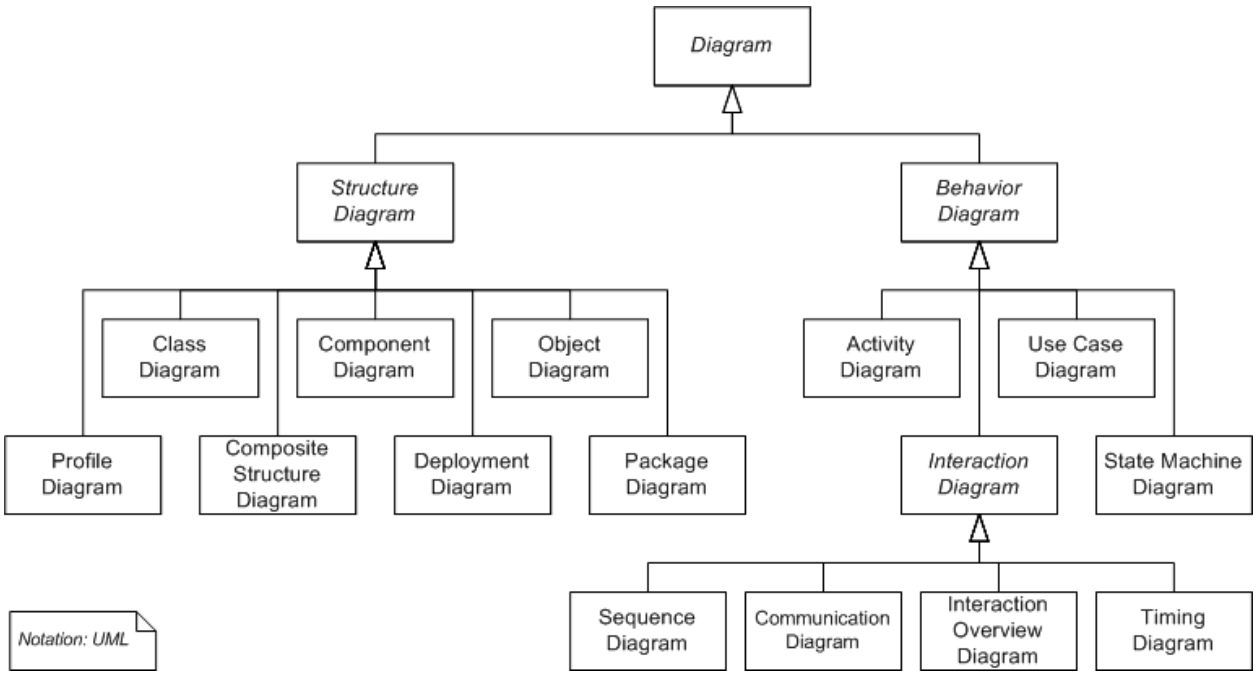
В основе UML лежит принцип многомодельности.

Этот принцип гласит, что никакая единственная модель не может с достаточной степенью адекватности описывать различные аспекты сложной системы.

Наиболее общими представлениями сложной системы принято считать статическое и динамическое представления, которые в свою очередь могут подразделяться на другие более частные представления.

В случае UML это означает взаимную конкретизацию и взаимное дополнение различными моделями друг друга.

UML диаграммы



2. Диаграммы вариантов использования

Диаграмма вариантов использования (Use case diagram) – диаграмма, на которой отражены отношения между акторами и вариантами использования.

Основная задача – предоставить возможность заказчику, конечному пользователю и разработчику совместно обсуждать функциональность и поведение системы.

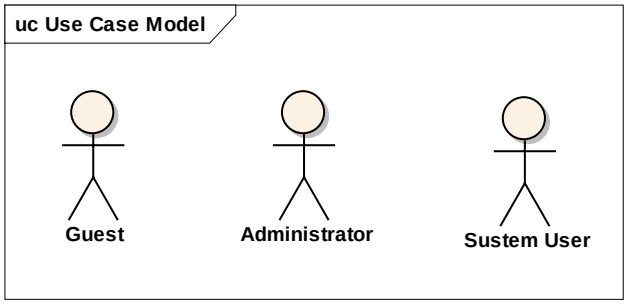
Основные правила:

- каждый вариант использования относится как минимум к одному действующему лицу;
- каждый вариант использования имеет инициатора;
- каждый вариант использования приводит к результату.

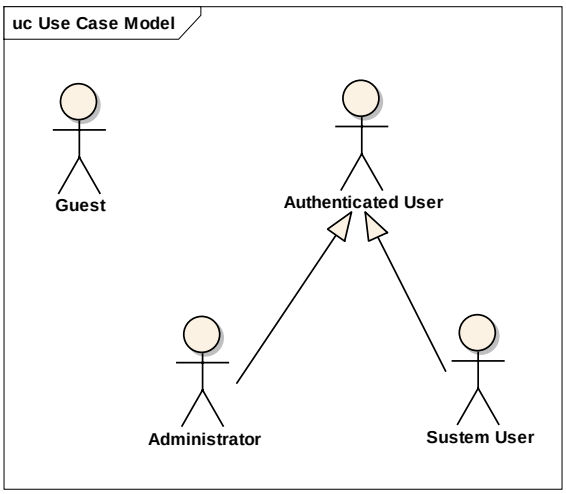
Пример диаграммы вариантов использования

Допустим, у нас есть ПО для составления и проверки отчётов. Выделим три роли:

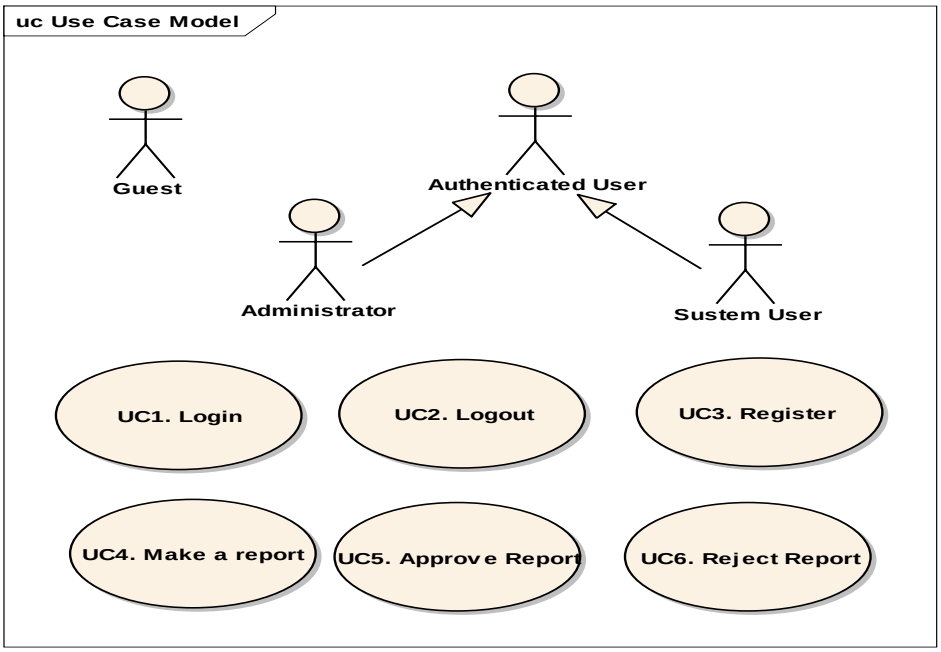
- гость (некто, не идентифицировавший себя);
- пользователь (создаёт отчёты);
- администратор (принимает отчёты).



Общее для пользователя и администратора вынесем в отдельную роль – авторизованный пользователь.

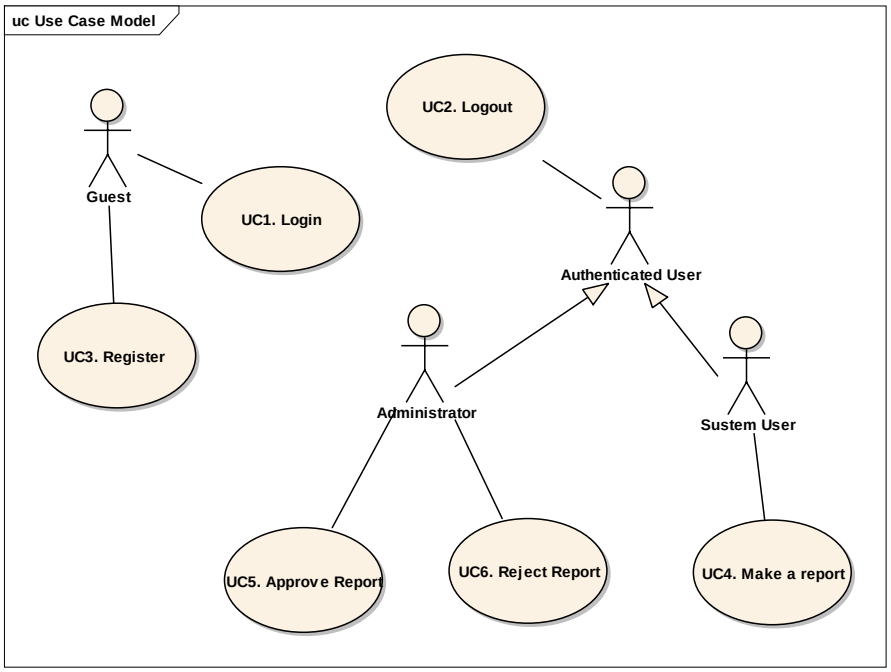


Перечислим действия, которые ПО позволяет выполнить представителям всех ролей.



Соединяем роли и действия (варианты использования).

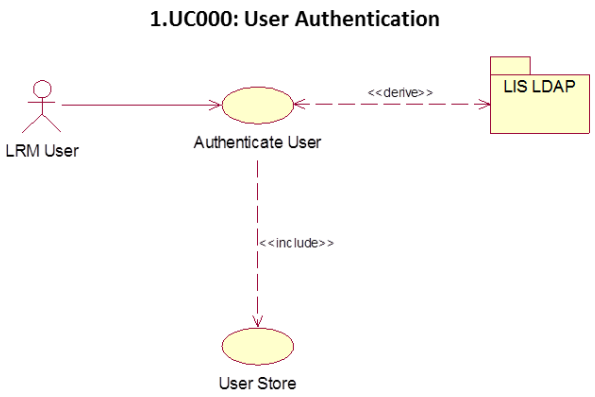
Если останется «непривязанный» вариант использования или актор – диаграмма составлена некорректно.



Каждый вариант использования затем расписывают в виде алгоритма взаимодействия «актор-система».

Это уже не относится к UML, но является важной частью бизнес-анализа,

позволяющей составить качественные тест-кейсы.



U.C. name:	Authenticate User	Code:	UC000
Actors:	LRM user		
Precondition:	User has successfully connected to LRM Home page		
Post condition:	The LRM User logs on the system.		
User		**System**	
1. User wants to log on the system.		2. Log on page is shown in LRM system	
3. Introduces his name and password		4. Connect (if it is not yet) with LIS LDAP System	
		5. Check user's name and password with LIS LDAP system	
		6. LIS LDAP System's OK answer received	
		7. Stores User rights for later use	
		8. Writes system log	
		9. Sends OK feedback page to user	
10. User logs on system			
Extensions			
		4a. It is impossible to connect with LIS LDAP System	
		1. Sends error with system page to user	
		2. Writes system log	
		6a. LIS LDAP System's bad answer received	
		1. User is not allowed to log on	
		2. Writes system log	
		3. Sends error in name or password page to user	
4. User reintroduces his name and password			
		5. Resume to step 5	
10a. User can not log on system			
1. User wants to contact with system administrator		message page is shown	
3. User fill fields		4. System checks fields	
		5. Fields are OK, else return error	
6. User send message		7. Message is delivered to system administrator	

3. Диаграммы классов

Диаграмма классов (Class diagram) – статическая диаграмма, описывающая структуру системы.

Она демонстрирует классы системы, их атрибуты, методы и зависимости между классами.

Существуют разные точки зрения на построение диаграмм классов в зависимости от целей их применения:

- концептуальная – диаграмма описывает модель предметной области, в ней присутствуют только классы прикладных объектов;
- точка зрения спецификации – диаграмма применяется при проектировании информационных систем;
- точка зрения реализации – диаграмма содержит классы, используемые непосредственно в программном коде (при использовании объектно-ориентированных языков программирования).

Имя класса должно быть уникальным в пределах пакета, который описывается некоторой совокупностью диаграмм классов. Оно указывается в первой верхней секции прямоугольника.

Во второй сверху секции прямоугольника записываются его атрибуты. Каждому атрибуту класса соответствует отдельная строка текста, которая состоит из:

- квантора видимости атрибута;
- имени атрибута;
- кратности;
- типа значений атрибута (и, возможно, его исходного значения).

Квантор видимости может принимать одно из трёх возможных значений:

- «+» – общедоступный (public) атрибут: доступен из любого другого класса пакета, в котором определена диаграмма.

- «#» – защищённый (protected) атрибут: недоступен для всех классов, за исключением подклассов данного класса.
- «-» – закрытый (private) атрибут: недоступен для всех классов без исключения, кроме того класса, где он объявлен.

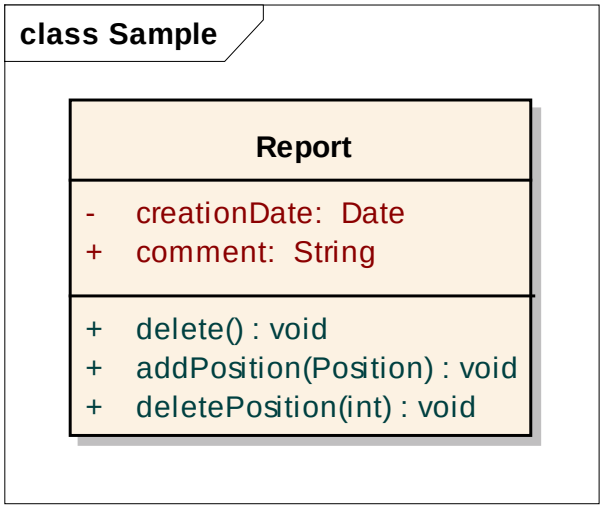
В третьей сверху секции прямоугольника записываются операции (методы, method) класса.

Совокупность операций характеризует функциональный аспект поведения класса.

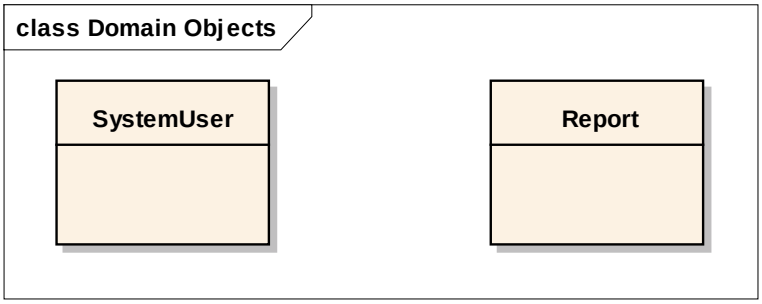
Каждой операции класса соответствует отдельная строка, которая состоит из:

- квантора видимости;
- имени;
- типа возвращаемого значения.
- Логика кванторов видимости операции аналогична логике кванторов видимости атрибутов.

Следующий пример показывает класс «Отчёт», описанный согласно только что рассмотренным правилам.

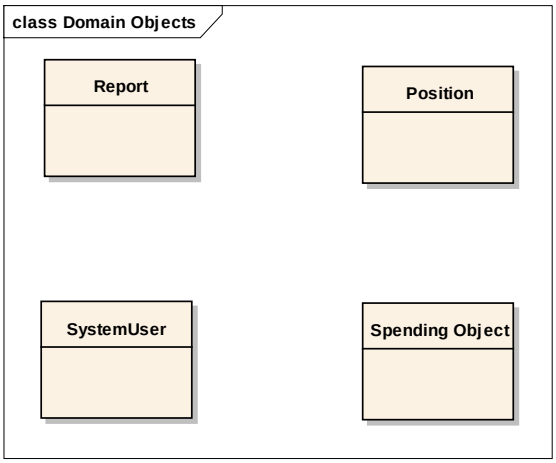


Расширяем модель, добавляя туда объект «Пользователь».



И снова расширяем модель, добавляя туда информацию о том, что отчёт

состоит из «Позиций», каждая из которых ссылается на «Трату» («10 карандашей по 7 рублей и 3 стирки по 2 рубля»).

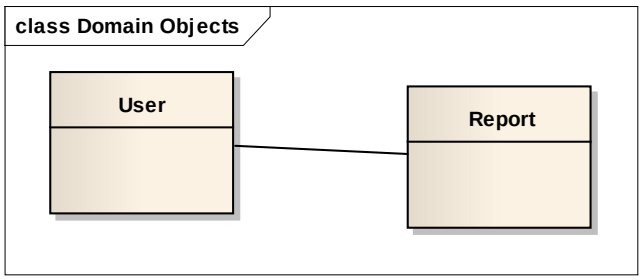


Связи

Ассоциация (association) показывает, что объекты одной сущности (класса) связаны с объектами другой сущности.

Двойные ассоциации (с двумя концами) представляются линией, соединяющей два классовых блока.

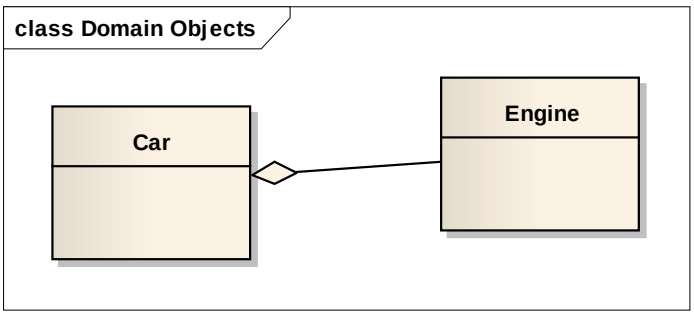
В представлении однонаправленной ассоциации добавляется стрелка, указывающая на направление ассоциации.



Агрегация (aggregation) – разновидность ассоциации при отношении между целым и его частями.

Одно отношение агрегации не может включать более двух классов (контейнер и содержимое).

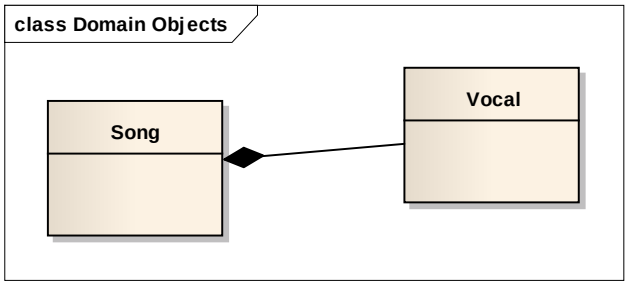
Графически агрегация представляется пустым ромбиком на блоке класса и линией, идущей от этого ромбика к содержащемуся классу.



Композиция (composition) – более строгий вариант агрегации (известна также как «агрегация по значению»).

Композиция имеет жёсткую зависимость времени существования экземпляров класса контейнера и экземпляров содержащихся классов.

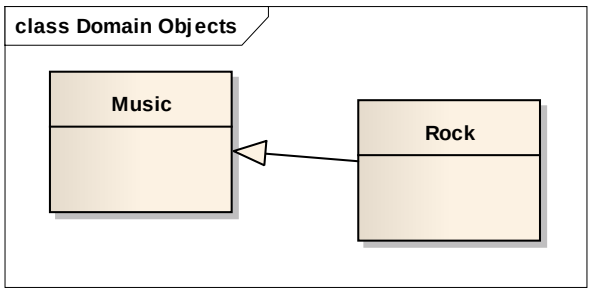
Если контейнер будет уничтожен, то всё его содержимое будет также уничтожено. Графически представляется как и агрегация, но с закрашенным ромбиком.



Обобщение (generalization) показывает, что один из двух связанных классов (подтип) является более частной формой другого (надтипа), который называется обобщением первого.

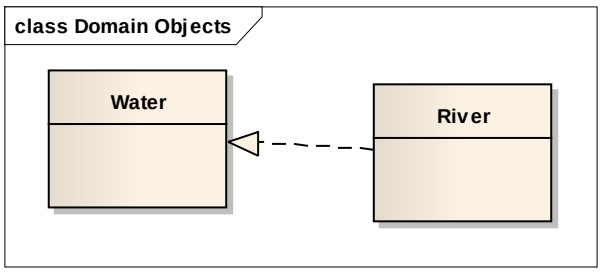
Это означает что любой экземпляр подтипа является также экземпляром надтипа.

Графически обобщение представляется линией с пустым треугольником у надтипа.



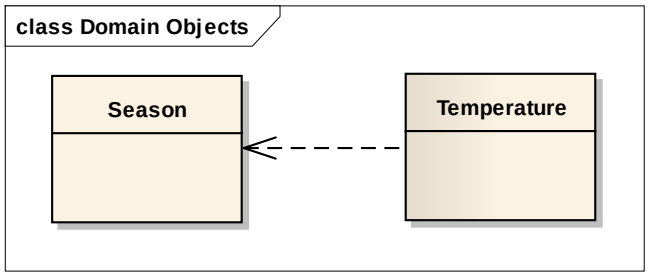
Реализация (realisation) – отношение между двумя элементами модели, в котором один элемент (клиент) реализует поведение, заданное другим (поставщиком).

Графически реализация представляется также как и генерализация, но с пунктирной линией.



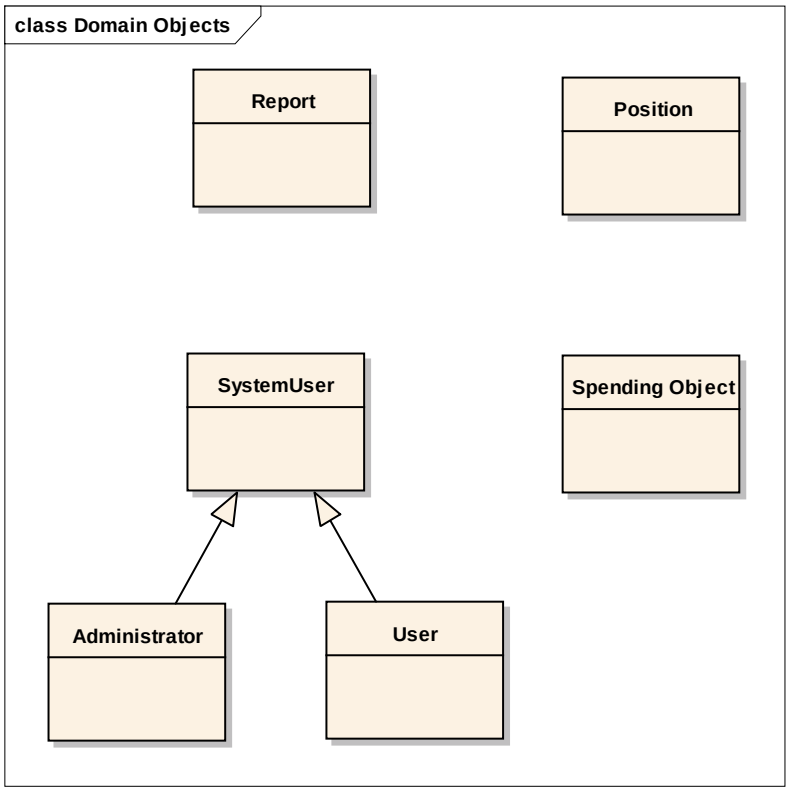
Зависимость (dependence) – отношение использования, при котором изменение в спецификации одного объекта влечёт за собой изменение другого объекта (причём обратное не обязательно).

Графически представляется пунктирной стрелкой, идущей от зависимого элемента к тому, от которого он зависит.

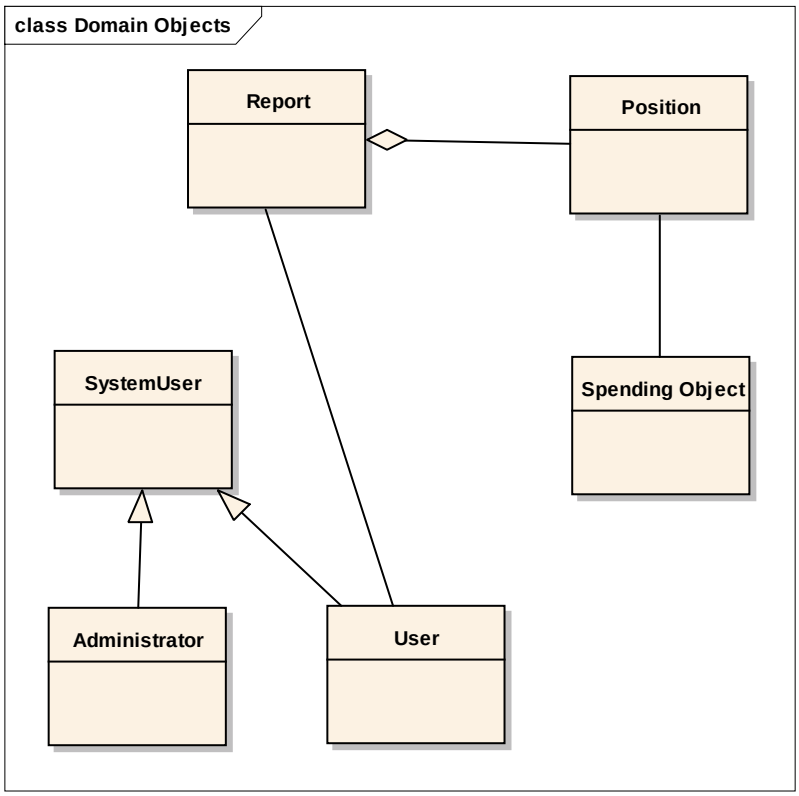


Продолжение примера с отчётом

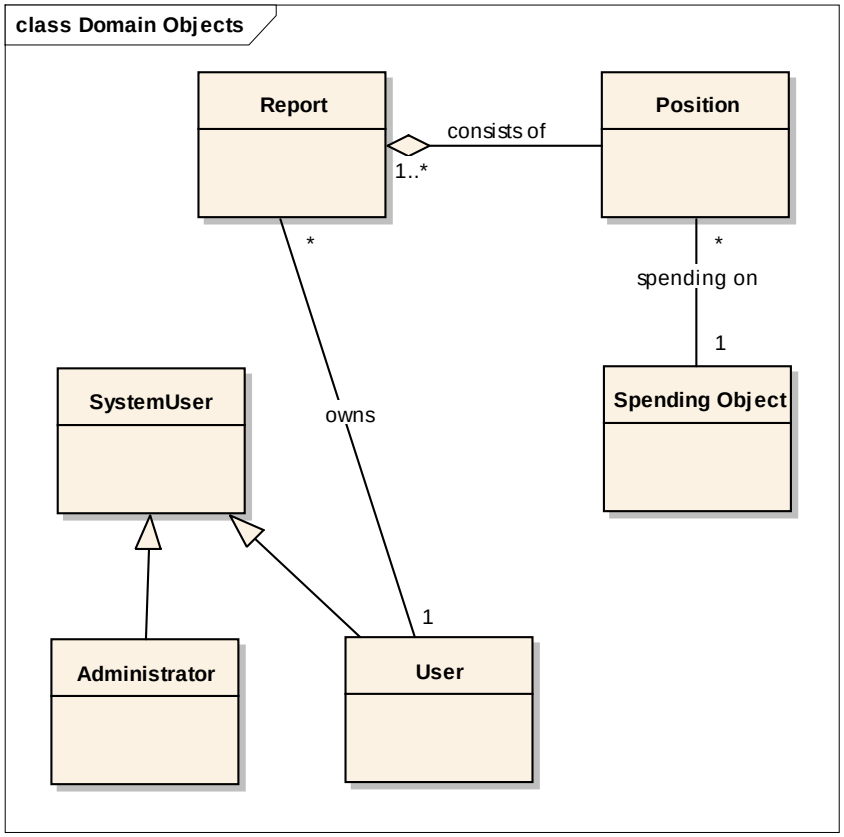
Записи о «Администраторе» и «Пользователе» расширяют понятие «Зарегистрированный пользователь».



Пользователь ассоциирован с его отчётом; отчёт содержит в себе позиции, которые ссылаются на объект траты.



Уточняем мощности связей и кратко описываем семантику связей.



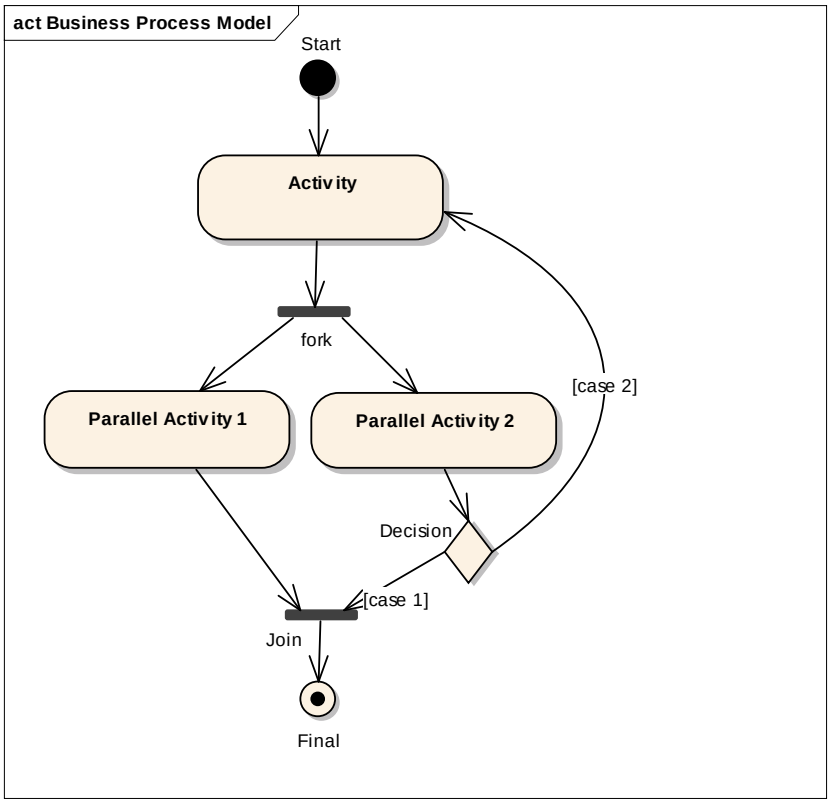
4. Диаграммы деятельности

Диаграмма деятельности (Activity diagram) – показывает декомпозицию некоторой деятельности на её составные части.

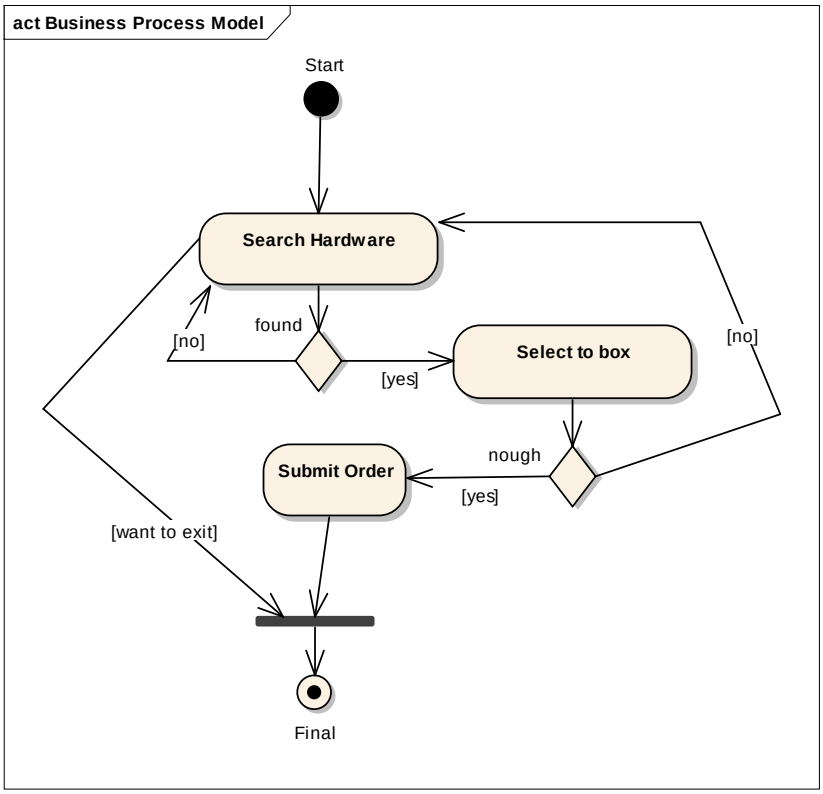
Диаграммы деятельности используются при моделировании бизнес-процессов, технологических процессов, последовательных и параллельных вычислений.

Аналогом диаграмм деятельности являются схемы алгоритмов.

Это просто обобщённый пример диаграммы деятельности:



Ещё один обобщённый пример диаграммы деятельности:



5. Диаграммы автоматов

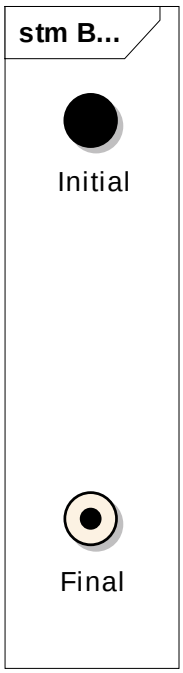
Диаграмма автомата (State Machine diagram) – представляет конечный автомат с простыми состояниями, переходами и композитными состояниями.

Конечный автомат (state machine) – спецификация последовательности состояний, через которые проходит объект или взаимодействие в ответ на события своей жизни, а также ответные действия объекта на эти события.

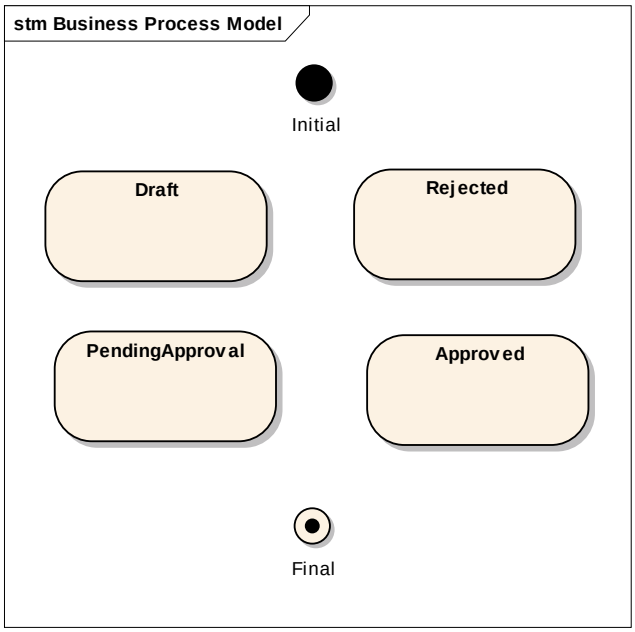
Главное предназначение этой диаграммы – описать возможные последовательности состояний и переходов, которые в совокупности характеризуют поведение элемента модели в течение его жизненного цикла.

Продолжаем пример на основе объекта «Отчёт».

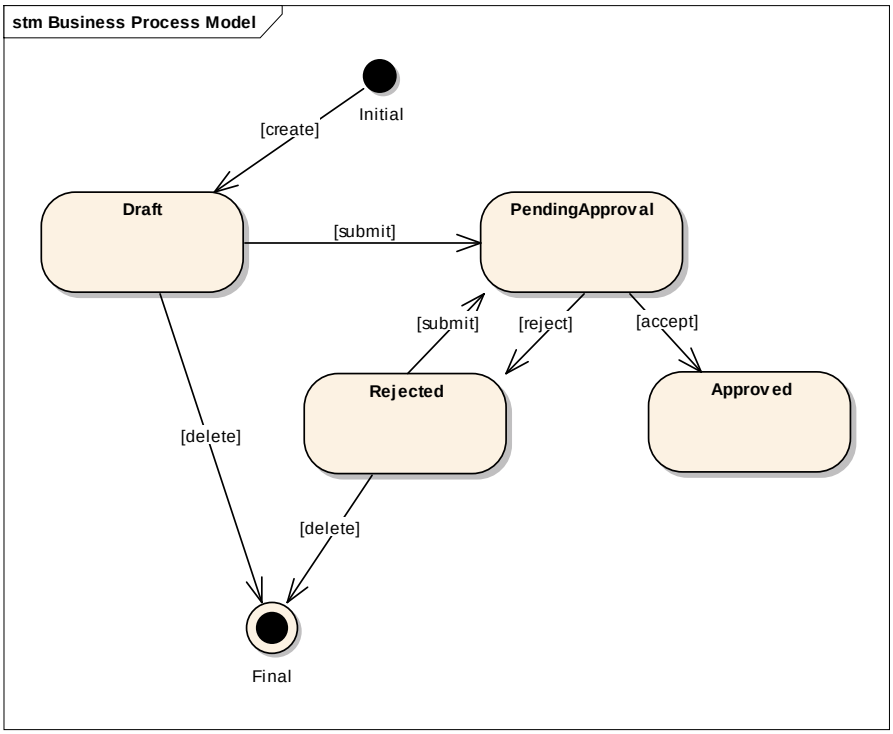
Рисуем точки начала и конца («ещё не существования» и «уже не существования»).



Перечисляем состояния отчёта.



Описываем переходы между состояниями.



6. Диаграммы последовательности

Диаграмма последовательности (Sequence diagram) – изображает упорядоченное во времени взаимодействие объектов.

Основными элементами диаграммы последовательности являются обозначения объектов (прямоугольники), вертикальные линии (lifeline),

отображающие течение времени при деятельности объекта, и стрелки, показывающие выполнение действий объектами.

Линия жизни объекта (object lifeline) изображается пунктирной вертикальной линией, ассоциированной с единственным объектом на диаграмме последовательности.

Линия жизни служит для обозначения периода времени, в течение которого объект существует в системе и может участвовать во всех её взаимодействиях.

Если объект существует в системе постоянно, то и его линия жизни должна продолжаться по всей плоскости диаграммы последовательности от самой верхней ее части до самой нижней.

В процессе функционирования объектно-ориентированных систем объекты могут находиться в активном состоянии или в состоянии пассивного ожидания сообщений от других объектов.

Чтобы явно выделить подобную активность объектов, в языке UML применяется специальное понятие, получившее название фокуса управления (focus of control). Фокус управления изображается в форме вытянутого узкого прямоугольника.

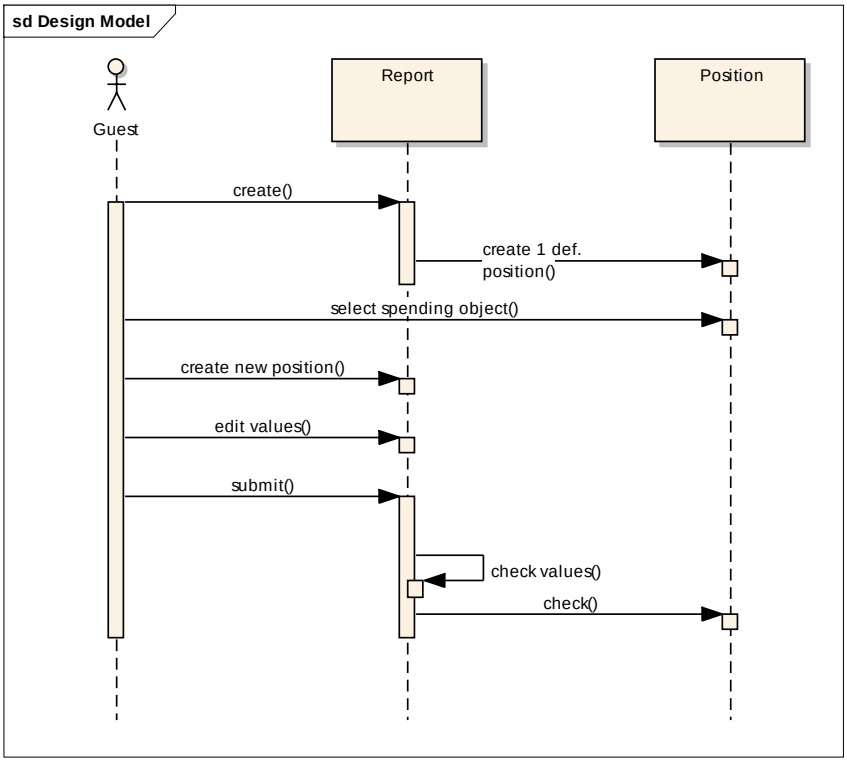
Каждое взаимодействие описывается совокупностью сообщений, которыми участвующие в нём объекты обмениваются между собой.

В этом смысле сообщение (message) представляет собой законченный фрагмент информации, который отправляется одним объектом другому.

При этом прием сообщения инициирует выполнение определённых действий, направленных на решение отдельной задачи тем объектом, которому это сообщение отправлено.

Сообщения могут быть синхронными и асинхронными.

Пример диаграммы последовательности:



7. Диаграммы развёртывания

Диаграмма развёртывания (Deployment diagram) – служит для моделирования работающих узлов (node) и артефактов, развёрнутых на них.

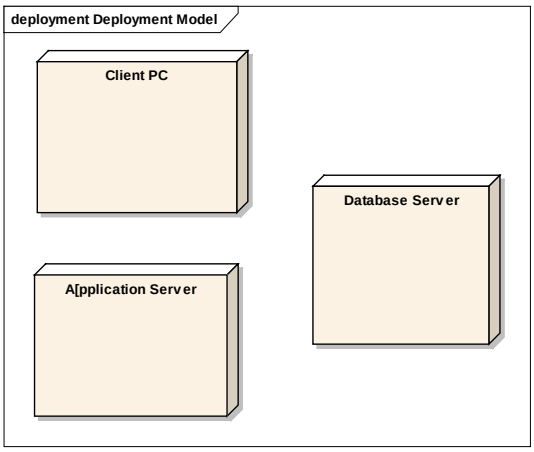
Физическое представление программной системы не может быть полным, если отсутствует информация о том, на какой платформе и на каких вычислительных средствах она реализована.

Диаграмма развёртывания показывает наличие физических соединений – маршрутов передачи информации между аппаратными устройствами, задействованными в реализации системы.

Узел (node) представляет собой физически существующий элемент системы, обладающий вычислительным ресурсом.

В качестве вычислительного ресурса узла может рассматриваться наличие памяти и/или процессора.

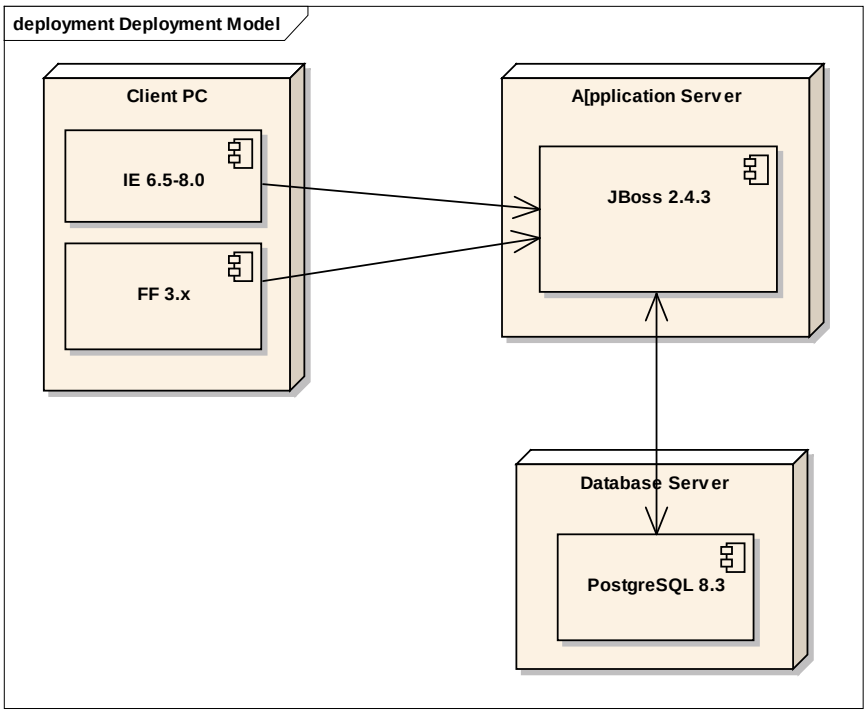
Графически на диаграмме развертывания узел изображается в форме трёхмерного куба.



Можно явно указать компоненты, которые размещаются на отдельном узле:

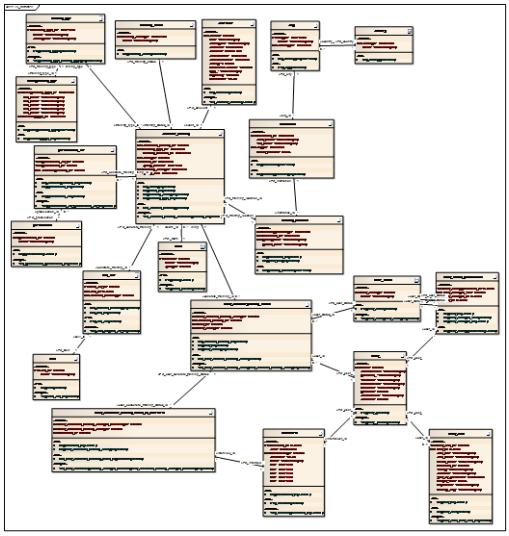
- разделить графический символ узла на две секции горизонтальной линией (разделяет имя узла и размещённые на нём компоненты);
- использовать узлы с вложенными изображениями компонентов (только исполняемых).

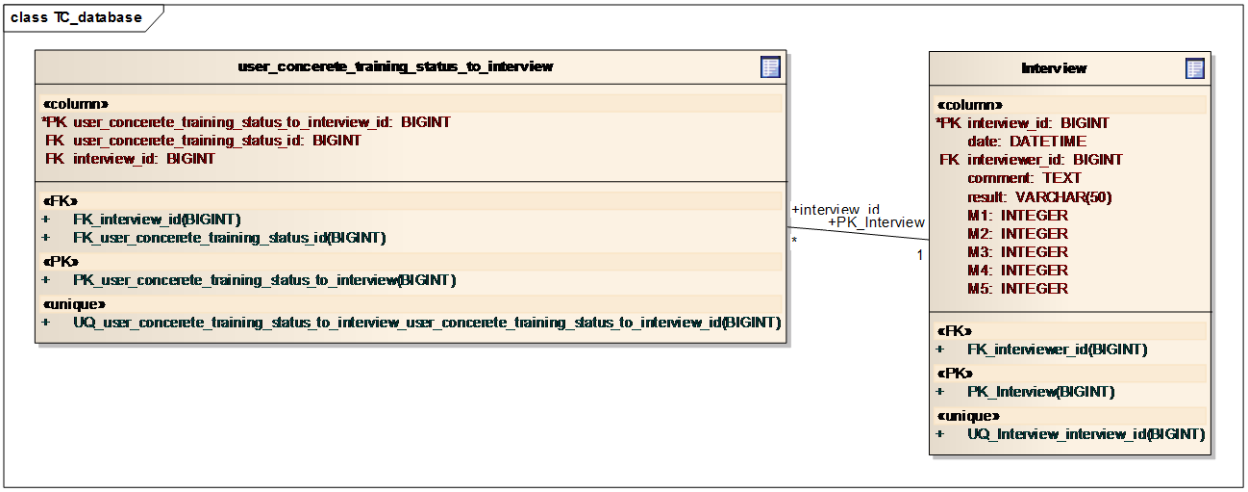
Соединения являются разновидностью ассоциации и изображаются отрезками линий, указывая на необходимость организации физического канала обмена информацией между узлами.



8. Описание баз данных с помощью UML

Использование UML-нотации позволяет достаточно эффективно описывать и модели реляционных баз данных, например:

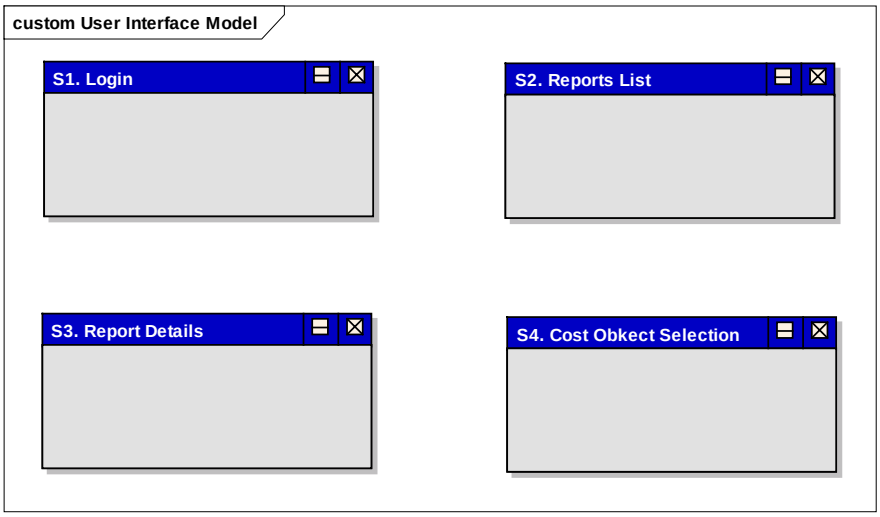




9. Полезное дополнение к UML: карты навигации

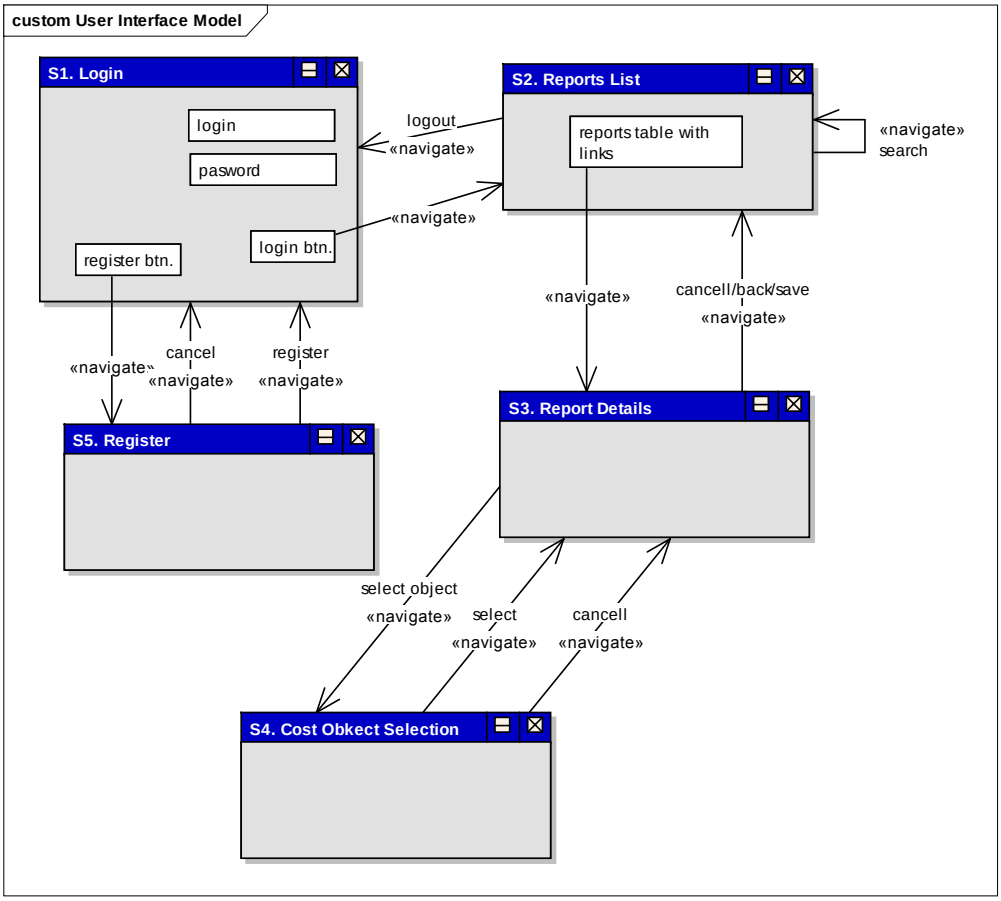
Карта навигации (Navigation Map) представляет собой структуру элементов интерфейса пользователя вместе с возможными путями перемещения по ней.

Выделим в системе экраны.



Обозначим переходы между экранами.

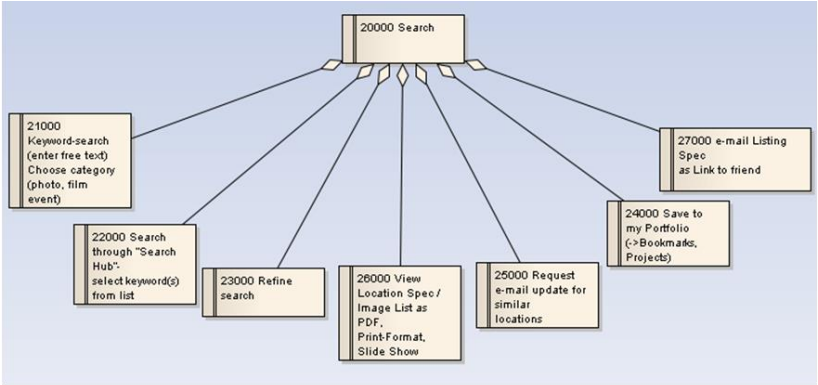
Эта диаграмма должна быть согласована с описанием вариантов использования.


custom User Interface Model
S1. Login
login
pasword
register btn.
login btn.
S2. Reports List
reports table with links
logout
«navigate»
«navigate»
«navigate»
search
«navigate»
cancell/back/save
«navigate»
«navigate»
cancel
«navigate»
register
«navigate»
S5. Register
S3. Report Details
select object
«navigate»
select
«navigate»
cancell
«navigate»
S4. Cost Obkect Selection

10. Применение UML в тестировании

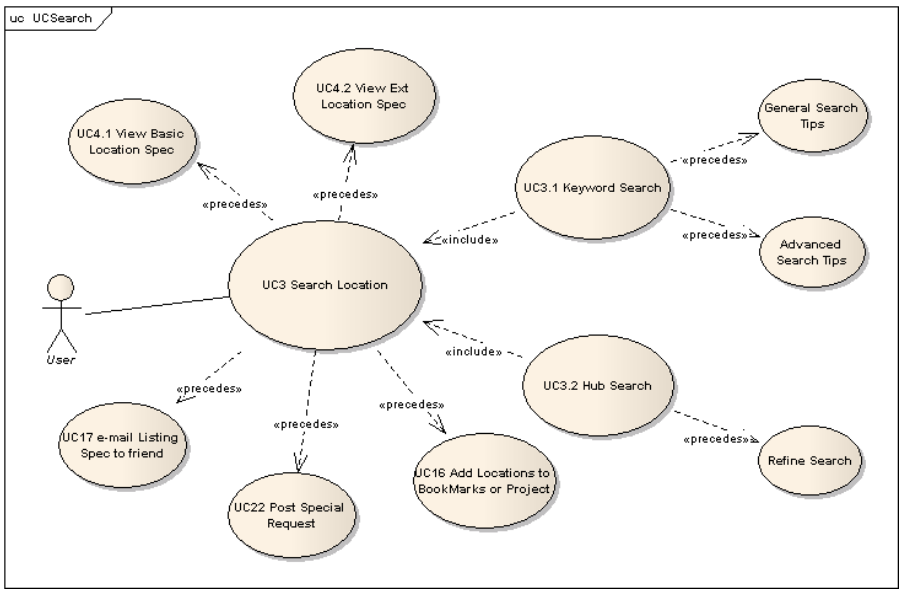
Следуем логике «Требования → варианты использования → навигация → тест-кейсы».

Визуализация требований (пока «вне UML») + описание вариантов использования:

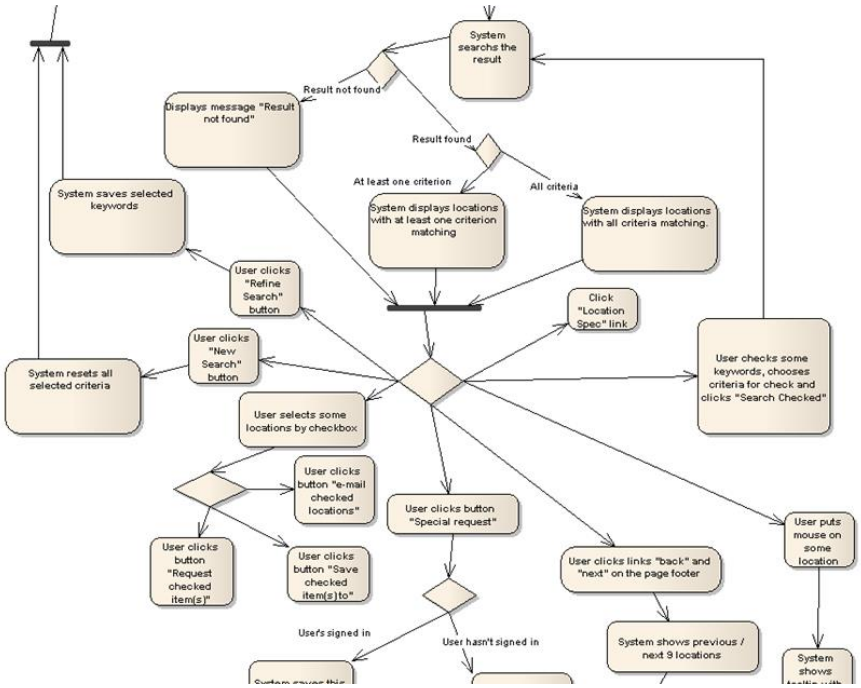


ID	Use Case	Actor	Functionality	Refer to Folder Document Page
…				
26 000	Search	Seeker	View Location Spec / Image List as PDF, Print-Format, Slide Show	Search Process1. Detailed Search Location Page 24, 26, 28
…				

Полноценное описание вариантов использования:



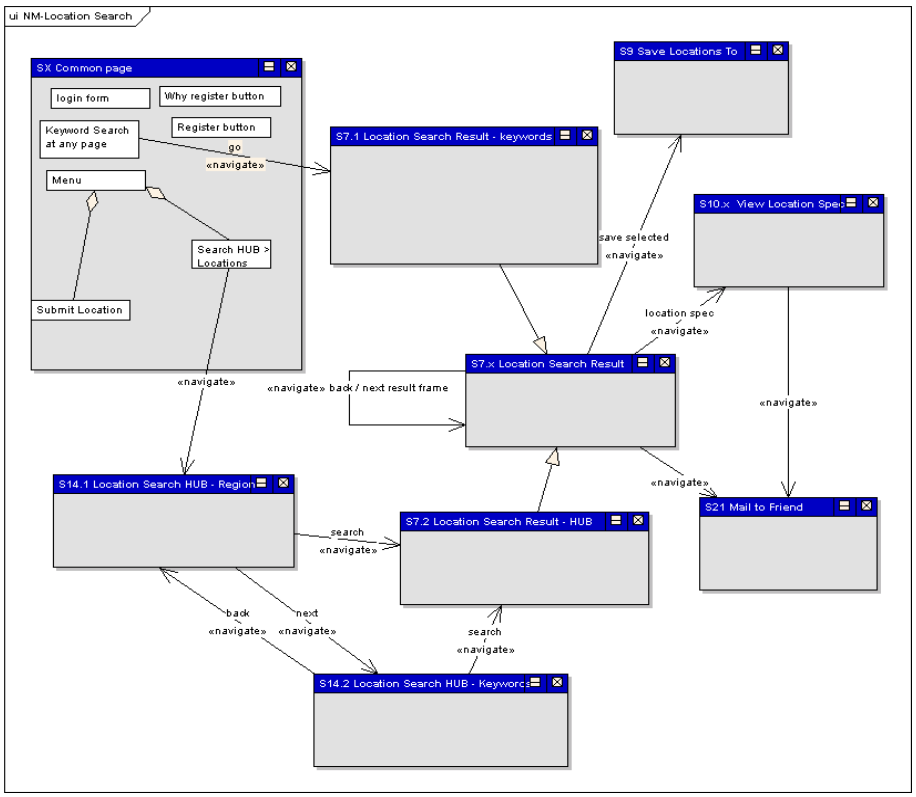
Описание жизненных циклов объектов:



Подробно расписанные варианты использования:

U.C. name:	View Basic Location Spec	Code:	*UC4.1*
Actors:	Anonymous User/Seeker/Host		
Precondition:	User hasn't requested/reserved/booked location. User is not the owner of location.		
	User clicks button "Location Spec"		
Post condition:	Location info is showed for user		
User		**System**	
		1. New page generation. Opened in separate window.	
		1a. If it is Location with Info & Images. User is not signed in. **Screen: S10.1.**	
		1b. If it is Managed Location. User is not signed in. **Screen: S10.2**	
		1c. If it is Location with Info & Images. User is signed in. **Screen: S10.3.**	
		1d. If it is Managed Location. User is not signed in. **Screen: S10.4**	
Extensions			
2a. User may click button "Print"		3. Go to **UC 4.2.2**	
2b. User may click button "View as PDF"		4. Go to **UC 4.2.1**	
2c. User may click link "show all location by same location provider"		5. Displays Search results page with locations by same location provider. **Screen S7.2**	
2d. User may click link "show all location in same area"		6. Displays Search results page with locations in same area. **Screen S7.2**	
2e. User may click link "show all location with matching keywords"		7. Displays Search results page with locations with matching keywords. **Screen S7.1**	
. User may click link "View images list"		8. Go to **UC 4.2.3**	
Notes			
On location spec 4 images with the highest rating are displayed			

Выявление экранов и навигации:



Теперь можно писать тесты (для каждого варианта использования на основании информации о сценариях, формах и т.п.):

No.	Use case code	Requirement No.	Module	Sub-Module/Screen	Test Description	Expected Results
6.1.17.	UC 4.1, 4.2	26 000		Location spec	Click link “show all location by same location provider” at Location spec page.	Search results page with locations by same location provider is displayed. [S7.2]
6.1.18.	UC 4.1, 4.2	26 000		Location spec	Click link “show all location in the same area" at Location spec page.	Search results page with locations in same area is displayed. [S7.2]
6.1.19.	UC 4.1, 4.2	26 000		Location spec	Click link “show all location with matching keywords” at Location spec page.	Search results page with locations with matching keywords is displayed. [S7.1]
6.1.20.	UC 4.2.2	26 000		Location spec	**Print format** Click "Print" button at Location spec page.	Location spec info is shown as optimized for printing page.
6.1.21.	UC 4.2.1	26 000		Location spec	**View as PDF** Click button “View as PDF” at Location spec page.	PDF with info available for this user type is generated.