

ПРОГРАММНАЯ ИНЖЕНЕРИЯ

МОДЕЛИРОВАНИЕ ПРЕЦЕДЕНТОВ

ОСНОВЫ ПРЕЦЕДЕНТОВ

ЧТО ТАКОЕ МОДЕЛИРОВАНИЕ ПРЕЦЕДЕНТОВ?

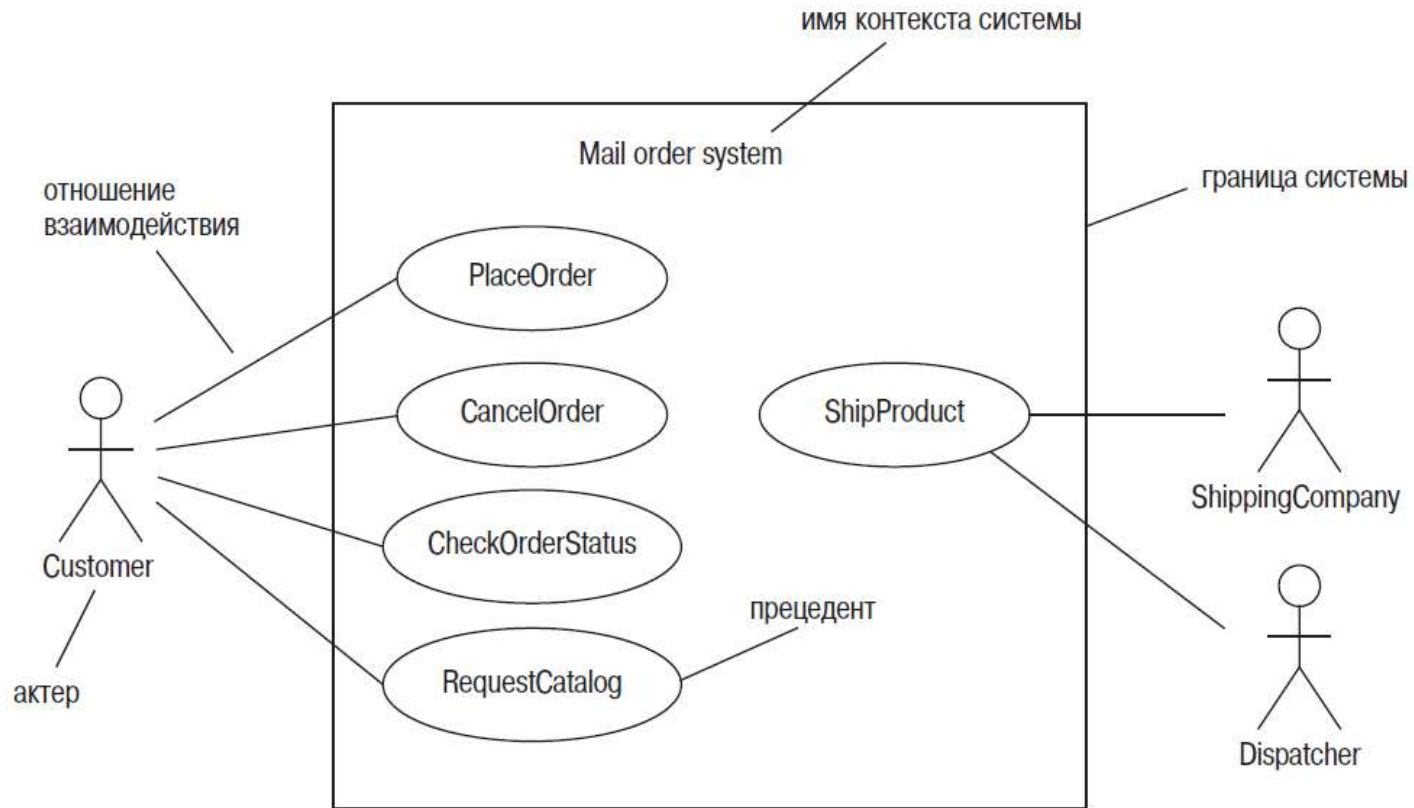
- ◎ **Моделирование прецедентов – это форма выработки требований.**
- ◎ **Процесс моделирования прецедентов:**
 1. Установить границы системы
 2. Выявить актеров
 3. Выявить прецеденты
 4. Повторить, пока не стабилизируются

МОДЕЛЬ ПРЕЦЕДЕНТОВ

Состоит из 4-х компонентов:

1. **Граница системы** – прямоугольник, очерчивающий прецеденты для обозначения границы моделируемой системы
2. **Актеры** – роли, выполняемые людьми или сущностями, использующими систему
3. **Прецеденты** - то, что актеры могут делать с системой
4. **Отношения** – отношения между актерами и прецедентами

ПРИМЕР ДИАГРАММЫ ПРЕЦЕДЕНТОВ



ГРАНИЦЫ СИСТЕМЫ

- ◎ **Границы системы (контекст системы в терминах UML)** отделяет систему от всего остального мира.
- ◎ Из-за неясности границ системы возникают серьезные проблемы при проектировании.
- ◎ Контекст изображается в виде прямоугольника с именем системы. *Актеры размещаются вне границ блока, а прецеденты – внутри*

АКТЕРЫ

- ◎ **Актеры** – это роли, исполняемые сущностями, непосредственно взаимодействующими с системой.
- ◎ Одна и та же роль может исполняться многими разными сущностями одновременно либо последовательно во времени.
- ◎ Например, одну роль `Customer` (покупатель) может исполнять несколько реальных человек (сущностей). Некоторые из них могут исполнять роль `System Administrator`.



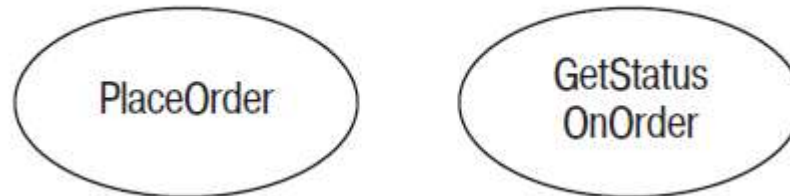
ИДЕНТИФИКАЦИЯ АКТЕРОВ

- ◎ Чтобы выявить актеров, спросите: «Кто или что использует или взаимодействует с системой?»:
 - ◎ Кто устанавливает систему?
 - ◎ Кто или что запускает и выключает систему?
 - ◎ Кто обслуживает систему?
 - ◎ Какие системы взаимодействуют с данной системой?
 - ◎ Кто или что получает и предоставляет информацию системе?
 - ◎ Происходит ли что-нибудь в точно установленное время?



ПРЕЦЕДЕНТЫ

- ◎ **Прецедент** – это описание последовательности действий, включая альтернативные и ошибочные последовательности, которые система, подсистема или класс могут осуществлять, взаимодействуя с внешними актерами.
- ◎ Прецедент – это что-то, что должна делать система по желанию актера. Это «**вариант использования**» системы конкретным актером.



ИДЕНТИФИКАЦИЯ ПРЕЦЕДЕНТОВ

- ⊙ Чтобы найти прецедент, надо спросить: «Как каждый из актеров использует систему?» и «Что система делает для каждого актера?»
- ⊙ Во время идентификации прецедентов могут обнаружиться некоторые новые актеры.

ДЕТАЛИЗАЦИЯ ПРЕЦЕДЕНТОВ

ДЕТАЛИЗАЦИЯ ПРЕЦЕДЕНТА

- ◎ Детализация прецедента – это точное определение каждого прецедента.
- ◎ Детализацию всех прецедентов можно производить параллельно, постепенно повышая уровень детализации.
- ◎ Итог: прецедент с спецификацией.

ПРИМЕР СПЕЦИФИКАЦИИ ПРЕЦЕДЕНТА

имя прецедента	{ { { { { { { { { {	Прецедент: PaySalesTax
идентификатор прецедента		ID: 1
краткое описание		Краткое описание: Выплата налога с оборота в Налоговое управление по окончанию налогового периода.
актеры, вовлеченные в прецедент		Главные актеры: Time (Время)
		Второстепенные актеры: TaxAuthority (налоговое управление)
состояние системы до начала прецедента		Предусловия: 1. Конец налогового периода.
фактические этапы прецедента		Основной поток: / неявный актер Time 1. Прецедент начинается в конце налогового периода. 2. Система определяет сумму Налога с оборота, которую необходимо выплатить Налоговому управлению. 3. Система посылает электронный платеж в Налоговое управление.
		Постусловия: 1. Налоговое управление получает соответствующую сумму Налога с оборота.
		Альтернативные потоки: Нет.
состояние системы после окончания прецедента		
альтернативные потоки		

ОСНОВНОЙ ПОТОК

- ◎ Основной поток описывает «идеальный» ход развития событий в прецеденте.
- ◎ Альтернативные потоки могут перехватывать ошибки, ответвления и прерывания основного потока.
- ◎ Основной поток *всегда начинается с действий главного актера*, направленных на инициацию прецедента.

ЗАПИСЬ ОСНОВНОГО ПОТОКА

- ◎ Удачным способом начала потока можно считать следующую форму записи:

Прецедент начинается, когда <актер> <действие>.

- ◎ Каждый этап потока прецедента должен быть выражен в следующей форме:

<номер> <ктолибо> <совершает некоторое действие>.

- ◎ Использование пассивного залога для описания этапа является неверным: «Вводятся данные покупателя»

КЛЮЧЕВЫЕ СЛОВА В ОПИСАНИИ ОСНОВНОГО ПОТОКА

- ◎ Альтернативные потоки могут быть заменены с помощью ключевого слова «Если»

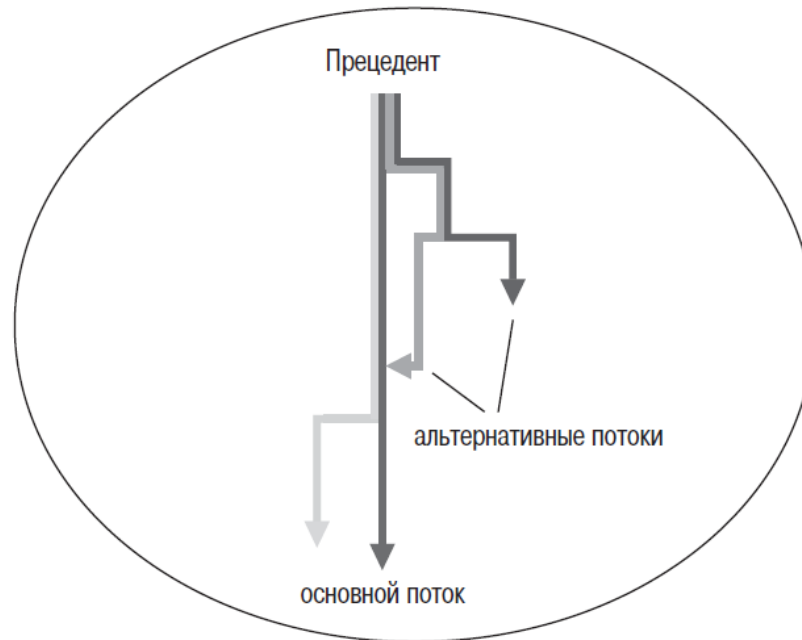
Основной поток:

1. Прецедент начинается, когда Покупатель выбирает товарную позицию в корзине.
2. Если Покупатель выбирает «удалить позицию».
 - 2.1. Система удаляет позицию из корзины.
3. Если Покупатель вводит новое количество.
 - 3.1. Система обновляет количество товаров в корзине.

- ◎ Аналогично можно использовать ключевые слова «Пока ...» или «Для всех элементов...» для описания циклов или повторений

АЛЬТЕРНАТИВНЫЕ ПОТОКИ

- © Могут не возвращаться в основной поток после выполнения, т.к. обрабатывают ошибки, исключительные ситуации и т.п.



ДОКУМЕНТАЦИЯ АЛЬТЕРНАТИВНЫХ ПОТОКОВ

Альтернативные потоки:

InvalidEmailAddress

InvalidPassword

Cancel

Обычно альтернативные потоки документируются отдельно.

Для них создается аналогичная спецификация, только используется ID 2-го уровня

Альтернативный поток: CreateNewCustomerAccount:InvalidEmailAddress
ID: 5.1
Краткое описание: Система сообщает Покупателю, что он ввел недействительный адрес электронной почты.
Главные актеры: Покупатель
Второстепенные актеры: Нет.
Предусловия: 1. Покупатель ввел недействительный адрес электронной почты.
Альтернативные потоки: 1. Альтернативный поток начинается после шага 2.2 основного потока. 2. Система сообщает Покупателю, что он ввел недействительный адрес электронной почты.
Постусловия: Нет.

ОТОБРАЖЕНИЕ ТРЕБОВАНИЙ

ОТОБРАЖЕНИЕ ТРЕБОВАНИЙ

- При отображении требований устанавливаются взаимосвязи между моделью требований и моделью прецедентов.
- Может проводиться как с помощью специального ПО (RequisitePro, DOORS) или вручную.

		Прецедент			
		П ₁	П ₂	П ₃	П ₄
Требование	T1	X			
	T2		X	X	
	T3			X	
	T4				X
	T5	X			

ПРИМЕНЕНИЕ МОДЕЛИ ПРЕЦЕДЕНТОВ

- ◎ Применяются, когда:
 - ◎ преобладают функциональные требования;
 - ◎ много типов пользователей (много актеров);
 - ◎ в системе много интерфейсов (много актеров).

- ◎ Не стоит применять, если:
 - ◎ преобладают нефункциональные требования;
 - ◎ в системе мало пользователей;
 - ◎ в системе мало интерфейсов.

- ◎ Например, сложно описать модель прецедентов для встроенных систем, систем со сложными алгоритмами.