Лабораторная работа N1
Описание требований в контексте модели прецедентов

Задание. Описать прецеденты в текстовом формате для тематики курсового проекта. Один прецедент описать в развернутом виде и 1-2 прецедента – в сжатом.

Обосновать выбор прецедентов.

Сведения из теории

В этой работе описываются категории требований в контексте модели FURPS+.

Требования (requirements) — это возможности или условия, которым должна соответствовать система или проект. Основная задача этапа определения требований — найти, обсудить и зафиксировать, что действительно требуется от системы в форме, понятной и клиентам и членам команды разработчиков.

В рамках унифицированного процесса провозглашается ряд идей, к числу которых относится управление требованиями. Это не означает необходимости окончательного определения и стабилизации требований на первой фазе проекта. Наоборот, с учетом постоянно меняющихся требований заказчиков и руководителей проекта предлагается "систематизированный подход к поиску, документированию, организации к отслеживанию изменчивых требований к системе". То есть к формулировке требований нужно относиться очень ответственно. Обратите внимание на слово "изменчивые" в предыдущем предложении. Изменение требований считается главным "двигателем" прогресса. Отметим также слово "поиск". Предполагается, что требования должны постоянно корректироваться в процессе описания прецедентов и специальных семинаров.

На рис. 1 показаны результаты исследования факторов риска для программных проектов. Согласно этим исследованиям, 37% риска связано с требованиями, следовательно, удельный вес проблем с требованиями среди общих проблем программных проектов очень высок. Поэтому очень важно грамотно организовать управление требованиями. В рамках итеративного процесса изменение требований и обратная связь рассматриваются как главные «двигатели» прогресса.

Типы требований

В рамках UP, согласно модели FURPS+, требования делятся на следующие категории.

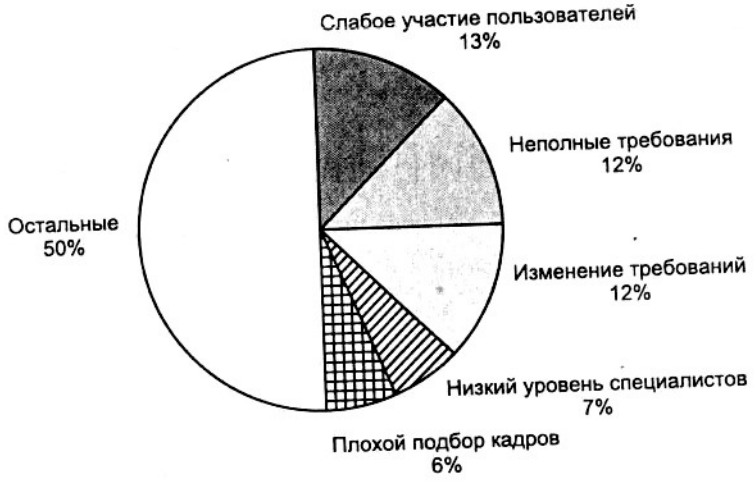


Рис. 1. Факторы риска для программных проектов

Функциональные требования - свойства, возможности, безопасность.

Удобство - человеческий фактор, справочная система, документация.

Надежность - частота сбоев, возможность восстановления и предсказуемость

Производительность - время отклика, точность, доступность, использование

Возможность поддержки - адаптивность, возможность поддели, соответствие международным стандартам, возможность конфигурирования

Символ "+" в названии FURPS+ означает дополнительные (не определяющие) факторы, к которым относятся следующие.

Реализация - требования к ресурсам, языки и средства, аппаратное обеспечение.

Интерфейс - ограничения, накладываемые необходимостью взаимодействия с внешними системами.

Операции - управление системой и ее параметры.

Юридические вопросы - авторское право и т.п.

Категории FURPS+ полезно использовать при формулировке требований, чтобы не упустить важные аспекты жизнедеятельности системы.

Некоторые из этих требований (удобство, надежность, производительность и возможность поддержки) называются ***атрибутами качества*** (quality attributes). Обычно требования делят на две большие категории: ***функциональные*** (относящиеся к поведению) и ***нефункциональные*** (все остальные).

Функциональные требования исследуются и формулируются в процессе разработки модели прецедентов, которая будет описана в следующей главе, а также в процессе осмысления видения системы. Остальные требования формулируются при более детальном описании прецедентов или в дополнительной спецификации. Требования высокого уровня фиксируются в документе "Видение". Затем они уточняются и конкретизируются в последующих документах. В словаре терминов

разъясняются термины, фигурирующие при описании требований. В рамках UP в этом словаре должно содержаться понятие ***"словарь данных"***, в котором описываются требования к данным, а именно правила верификации, допустимые значения и т.п. Механизм выяснения требований обеспечивает прототипирование.

Как будет видно дальше, требования к качеству оказывают существенное влияние на архитектуру системы. Например, требования высокой производительности или надежности влияют на выбор программного обеспечения и аппаратных средств, а также на конфигурацию системы. Наиболее важным "лейтмотивом" при разработке программных систем является необходимость адаптивности с целью отслеживания быстро изменяющихся функциональных требований.

Описание прецедентов — повествовательных историй об использовании системы — отличный метод осмысления и формулировки требований.

Задачи и описания

У потребителей и конечных пользователей есть свои задачи (которые в контексте UP называют потребностями), решение которых должна обеспечить компьютерная система. Эти задачи могут варьироваться от регистрации торговых операций до оценивания объемов добычи нефти из перспективных месторождений. Существует несколько способов выделения этих задач или системных требований. Наилучшие из них достаточно просты и доступны, поскольку это облегчает участие конечных пользователей в определении требований к системе. А участие в этом процессе потребителей снижает риск провала проекта.

Прецеденты — это механизм упрощения этапа формулировки требований для всех заинтересованных лиц. По существу это рассказы об использовании системы в процессе решения поставленных задач. Вот пример сжатого формата описания прецедента.

> **Обработка продажи (process sale).** Покупатель подходит к кассе с выбранными товарами. Кассир с помощью POS-системы регистрирует каждый товар. Система отображает информацию о каждом наименовании, товара и вычисляет общую сумму. Покупатель вводит требуемую информацию; система ее верифицирует и регистрирует. Система выполняет инвентаризацию. Покупатель получает товарный чек и покидает магазин с покупками.

Зачастую прецеденты нужно продумывать гораздо детальнее. Но основная идея состоит в исследовании и формулировке функциональных требований путем написания историй "из жизни системы". Эти истории помогают сформулировать различные задачи и представляют собой сценарии использования системы. На первый взгляд, описать прецеденты не сложно, хотя зачастую достаточно трудно

определить, что требуется от системы и описать это на нужном уровне детализации.

О прецедентах написано достаточно много. Но все равно существует опасность, что умные творческие люди могут слишком усложнить изначально простую идею. Зачастую начинающие специалисты по описанию прецедентов слишком увлекаются созданием диаграмм прецедентов, описанием их взаимосвязей, составлением пакетов прецедентов, рассмотрением необязательных атрибутов и т.д., а не написанием историй. Сила механизма прецедентов состоит в возможности масштабировать уровень сложности и формальности описания в зависимости от реальных потребностей.

Прецеденты и ощутимый результат

Сначала введем некоторые неформальные определения. *Исполнителем* (actor) будем называть сущность, обладающую поведением, например, человека (идентифицируемого по роли), компьютерную систему или организацию, например кассира.

Сценарий (scenario)— это специальная последовательность действий или взаимодействий между исполнителями и системой. Его иногда также называют ***экземпляром прецедента*** (use case instance). Это один конкретный сценарий использования системы либо один проход прецедента, например, сценарий успешной покупки товаров за наличный расчет, либо сценарий неудачного завершения покупки из-за прерванной транзакции по обработке данных кредитной карточки.

Неформально, ***прецедент*** (use case) — это набор взаимосвязанных успешных и неудачных сценариев, описывающий использование системы исполнителем для решения одной из задач. Например, рассмотрим свободный формат прецедента, включающего некоторые альтернативные сценарии.

Возврат товара (Handle Returns)

> ***Основной успешный сценарий.*** Покупатель подходит к кассе с товарами, подлежащими возврату. Кассир использует POS-систему для регистрации каждого возвращаемого товара...
>
> ***Альтернативные сценарии.*** Если в авторизации кредитной карточки отказано, кассир информирует об этом покупателя и предлагает ему другой способ оплаты покупки.
>
> Если идентификатор товара в системе не обнаружен, система уведомляет об этом кассира и предлагает ему вручную ввести идентификационный код (возможно, штрих-код поврежден и его сложно считать).
>
> Если у системы возникают сложности при коммуникации с внешней системой вычисления налога,...

Несколько другое, но подобное определение прецедента приводится в RUP.

Прецедент — это набор сценариев использования, в котором каждый экземпляр сценария представляет собой последовательность действий, выполняемых системой для достижения ощутимого для конкретного исполнителя результата.

Фраза *"ощутимый результат"* несколько туманна, но очень важна, поскольку она указывает на то, что поведение системы должно быть ощутимо для пользователя.

> Основное внимание при описании прецедента нужно сконцентрировать на вопросе: "Как использование системы обеспечивает ощутимый для пользователя результат или решает его задачу?", а не на обдумывании системных требований в терминах свойств или функций.

На первый взгляд, требование обеспечения ощутимого результата может показаться неочевидным. Однако в сфере программного обеспечения известно множество неудачных проектов, которые провалились из-за невыполнения реальных задач пользователей. К такому отрицательному результату может при вести подход к описанию системных требований в виде списка свойств и функций системы, поскольку он не заставляет заинтересованных лиц рассматривать требования в более широком контексте достижения некоторого ощутимого результата или некоторой цели. В отличие от этого подхода, в контексте прецедентов свойства и функции системы ориентированы на достижение цели.

В этом состоит основная идея, сформулированная Якобсоном относительно прецедентов: требования должны быть направлены на получение ощутимого результата и достижения целей.

Прецеденты и функциональные требования

Прецеденты, в основном, это функциональные требования, указывающие на то, что должна делать система. В контексте типов требований, определяемых моделью FURPS+, основное внимание уделяется функциональным требованиям. Однако остальные типы требований тоже могут быть связаны с прецедентами. В рамках UP и большинства других современных методов прецеденты являются основным механизмом, рекомендуемым для их определения и исследования. Прецеденты определяют пожелания или соглашения относительно поведения системы.

Итак, прецеденты — это требования (хотя и не все требования). Некоторые считают требованиями только список функций и свойств типа "система должна...". На самом деле это не так. Ключевая идея использования прецедентов как раз и

состоит (обычно) в снижении роли списка требований в старом понимании этого слова. Теперь в качестве функциональных требований выступают сами прецеденты.

Описания прецедентов — это текстовые документы, а не диаграммы. Моделирование прецедентов — это процесс написания текста, а не рисования. Однако для иллюстрации имен прецедентов и исполнителей, а также их взаимоотношений в UML определены обозначения для диаграммы прецедентов.

Типы и форматы прецедентов

Прецеденты типа "черный ящик" и системные обязанности

Прецеденты типа "черный ящик" (black-box use cases) — это самый типичный и рекомендуемый тип прецедентов. Они не описывают внутреннюю работу системы, ее компоненты или дизайн. Наоборот, системе вменяются некоторые *обязанности* (responsibilities). Этот метафорический термин широко применяется в объектно-ориентированном проектировании: программные элементы имеют обязанности и взаимодействуют с другими элементами со своими обязанностями. Определяя обязанности системы через прецеденты типа "черный ящик", можно указать, *что* должна делать система (функциональные требования), не расписывая, как это делать (не выполняя проектирование). Вообще, термины "анализ" и "проектирование" зачастую сводятся к вопросам "что" и "как". Это важные вопросы в хорошей программкой разработке. В процессе анализа требований нужно избегать принятия решений "как", а описывать лишь внешнее поведение системы как черного ящика. Позднее, на этапе проектирования, создается решение, удовлетворяющее разработанной спецификации.

Стиль черного ящика	Другой стиль
Система регистрирует покупку	Система записывает сведения о покупке в базу данных. или, еще хуже Система генерирует оператор SQL INSERT для данной продажи...

Степень формализации

Прецеденты описываются в различных форматах, в зависимости от потребностей. Помимо типов "черного ящика" и "белого ящика", выделяют несколько степеней формализации описания прецедентов.

- ***Сжатый*** — аннотация в виде одного абзаца. Обычно она описывает только главный успешный сценарий. Пример такого описания приведен выше для

прецедента Оформление продажи (Process Sale).

• ***Свободный*** — неформальный стиль описания. Описание прецедента занимает несколько абзацев и охватывает различные сценарии. Примером такого описания является рассмотренный выше прецедент Возврат товара.

• ***Развернутый*** — наиболее подробный стиль описания. При таком подходе детально описываются все шаги и варианты развития сценария, а также предусловия и результаты. Рассмотрим пример развернутого описания прецедента для системы Алтын.

Пример развернутого описания прецедента

Оформление продажи

Развернутые описания прецедентов структурированы и содержат большое количество деталей. Их полезно использовать для углубления понимания целей, задач и требований. Для примера POS-системы Алтын такие описания можно обсуждать на семинарах по определению требований на начальной стадии проекта вместе с системным аналитиком, экспертами предметной области и разработчиками.

Формат usecases.org

Для развернутого описания прецедентов существуют различные шаблоны форматирования. Однако чаще всего используется шаблон, приведенный на Web-узле www.usecases.org. Этот стиль проиллюстрирован в следующем примере.

Этот пример детального описания прецедента относится к рассматриваемой в данной книге системе Алтын и отражает множество типичных элементов и вопросов.

Прецедент П1. Оформление продажи

Основной исполнитель. Кассир.

Заинтересованные лица и их требования

- *Кассир.* Хочет точно и быстро ввести данные, не допуская ошибок в платеже, поскольку недостача вычитается из его зарплаты.

- *Продавец.* Хочет получить свои комиссионные от продажи.

- *Покупатель.* Хочет купить товары и быстро оформить покупку с минимальными усилиями. Хочет получить подтверждение факта покупки для возможного возврата товара.

- *Компания*. Хочет аккуратно записать транзакцию и удовлетворить интересы покупателя. Хочет удостовериться, что служба авторизации платежей

зафиксировала все данные о платеже. Заинтересована в обеспечении устойчивости к сбоям; хочет продолжать регистрировать продажи, даже если серверные компоненты (например, служба удаленной проверки кредитоспособности) недоступны. Хочет автоматически обновлять бухгалтерскую документацию и вести складской учет.

- *Государственные налоговые службы*. Хотят получать налог от каждой продажи.

Предусловия. Кассир идентифицирован и аутентифицирован.

Результаты (Постусловия). Данные о продаже сохранены. Налоги корректно вычислены. Бухгалтерские и складские данные обновлены. Комиссионные начислены. Чек сгенерирован. Авторизация платежа выполнена.

Основной успешный сценарий (или основной процесс)

1. Покупатель подходит к кассовому аппарату PQS-системы с выбранными товарами.
2. Кассир открывает новую продажу.
3. Кассир вводит идентификатор товара,
4. Система записывает наименование товара и выдает его описание, цену и общую стоимость. Цена вычисляется на основе набора правил. Кассир повторяет действия, описанные в пп. 3-4, для каждого наименования товара.
5. Система вычисляет общую стоимость покупки с налогом.
6. Кассир сообщает покупателю общую стоимость и предлагает оплатить покупку.
7. Покупатель оплачивает покупку, система обрабатывает платеж,
8. Система регистрирует продажу и отправляет информацию о ней внешней бухгалтерской системе (для обновления бухгалтерских документов и начисления комиссионных) и системе складского учета (для обновления данных).
9. Система выдает товарный чек.
10. Покупатель покидает магазин с чеком и товарами (если он что-то купил).

Расширения (или альтернативные потоки)

*а. При каждом выходе системы из строя.

Для ввода системы в строй и корректной обработки платежа нужно обеспечить восстановление всех транзакций и событий с любого шага сценария.

1. Кассир перезапускает систему, регистрируется и предлагает восстановить предыдущее состояние.

2. Система восстанавливает предыдущее состояние. 2а. Система определяет аномалию, повлекшую сбой.

2.1. Система уведомляет об ошибке кассира, регистрирует ошибку и переходит в начальное состояние.

2.2. Кассир начинает новую продажу.

3а. Неправильный идентификатор.

1. Система уведомляет об ошибке и отменяет ввод данного наименования товара.

3б. В рамках одной категории существует несколько различных наименований товара и идентифицировать конкретное наименование не нужно (например, 5 пакетов леденцов).

1. Кассир может ввести идентификатор категории товара и количество единиц.

3.6а. Покупатель просит кассира отменить покупку одного из товаров,

1. Кассир вводит идентификатор товара для удаления из продажи.

2. Система отображает обновленную промежуточную стоимость,

3.6б. Покупатель просит кассира отменить продажу.

1. Кассир отменяет продажу

3.6в. Кассир приостанавливает продажу.

1. Система записывает сведения о продаже таким образом, чтобы они были доступны с любого терминала POS-системы.

4а. Сгенерированная системой цена товара не устраивает покупателя (например, у него есть дисконтная карта и он рассчитывает на более низкую цену товара).

1. Кассир вводит команду об изменении цены.

2. Система вычисляет новую цену.

5а. Система выявляет сбой при коммуникации с внешней службой вычисления налога.

1. Система перезапускает службу с данного узла PQS-системы и продолжает работу.

….

7а. Оплата наличными.

1. Кассир вводит предложенную покупателем сумму.

2. Система вычисляет положенную сдачу и открывает кассу с наличностью.

3. Кассир складывает поученные деньги и выдает сдачу покупателю.

4. Система регистрирует платеж наличными.

7б. Оплата по кредитной карточке.

1. Покупатель вводит информацию о своей кредитной карточке.

2.Система отправляет запрос на авторизацию платежа внешней системе службы авторизации платежей и запрашивает подтверждение платежа.

…

Специальные требования

- Сенсорный экран с интерфейсом пользователя для большого плоского монитора. Текст должен быть виден с расстояния один метр.

- Отклик службы авторизации в 90% случаев приходит в течение 30 секунд.

- Каким-то образом нужно обеспечить робастное восстановление информации в.случае сбоя при

доступе к удаленным службам, таким как система складского учета.

- Возможность локализации {представления на различных языках) отображаемого текста.

- Возможность добавления новых бизнес-правил на шагах 3 и 7 в процессе

функционирования системы.

Список технологий и типов данных

3а. Идентификатор товара считывается со штрих-кода (при наличии последнего) лазерным сканером или вводится с клавиатуры.

3б. Идентификатор товара может определяться по схемам кодирования UPC, EAN, JAN или SKU.

7а. Информация об открытом кредите вводится с помощью считывающего устройства или с клавиатуры.

7б. Подпись при оплате чеком ставится на бумажном документе. Однако ожидается, что в течение двух лет большинство покупателей будут требовать цифровые устройства считывания подписи.

Частота использования: почти постоянно.

Открытые вопросы

- Изучить законодательство по налогообложению.
- Исследовать вопрос восстановления удаленных служб.
- Какая настройка потребуется для различных типов магазинов.
- Должен ли кассир снимать кассу при выходе из системы.
- Может ли пользователь сам использовать устройство считывания данных с карточки или это должен делать кассир.

Этот прецедент скорее является иллюстративным, чем исчерпывающим (хотя и основывается на реальных требованиях к POS-системе). Тем не менее, он описан достаточно подробно и позволяет составить реалистичное представление о развернутом формате описания прецедентов и их сложности. Этот пример может служить моделью для многих других прецедентов.

Представление в виде двух колонок

Некоторые предпочитают оформлять прецеденты в виде двух колонок, обращая внимание на факт взаимодействия исполнителей и системы. Вот как выглядит рассмотренное выше описание, представленное в виде двух колонок.

Прецедент П1. Оформление продажи

Основной исполнитель:...

... как и ранее...

Основной успешный сценарий

Действие исполнителя	Отклик системы
1. Покупатель подходит к кассовому аппарату POS-системы с выбранными товарами	

2. Кассир открывает новую продажу	
3.Кассир вводит идентификатор товара	4. Система записывает наименование товара и выдает его описание, цену и общую стоимость. Цена вычисляется на основе набора правил.
Кассир повторяет действия, описанные в пп. 3-4 для каждого наименования товара.	5. Система вычисляет общую стоимость покупки с налогом
6. Кассир сообщает покупателю общую стоимость и предлагает оплатить покупку.	
7. Покупатель оплачивает покупку.	8. Система обрабатывает платеж.
	9. Система регистрирует продажу и отправляет информацию о ней внешней бухгалтерской системе (для обновления бухгалтерских документов и начисления комиссионных) и системе складского учета (для обновления данных). Система выдает товарный чек.

Какой формат лучше?

Однозначного ответа на этот вопрос не существует. Главная задача — детально описать основной успешный сценарий и его расширения в некоторой стандартной форме.

Пояснения

Вводные элементы

В описание прецедента можно добавлять различные вводные элементы, из них, которые важны для изложения последующего сценария, следует пом щать в начало описания. Второстепенный материал можно располагать в KOHI сценария. Например, в описание можно добавить следующий вводный элемент.

Главный исполнитель. Основной исполнитель, вызывающий системные службы для достижения цели. Заинтересованные лица и их потребности

Этот список играет более важную роль, чем это кажется на первый взгляд. С его помощью можно понять, что должна делать система. Приведем цитату.

“Система реализует соглашение между заинтересованными лицами. Поведение системы описывается с помощью прецедентов... Прецедент, как соглашение о

поведении, включает все возможные аспекты поведения, связанные с удовлетворением запросов заинтересованных лиц" (Cockburn A.)

Эта цитата дает ответ на вопрос, что нужно описывать в прецеденте. Там нужно описывать все, что служит удовлетворению запросов заинтересованнь лиц. Кроме того, начиная описание прецедента с перечня заинтересованных лиц

По окончании обработки расширения по умолчанию выполняется возврат к основному сценарию, если в расширении не предусмотрен другой ход событий (например, завершение работы системы).

Иногда некоторые расширения оказываются очень сложными, например, платеж по кредитной карточке. В этом случае расширение можно выделить в отдельный прецедент.

Вот пример, демонстрирующий описание сбоя при реализации расширения.

76. Оплата по кредитной карточке.

1. Покупатель вводит информацию о своей кредитной карточке.

2. Система отправляет запрос на авторизацию платежа внешней системе службы авторизации платежей и запрашивает подтверждение платежа. 2а. Система определяет сбой при взаимодействии с внешней системой.

1. Система сигнализирует об ошибке кассиру.

2. Кассир просит покупателя изменить тип платежа.

3....

Если нужно описать условия, которые могут возникнуть в любой момент, то в обозначении пункта можно использовать символ * (см. ниже).

*а. При каждом выходе системы из строя.

Для ввода системы в строй и корректной обработки платежа нужно обеспечить восстановление всех транзакций и событий с любого шага сценария.

1. Кассир перезапускает систему, регистрируется и предлагает восстановить предыдущее состояние.

2. Система восстанавливает предыдущее состояние.

Специальные требования

В этот раздел заносятся нефункциональные требования, атрибуты качества или ограничения, связанные с данным прецедентом. Сюда относятся характеристики производительности, надежности, удобства использования и конструктивные ограничения (например, на устройства ввода-вывода).

Специальные требования

- Сенсорный экран с интерфейсом пользователя для большого плоского монитора. Текст должен быть виден с расстояния один метр.

- Отклик службы авторизации в 90% случаев приходит в течение 30 секунд.

- Возможность локализации (представления на различных языках)

отображаемого текста. Возможность добавления новых бизнес-правил на шагах 3 и 7 в процессе функционирования системы.

Запись этих условий при описании прецедента — классический совет разработчиков унифицированного процесса. Однако на практике многие специалисты помещают эти требования в едином общем документе, например, в дополнительной спецификации. Тогда их удобнее читать и осмысливать, поскольку обычно они рассматриваются в общем контексте в процессе анализа архитектуры.

Список технологий и типов данных

Зачастую при реализации проекта важно не что сделать, а как. Перечень используемых технологий тоже приводится в описании прецедента. Типичным примером такой ситуации являются технические ограничения, выдвигаемые заинтересованными лицами для технологий ввода и вывода. Например, заказчик может потребовать, чтобы POS-система поддерживала ввод данных кредитной карточки с клавиатуры и с помощью считывающего устройства. Заметим, что здесь приводятся лишь примеры проектных решений и ограничений, появляющихся на ранней стадии реализации проекта. В целом, такой конкретизации следует избегать, однако иногда она бывает неизбежна, особенно в отношении технологий ввода/вывода.

Нужно также указать возможные варианты типов данных, например, различные кодировки идентификаторов товаров.

Список технологий и типов данных

3а. Идентификатор товара считывается со штрих-кода (при наличии последнего) лазерным сканером или вводится с клавиатуры.

3б. Идентификатор товара может определяться по схемам кодирования UPC, EAN, JAN или SKU.

7а. Информация об открытом кредите вводится с помощью считывающего устройства или с клавиатуры.

7б. Подпись при оплате чеком ставится на бумажном документе. Однако ожидается, что в течение двух лет большинство покупателей будут требовать цифровые устройства считывания подписи.

Совет. В этом разделе не должно быть много пунктов, описывающих различные действия для разных ситуаций. Если это необходимо, перенесите эти описания в раздел расширений.

Задачи и рамки прецедента

Как выделить прецедент? Зачастую определить правильный (а точнее, полезный) прецедент очень сложно. Каждую задачу можно рассматривать на разных уровнях детализации, начиная от конкретных простых действий и заканчивая деятельностью на уровне предприятия.

На каком же уровне детализации следует формулировать прецеденты?

Прецеденты для элементарных бизнес-процессов

Какой из следующих пунктов можно считать прецедентом?

- Переговоры с поставщиком
- Обработка возврата товара
- Регистрация

Можно сказать, что все это прецеденты на разных уровнях детализации, выделенные в зависимости от рамок системы, исполнителей и задач.

Вместо вопроса "Что такое корректный прецедент?" целесообразнозно задать себе вопрос: "На каком уровне следует рассматривать прецеденты в процессе анализа требований к приложению?".

Совет. В процессе анализа требований в компьютерному приложению следует сосредоточиться на уровне элементарных бизнес-процессов (EBP-Elementary Business Processes)

Термин EBP заимствован из области исследования бизнес-процессов и определяется следующим образом.

"Элементарный бизнес-процесс — это задача, выполняемая одним человеком в одном месте в одно время в ответ на некоторое бизнес-событие, добавляющая измеряемое бизнес-значение и переводящая данные в некоторое устойчивое состояние, например, подтверждение платежа по кредитной карточке или распоряжение брокеру при изменении цен".

Однако общая идея этого определения такова. Прецедент — это не один маленький шаг, такой, например, как удаление товара или печать документа. Основной сценарий прецедента обычно включает пять-десять шагов. Описываемый сценарий выполняется не в течение нескольких дней или сеансов, как, например, переговоры с поставщиками. Это задача, выполняемая в течение одного сеанса. Реализация прецедента может длиться от нескольких минут до нескольких дней. Как и при определении унифицированного процесса, основное внимание уделяется получению ощутимого (измеримого) результата и переходу системы и данных в устойчивое состояние.

Обоснованные отклонения от элементарного бизнес-процесса

Несмотря на то, что основные прецеденты приложения должны сответствовать элементарным бизнес-процессам, зачастую полезно прецеденты более низкого уровня, представляющие подзадачи или последовательности действий в рамках основного прецедента. То есть могут существовать прецендженты, не соотвествующие EBP.

Прецеденты и задачи

Исполнители имеют свои задачи (или потребности), для решения которых они используют систему. Поэтому прецеденты уровня EBP еще называют прецедентами уровня *задач пользователя* (user goal). Это делается для того, чтобы обратить внимание на реализацию потребностей пользователей системы или основного исполнителя,

Отсюда следует алгоритм выделения прецедентов.

1. Выделить задачи (цели) пользователей.

2. Определить для каждой из них отдельный прецедент.

При таком подходе несколько смещаются акценты аналитиков. Вместо вопроса "Каковы прецеденты для данной системы?" возникает вопрос "Каковы задачи исполнителей?". Чтобы отобразить эту взаимосвязь, имя прецедента должно соответствовать названию задачи. Например, задаче электронного оформления продажи должен соответствовать прецедент **Оформление продажи**.

Заметим, что такая симметрия позволяет оценить адекватность уровня выделения прецедентов и задач пользователя в соответствии с правилом выделения элементарных бизнес-процессов.

Совет. Ключевая идея состоит в том, чтобы для выделения прецедентов исследовать задачи исполнителей.

Допустим, мы собрались на семинаре для формулировки требований. На этом семинаре основной вопрос можно формулировать двумя способами.

• Что делает система?

• Каковы задачи исполнителей?

Ответы на первый вопрос, скорее, отражают текущие решения и процедуры, а также сложность этих процедур.

Ответы на второй вопрос, особенно в сочетании с исследованием целей более высокого уровня ("в чем задача этой задачи?"), открывают видение новых и более эффективных решений, акцентируют внимание на получении ощутимого результата и позволяют глубже понять основные потребности заинтересованных лиц в контексте обсуждаемой системы.

Пример: использование рекомендаций в соответствии с EBP

Допустим, вы являетесь системным аналитиком и отвечаете за формулировку требований к системе Алтын. Для этого вам нужно исследовать задачи пользователей. На семинаре по формулировке требований может состояться такой диалог.

Системный аналитик: "Каковы ваши задачи в контексте использования POS-системы"

Кассир: "Во-первых, быстро зарегистрироваться. Во-вторых, оформлять продажи."

Системный аналитик: "Какая задача более высокого уровня приводит, на ваш взгляд, к необходимости выделения отдельной задачи регистрации?"

Кассир: "Мне необходимо "представиться" системе, а она должна проверить, имею ли я право ею пользоваться."

Системный аналитик: "Какова еще более глобальная задача?"

Кассир: "Предотвратить утечку или повреждение данных."

Обратите внимание на стратегию аналитика выстроить иерархию целей и выявить таким образом нужный уровень для элементарного бизнес-процесса. Это позволяет лучше понять мотивацию действий исполнителей и их задачи.

Предотвращение утечки данных — это цель более высокого уровня, чем задача пользователя.

Цель следующего уровня иерархии ("представиться" системе и выполнить аутентификацию) несколько ближе к задачам пользователя. Но относится ли она к уровню элементарных бизнес-процессов? Ее решение не добавляет ощутимого результата или измеримого бизнес-значения. Если на вопрос о том, чем кассир занимался сегодня на работе, прозвучит ответ: "Я 20 раз зарегистрировался!", то вряд ли такой ответ устроит начальника. Значит, это второстепенная задача, Которая служит достижению важной цели, но не относится к уровню EBP. Уровню EBP точнее всего соответствует задача оформления продажи.

В качестве другого примера можно рассмотреть процесс регистрации выручки, когда кассир задвигает ящик с наличностью и закрывает его в системе, регистрируется и вводит в систему сумму выручки. Регистрация выручки — это прецедент уровня EBP (или уровня задач пользователя), а регистрация в системе — это один из его шагов, выполняющий вспомогательную задачу, а не отдельный прецедент.

Вспомогательные задачи и прецеденты

Несмотря на то, что задача "представиться" системе и выполнить аутенти-

фикацию (или задача регистрации) не была отнесена к уровню задач пользователя, она все же является целью, но на более низком уровне. Такие задачи называют ***вспомогательными*** (subfunctional goal), поскольку они призваны обеспечивать выполнение задач пользователя. Для вспомогательных задач отдельные прецеденты создаются очень редко, хотя специалисты по написанию прецедентов зачастую рекомендуют улучшать (обычно упрощать) набор прецедентов.

Для вспомогательных задач писать прецеденты не запрещается, однако это не всегда нужно, поскольку при этом усложняется модель прецедентов. Количество вспомогательных задач для системы может исчисляться сотнями, но стоит ли создавать так много прецедентов?

Важно понимать, что с увеличением числа прецедентов возрастает сложность задачи формулировки и управления требованиями, а значит, увеличивается также время решения этой задачи,

Основным мотивом написания прецедента для вспомогательной должна служить повторяемость этой задачи или ее важное значение как предусловия для множества других прецедентов. Этому принципу удовлетворяет задача авторизации, которая обеспечивает предусловие для большинства, если не всех остальных прецедентов уровня задач пользователя.

Таким образом, вполне логично создать прецедент Аутентификация пользователя.

Составные задачи и прецеденты

Задачи могут принадлежать к различным уровням сложности и являться составными, начиная от уровня предприятия ("быть прибыльным"), до задач среднего уровня ("регистрация торговых операций") и вспомогательных задач в рамках приложения ("проверка правильности ввода").

Аналогично, и прецеденты могут относиться к различным уровням сложности и состоять из прецедентов более низкого уровня.

Наличие нескольких уровней сложности задач и прецедентов может ввести в заблуждение при определении соответствующего уровня для основных прецедентов. Для отсеивания слишком детальной информации следует использовать принцип нахождения элементарных бизнес-процессов.

Определение основных исполнителей, задач и прецедентов

Прецеденты предназначены для удовлетворения потребностей основных исполнителей. Поэтому для выделения прецедентов используется следующая

процедура.

1. Определите рамки системы: является ли она программным приложением, аппаратно-программным комплексом, включает ли в себя своих пользователей или всю организацию?

2. Идентифицируйте основных исполнителей, потребности (цели) которых удовлетворяются с помощью системы.

3. Для каждого исполнителя определите его задачи. Составьте иерархию в соответствии с рекомендациями по выделению EBP.

4. Определите прецеденты, удовлетворяющие потребности каждого исполнителя, и присвойте им имена в соответствии с задачами. Обычно основные прецеденты соответствуют задачам пользователей, за одним исключением, о котором речь пойдет ниже.

Шаг 1. Определение рамок системы

Для данного прецедента разрабатываемой системой является сама POS-система. Все, что находится за ее пределами, включая кассира, службу авторизации платежей и т.д., в эти рамки не включается.

Для определения рамок системы следует, в первую очередь, указать, что к ней не относится, т.е. определить внешних основных и вспомогательных исполнителей. После идентификации внешних исполнителей рамки системы очерчиваются более четко. Например, возлагается ли на систему полная ответственность за авторизацию платежей? Нет, эту задачу выполняет внешний исполнитель — служба авторизации платежей.

Шаги 2 и 3. Определение основных исполнителей и задач

Нельзя однозначно указать последовательность определения исполнителей и задач. Обычно на семинаре по определению требований методом мозгового штурма идентифицируются и те и другие артефакты. Иногда исполнители определяются после формулировки задач, а иногда наоборот.

В процессе мозгового штурма основное внимание следует уделить определению основных исполнителей, поскольку это расширит возможности для дальнейшего исследования.

Вопросы, задаваемые при определении исполнителей и задач

При определении основных исполнителей и задач пользователей следует ответить на следующие вопросы, чтобы не упустить из виду некоторые неочевидные моменты.

- Кто запускает и выключает систему?
- Кто является системным администратором?
- Кто осуществляет управление пользователями и безопасностью?
- Относится ли время к числу исполнителей, другими словами, должна ли система выполнять какие-либо действия в ответ на события времени?
- Существует ли процесс мониторинга, благодаря которому система перезапускается в случае сбоя?
- Кто контролирует деятельность и производительность системы?
- Как выполняется обновление программного обеспечения?
- Кто анализирует журналы регистрации? Можно ли обеспечить удаленный доступ к ним?

Основные и вспомогательные исполнители

Напомним, что основные исполнители — это те, чьи потребности удовлетворяются с помощью системы. Для решения своих задач они используют систему. В отличие от них, *вспомогательные исполнители* (supporting actor) занимаются обслуживанием системы. Пока сосредоточимся на идентификации основных исполнителей.

Напомним также, что основными исполнителями, среди прочего, могут быть другие компьютерные системы.

Совет. Отсутствие внешних компьютерных систем среди основных исполнителей должно насторожить разработчиков.

Перечень исполнителей и их задач

Составьте список основных исполнителей и их задач. В терминах артефактов унифицированного процесса этот список должен быть разделом артефакта «Видение». Рассмотрим следующую таблицу.

Исполнитель	**Задачи**	**Исполнитель**	**Задачи**
Кассир	Оформляет продажи Оформляет кредиты Выполняет возврат товара Регистрирует выручку	Системный администратор	Добавляет пользователей. Изменяет параметры пользователей. Удаляет пользователей. Управляет безопасностью. Управляет системными таблицами.
Менеджер	Включает систему	Система анализа	Анализирует информацию о продажах и оценивает

	Выключает систему	торговой деятельности	производительность

Система анализа торговой деятельности – это удаленное приложение, которое достаточно часто будет запрашивать данные от каждого узла POS-системы по сети.

"Суровая действительность"

Этот перечень выглядит достаточно просто и понятно. Однако на самом деле создать его довольно сложно. На это уходит масса усилий и требуется несколько интенсивных семинаров, организованных по методу мозгового штурма. Вернемся к приведенному выше примеру, иллюстрирующему применение правила EBP к задаче регистрации в системе. На семинаре, посвященном составлению этого списка, кассир может предложить рассматривать регистрацию как одну из задач уровня пользователя. Системный аналитик, углубляясь в проблему, может расширить эту задачу до уровня "идентификации и аутентификации" пользователя. Затем он может осознать, что эта задача не относится к уровню EBP и исключить ее из списка задач пользователя. На самом деле, в реальности все будет выглядеть несколько иначе, поскольку опытный аналитик, как правило, знает набор эвристик, сформулированных на основе его прежнего опыта, одна из которых сводится к следующему: регистрация пользователя редко относится к уровню задач EBP. Поэтому он достаточно быстро исключит ее из рассмотрения.

Основной исполнитель и задачи системы зависят от ее рамок.

Почему основным исполнителем для прецедента **Оформление продажи** является кассир, а не покупатель? Почему покупатель не включен в список исполнителей?

Ответ определяется рамками разрабатываемой системы. Если предприятие или торговую организацию рассматривать как ***агрегатную систему***, то для нее основным исполнителем должен являться покупатель, задача которого — приобретение товаров или услуг. Однако с точки зрения самой POS-системы (которая определяет рамки системы для данного прецедента), основным исполнителем является кассир, задача которого — обслуживание продаж.

Определение исполнителей и задач путем анализа событий

Для определения исполнителей, их задач и прецедентов можно также использовать внешние события. Что это означает? Зачастую к одному и тому же уровню EBP или прецеденту, например, относится целая группа событий.

Внешнее событие	**Инициатор**	**Задача**

Ввод информации о наименовании товара	Кассир	Оформить продажу
Ввод информации о платеже	Кассир или покупатель	Оформить продажу
…		

Рис. 2. Основные исполнители и их задачи при изменении рамок системы

Шаг 4. Определение прецедентов

Как правило, каждой задаче пользователя соответствует один прецедент уровня EBP. Его имя должно соответствовать названию задачи, например, задаче оформления продажи должен соответствовать прецедент **Оформление продажи**.

Как правило, имя прецедента начинается с существительного, описывающего действие.

Типичным исключением из правила соответствия задач и прецедентов является прецедент, решающий четыре задачи — создание, восстановление, обновление и удаление. Обычно такой прецедент называется Управление <*чем-либо*>. Например, задачи "изменение информации о пользователях", "удаление пользователей" и т.д. решаются в рамках прецедента **Управление пользователями**.

Определение прецедентов выполняется в несколько этапов, одни из которых занимают несколько минут (например, присвоение имен прецедентам), а другие — по несколько дней или недель (развернутое описание). В последующих разделах этой главы, посвященных унифицированному процессу, эти этапы будут рассмотрены в контексте итеративной разработки.

Если прецеденты описаны плохо

Для описания прецедентов для POS-системы Алтын в течение нескольких итераций созывались семинары по определению требований, постепенно количество прецедентов увеличивалось, требования уточнялись и дорабатывались с учетом обратной связи. В процессе описания прецедентов активно участвовали эксперты предметной области, кассиры и программисты. Между ними не было никаких посредников, все общение происходило напрямую.

Хорошо, но не очень. Спецификация требований создает лишь иллюзию корректности. Можно с уверенностью утверждать, что прецеденты и другие требования в этом документе не корректны. В описании не хватает важной информации и содержатся неверные утверждения. Для решения этой проблемы мы не призываем вас следовать рекомендациям однопроходного процесса разработки и приложить

больше усилий к полному описанию требований на начальных этапах разработки, хотя, безусловно, эту работу нужно выполнить максимально добросовестно. Однако этого не достаточно.

Здесь требуется другой подход. В основном проблема решается за счет итеративности процесса разработки, но в дополнение к нему иногда требуется *постоянное личное общение* — ежедневное обсуждение между всеми разработчиками при участии специалиста по предметной области, который уполномочен принимать решения о требованиях к системе. Нужен некто, к кому программисты могут подойти и за несколько секунд снять возникшие у них вопросы. Например, в рамках подхода XP существует отличный принцип: пользователь должен постоянно находиться среди участников проекта, в том же помещении.

Описание прецедентов, относящихся к интерфейсу пользователя, в свободном стиле

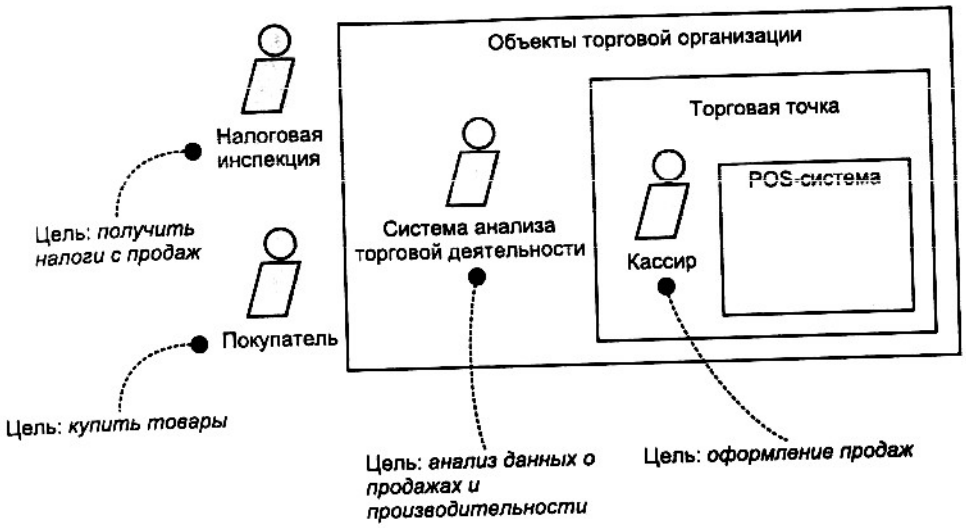
Исследование ***целей***, а не обязанностей и процедур позволяет сосредоточить внимание на основных требованиях. Например, на одном из семинаров кассир может сказать, что одной из его целей является регистрация в системе. При этом он может иметь в виду элементы интерфейса пользователя, соответствующие диалоговые окна, ввод идентификатора и пароля. Однако все это — механизмы достижения цели, а не сама цель. Изучая иерархию целей (отвечая на вопрос "какова цель этой задачи?"), системный аналитик приходит к формулировке, независимой от механизма реализации: "идентифицировать себя и выполнить аутентификацию", или на более высоком уровне — "предотвратить утечку информации".

Такой процесс исследования позволяет получить новые и более эффективные решения. Например, в настоящее время достаточно распространены и недорого стоят клавиатура и мышь с устройствами считывания биометрической информации, в частности отпечатков пальцев. Если целью является идентификация и аутентификация, то почему бы для ее достижения не использовать эффективное и быстрое средство считывания биометрических данных с клавиатуры? Однако отвечая на этот вопрос, следует принимать во внимание удобство использования. В данном случае придется установить профили типичных пользователей. А если их пальцы чем-то испачканы? А если они травмированы?

Базовый стиль описания

Эта идея была сформулирована в различных рекомендациях типа "не уделяйте внимания вопросам интерфейса пользователя, сосредоточьте внимание на

содержательной стороне вопроса"



Если описание не содержит подробной информации о реализации пользовательского интерфейса, а основное внимание в нем сосредоточено на содержательных моментах, то такой стиль описания Константин называет *базовым* (essential).[a]

Базовый стиль описания предполагает изложение на уровне *намерений* пользователя и *обязанностей* системы, а не на уровне их конкретных действий. При таком стиле описания не нужно углубляться в детали технологии и механизма реализации, особенно при рассмотрении вопросов, связанных с интерфейсом пользователя.

Совет. Описывайте прецеденты в базовом стиле. Не уделяйте внимания интерфейсу пользователя, а сосредоточьтесь на содержательной стороне вопроса.

Прецедент Оформление продажи был выдержан в базовом стиле описания.

Заметим, что понятия *цель* и *намерение* являются синонимами. Этим подтверждается взаимосвязь между *базовым* стилем описания и ориентацией *на цели* (задачи). Действительно, многие намерения исполнителей можно трактовать как вспомогательные цели.

Исполнители

Исполнитель (actor) — это сущность, обладающая поведением. К числу исполнителей может относиться и сама рассматриваемая система, если она вызывает службы других систем. В прецеденте могут участвовать основные и вспомогательные (второстепенные) исполнители. Исполнителями являются не только люди, но и организации, машины и программы. Существует три типа внешних по отношению к разрабатываемой системе исполнителей.

- Основной исполнитель (primary actor) — его задачи выполняются с

использованием системы. Примером основного исполнителя является кассир.

- Зачем его идентифицировать? Чтобы определить цели пользователя, на основе которых формулируются прецеденты.

• Вспомогательный исполнитель (supporting actor) — обслуживает систему (например, предоставляет информацию). Примером вспомогательного исполнителя является служба авторизации платежей.

- Зачем его идентифицировать? Чтобы определить внешние интерфейсы и протоколы.

• Закулисный исполнитель (offstage actor) — заинтересован в реализации прецедента, но не является основным или вспомогательным исполнителем. Примером закулисного исполнителя является налоговая служба.

- Зачем его идентифицировать? Чтобы удостовериться, что *все* интересы определены и удовлетворены. Интересы закулисных исполнителей обычно не очевидны и их легко упустить из виду, если не идентифицировать

Диаграммы прецедентов

На языке UML существует система обозначений для диаграммы прецедентов, иллюстрирующей имена прецедентов, исполнителей и взаимосвязи между ними (рис. 3).

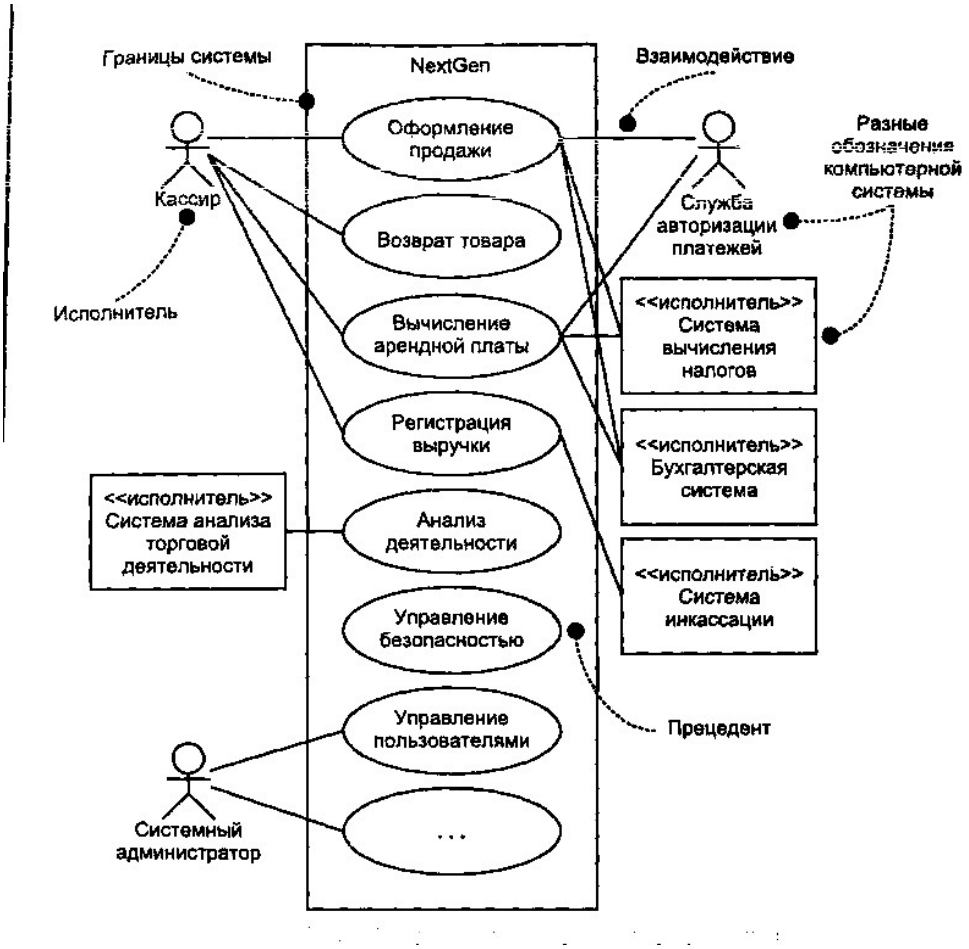


Рис. 3. *Фрагмент диаграммы прецедентов*

Диаграммы прецедентов и их взаимосвязей имеют второстепенное значение при работе над прецедентами. Прецеденты — это текстовые документы. Работать над прецедентами — значит составлять текстовые описания.

Обычно новички в области моделирования прецедентов начинают с составления диаграмм прецедентов и их взаимоотношений, а не с составления текстовых описаний. Однако специалисты по написанию прецедентов мирового класса, *к* числу которых относятся Андерсон (Andersen), Фовлер (Fowler), Кокбурн (Cockburn) и др., основное внимание уделяют составлению текстовых описаний, а не диаграмм, В таком ракурсе диаграмма лишь иллюстрирует способы использования системы внешними исполнителями.

Совет. Стройте простые диаграммы прецедентов в соответствии со списком исполнителей и их задач.

Система обозначений для диаграммы прецедентов

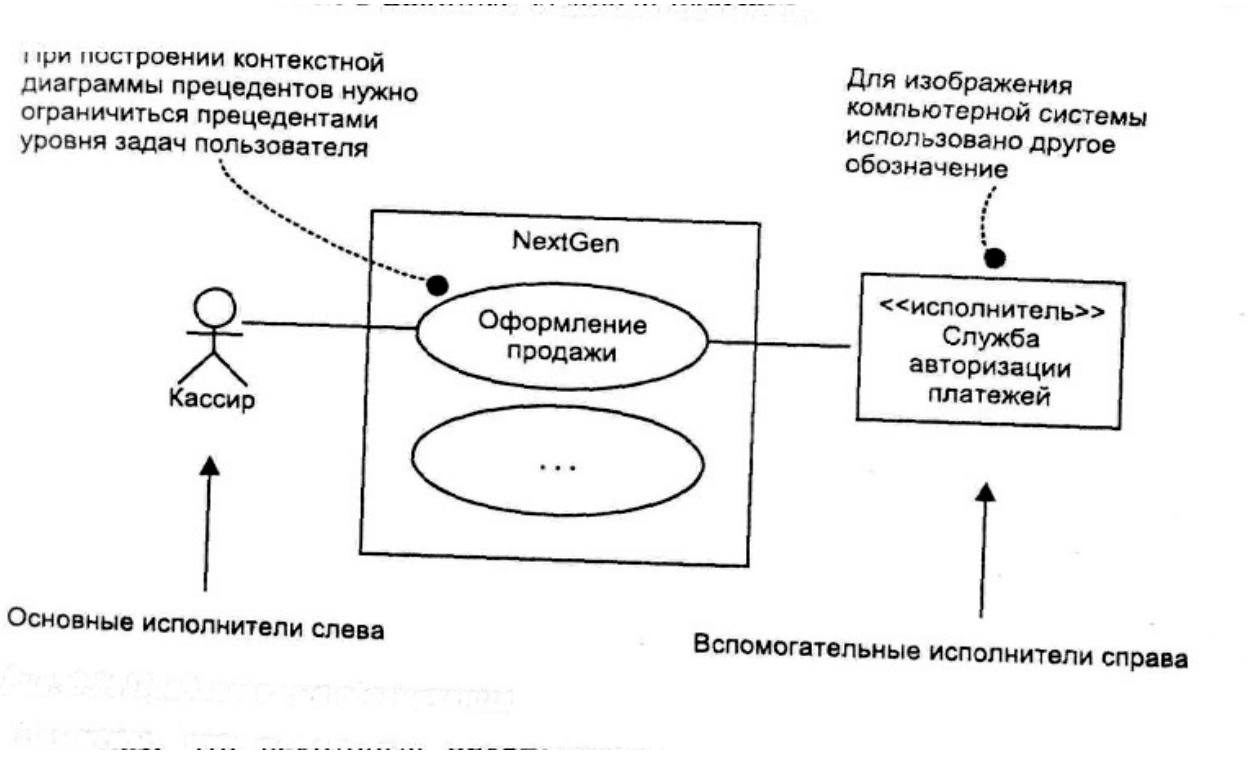


Рис. 4. Предлагаемые обозначения

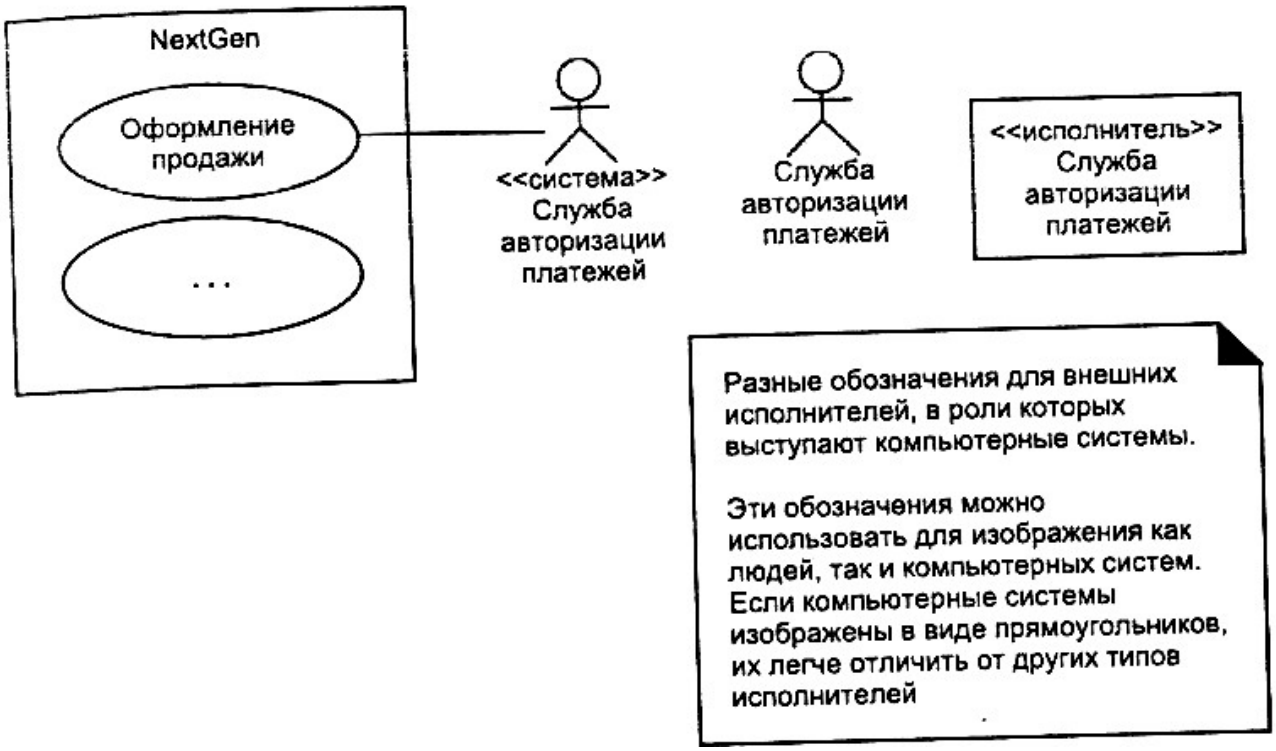


Рис. 5. Другие обозначения исполнителей

Требования и списки низкоуровневых свойств

Основным мотивом описания прецедентов является определение требований в контексте задач (целей) и сценариев использования системы. Это очень хорошо, поскольку позволяет углубить понимание системы. Однако прецеденты — это не единственные артефакты формулировки требований. Некоторые нефункциональные требования, правила и другие элементы лучше описать в дополнительной спецификации, рассмотрению которой будет посвящена следующая глава.

Основная идея — заменить детальные списки низкоуровневых свойств (которые обычно составлялись при использовании традиционного метода формулировки требований) прецедентами (за некоторыми исключениями). Эти списки были сгруппированы по функциональному назначению и выглядели примерно следующим образом.

Идентификатор	**Свойство**
Свойство 1.9	Система будет позволять вводить новые идентификаторы
Свойство 2.4	Система будет регистрировать платежи по кредитной карточке в бухгалтерской системе

Такие списки тоже могут оказаться полезными, однако один подобный список занимает не полстраницы, а десятки или сотни страниц, Это вызывает некоторые

проблемы, решить которые позволяют прецеденты. К числу проблем относятся следующие,

- Длинные, подробные списки функций не отражают требований в общем контексте. Различные функции и свойства организованы в обычный список разрозненных элементов. А при описании прецедентов требования рассматриваются в контексте задач и целей системы.
- Если используются и описания прецедентов и списки функций, то происходит дублирование информации. На это затрачивается больше труда, требуется больше усилий для их осмысления и согласования.

Совет. Старайтесь заменить подробные списки низкоуровневых свойств описанием прецедентов.

Списки высокоуровневых свойств системы вполне допустимы

Достаточно часто функциональные свойства системы описываются в кратких списках свойств высокого уровня в рамках документа "В*и*дение". Это вполне оправдано. В отличие от списка низкоуровневых свойств, перечисленных на 100 страницах, список основных свойств системы должен включать не более нескольких десятков элементов. В нем перечислены лишь функциональные свойства системы, не упомянутые при описании прецедентов.

Список свойств системы

- Регистрация продаж.
- Авторизация платежей (кредитных, дебитных, чековых).
- Системное администрирование для управления пользователями, безопасностью, таблицами кодов и констант и т.п.
- Автоматическая обработка информации о продажах в фоновом режиме в случае отказа внешних компонентов.
- Взаимодействие в реальном времени с внешними системами (на основе промышленных стандартов), включая систему складского учета, вычисления налоговых отчислений, бухгалтерскую систему, систему управления человеческими ресурсами и службы авторизации платежей.
- Определение и выполнение настраиваемых бизнес-правил в фиксированных точках сценария.
- ...

Когда нужен подробный список свойств?

Иногда прецедентов действительно недостаточно. Для некоторых систем нужно разработать детальный список свойств. Например, серверы приложений, системы управления базами данных и другие базовые системы нужно рассматривать в

терминах *свойств* ("Необходима поддержка следующей версии XML"). Прецеденты не годятся для разработки таких приложений.

Прецеденты в рамках унифицированного процесса

Прецеденты играют жизненно важную роль при реализации унифицированного процесса, поскольку вся разработка в рамках этого подхода осуществляется *под управлением прецедентов* (use-case driven development). Это означает следующее.

- Требования в основном формулируются при описании прецедентов (в модели прецедентов). Остальные требования (если таковые существуют) являются либо техническими (например, список функций), либо второстепенными.
- Прецеденты — важный этап итеративного планирования. На каждой итерации реализуются некоторые сценарии или целые прецеденты. Поэтому описания прецедентов вносят существенный вклад в оценивание результата.
- Разработка приложения состоит в реализации прецедентов. То есть группа разработчиков продумывает способы взаимодействия объектов или архитектуру подсистем для реализации прецедентов.

В рамках UP выделяют два вида прецедентов: системные и бизнес-прецеденты. ***Системные прецеденты*** (system use-case) — это такие, которые рассматривались в этой главе, например Оформление продажи. Они создаются в рамках дисциплины "Требования" и являются частью модели прецедентов.

Бизнес-прецеденты (business use-case) используются гораздо реже. При необходимости они создаются в рамках дисциплины "Бизнес-моделирование" как часть крупномасштабного бизнес-процесса или для облегчения понимания контекста новой системы. Они описывают последовательность действий в целом, выполняемых ***бизнес-исполнителем*** (business actor) (исполнителем в бизнес-среде, например, потребителем). В частности, для ресторана можно выделить бизнес-прецедент **Приготовление блюда**.

Прецеденты и спецификация требований в рамках итеративного процесса

В этом разделе мы вернемся к основной идее UP и итеративной разработки. Рассмотрим временные рамки, выделяемые на формулировку требований на каждой итерации. В табл. 2 представлен пример (не претендующий на рекомендацию) определения требований в процессе реализации стратегии UP.

Обратите внимание, что группа технической разработки приступает к созданию

ядра системы в тот момент, когда детализированы лишь 10% требований, далее следует намеренная задержка, и формулировка требований откладывается до конца первой итерации стадии развития.

В этом состоит ключевое отличие от однопроходного процесса — промышленная разработка ядра системы начинается достаточно быстро, задолго до полного завершения формулировки требований.

Обратите внимание, что в конце первой итерации стадии развития проводится второй семинар по формулировке требований, по окончании которого сШ7о прецедентов описаны подробно. При таком планировании можно учесть обратную связь и оценки реализованной части системы. Обратная связь поступает от пользователей, специалистов по тестированию и позволяет "познать непознанное". То есть в процессе разработки системы очень быстро возникают вопросы, требующие немедленного решения.

Прецеденты начальной фазы

На начальной стадии не все прецеденты описываются детально. В этот период для описания прецедентов целесообразно организовать двухдневный семинар. В первой половине первого дня можно определить задачи и круг заинтересованных лиц, а также обсудить, какие задачи относятся к проекту, *а* какие выходят за его рамки. С помощью компьютерного проектора можно составить и демонстрировать таблицу исполнителей и их задач. Можно приступить к построению диаграммы прецедентов. В течение нескольких часов вполне реально определить порядка 20 задач пользователей (и соответствующих им прецедентов), в том числе прецеденты **Оформление продажи, Возврат товара** и т.д. За это время большинство интересных, сложных прецедентов с высоким уровнем риска должны быть описаны в сжатом формате. На описание каждого такого прецедента понадобится около 2 минут. Разработчики могут приступить к составлению высокоуровневой схемы функциональности системы.

После этого 10-20% прецедентов, представляющих сложные функции или высокую степень риска, нужно описать подробно. При этом разработчики лучше оценят сложность проекта, его рамки, выявят проблемы, возникающие в процессе разработки. Самыми важными прецедентами являются, пожалуй, **Оформление продажи** и **Возврат товара.**

Для описания прецедентов нужно использовать средства управления требованиями со встроенным текстовым процессором, а результаты работы желательно де-

монстрировать участникам семинара с помощью проектора. Для этих прецедентов нужно составить список заинтересованных лиц и их интересов. Это поможет глубже понять неочевидные функциональные и основные нефункциональные требования, в частности к надежности и производительности системы.

Задачей такого анализа является не исчерпывающее описание прецедентов, а более глубокое понимание проблемы.

Спонсоры проекта должны решить, стоит ли вкладывать деньги в дальнейшие исследования (на стадии развития). На начальной стадии не ставится задача напрямую исследовать целесообразность капиталовложений. На этом этапе нvжно определить масштаб проекта, основные риски, реалистичность этого проекта — т.е. получить исчерпывающую информацию для принятия последующих решений о продолжении или прекращении проекта.

Предположим, начальная фаза проекта Алтын заняла 5 дней. В результате двухдневного семинара и последующей работы по анализу прецедентов было принято решение о целесообразности проекта.

Прецеденты на стадии развития

Фаза развития включает несколько итераций фиксированной длительности (скажем, четыре недели), в течение которых реализуются наиболее рискованные, значимые и архитектурно важные части системы, а также идентифицируется и проясняется большая часть требований. Благодаря наличию обратной связи разработчики лучше понимают требования, постепенно уточняют и конкретизируют их. На каждой итерации желательно провести двухдневный семинар. Однако на каждом семинаре не нужно обсуждать все прецеденты. Их нужно упорядочить по приоритетности и сначала рассматривать самые важные прецеденты.

На каждом следующем семинаре нужно дорабатывать и уточнять основные требования. На начальных итерациях процесс изменения требований может происходить достаточно интенсивно. Однако в дальнейшем он стабилизируется. В итеративном процессе происходит постепенная доработка наиболее важных частей системы.

На каждом семинаре список задач пользователей и прецедентов постепенно уточняется. По окончании фазы развития большинство прецедентов должны быть описаны или переписаны в развернутом формате (примерно 80-90%). Если для POS-системы выделено 20 прецедентов, то как минимум 15 наиболее сложных и рискованных из них должны быть определены полностью.

Заметим, что на стадии развития разработка части системы доводится до программной реализации. Поэтому по окончании этой фазы команда разработчиков системы Алтын будет иметь не только достаточно полно определенные прецеденты, но и качественный исполняемый код.

Прецеденты на стадии конструирования

Этап конструирования состоит из некоторого числа итераций фиксированной длительности (например, 20 итераций по 2 недели каждая), в течение которых основное внимание уделяется доработке системы, поскольку наиболее принципиальные вопросы уже решены на стадии развития. На этом этапе тоже некоторое внимание уделяется описанию прецедентов, однако значительно Меньшее, чем на стадии развития. К этому моменту большая часть функциональных и нефункциональных требований должна быть постепенно стабилизирована. Это не означает необходимости замораживания требований и завершения исследований. Однако теперь степень модификации требований от итерации к итерации значительно снижается.

Пример: прецеденты начальной фазы проекта Алтын

На начальной стадии не все прецеденты описываются детально. Модель прецедентов на этом этапе может быть разработана на следующем уровне детализации.

В развернутом формате	**В свободном формате**	**Краткое описание**
Оформление продажи Возврат товара	Вычисление арендной платы Анализ торговой деятельности	Регистрация выручки Управление пользователями Запуск системы Завершение работы Управление системными таблицами

Определение остальных требований

Введение

Для определения требований одного описания прецедентов недостаточно. Необходимо определить и другие виды требований, в частности, соглашения о лицен-

зировании, возможностях поддержки системы и т.д. Все эти требования описываются в ***дополнительной спецификации*** (supplementary specification).

В ***словарь терминов*** (glossary) включаются термины и определения. Он также может служить словарем данных.

Документ "В*и*дение" определяет видение проекта. В нем кратко описываются основные цели проекта, проблемы, указывается круг заинтересованных лиц, их потребности, а также основные идеи предложенного решения.

"Документ "В*и*дение" определяет точку зрения заинтересованных лиц на разрабатываемый продукт в терминах их основных потребностей и свойств продукта. Этот документ содержит описание неочевидных основных требований и составляет основу для более детального описания технических требований".

Примеры для системы Алтын

Целью последующих примеров является не подробное составление документов "Видение", "Словарь терминов" п "Дополнительная спецификация". Задачей этой книги является изучение объектного проектирования, анализа требований в контексте описания прецедентов и объектно-ориентированного анализа, а не изучение проблематики POS-систем или разработка конкретной системы. Поэтому основной пример этой книги будет упомянут только в некоторых разделах, чтобы сохранить последовательность изложения материала, выделить основные моменты и обеспечить быстрое продвижение вперед.

Пример Алтын: фрагмент дополнительной спецификации

Дополнительная спецификация

Введение

В этом документе описаны все требования к POS-системе Алтын, не вошедшие в описание прецедентов.

Функциональность

(Имеющая отношение *ко многим прецедентам)*

Регистрация событий и обработка ошибок

Все ошибки регистрируются на постоянном носителе.

Подключаемые бизнес-правила

Необходимо обеспечить возможность настройки функциональности системы в различных точках сценариев нескольких прецедентов (эти точки нужно определить) на основе заданных правил.

Безопасность

Необходимо выполнять аутентификацию всех пользователей.

Удобство использования

Человеческие факторы

Пользователь POS-системы будет работать с большим монитором, поэтому необходимо следующее.

- Текст должен быть виден с расстояния 1 м
- Нужно избегать мерцающих цветов

Быстрая, простая и корректная обработка информации — вот главные принципы системы автоматизации торговли, поскольку пользователь хочет поскорее совершить покупку. В противном случае ему не понравится этот магазин (и продавец). Кассир зачастую смотрит не на экран компьютера, а на покупателя или товары. Поэтому предупреждающие сообщения нужно сопровождать звуковыми сигналами, а не только графически отображать на экране.

Надежность

Возможность восстановления информации

При сбоях в работе внешних систем (службы авторизации платежей, бухгалтерской системы и т.д.) нужно обеспечить возможность локальной обработки данных (их сохранение и последующую передачу внешним системам). Этот вопрос требует более детальной проработки.

Производительность

Покупатель хочет сделать покупку *как можно скорее.* Задержка этого процесса может быть связана с внешней службой авторизации. Наша задача — выполнить авторизацию не более чем за 1 минуту в 90% случаев.

Возможности поддержки

Адаптация системы

Различные пользователи POS-системы Алтын могут устанавливать свои бизнес-правила для обработки данных о продажах. Поэтому в нескольких заранее определенных точках сценария (например, при инициализации новой продажи или при добавлении нового наименования товара) нужно обеспечить возможность подключения бизнес-правил.

Конфигурирование

Сетевые конфигурации различных пользователей POS-системы могут отличаться. Могут использоваться архитектуры "тонкого" и "толстого" клиентов, двухуровневые и многоуровневые архитектуры и т.д. Кроме того, конфигурация ресурсов каждого клиента может изменяться со временем, отражая производственные потребности и потребности в производительности. Следовательно, система должна быть настраиваемой и отражать потребности пользователей. Этот вопрос требует тщательной дополнительной проработки, изучения степени гибкости и способов ее достижения.

Ограничения

Руководство проекта Алтын настаивает на применении технологии Java, поскольку это улучшит возможности по поддержке системы и ее переходу на различные платформы, а также обеспечит простоту разработки,

Приобретаемые компоненты

- Система вычисления налоговых платежей. Разрабатываемая система должна

поддерживать работу с подключаемыми внешними системами различных стандартов для разных стран.

Бесплатные компоненты на основе открытого кода

В целом, рекомендуется максимальное использование в этом проекте компонентов на основе открытого кода в рамках Java-технологии.

Несмотря на то, что пока преждевременно определять проектные решения, предлагаются следующие варианты.

- Контур регистрации Jlog

Интерфейсы

Важные интерфейсы и аппаратные средства

- Сенсорный монитор (воспринимаемый операционной системой как обычный монитор; прикосновения обрабатываются как события мыши).
- Лазерный сканер для считывания штрих-кодон (ои«чмп подключаемый к специальной клавиатуре; считанный код оЬраЬатывается как последовательность нажатия клавиш).
- Устройство для печати чеков.
- Устройство считывания данных с кредитной/дебитной карточки.
- Устройство считывания подписи (не в первой версии системы).

Бизнес-правила

Имя	Правило	Возможность изменения
ПРАВ1	Для платежей по кредитной карточке требуется подпись	Подпись покупателя будет нужна, но через 2 года большинство пользователей захотят применять цифровое устройство для ввода подписи, *з,* через пять лет может понадобиться поддержка уникальной цифровой закодированной подписи, введенной в настоящее время в США
ПРАВ2	Правила вычисления налоговых платежей. С продаж отчисляются налоги. Правила налогообложения изложены в официальных документах	
ПРАВ3	При возврате товара, купленного по кредитной карточке, возврат денег осуществляется не наличными, а путем перевода на кредитную карточку	

Вопросы законодательства

Рекомендуется использование бесплатных компонентов на основе открытых систем.

Высокая вероятность изменения. В каждой торговой организации используются свои правила, которые могут изменяться ежедневно и ежечасно

Информация из предметной области

Ценовая политика

Помимо описанных выше правил ценообразования, каждому товару могут соответствовать две цены; *исходная* и *постоянная сниженная цена.* Указанная цена товара (без вычисления положенных скидок) соответствует постоянной сниженной цене, если таковая имеется. Однако организации сохраняют исходную цену даже при наличии постоянной сниженной для бухгалтерской отчетности.

Обработка платежей по кредитной и дебитной карточке

Если электронные платежи по кредитной или дебитной карточке подтверждены службой авторизации платежей, то за них несет ответственность эта служба, а не сам покупатель. Следовательно, для каждого платежа продавец должен зафиксировать получение денежных сумм от службы авторизации. Обычно перевод денег за все выполненные в текущий день платежи на счет торговой организации выполняется в ночное время, при этом за каждую транзакцию служба взимает небольшой взнос.

Вычисление налогов

Налоги могут вычисляться по сложным схемам, и суммы отчислений могут часто изменяться на правительственном уровне. Поэтому желательно возложить задачу вычисления налоговых платежей на отдельную программу. Налоги могут взиматься городскими, региональными властями или на уровне государства, Некоторые ставки налогов могут зависеть от категории покупателя или назначения товара (например, товары для детей или фермеров).

Дополнительная спецификация (комментарии)

В дополнительной спецификации содержатся требования, ограничения и другая информация, не вошедшая в описание прецедентов или словарь терминов, включая атрибуты качества и специальные требования. Заметим, что требования, связанные с прецедентами, должны быть представлены в описаниях прецедентов в разделе "Специальные требования", однако некоторые предпочитают также включать их в дополнительную спецификацию. В дополнительную спецификацию можно включать следующие элементы.

- Требования согласно модели FURPS+ — функциональные, требования к удобству использования, надежности, производительности и возможности поддержки.

- Отчеты.
- Ограничения на аппаратные и программные средства (операционные и сетевые системы и т.д.).
- Ограничения, накладываемые на процесс разработки (например, процесс или средства разработки).
- Другие ограничения проектирования или реализации.
- Международные соглашения (единицы измерения, языки и т.д.). В Документация (пользовательская, руководство по установке и администрированию) и справочная информация.
- Соглашения о лицензировании или другие юридические соглашения.
- Разбиение на пакеты.
- Стандарты (технические, обеспечения качества и безопасности).
- Физические требования к окружению (например, температурный режим эксплуатации или ограничения на вибрацию).
- Операционные требования (например, способ обработки ошибок, частота архивации).
- Информация из предметной области (например, о полном цикле обработки платежа по кредитной карточке).

Ограничения (constraints) относятся не к поведению системы, а к проектированию. Они тоже являются требованиями, но название указывает на их ограничительный характер. Приведем следующий пример.

Необходимо использовать продукты Oracle (поскольку с этой компанией существует лицензионное соглашение).

Система должна работать в операционной системе Linux (она бесплатна).

Не стоит принимать проектные решения и устанавливать ограничения на ранних этапах разработки, особенно на начальной стадии, когда имеется слишком мало информации о системе. Однако некоторые ограничения неизбежны, например, интерфейсы с внешними системами или юридические ограничения.

Атрибуты качества

Некоторые требования называются ***атрибутами качества*** (quality attributes) системы. К ним относятся удобство использования, надежность и т.д. Заметим, что это качественные характеристики системы, однако не все они должны обеспечивать высокое качество соответствующего процесса (слово "качество" имеет два разных значения). Например, возможности поддержки (качественная характеристика) могут быть низкими, если данный программный продукт не предназначен для долгосрочной эксплуатации.

Атрибуты качества делятся на два типа.

1. Наблюдаемые в процессе работы системы (функциональность, удобство

использования, надежность, производительность и т.д.).

2. Ненаблюдаемые в процессе работы системы (возможность поддержки, удобство тестирования и т.д.).

Функциональность системы определяется в описании прецедентов (например, требования к производительности указываются при описании прецедента **Оформление продажи)**.

Другие атрибуты качества, согласно модели FURPS+, определяются в дополнительной спецификации.

Несмотря на то, что функциональность системы относится к качественным характеристикам, ее зачастую не включают в число атрибутов качества. Однако термин "атрибуты качества" не является синонимом термина "нефункциональные требования". Последнее понятие гораздо шире, поскольку включает в себя все характеристики, за исключением функциональности (например, лицензионные соглашения или соглашения о пакетировании).

Атрибуты качества (а значит, и дополнительная спецификация, в которой они описываются) относятся к сфере интересов системного архитектора, поскольку архитектурный анализ и проектирование подразумевают идентификацию и обоснование атрибутов качества в контексте функциональных требований. Например, предположим, что одним из атрибутов качества системы Алтын должна быть ее абсолютная надежность при сбоях внешних систем. С точки зрения архитектора, это требование накладывает отпечаток на крупномасштабные проектные решения.

Все атрибуты качества взаимосвязаны. В качестве простого примера можно рассмотреть требования "высокой надежности" и "простоты тестирования". Эти требования несколько противоречивы, поскольку для распределенной системы существует множество причин и способов выхода из строя.

Бизнес-правила

Бизнес-правила или правила предметной области определяют процессы в предметной области. Это требования не к конкретному приложению, а ко всем приложениям для данной предметной области. Типичными бизнес-правилами являются политика компаний, а также физические или государственные законы.

Бизнес-правила влияют на другие требования к системе, обычно определяемые прецедентами, поскольку они проясняют многие неясные при описании прецедентов моменты. Например, если при описании прецедента Оформление продажи кому-то взбредет в голову осуществлять платежи по кредитной карточке без подписи

покупателя, то такая возможность будет пресечена наличием бизнес-правила, учитывающего требования служб авторизации платежей (ПРАВ1).

Предупреждение. Правила — это не требования к приложению. Не включайте в список правил свойства системы. Правила описывают процессы из предметной области, а также их ограничения, а не свойства приложения.

Информация из предметной области

Зачастую очень важно внести в дополнительную спецификацию некоторую информацию из предметной области, имеющую отношение к разрабатываемой системе (продажам и бухгалтерскому учету, геофизических свойствах потоков нефти, воды или газа и т.д.) и позволяющую разработчикам глубже понять происходящие процессы. В этот раздел можно включать ссылки на литературу, мнения экспертов, формулы, законы к другие ссылки.

Пример для системы Алтын: видение (фрагмент)

Введение

Нам видится надежное приложение автоматизации розничной торговли следующего поколения (POS-система Алтын), обеспечивающее гибкую поддержку различных бизнес-правил, механизмы поддержки различных терминалов и интерфейсов пользователя, а также интеграцию с различными внешними вспомогательными системами. *Анализ в этом примере носит иллюстративный характер.*

Позиционирование

Экономические предпосылки

Существующие программные продукты не обеспечивают настройку на потребности различных пользователей, в частности добавление различных бизнес-правил или поддержку разных сетевых архитектур (например, на основе "толстого" или "тонкого" клиента, двух-, трех- или четырехуровневые архитектуры). Кроме того, они плохо масштабируются. Ни одна из известных систем не обеспечивает автоматический переход из интерактивного в автономный режим при сбоях внешних систем, Отсутствует простая возможность интеграции с внешними системами. Существующие системы не поддерживают новые терминальные технологии. Негибкость существующих систем открывает новую нишу на рынке программного обеспечения POS-систем.

Формулировка проблемы

Традиционные POS-системы не обладают гибкостью, неустойчивы к сбоям и не обеспечивают интеграцию с внешними системами. Это приводит к проблемам с оформлением продаж, несоответствию программного обеспечения экономическим потребностям предприятий, невозможности точной и своевременной обработки данных и поддержки планирования. Эти проблемы касаются кассиров, менеджеров

по продажам, системных администраторов и руководителей предприятий.

Место системы

Указать, для кого предназначена система, описать ее свойства и отличия от продуктов конкурирующих организаций.

Заинтересованные лица

Необходимо определить, для кого предназначена система и каковы проблемы заинтересованных лиц.

Демографические особенности рынка...

Заинтересованные лица, не являющиеся пользователями системы...

Пользователи системы...

Основные задачи высокого уровня и проблемы заинтересованных лиц

Необходимо объединить информацию из списка исполнителей и задач, *а также из раздела* описания *прецедентов, отражающего потребности заинтересованных лиц.*

Задачи уровня пользователя

Сюда можно включить список исполнителей и их задач, разработанный в процессе моделирования прецедентов, либо более сжатую информацию. Пользователи (и внешние системы) используют данную систему в таких целях.

Кассир. Оформляет продажи, возврат товаров, регистрирует выручку.

Системный администратор. Управляет пользователями, безопасностью и системными таблицами.

Менеджер. Осуществляет запуск и завершает работу системы.

Система анализа торговой деятельности. Анализирует данные о продажах.

Окружение... Обзор

Перспективы продукта

Система Алтын обычно будет устанавливаться в магазинах, при использовании мобильных терминалов они будут располагаться вблизи сети магазинов, либо внутри магазинов. Система будет обслуживать пользователей и взаимодействовать с другими системами, как показано на рис. Видение 1. *Диаграмма строится на основе диаграммы прецедентов.*

Контекстные диаграммы можно строить в различных форматах с разной степенью детализации, но все они отражают взаимодействие внешних исполнителей с системой.

Преимущества системы

Подобно перечню исполнителей и их задач, в этой таблице указаны задачи, их решения и преимущества, однако на более высоком уровне, чем при описании

прецедентов.

Здесь описывается основное значение и отличительные свойства продукта.

Свойство	**Преимущества для заинтересованных лиц**
Система будет обеспечивать всю основную функциональность, необходимую торговым организациям, включая обработку информации о продажах, авторизацию платежей, оформление возврата платежей	Быстрая работа торговых точек в автоматическом режиме

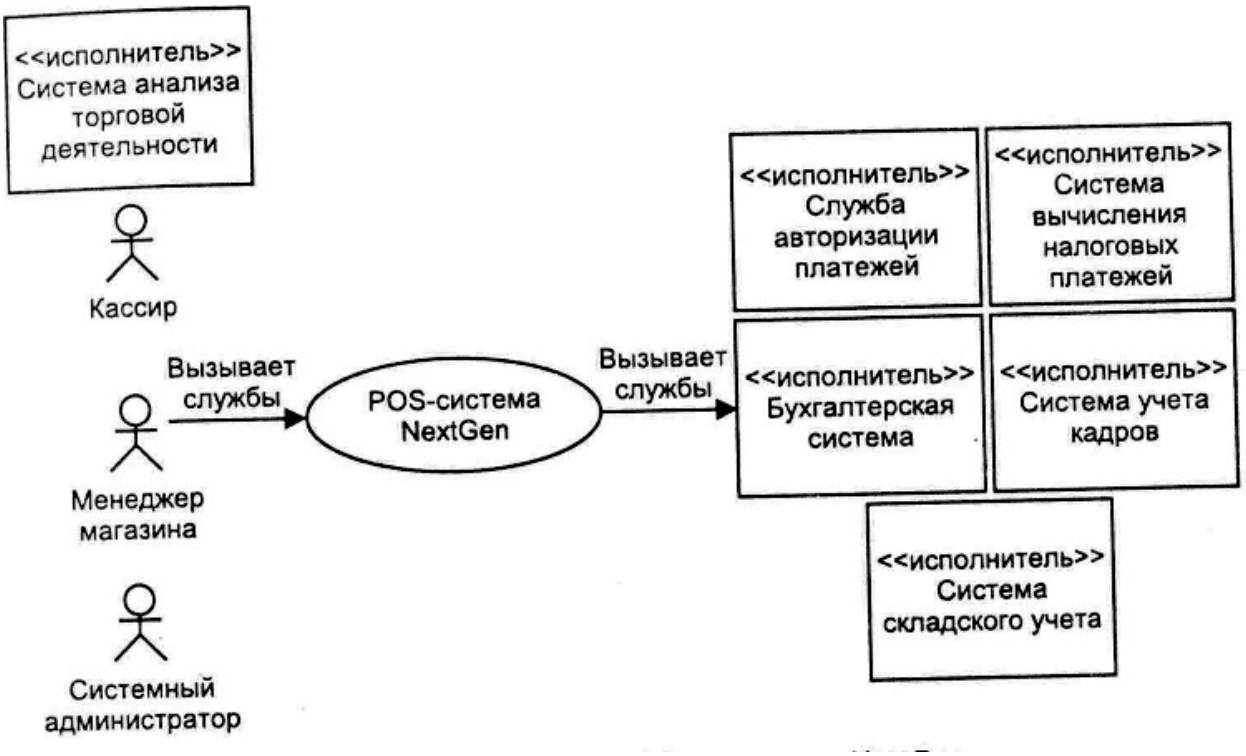


Рис. Видение. Контекстная диаграмма POS-системы Алтын

***Предположения* и зависимости...**

Стоимость и ценообразование...

Лицензирование и установка...

Основные свойства системы

Как было упомянуто выше, свойства системы описываются сжато путем перечисления основных функции.

Оформление продаж.

Авторизация платежей (по кредитной или дебитной карточке, чеком).

Системное администрирование и управление пользователями, безопасностью,

таблицами констант

Автоматический переход в автономный режим работы при выходе из строя *внешних систем*

Транзакции в реальном времени на основе промышленных стандартов с внешними системами, включая бухгалтерскую систему, систему складского учета, учета человеческих ресурсов, вычисления налогов, службы авторизации платежей. Определение и выполнение настраиваемых бизнес-правил в фиксированных точках выполнения сценариев.

Другие требования и ограничения

Видение (комментарий)

Ту ли проблему мы решаем? Формулировка проблемы

На начальной стадии разработки при определении требований необходимо кратко сформулировать суть проблемы. Это позволит избежать недопонимания между заинтересованными лицами. Зачастую разные участники проекта ставят перед собой различные задачи.

В шаблоне RUP для формулировки проблемы предлагается использовать не текстовый, а табличный формат.

Проблема состоит в	
Затрагивает	
Влияет на	
Ее успешное решение основывается на	

Основные высокоуровневые цели и потребности заинтересованных лиц

В этой таблице указываются задачи и проблемы более высокого уровня, чем при описании прецедентов, а также нефункциональные цели, затрагивающие несколько различных прецедентов, например, следующие.

- Необходимо обеспечить бесперебойное оформление продаж.
- Необходимо обеспечить возможность корректировки бизнес-правил.

Каковы основные проблемы и цели?

Заинтересованные лица зачастую формулируют свои потребности в форме предполагаемых решений, например: "Нам необходим штатный программист для настройки бизнес-правил при их изменении". Такие решения иногда оправданны, по-

скольку заинтересованные лица хорошо понимают проблемы своей предметной области и возможные пути их решения. Однако иногда возникают попытки предложить решения, которые являются неоптимальными или выходят за рамки основной проблемы.

Поэтому проблему и основные цели должен исследовать системный аналитик. Как было описано в предыдущих главах, он сможет выстроить иерархию проблем и определить приоритеты их решения.

Методы групповой работы

В процессе определения высокоуровневых требований и целей большое внимание уделяется групповой работе. Вот некоторые приемы групповой работы, облегчающие процесс изучения проблемы и определения основных задач: построение диаграмм Парето, многовариантное голосование, точечное голосование, номинальный групповой процесс, мозговой штурм и группировка по признаку подобия. Информацию об этих и других приемах можно почерпнуть из Internet. Автор предпочитает применять несколько перечисленных методик в процессе одного семинара для изучения общих проблем и требований с разных точек зрения.

Прецеденты — не единственный способ выражения функциональных требований.

Совет. В документ "Видение" желательно включать менее 50 свойств. Если их больше, попробуйте сгруппировать некоторые из них и сформулировать на более высоком уровне абстракции.

Другие требования в документе "Видение"

В документе "Видение" системные свойства вкратце отражают функциональные требования, которые более детально изложены в описании прецедентов. В этом документе можно также отразить другие требования (например, к надежности и удобству в использовании), которые более подробно описаны в разделе "Специальные требования" модели прецедентов и в дополнительной спецификации. Однако здесь возникает риск неоправданного дублирования. Например, в шаблонах продуктов, поддерживающих RUP, содержатся идентичные или схожие разделы требований к надежности, производительности, удобству в использовании и т.д. При таком дублировании очень сложно отслеживать изменения во всех документах. Кроме того, при этом нужен одинаковый уровень детализации дополнительной спецификации и документа "Видение". То есть краткие и детальные описания требований становятся идентичными.

Совет. Старайтесь избегать дублирования требований в дополнительной спецификации, документе "Видение" и описании прецедентов. Лучше сформулировать их в дополнительной спецификации или описании прецедентов, а в документе "Видение" сделать ссылку на эти описания.

Этот совет позволит упростить разработку моделей. Однако если вы предпочитаете использовать стандартные шаблоны, то это тоже не запрещается.

Видение, свойства или прецеденты — что раньше?

О порядке разработки артефактов говорить неуместно. Различные артефакты, относящиеся к требованиям, создаются параллельно. Тем не менее, можно порекомендовать такую последовательность разработки.

1. Создайте краткий черновой вариант документа "Видение".
2. Идентифицируйте задачи пользователей и соответствующие прецеденты.
3. Опишите некоторые прецеденты и приступите к разработке дополнительной спецификации.
4. На основе полученной информации уточните документ "Видение".

Комментарии: словарь терминов

В простейшем варианте ***словарь терминов*** (glossary) представляет собой список важных понятий и их определение. Очень часто возникает ситуация, когда технический или другой термин используется заинтересованными лицами в несколько различных значениях. Такие противоречия необходимо разрешить для облегчения общения и однозначной формулировки требований.

Совет. Начинайте разработку словаря терминов на ранних этапах выполнения проекта. Мне приходилось работать в коллективах, где один и тот же термин по-разному трактовался различными разработчиками.

В словарь терминов нужно вносить не все возможные понятия, а только неясные, неоднозначные или требующие дополнительного изучения, например, информацию о форматировании или правила верификации.

Словарь терминов в роли словаря данных

В рамках UP словарь терминов также играет роль *словаря данных* (data dictionary) — документа, содержащего информацию о данных. На начальной стадии проекта словарь терминов включает лишь основные термины и их описание, а на стадии развития его можно превратить в словарь данных.

К атрибутам терминов относятся следующие.

- Синонимы
- Описание
- Формат (тип, длина, единицы измерения)
- Взаимосвязи с другими элементами
- Диапазон значений
- Правила проверки корректности

Составные термины

В словарь терминов заносятся не только простейшие понятия, такие как "цена товара". В него необходимо включать и сложные элементы, например, "продажа" (это понятие включает в себя другие элементы, такие как дата и место продажи), а также аббревиатуры, используемые для описания передачи данных между исполнителями в рамках одного прецедента. Например, рассмотрим следующий фрагмент описания прецедента Оформление продажи.

Система отправляет *запрос на авторизацию платежа* внешней службе авторизации платежей и запрос на подтверждение платежа. Термин "запрос на авторизацию платежа" обозначает набор данных, содержание которых необходимо пояснить в словаре терминов.

Надежные спецификации — нет ли здесь противоречия?

При описании требований может сложиться впечатление, что реальные требования уже хорошо определены и вполне понятны и их можно использовать для надежного планирования и объективного оценивания состояния проекта. Однако это иллюзия. Разработчики программного обеспечения из собственного опыта знают, насколько это не так. Об этом свидетельствует и высказывание Гете, выбранное в качестве эпиграфа к данной главе.

На самом деле реальное значение имеет лишь соответствие программы тестам, определенным пользователями и заинтересованными лицами, а также выполнение поставленных задач (которые зачастую нельзя четко сформулировать до начала работы с программой).

Создание дополнительной спецификации и документа "Видение" — это упражнение, позволяющее несколько прояснить поставленные задачи, обосновать назначение продукта и описать основные идеи. Однако эти документы, как и любой артефакт дисциплины определения требований; не являются надежной

спецификацией. Только в процессе написания кода, его тестирования, получения обратной связи, взаимодействия с пользователями и потребителями можно составить реальное представление о сущности системы.

Это не призыв к отказу от анализа и быстрому написанию кода, а лишь рекомендации по правильному восприятию требований и их постоянной доработке.

Не слишком ли много UML на начальной стадии проекта?

Основная задача начальной фазы — накопить достаточно информации для составления общего видения проекта, принятия решения о его достижимости и целесообразности. Поэтому на этом этапе не требуется создавать подробных диаграмм в системе обозначений языка UML за исключением простых диаграмм прецедентов. На данном этапе необходимо сосредоточить внимание на осмыслении масштаба проекта и 10% требований. Разрабатываемые документы являются текстовыми. Большая часть диаграмм приходится на следующую фазу — фазу развития.