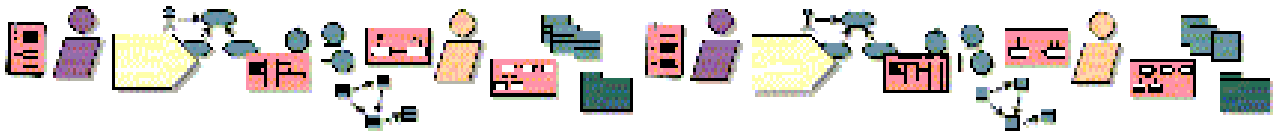


Поток работ «Требования» (окончание)

Артефакты интерфейса пользователя

Артефакты интерфейса пользователя занимают некоторое промежуточное положение между анализом и проектированием и управлением требованиями.

С одной стороны, эти артефакты содержат некоторые элементы проекта: подробное описание порядка взаимодействия пользователя с будущей системой, включающее рекомендации по применимости (иллюстрированные сценарии), наметки архитектуры интерфейса (граничные классы и связи между ними) и даже внешний вид диалоговых окон (прототипы).

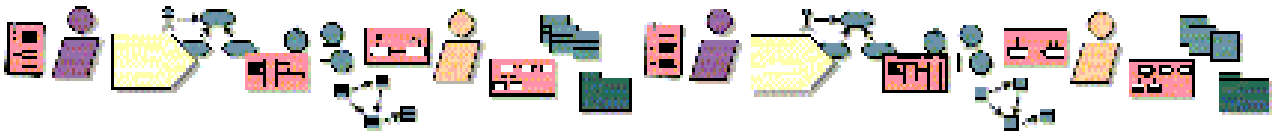
С другой стороны, именно при моделировании и прототипировании интерфейса важно возможно более тесно взаимодействовать с потенциальными пользователями системы. При этом применимость системы может исследоваться адресно, помогая раскрыть все предварительно неоткрытые требования и уточнить определения уже идентифицированных ранее.

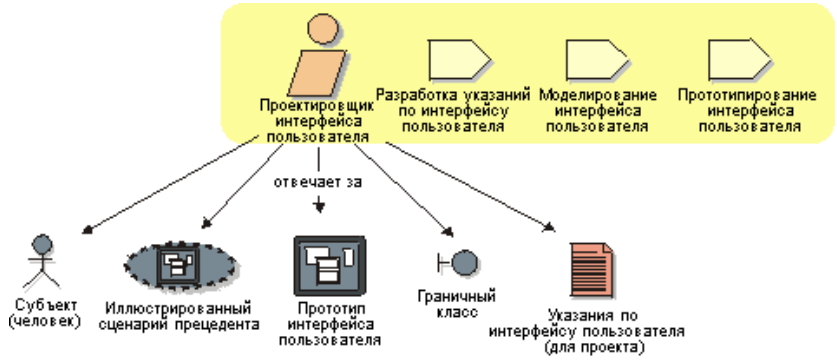
Создание и поддержка артефактов интерфейса пользователя выполняются в потоке работ Требования (см. иллюстрацию на стр. 11-6), часть Уточнение определения системы (см. стр. 11-16). Ответственность за эту работу возлагается на Проектировщика интерфейса пользователя.

Работник: Проектировщик интерфейса пользователя

Проектировщик интерфейса пользователя выполняет и координирует прототипирование и проектирование интерфейса пользователя. Сюда входит:

- фиксация требований к интерфейсу пользователя, включая требования применимости;
- формирование прототипов интерфейса пользователя;
- включение других совладельцев интерфейса пользователя, типа конечных пользователей, в обзор практичности и в сеансы тестирования удобства использования;
- рассмотрение и обеспечение соответствующей обратной связи при окончательном выполнении интерфейса пользователя (созданного другими разработчиками, то есть проектировщиками и конструкторами).





В этой главе мы будем рассматривать действия Моделирование и Прототипирование интерфейса пользователя. Действие Разработка указаний по интерфейсу пользователя мы сейчас рассматривать не будем, т.к. оно выполняется в потоке работ Среда и будет рассмотрено в свое время.

Обратите внимание, что проектировщик интерфейса пользователя не должен сам выполнять интерфейс. Вместо этого, он должен сосредоточиться и заниматься только проектированием и «формированием видения» интерфейса, потому что:

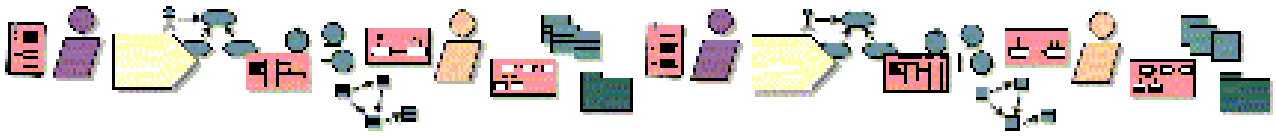
- Навыки, необходимые проектировщику интерфейса пользователя, часто должны быть усовершенствованы и оптимизированы для текущего проекта и вида приложения (с потенциально уникальными требованиями применимости), а это требует времени и сосредоточенности.
- Должен быть исключен риск «смешения зависимости», то есть проектировщик интерфейса пользователя не должен находиться под влиянием соображений выполнимости (в противоположность соображениям применимости).

Моделирование интерфейса пользователя

Целью этого действия является создание модели интерфейса пользователя, которая поддерживает обоснование его применимости.

Для каждого прецедента, приоритетного с точки зрения применения в текущей итерации, выполняются следующие шаги:

- Описание характеристик связанных субъектов
- Создание иллюстрированного сценария прецедента
- Описание потока событий иллюстрированного сценария
- Формулирование требования применимости к иллюстрированному сценариию прецедента
- Поиск необходимых граничных классов иллюстрированного сценария прецедента
- Описание взаимодействий между граничными объектами и субъектами



- Дополнение диаграммы иллюстрированного сценария прецедента
- Ссылка на прототип интерфейса пользователя из иллюстрированного сценария прецедента

Для идентифицированных граничных классов выполняются следующие шаги:

- Описание ответственности граничных классов
- Описание атрибутов граничных классов
- Описание связей между граничными классами
- Описание требований применимости к граничным классам
- Представление граничных классов на общих диаграммах классов
- Оценка результатов

Обратите внимание, что эти шаги представлены в логическом порядке, но Вам, вероятно, придется переставлять их или исполнять некоторых из них параллельно. Кроме того, некоторые шаги необязательны - это зависит от сложности конкретного интерфейса пользователя.

Прототипирование интерфейса пользователя

Целью этого действия является создание прототипа интерфейса пользователя.

Для каждого иллюстрированного сценария прецедента, который будет смоделирован в текущей итерации, выполняются следующие шаги:

- Проектирование прототипа интерфейса пользователя
- Выполнение прототипа интерфейса пользователя
- Установка обратной связи с прототипом интерфейса пользователя

Так же, как и в предыдущем действии, эти шаги перечислены в некотором логическом порядке, который может меняться.

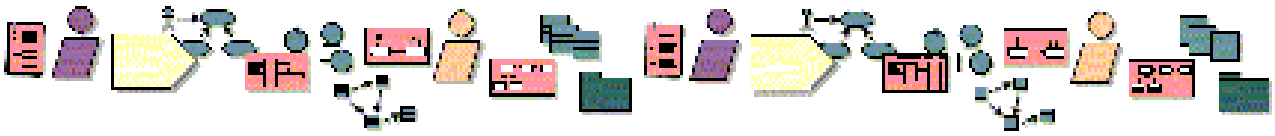
Сначала у Вас будет уходить много времени на проектирование прототипа; потом, когда Вы больше узнаете о нем, основное время будет уходить на выполнение и поддержку обратной связи.

Артефакт: Иллюстрированный сценарий прецедента

Краткий обзор

Иллюстрированный сценарий прецедента – это логическое и концептуальное описание того, как прецедент обеспечивается интерфейсом пользователя, включая взаимодействия между субъектом(ами) и системой.

Иллюстрированные сценарии прецедентов используют:



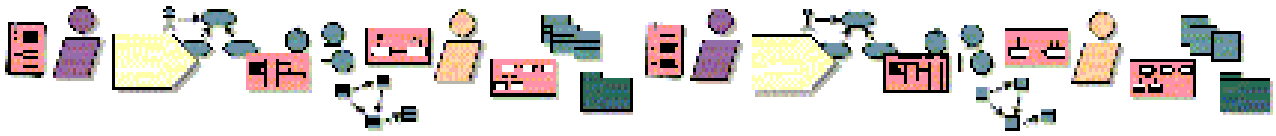
- **проектировщики интерфейса пользователя** при формировании модели интерфейса пользователя;
- **проектировщики** граничных объектов, участвующих в иллюстрированном сценарии прецедента, чтобы понять роли объектов в прецедентах и то, как эти объекты взаимодействуют. Они используют эту информацию при проектировании и выполнении граничных объектов (то есть при создании интерфейса пользователя);
- **те, кто проектируют следующую версию системы**, чтобы понять, как система выполняет поток событий в терминах граничных объектов. Например, изменение может затрагивать ограниченное число прецедентов, в этом случае проектировщики должны видеть реализацию их потоков событий;
- **тестировщики**, которые проверяют прецеденты системы;
- **руководитель** при планировании и отслеживании работ анализа и проектирования.

Толкование

Иллюстрированные сценарии прецедентов используются для выяснения и обсуждения требований интерфейса пользователя, включая требования применимости. Они обеспечивают высокоуровневое представление интерфейса пользователя, и разрабатываются намного быстрее, чем фактический интерфейс пользователя. Таким образом, иллюстрированные сценарии прецедентов могут использоваться для создания и обсуждения нескольких версий интерфейса пользователя прежде, чем он будет прототипироваться, разрабатываться и реализовываться.

Иллюстрированный сценарий прецедента описывается в терминах граничных классов и их статических и динамических связей, типа объединений, ассоциаций и ссылок. Каждый граничный класс – это, в свою очередь, высокоуровневое представление окна или подобной конструкции в интерфейсе пользователя. Преимущества такого подхода состоят в следующем:

- Он обеспечивает высокоуровневое представление статических связей между окнами, типа иерархии вместимостей окон, и других зависимостей между объектами в интерфейсе пользователя.
- Он обеспечивает высокоуровневое представление динамических связей между окнами, типа путей передвижения по окнам и других навигационных путей между объектами в интерфейсе пользователя.
- Он обеспечивает способ фиксации требований к каждому окну или подобной конструкции в интерфейсе пользователя, описывая соответствующий граничный класс. Описание каждого граничного класса определяет ответственности, атрибуты, связи и т.д., которые имеют прямое отображение в соответствующей конструкции интерфейса пользователя.
- Он обеспечивает трассировку к определенному прецеденту, и, как следствие, безшовную интеграцию с управляемым прецедентами подходом в разработке программного обеспечения. В результате интерфейс пользователя будет управляться прецедентами, которые должны быть



обеспечены системой, и ролями субъектов (пользователей) и их ожиданиями от этих прецедентов.

Иллюстрированный сценарий прецедента в модели анализа трассируется (взаимно-однозначно) к прецеденту в модели прецедентов.

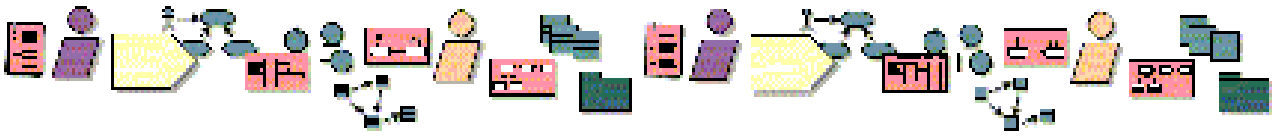
Работа над иллюстрированным сценарием прецедента обычно выполняется в следующей последовательности:

- Начальное описание потока событий иллюстрированного сценария
- Расширение описания аспектами применимости
- Создание диаграмм граничных классов
- Создание диаграмм взаимодействия граничных объектов
- Дополнение описания иллюстрированного сценария объектами из диаграмм
- Фиксация требований применимости
- Поддержка обратной связи с прототипом интерфейса пользователя

Начальное описание потока событий иллюстрированного сценария

Ниже представлены рекомендации того, как описывать поток событий иллюстрированного сценария:

- **Начните с разъяснения самого прецедента, а не его интерфейса пользователя.** Сохраните описание независимым от интерфейса пользователя, особенно, если прецедент неисследован. Впоследствии, когда прецедент будет понят, поток событий иллюстрированного сценария может быть расширен аспектами применимости интерфейса.
- **Сохраните описания действий краткими.** Описание не должно быть очень большим, потому что оно предназначено только для проектировщиков интерфейса пользователя. Описания действий как последовательности шагов обеспечивают лучший обзор, так как это делает описание короче. Например, когда Вы описываете, как прецедент обменивается данными с субъектом, постарайтесь сделать это кратко, но полно; передаваемые данные Вы можете перечислять в конце предложения в круглых скобках: «создать запись о человеке (имя, адрес, телефон)».
- **Избегайте последовательностей и режимов.** Субъекты (люди) часто могут выполнять взаимодействия с прецедентом в различных последовательностях, особенно в системах с интенсивным интерфейсом, где пользователь управляет системой. Последовательности часто



подразумевают режимы интерфейса пользователя. Вы должны избегать режимов, где это возможно. Вы должны определять последовательности только тогда, когда это обязательно в прецеденте. Например, что пользователь должен идентифицировать себя прежде, чем он может забрать деньги из банкомата, или что система должна показать счет пользователю прежде, чем пользователь сможет принять или отказаться от него.

- **Сохраняйте совместимость с прецедентом.** Так как иллюстрированный сценарий может описываться более или менее параллельно с соответствующим прецедентом, эти два артефакта должны сохранять совместимость и предоставлять обратную связь друг другу. В частности, поток событий иллюстрированного сценария должен быть совместим с потоком событий соответствующего прецедента. Обратите внимание, что это часто требует отлаженной связи между автором прецедента, ответственным за прецедент, и проектировщиком интерфейса пользователя, ответственным за иллюстрированный сценарий прецедента.

Пример:

Ниже приводится пример начального описания потока событий иллюстрированного сценария для прецедента Управление входящими сообщениями почты, прежде, чем это описание будет расширено аспектами применимости.

a) Прецедент начинается, когда пользователь почты требует отобразить сообщения почты и система отображает сообщения.

b) Пользователь почты может выполнить один или более из следующих шагов:

c) Упорядочить почтовые сообщения по отправителям или по темам.

d) Прочитать текст почтового сообщения.

e) Сохранить почтовое сообщение как файл.

f) Сохранить приложение к сообщению как файл.

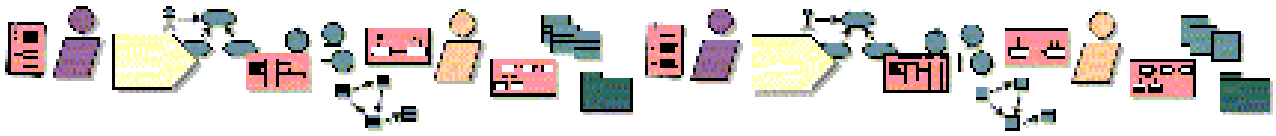
g) Прецедент заканчивается, когда пользователь почты подает команду о завершении работы с почтовыми сообщениями.

По возможности используйте текст пошагового описания соответствующего прецедента как шаблон для создания начального описания потока событий иллюстрированного сценария.

Расширение описания аспектами применимости

Ориентиры

Действительно пригодная для использования система не только помогает пользователю, автоматизируя простые и повторяющиеся задачи, но также показывает ориентиры, обычно (неявно) предоставляя необходимую информацию для задач, которые не могут быть автоматизированы. Такие ориентиры могут обеспечиваться, например, справочными надписями или контекстной экранной справкой.



Эта важная для проектирования интерфейса пользователя информация может быть предоставлена путем указания на потребность в таком ориентировании в определенных точках потока событий прецедента.

Проектировщик интерфейса пользователя должен просмотреть поток событий и рассмотреть на каждом шаге следующие проблемы:

- В каких ориентирах может нуждаться пользователь?
- Какие ориентиры могла бы обеспечить система?
- Что может быть представлено как ориентиры и что не должно?

Описание потока событий может идентифицировать основные функциональные возможности системы. Пометки о полезных ориентирах помогают идентифицировать «необязательные» функциональные возможности, которые не критичны для пользователя в смысле ограничения его способности выполнить свою работу, но они могут помочь ему выполнить эту работу, снабжая его полезной информацией. Таким образом, что-либо, что может помочь найти такие необязательные функциональные возможности, должно быть представлено как полезные ориентиры. Однако, Вы не должны представлять как полезные ориентиры то, что только идентифицирует функциональные возможности, которые мы все равно нашли бы, используя хорошую практику формирования интерфейса пользователя (например, не представляйте как полезные ориентиры то, что система должна осуществлять обратную связь с пользователем при его действиях или показывать пользователю различные опции, которые он может выбрать, и т.д.).

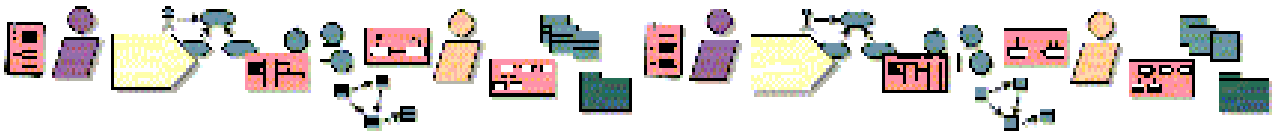
Полезные ориентиры можно использовать также и для того, чтобы сообщить о том, что не нужно показывать при формировании интерфейса, чтобы не загружать пользователя ненужной информацией.

Полезные ориентиры – это не «требования» в таком же смысле, как поток событий, это – скорее «пожелание» или «рекомендация». Когда Вы идентифицируете и описываете полезные ориентиры, Вы должны думать не в терминах того, что должна делать система, а в терминах того, что может понадобиться пользователю в дополнение к этому. Помните, что ориентиры не абсолютно необходимы, это просто средство улучшения применимости.

Пример:

Ниже приводится фрагмент описания потока событий иллюстрированного сценария для прецедента Управление входящими сообщениями почты, расширенного ориентирами (текст в пределах []).

а) Прецедент начинается, когда пользователь почты требует отобразить сообщения почты и система отображает сообщения. [Пользователь должен иметь способ различать новые, прочитанные и непрочитанные сообщения; кроме того, пользователь должен видеть отправителя, тему и приоритет каждого сообщения.]



Ориентиры часто необходимы в действиях, когда пользователь должен принять решение. В следующих ситуациях часто используют полезные ориентиры:

- Они нетривиальны для пользователя.
- Они затрагивают человеческий ресурс или деловое окружение системы (люди или деловое окружение системы обычно означают деловую сферу, пользователя и задачу, которую пытается выполнить пользователь).
- Они понадобятся после того, как прецедент закончится.

Средние значения атрибутов и объемы объектов

Часто важно фиксировать средние значения атрибутов и объемы объектов, которые должны управляться или представляться пользователю. Тогда интерфейс пользователя может быть соптимизирован для этих средних значений и объемов.

Пример:

Ниже приводится фрагмент описания потока событий иллюстрированного сценария для прецедента Управление входящими сообщениями почты, расширенное средними значениями атрибутов и объемами (текст в пределах {}).

> d) Прочитать текст почтового сообщения. {Текст сообщения в среднем содержит 100 символов.}

Средняя интенсивность использования

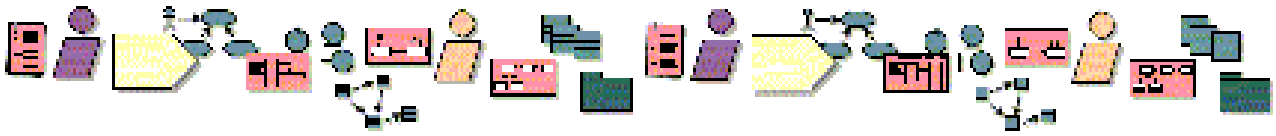
Средняя интенсивность использования фиксируется, чтобы найти действия, которые используются интенсивно в противоположность действиям, которые используются редко. В результате мы найдем в прецеденте и обычно используемые и редкие последовательности (потоки). Это – важная информация, которая поможет отсортировать по приоритетам, определить интенсивно используемые части интерфейса пользователя и сосредоточиться на его навигационных иерархиях (например, сформировать закладки или дополнительные инструментальные панели для выполнения наиболее частых действий).

Пример:

Ниже приводится фрагмент описания потока событий иллюстрированного сценария для прецедента Управление входящими сообщениями почты, расширенное средними значениями частоты использования (текст в пределах ()).

> d) Прочитать текст почтового сообщения. (Выполняется больше, чем в 75 % случаев.)
>
> e) Сохранить почтовое сообщение как файл. (Выполняется меньше, чем в 5 % случаев.)

Вывод из этого описания – это то, что шаг d) нуждается в особой поддержке интерфейса



пользователя.

Результирующее описание потока событий иллюстрированного сценария

Пример:

Ниже приводится пример финального описания потока событий иллюстрированного сценария для прецедента Управление входящими сообщениями почты, расширенное различными аспектами применимости.

a) Прецедент начинается, когда пользователь почты требует отобразить сообщения почты и система отображает сообщения. [Пользователь должен иметь способ различать новые, прочитанные и непрочитанные сообщения; кроме того, пользователь должен видеть отправителя, тему и приоритет каждого сообщения.] {Среднее число одновременно показываемых непрочитанных сообщений почты – около 100; в 90 % случаев строка описания содержания сообщения – меньше 40 символов.}

b) Пользователь почты может выполнить один или более из следующих шагов:

c) Упорядочить почтовые сообщения по отправителям или по темам. (Выполняется больше, чем в 60 % случаев.)

d) Прочитать текст почтового сообщения. {Текст сообщения в среднем содержит 100 символов.} (Выполняется больше, чем в 75 % случаев.)

e) Сохранить почтовое сообщение как файл. (Выполняется меньше, чем в 5 % случаев.)

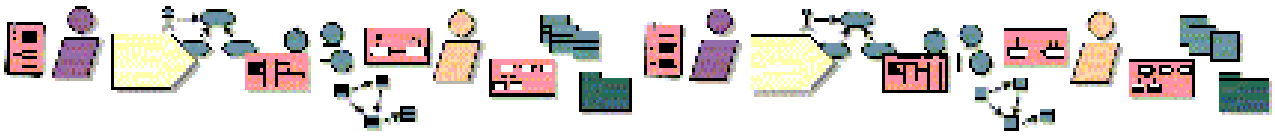
f) Сохранить приложение к сообщению как файл. [Пользователь должен быть способен видеть типы файлов приложений.] {В 95 % случаев сообщения имеют меньше двух приложений.}

g) Прецедент заканчивается, когда пользователь почты подает команду о завершении работы с почтовыми сообщениями.

Таким образом, основная идея потока событий иллюстрированного сценария состоит в том, чтобы расширить описание потока событий различными аспектами применимости; затем эта информация используется при проектировании пригодного для использования интерфейса пользователя. Заметьте также, что аспекты применимости, которые показаны в примере выше, могут заменяться или расширяться другими аспектами, в зависимости от потребностей специфического типа приложения или технологии использования интерфейса пользователя.

Создание диаграмм граничных классов

Иллюстрированный сценарий прецедента реализуется граничными классами и их взаимодействующими объектами. Для отображения граничных классов (вместе с их связями), участвующих в иллюстрированном сценарии прецедента, мы создаем диаграмму классов и рассматриваем ее как часть описания иллюстрированного сценария прецедента.



Пример:

Ниже приводится диаграмма классов, принадлежащая иллюстрированному сценарию прецедента Управление входящими сообщениями почты:

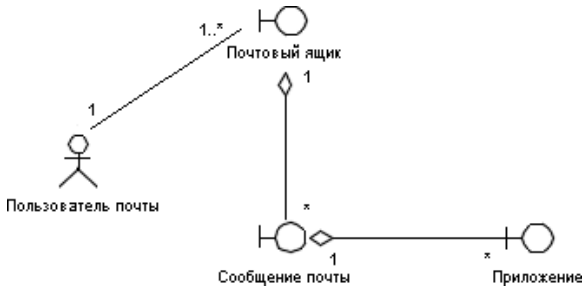


Диаграмма классов включает субъект Пользователь почты и граничные классы Почтовый ящик, Сообщение почты и Приложение. Мы объясняем здесь, что Пользователь почты взаимодействует со всеми объектами, содержащимися в иерархии объединений под Почтовым ящиком.

Создание диаграмм взаимодействия граничных объектов

Для отображения граничных объектов, участвующих в иллюстрированном сценарии прецедента, и их взаимодействия с пользователем мы используем диаграммы последовательности или сотрудничества. Это полезно для прецедентов со сложными последовательностями или потоками событий.

Пример:

Ниже приводится диаграмма сотрудничества, принадлежащая иллюстрированному сценарию прецедента Управление входящими сообщениями почты:

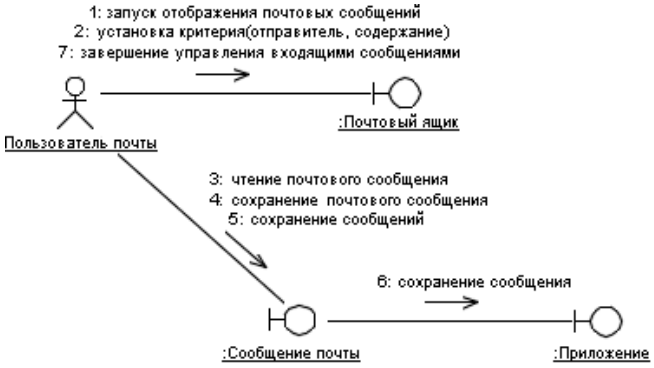
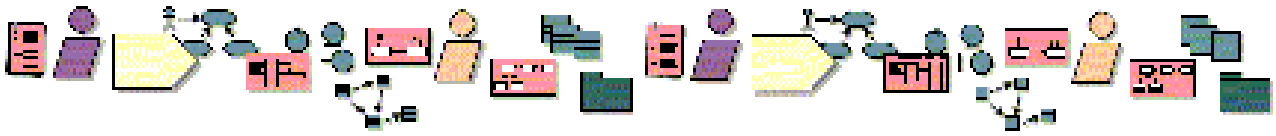


Диаграмма сотрудничества содержит субъект Пользователь почты и граничные объекты Почтовый ящик, Сообщение почты и Приложение, участвующие в иллюстрированном сценарии прецедента, реализующем прецедент Управление входящими сообщениями почты.



Дополнение описания иллюстрированного сценария объектами диаграмм

В случае необходимости, диаграмма иллюстрированного сценария прецедента может разъясняться с использованием потока событий иллюстрированного сценария как дополнительного текстового описания. Это можно сделать расширением описания за счет включения в иллюстрированный сценарий используемых граничных классов.

Пример:

Ниже приводится пример описания потока событий иллюстрированного сценария для прецедента Управление входящими сообщениями почты, расширенное граничными классами (текст в пределах «»).

a) Прецедент начинается, когда пользователь почты требует отобразить сообщения почты и система отображает сообщения. [Пользователь должен иметь способ различать новые, прочитанные и непрочитанные сообщения; кроме того, пользователь должен видеть отправителя, тему и приоритет каждого сообщения.] {Среднее число одновременно показываемых непрочитанных сообщений почты – около 100; в 90 % случаев строка описания содержания сообщения – меньше 40 символов.} «Почтовый ящик»

b) Пользователь почты может выполнить один или более из следующих шагов:

c) Упорядочить почтовые сообщения по отправителям или по темам. (Выполняется больше, чем в 60 % случаев.) «Почтовый ящик»

d) Прочитать текст почтового сообщения. {Текст сообщения в среднем содержит 100 символов.} (Выполняется больше, чем в 75 % случаев.) «Сообщение почты»

e) Сохранить почтовое сообщение как файл. (Выполняется меньше, чем в 5 % случаев.) «Сообщение почты»

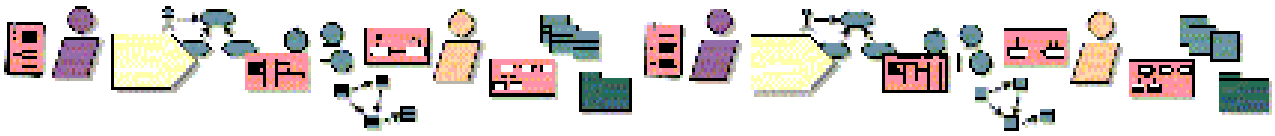
f) Сохранить приложение к сообщению как файл. [Пользователь должен быть способен видеть типы файлов приложений.] {В 95 % случаев сообщения имеют меньше двух приложений.} «Сообщение почты» «Приложение»

g) Прецедент заканчивается, когда пользователь почты подает команду о завершении работы с почтовыми сообщениями. «Почтовый ящик»

Фиксация требований применимости

Требования применимости могут определять уровень применимости интерфейса пользователя. Эти требования могут быть найдены, например, в артефакте Дополнительная спецификация. Требования применимости должны устанавливать не то, что, как Вы полагаете, должна делать система, а самый низкий уровень применимости, которого должна достичь система, чтобы использоваться.

То, чего должна достичь система, чтобы использоваться, зависит главным образом от того, что является альтернативой к использованию системы. Разумно требовать, чтобы система была



существенно более пригодна для использования, чем альтернативы. В качестве альтернатив можно предложить использовать:

- Существующие ручные процедуры.
- Наследуемую систему(ы).
- Конкурирующее изделие.
- Более раннюю версию(и) системы.

Требования применимости могут исходить также из потребности экономно внедрять новую систему: если заказчик должен заплатить $3 000 000 за новую систему, он может предложить наложить требования применимости, которые подразумевают, что он сохранит $1 000 000 ежегодно из-за уменьшенной рабочей нагрузки на свой персонал.

Требования применимости в иллюстрированном сценарии прецедента обычно определяют:

- Максимальное время выполнения – как много времени должно потребоваться обученному пользователю, чтобы выполнить обычный сценарий прецедента.
- Максимальная норма ошибки – сколько ошибок в среднем совершит обученный пользователь при выполнении обычного сценария прецедента.

Единственные допустимые ошибки – это те, которые не являются неисправимыми, и не будут оказывать отрицательного воздействия на организацию, типа потерь в бизнесе или нанесения ущерба аппаратным средствам. Если единственное последствие ошибки – то, что она требует времени для ее идентификации, это затронет измеренное время выполнения.

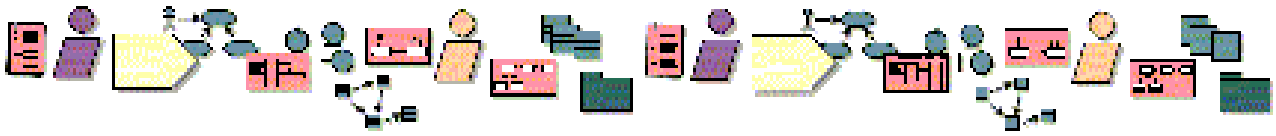
Время обучения должно быть измерено как время, которое нужно затратить прежде, чем пользователь сможет выполнять сценарий быстрее, чем указанное максимальное время выполнения.

Обратите внимание, что требования применимости не должны работать как цель, то есть, верхний предел. Требования применимости должны определять самую низкую приемлемую абсолютную применимость системы. Таким образом, Вы не обязательно должны прекратить улучшать применимость, когда требования применимости выполнены.

Пример:

Ниже приводится пример описания требований применимости для прецедента Управление входящими сообщениями почты, как оно представлено в соответствующем иллюстрированном сценарии прецедента.

- Пользователь почты должен быть способен упорядочить почтовые сообщения одним щелчком мыши.
- Пользователь почты должен быть способен скроллировать тексты почтовых сообщений нажатием одной кнопки клавиатуры.



- Пользователь почты не должен получать сигнал о входящем сообщении почты при чтении существующего сообщения.

Поддержка обратной связи с прототипом интерфейса пользователя

Для дальнейшего разъяснения иллюстрированного сценария прецедента последний может ссылаться на элементы (например, окна) прототипа интерфейса пользователя, соответствующего участвующим в нем граничным классам. Это может быть полезно, если части прототипа интерфейса пользователя уже существуют и иллюстрированный сценарий описан.

Синхронизация

Иллюстрированные сценарии прецедентов производятся сразу, как только соответствующие прецеденты будут отсортированы по приоритетам, чтобы иметь возможность рассмотреть их с позиций применимости. Иллюстрированный сценарий прецедента создается до моделирования и реализации интерфейса пользователя, то есть и в потоке работ Требования и в потоке работ Анализ и проектирование.

Создание и уточнение артефакта Иллюстрированный сценарий прецедента происходит в результате действия Моделирование интерфейса пользователя.

Этот артефакт предоставляет исходную информацию для действия Прототипирование интерфейса пользователя.

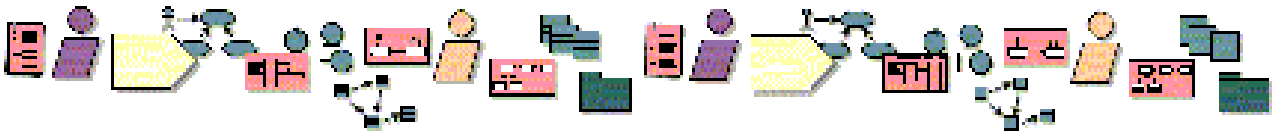
Проектировщик интерфейса пользователя отвечает за целостность иллюстрированного сценария прецедента и гарантирует, что:

- описание иллюстрированного сценария потока событий читабельно и соответствует цели;
- диаграммы, описывающие иллюстрированный сценарий прецедента, понятны и соответствуют цели;
- требования применимости читабельны и соответствуют цели; они правильно фиксируют требования применимости соответствующего прецедента в модели прецедентов;
- трассировка зависимости к соответствующему прецеденту в модели прецедентов правильна;
- связи (связи-ассоциации, связи включения и расширения) соответствующего прецедента в модели прецедентов в пределах иллюстрированного сценария прецедента обработаны правильно.

Очень полезно, чтобы проектировщик интерфейса, ответственный за иллюстрированный сценарий прецедента, был также ответственным за граничные классы и связи, которые используются в иллюстрированном сценарии.

Использование

Иллюстрированные сценарии прецедентов используются, прежде всего, в начальных стадиях разработки, прежде, чем интерфейс пользователя будет смоделирован, как инструмент



«рассуждения», чтобы зафиксировать требования к интерфейсу пользователя.

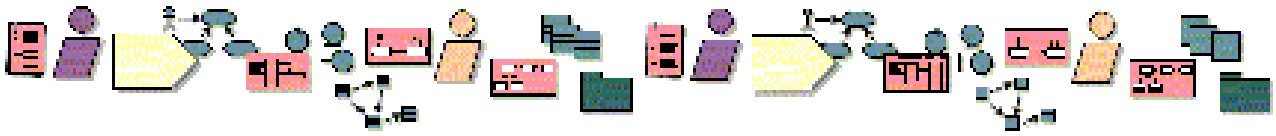
Иллюстрированные сценарии прецедента часто рассматриваются как переходные артефакты, только как проект, и могут не поддерживаться после прототипирования быстродействия или выполнения интерфейса пользователя. Однако в некоторых случаях может быть полезно поддерживать иллюстрированные сценарии прецедентов в течение ряда итераций, например, если имеются сложные требования, представленные в интерфейсе пользователя, которые требуют времени (более, чем несколько итераций), чтобы стать понятными.

Иллюстрированные сценарии прецедентов должны описываться с учетом того, что читателями этих описаний будут только проектировщики прецедента, так как иллюстрированные сценарии прецедента концептуальны и высокоуровневы, и могут оказаться непонятными другим читателям. Вместо этого, рассматривать и обсуждать, в том числе с другими совладельцами типа конечных пользователей, нужно их конкретные проявления (то есть непосредственно интерфейс пользователя или его прототип). Однако, иллюстрированные сценарии прецедентов могут использоваться проектировщиком интерфейса пользователя для ссылок при испытаниях использования, чтобы сфокусировать внимание на некоторых аспектах тестирования (например, на сложных последовательностях взаимодействий).

Компромисс в вышеуказанной рекомендации: если разработать иллюстрированные сценарии существенно дешевле, чем фактический прототип интерфейса пользователя, может быть разумно воспользоваться их наличием и рассмотреть иллюстрированные сценарии прежде, чем реализовывать прототип. Для этого важно, чтобы описания иллюстрированных сценариев были ясными и самодостаточными так, чтобы они могли быть поняты пользователями; создание таких описаний может потребовать существенных ресурсов разработки.

В случаях, когда в потоке работ требований разработаны иллюстрированные сценарии прецедентов и создан прототип интерфейса пользователя, соответствующие граничные классы – это хорошая исходная информация для действий анализа. Однако иногда возникает потребность создавать иллюстрированные сценарии прецедента в течение анализа. Это делается для увязывания элементов проекта интерфейса пользователя с выполняемыми действиями, потому что:

- Некоторые проекты не создают прототип интерфейса пользователя; вместо этого они проектируют и выполняют непосредственно интерфейс пользователя без прототипа.
- Некоторые проекты создают прототип интерфейса пользователя, но только для небольшого числа прецедентов. Для остальной части прецедентов интерфейс пользователя не прототипируется.



Артефакт: Граничный класс

Краткий обзор

Граничный класс моделирует взаимодействие между одним или несколькими субъектами и системой.

Граничный класс представляет интерфейс между системой и некоторой сущностью вне системы: человеком или другой системой. Его назначение – обеспечить обмен информацией с окружающим миром и защитить систему от изменений в ее среде.

Толкование

Граничные классы используются для моделирования взаимодействия между системой и ее средой, то есть, ее субъектами. Граничные классы фиксируют следующие аспекты взаимодействия:

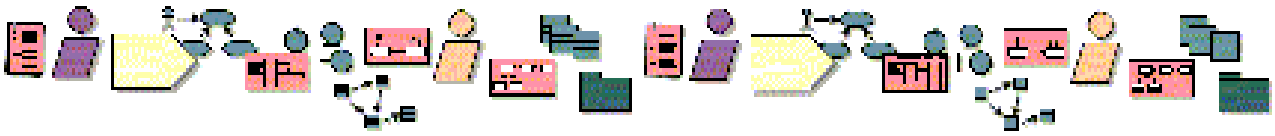
- согласование поведения субъекта с «внутренней организацией» системы;
- прием сообщений от субъекта системе, например, информации или запросов;
- вывод сообщений от системы субъекту, например, накопленную информацию или произведенные результаты.

Таким образом, граничные классы могут использоваться для фиксации требований к интерфейсу пользователя. Создание объектной модели пользовательского интерфейса заслуживает большого внимания, так как многие из создаваемых сегодня интерфейсов пользователя объектно-ориентированы, и наблюдается тенденция к еще большей объектной ориентации из-за ее выгод в части естественного и эффективного использования. Хотя многие из лучших проектных решений интерфейса пользователя были приняты непосредственно в ходе прототипирования и графической разработки, рассуждение относительно структуры и требований применимости к интерфейсу пользователя естественно делать в терминах объектной модели.

Использование граничных классов при моделировании интерфейса пользователя

Интерфейс пользователя, базирующийся на работе с окнами, может быть смоделирован граничными классами; для его описания на высоком уровне используйте следующие рекомендации:

- Используйте один граничный класс, чтобы представить одно главное окно.
- Используйте один граничный класс для представления каждого необходимого логического объекта (типа), управляемого субъектом в пределах главного окна; обратите внимание, что каждый такой объект также часто имеет свои собственные вторичные окна, типа окон свойств.
- Используйте ответственности граничных классов, чтобы описать необходимые операции для соответствующих окон и объектов.
- Используйте атрибуты граничных классов, чтобы описать свойства соответствующих окон и объектов.



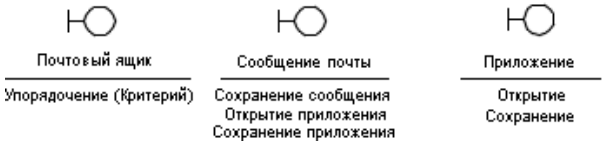
- Используйте связи между граничными классами, чтобы представить связи типа иерархии объединений и навигационных путей соответствующих окон и объектов.
- Используйте специальные требования к граничным классам, чтобы зафиксировать требования применимости и другие типы требований, как требования проектирования и выполнения к соответствующим окнам и объектам.

Примеры граничных классов

Следующие примеры можно привести для почтовой прикладной программы:

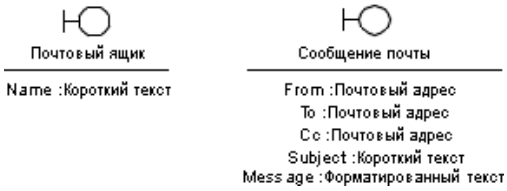
- Граничные классы:

- Ответственности граничных классов:



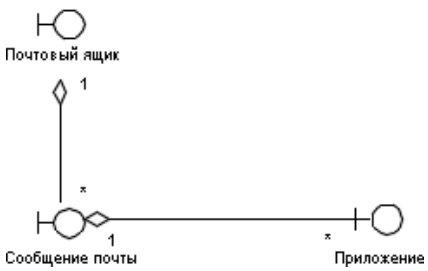
Ответственность Упорядочение (Критерий) состоит в том, чтобы упорядочить почтовые сообщения в окне почты согласно различным критериям, типа отправителя, содержания, времени получения и т.д.

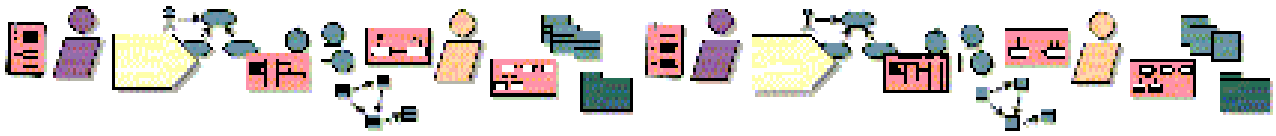
- Атрибуты классов:



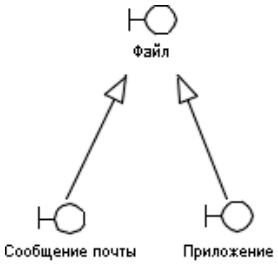
Обратите внимание на то, что типы атрибутов представлены здесь на концептуальном уровне; в фактическом интерфейсе пользователя они могут быть реализованы различными единицами измерения, типа сантиметров, количества символов и т.д..

- Связи объединения:





- Связи обобщения:



Примеры специальных требований к граничным классам

Специальными требованиями к граничным классам могут быть:

- Требования применимости, такие как квалификация пользователя, время обучения и вероятнось ошибки, связанные с граничным классом.
- Не функциональные требования, которые установлены для класса, но которые будут выполнены, когда соответствующий интерфейс пользователя будет разработан и реализован. Примером такого требования является то, что граничный класс должен быть реализован с использованием определенного компонента, типа управления ActiveX.

Требованием применимости для класса Сообщение почты могло бы быть:

- Пользователь почты не должен получать сигнал о входящем сообщении почты при чтении существующего сообщения.

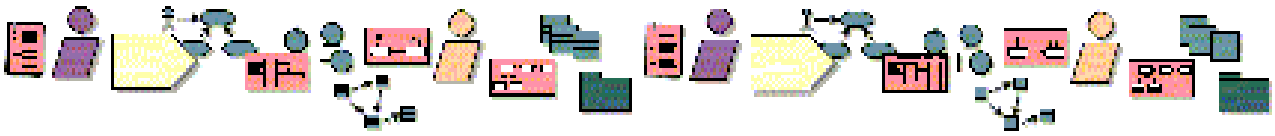
Требованием применимости для класса Почтовое окно могло бы быть:

- Пользователь почты должен иметь возможность упорядочить сообщения почты одиночным щелчком мыши.

Другие требования применимости имеют отношение к значениям атрибутов и размерам объектов. Они выставляются потому, что интерфейс пользователя часто должен обеспечивать читаемость определенного диапазона значений атрибута или пригодное для использования управление объектом, размеры которого находятся в определенном диапазоне. Тогда этот диапазон должен представляться как нормальный случай и обеспечиваться на 90 %. Интерфейс пользователя, в свою очередь, должен покрывать все возможные значения, включая превышающие диапазон, но может быть оптимизирован для значений в нормальном диапазоне.

Требованием применимости относительно значений атрибутов класса Сообщение почты могло бы быть:

- Тема сообщения должна быть способна содержать 40 символов без дефисов.



Связь субъектов и граничных классов

Так как граничные классы моделируют взаимодействия между субъектами и системой, мы поясняем это, сопоставляя каждый субъект с граничными классами, обрабатывающими взаимодействие с субъектом.

Для прикладной программы почты мы идентифицировали субъект Пользователь почты, который взаимодействует с Почтовыми ящиками.

В этом случае мы также обеспечиваем понимание из контекста того, что Пользователь почты взаимодействует со всеми объектами, содержащимися в иерархии объединений Почтового ящика.

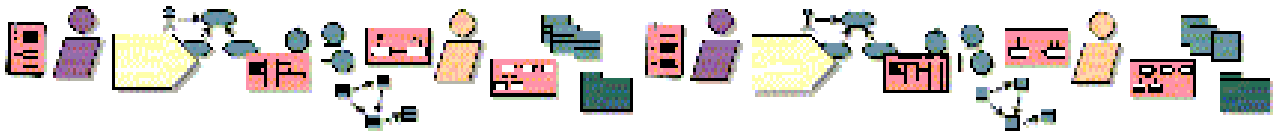
Критерии хорошего качества граничных классов и их связей

Проблема многих интерфейсов пользователя состоит в том, что они имеют слишком много окон и путей перемещения между окнами, которые становятся слишком длинными. Помимо самих дополнительных бесполезных действий, слишком длинные пути перемещения между окнами увеличивают вероятность того, что пользователь «потеряется» в системе. Идеально, если все окна можно открыть из одного первичного окна, часто называемого главным (main) окном. Тогда Вы имеете максимальную длину пути, равную двум шагам. Избегайте путей, имеющих длину более трех шагов.

Таким образом, при формировании интерфейса пользователя важно иметь для субъекта только одно главное окно, если возможно, содержащее все объекты, к которым должен обращаться субъект. Если это невозможно, окно должно обеспечить переход ко всем объектам, к которым должен обращаться субъект. Важно, что субъект проводит значительную часть своего «полезного времени», взаимодействуя с этим главным окном.

В результате эти соображения имеют следующее влияние на объектные модели интерфейса пользователя:

- Каждый субъект должен, если возможно, быть связан с только одним граничным классом (объединением), представляющим главное окно.
- Все граничные классы, используемые одним и тем же субъектом, должны быть связаны некоторым способом, часто через иерархию объединений, начинающуюся с граничного класса, представляющего главное окно.
- Иерархии объединений должны быть широкими и неглубокими, насколько это возможно.
- Граничные классы должны, как правило, быть частями одной (и только одной) иерархии объединений, так, чтобы субъект знал, где их найти.



- Нужно стремиться уменьшить насколько возможно количество иерархий объединения, потому что они усложняют субъектам понимание модели интерфейса.
- Если Вы имеете ассоциацию с множественностью один, и связанные классы – это части различных иерархий объединения, Вы должны рассмотреть замену ассоциации на объединение, если эти две иерархии могут быть объединены. Заметьте, однако, что это может привести к выбору между одной глубокой иерархией объединения и двумя более мелкими.

Критерии хорошего качества, представленные выше, находятся на весьма детальном уровне. Для более высокого уровня можно перечислить следующие критерии, которые не показывают так много деталей, но, по крайней мере, говорят о роли граничных классов:

- Граничные классы должны способствовать хорошей применимости системы.
- Граничные классы дожны сохранять высоко концептуальный уровень, насколько это возможно.
- Граничные классы должны обеспечить логичную модульность взаимодействий между системой и субъектом.
- Граничные классы должны быть единственными объектами, которые могут измениться, если субъекты изменяют способ, которым они воздействуют на систему.
- Граничные классы должны быть единственными объектами, которые могут измениться, если система изменяет способ, которым она обеспечивает вывод для субъектов.
- Граничные классы должны «знать» требования от других типов объекта (типа объектов контроллеров и сущностей) так, чтобы они могли быть реализованы и были пригодными для использования и эффективными «внутри системы».

Синхронизация

Граничные классы, относящиеся к применимости системы, идентифицируются и описываются в течение стадий Начало и/или Уточнение прежде, чем будет смоделирован, спроектирован и реализован интерфейс пользователя.

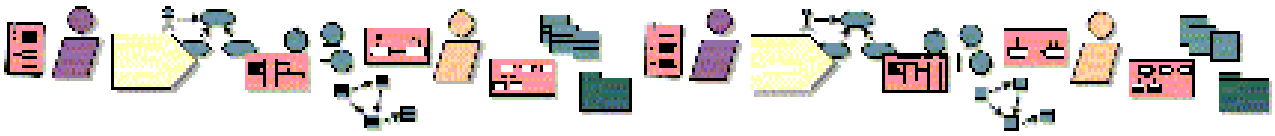
Создание и сопровождение артефактов Граничный класс происходит в действии Моделирование интерфейса пользователя.

Артефакт Граничный класс предоставляет исходную информацию для следующих действий:

- Прототипирование интерфейса пользователя
- Анализ прецедента

Проектировщик интерфейса пользователя или объектный аналитик ответственны за целостность граничного класса, гарантируя что:

- Класс отвечает требованиям, предъявленным в артефактах Реализация прецедента и



Иллюстрированный сценарий прецедента, в которых этот класс участвует.

- Класс настолько независим от других классов, насколько возможно.
- Свойства класса, включая его ответственности, однонаправленные связи и атрибуты, обоснованы и совместимы с друг другом.
- Роль класса в двусторонних связях, в которые он включен, ясна и интуитивно понятна.
- Видимость его элементов, прежде всего, атрибутов, правильна. Видимость может быть «public», «private» и так далее.
- Контекст его элементов, прежде всего операций и атрибутов, является правильным. Контекст – это «true» для контекста тип/класс и «false» для контекста объект/образец.
- Специальные требования читабельны и отвечают своим целям.
- Диаграммы, описывающие класс, читаемы и совместимы с другими свойствами.

Артефакт: Прототип интерфейса пользователя

Краткий обзор

Прототип интерфейса пользователя используют:

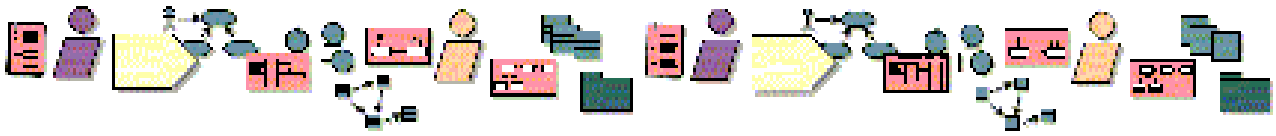
- **описатели прецедентов**, чтобы понять интерфейс пользователя для прецедента;
- **системные аналитики**, чтобы понять, как влияют друг на друга интерфейс пользователя и анализ системы;
- **проектировщики**, чтобы понять, как влияет интерфейс пользователя на «внутреннее устройство» системы;
- **те, кто тестируют классы**, для планирования действий испытания.

Прототип интерфейса пользователя может быть реализован как:

- Эскизы или рисунки на бумаге;
- Точечные изображения, выполненные с использованием инструментов рисования;
- Интерактивный выполнимый прототип (например, реализованный с использованием Microsoft® Visual Basic®).

В большинстве проектов Вы должны использовать все три способа в порядке, перечисленном выше. Так как обычно время их выполнения весьма различно (создавать и изменять рисунки намного быстрее, чем выполнимые программы), этот порядок позволяет вначале просто систематизировать сведения. Однако рисунки не отражают должным образом ограниченную площадь экрана; в рисунке проще поместить больше, чем вмещает площадь экрана.

Лучше всего окончательно определить интерфейс пользователя путем совместного



использования точечных рисунков и выполнимых программ. Это нужно сделать, так как Вы должны будете демонстрировать прототип не только проектировщикам прецедента, но и другим людям. Точечный рисунок может определить точный вид первичных окон, учитывая, что выполнимые программы могут аппроксимировать вид первичных окон и поддерживать их действия, также как вид и поведение вторичных окон. Естественно, если Вы можете с разумным усилием выполнить точный вид первичных окон в выполнимой программе, это лучше, чем объединение выполнимой программы с точечным рисунком. Если Вы не имеете достаточно ресурсов для создания выполнимой программы, Вы можете использовать точечные рисунки как конечное выполнение прототипа. В этом случае может быть полезно дополнить их иллюстрированными сценариями прецедента, описывающими их динамику; иначе велика вероятность того, что конструкторы интерфейса пользователя получат неправильную динамику.

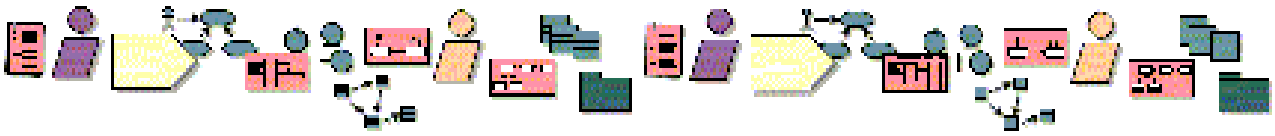
Заметьте, что не нужно стремиться достигнуть хорошей структуры и модуляризации исходного кода для выполнимого прототипа; вместо этого следует сосредоточиться на создании временного прототипа, который визуализирует существенные аспекты интерфейса пользователя и этим обеспечивает некоторые из его существенных действий. Кроме того, прототип, вероятно, несколько раз изменится при разработке и демонстрировании его другим, и эти изменения часто выполняются как дешевые заплаты. В результате исходный код прототипа часто имеет очень ограниченное значение, и не «эволюционизирует» в реальный код интерфейса пользователя.

Вообще, выполнимый прототип осуществить намного дешевле, чем выполнить реальный интерфейс пользователя. Ниже приведены некоторые различия между прототипом и реальным выполнением интерфейса пользователя:

- Прототипу нет необходимости поддерживать все прецеденты и сценарии. Вместо этого, прототип может поддерживать только небольшое число приоритетных прецедентов и/или сценариев.
- Первичные окна часто достаточно сложны в осуществлении; если Вы делаете продвинутый интерфейс пользователя, который действительно использует преимущества потенциальной визуализации, то может быть трудно найти готовые компоненты. Прежде, чем создавать новые компоненты, обычно Вы можете использовать примитивы, типа нажимной кнопки, кнопок переключения или выбора, как аппроксимацию того, как интерфейс пользователя будет отыскивать некоторый набор данных. Если возможно, сделайте несколько прототипов, показывая различные наборы данных, которые охватывают объекты средней величины.
- Имитируйте или игнорируйте все нетривиально осуществляемые оконные операции.
- Имитируйте или игнорируйте внутреннюю организацию системы, типа деловой логики, внешней памяти, множественных процессов и взаимодействия с другими системами.

Синхронизация

Прототип интерфейса пользователя формируется достаточно рано, в стадии Начало или в начале стадии Уточнение, и прежде, чем вся система (включая ее «реальный» пользовательский



интерфейс) будет проанализирована, спроектирована и реализована.

Обратите внимание, что главная цель создания прототипа интерфейса пользователя состоит в том, чтобы сформулировать и проверить функциональные возможности и применимость системы перед реальным проектированием и началом разработки. Таким способом Вы сможете гарантировать, что Вы строите правильную систему прежде, чем Вы затратите очень много времени и ресурсов на разработку.

Чтобы достичь этой цели, разработка прототипа должна быть существенно более дешевой, чем разработка реальной системы, при наличии достаточной возможности обеспечить значимые эксплуатационные испытания.

Создание и уточнение прототипа интерфейса пользователя происходит в результате действия Прототипирование интерфейса пользователя.

Артефакт Прототип интерфейса пользователя предоставляет исходную информацию для следующих действий:

- Детализация требований к программному обеспечению
- Разработка материалов поддержки
- Разработка учебных материалов

Проектировщик интерфейса пользователя отвечает за целостность прототипа интерфейса пользователя, гарантируя, что прототип способствует созданию пригодного для использования интерфейса пользователя, соответствует требованиям иллюстрированных сценариев прецедента и граничным объектам.

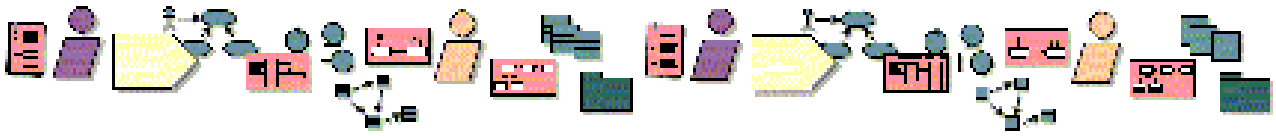
Проектирование прототипа интерфейса пользователя

Ниже перечислены различные аспекты проектирования прототипа интерфейса пользователя:

- Идентификация первичных окон
- Проектирование изображения первичных окон
- Проектирование операций первичных окон
- Проектирование окон свойств
- Проектирование операций, содержащих множественные объекты
- Проектирование различных особенностей

Не обязательно рассматривать все эти аспекты в прототипе. Часто имеет смысл оставить некоторые из них разработчикам, и не иметь с ними дело в прототипе. В этом случае мы рекомендуем Вам не трогать нижние аспекты, а сосредоточиться на верхней части списка.

Интерактивная версия Rational Unified Process содержит подробные рекомендации и много



хороших примеров прототипов интерфейса.

Обратная связь с пользователями

Очень важно продемонстрировать прототип интерфейса пользователя другим. Но, чтобы получить полезную обратную связь, Вы не должны проводить полное испытание готовности, где реальные пользователи выполняют реальные задачи с прототипом. Большинство из того, что Вы обнаружите при эксплуатационных испытаниях – это недостатки, вызванные «авторской слепотой». Это то, на что мог бы указать Вам любой, кто не был включен в конструирование интерфейса пользователя.

По мере продвижения проекта и выполнения прототипа, Вы демонстрируете проект все большему числу рецензентов, включая:

- других членов проекта
- внешних экспертов по эргономике
- пользователей

Демонстрация проекта другим членам проекта

Это – недооцениваемый способ демонстрации проекта. Он имеет очень быструю отдачу и занимает мало времени: члены проекта уже знакомы с прикладной программой, и они обычно очень доступны для спонтанной демонстрации. Делайте этот регулярно, чтобы ликвидировать последствия вашей «авторской слепоты».

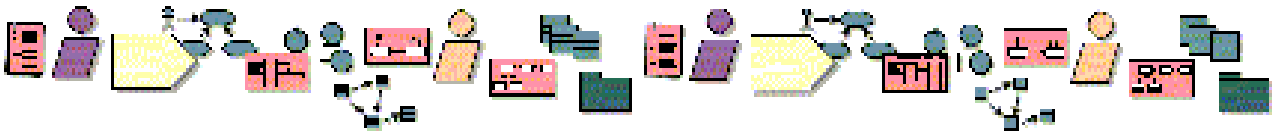
Демонстрация проекта внешним экспертам по эргономике

Хороший эксперт по эргономике может, основываясь на своем опыте, уменьшить объем работы, указывая на очевидные недостатки с точки зрения применимости и возможностей интерфейса пользователя. Таким образом, может быть, полезно включить внешних экспертов по эргономике на начальном этапе, чтобы Вы имели время для работы над новым направлением, предложенным экспертами.

Демонстрация проекта пользователям

Создание демонстрационных версий для пользователей - обычно полезное использование вашего времени. Делайте это столь часто, сколько возможно (доступ к пользователям часто ограничен), чтобы исправлять неправильные восприятия, полученные в течение сбора требований. Однако не демонстрируйте прототип пользователям прежде, чем Вы получите по крайней мере приличный растровый прототип главного первичного окна. И не демонстрируйте прототип одному и тому же пользователю больше одного раза; второй раз пользователь будет заражен вашими более ранними идеями конструирования (подобно «авторской слепоте»).

Также будьте осторожны в своих надеждах. Многие пользователи захотят исправить поведение интерфейса пользователя, то есть окон, позже, когда система будет сформирована.



Как демонстрировать прототип

Часто лучший способ демонстрации прототипа – просмотреть его сидя перед экраном вместе с человеком, которому Вы его демонстрируете. Пройдите через общий сценарий, например, нормальный поток прецедента с нормальными значениями, как описано в иллюстрированном сценарии прецедента. Поощряйте человека задавать вопросы и давать комментарии. Делайте себе заметки.

Другой, довольно дорогостоящий, путь демонстрации состоит в том, чтобы выполнить эксплуатационные испытания. В эксплуатационных испытаниях реальные пользователи выполняют реальные задачи с прототипом. Проблема состоит в том, чтобы получить надежные результаты:

- Вы должны иметь очень «готовый» прототип, функционирующий почти как законченный интерфейс пользователя. Когда Вы достигнете этого в своей разработке, обычно будет слишком дорого делать существенные изменения.
- Вы должны будете обучить пользователя. Большинство пользователей будет «улучшать промежуточные звенья»; если Вы хотите проверить, насколько хорош прототип для этих пользователей, Вы должны будете обучать их, в худшем случае, в течение нескольких недель.

По этим причинам, и если Вы не имеете очень щедрого бюджета для создания интерфейса пользователя, эксплуатационные испытания по максимуму должны быть выполнены в конце итерации (или ее части), в которой создан интерфейс пользователя.

На этом мы заканчиваем знакомство с потоком работ Требования. Пожалуй, единственная тема, которую мы не обсудили достаточно подробно – это управление изменением требований (см. стр. 11-18). Я думаю, что мы исправим это упущение при рассмотрении потока работ Управление конфигурацией и изменением.

Выводы

- Требование – это условие или возможность, которой должна соответствовать система.
- В типичном проекте Вы должны рассмотреть несколько типов требований, включая функциональные и не содержащие функциональности.
- Rational Unified Process определяет проект интерфейса пользователя как описание и визуальное отображение, которое позволяет отработать различные требования применимости.
- В основном потоке работ Требования различные работники анализируют проблему, выявляют потребности совладельцев, определяют систему, управляют контекстом проекта, детализируют систему и управляют изменяющимися требованиями.
- Инструментальные средства Rational поддерживают сбор данных и управление требованиями.