

# Техники тест-дизайна

## Техники обеспечения качества ПО:

- **Статические** – призваны предотвратить появление ошибок. Не требуют непосредственного использования ПО
- **Динамические** – призваны найти ошибки в ПО путем его использования

## Статические техники:

- **Рецензирование (review):**
  - неформальное рецензирование (informal review)
  - разбор (walkthrough)
  - технический анализ (technical review)
  - инспекция (inspection)
- **Статический анализ (static analysis):**
  - анализ [потока] данных (data [flow] analysis)
  - анализ потока управления (control flow analysis)
  - стандарты кодирования/метрики кода

## Инструменты статического анализа:

- компиляторы
- среды разработки (IDE)
- валидаторы кода (W3C HTML/CSS validator, FindBugs)

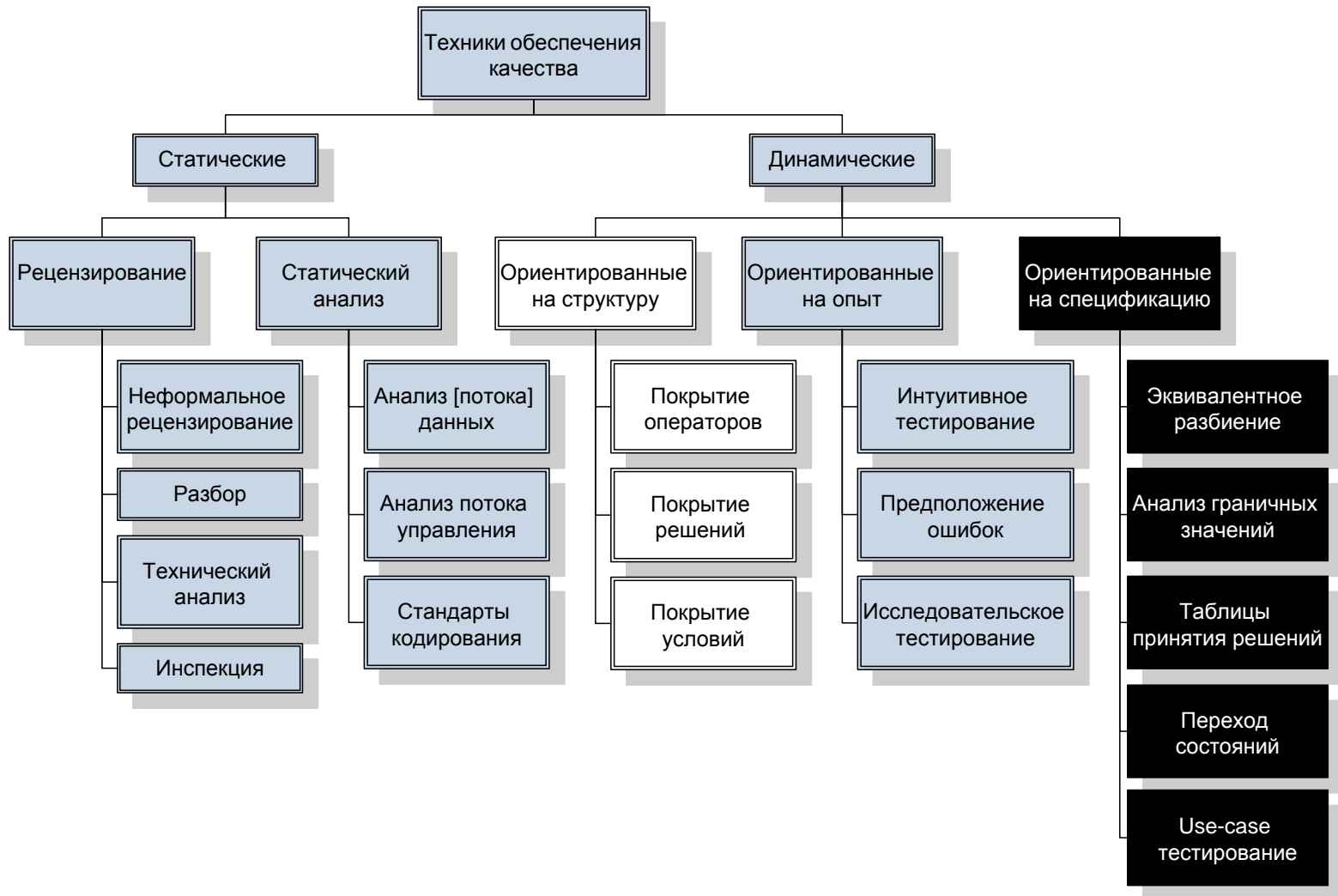
## Динамические техники:

- **Ориентированные на структуру (structure-based):**
  - покрытие операторов (statement coverage)
  - покрытие решений (decision coverage)
  - покрытие условий (condition coverage)
- **Ориентированные на опыт (experience-based):**
  - интуитивное тестирование (ad hoc testing)
  - предположение ошибок (error guessing)
  - исследовательское тестирование (exploratory testing)

## Динамические техники:

- **Ориентированные на спецификацию (specification-based):**
  - эквивалентное разбиение (equivalence partitioning)
  - анализ граничных значений (boundary value analysis)
  - таблицы принятия решений (decision tables)
  - переход состояний (state transition)
  - тестирование на основе сценариев использования (use-case testing)

# Классификация



# White-box техники



$$SC = \frac{S_{Exercised}}{S_{Total}} \cdot 100\%$$

- Black-box тестирование:  $SC = 60\%-70\%$
- $SC = 100\%$  - каждый оператор (statement) выполнен хотя-бы один раз

# Поккрытие операторов

Пример 1:

```
Read P
Read Q
If P+Q>100 Then
    Print "P>Q"
End If
If P>50 Then
    Print "P>50"
End If
```

TC1:

- $P = 60$
- $Q = 60$

# Поккрытие операторов

## Пример 2:

```
Read X
Read Y
If X=3 Then
    Print "Message X"
    If Y=2 then
        Print "Message Y"
        Else Print "Message Z"
    End If
Else Print "Message Z"
End If
```

## TC1:

- $X = 3$
- $Y = 2$

## TC2:

- $X = 3$
- $Y \neq 2$

## TC3:

- $X \neq 3$
- $Y = \forall$

$$DC = \frac{D_{Exercised}}{D_{Total}} \cdot 100\%$$

- Decision statements: IF, Case, Do-While, Repeat-Until, etc.
- Black-box тестирование: DC = 40%-60%
- DC = 100% - все возможные альтернативы были выбраны хотя-бы один раз
- DC = 100% → SC = 100%

# Поккрытие решений

Пример 3:

```
Read P
Read Q
If P+Q>100 Then
    Print "P+Q>100"
End If
If P>50 Then
    Print "P>50"
End If
```

**TC1:**

- $P = 60$
- $Q = 60$

**TC2:**

- $P = 10$
- $Q = 15$

# Покрывтие решений

## Пример 4:

```
Read X
Read Y
If X=3 Then
    Print "Message X"
    If Y=2 then
        Print "Message Y"
        Else Print "Message Z"
    End If
Else Print "Message Z"
End If
```

## TC1:

- $X = 3$
- $Y = 2$

## TC2:

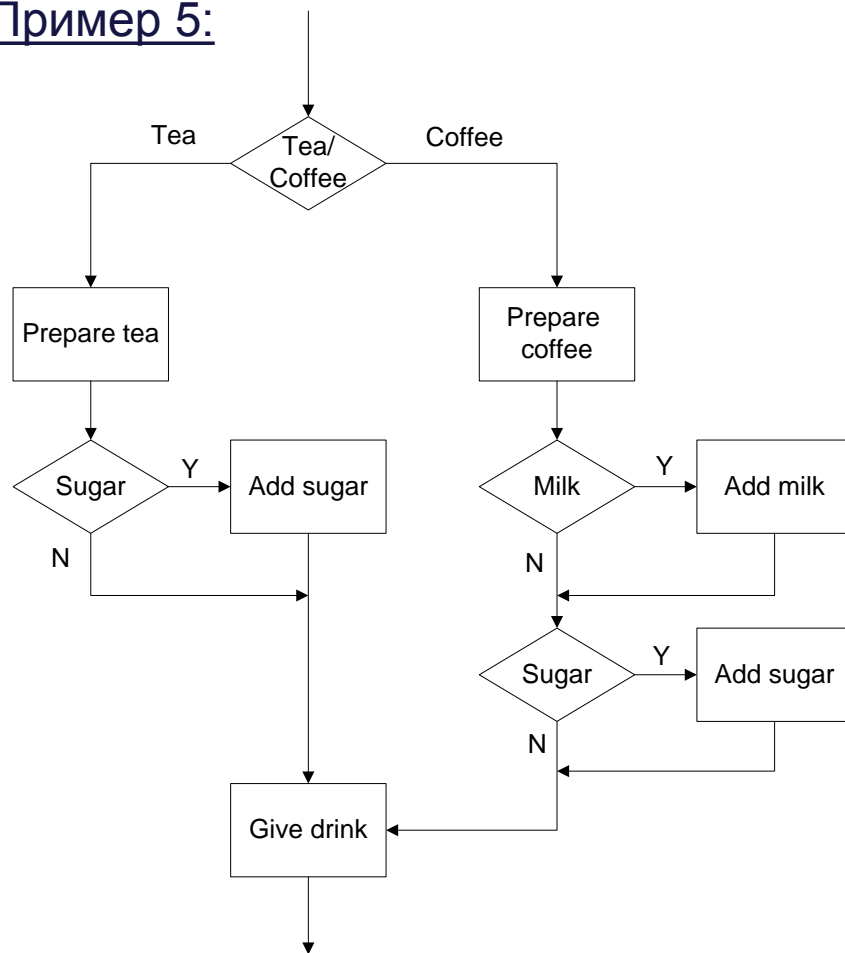
- $X = 3$
- $Y \neq 2$

## TC3:

- $X \neq 3$
- $Y = \forall$

# Покрытие решений

## Пример 5:



## Покрытие операторов:

**TC1:** Tea, Sugar

**TC2:** Coffee, Milk, Sugar

## Покрытие решений:

**TC3:** Tea, Sugar

**TC4:** Coffee, Milk, Sugar

# Покрытие условий

**Condition coverage** – каждое условие из решения принимает все возможные значения хотя бы один раз

**Condition/decision coverage** – CC+DC

**Modified condition/decision coverage** – CC+DC + каждое условие из решения должно повлиять на исход решения независимо от других условий

**Multiple condition coverage** – все комбинации условий в решении должны выполняться хотя бы один раз



# Покрытие условий

Пример 6:

```
If A or B Then  
    Print "Message Z"  
End If
```

**SC:** TF

**DC:** TF, FF

**CC:** TF, FT (TT,FF)

**CDC:** TT, FF

**Modified CDC:** TF, FT, FF

**Multiple CC:** TT, TF, FT, FF

# Покрытие условий

## Пример 7:

```
If (A or B) and C Then  
    Print "Message Z"  
End If
```

**SC:** TTT

**DC:** TTT, TTF

**CC:** TFT, FTF (TTT, FFF)

**CDC:** TTT, FFF

**Modified CDC:**

- **A:** FFT / TFT
- **B:** FFT / FTT
- **C:** TTF / TFT(FTT)
- **All:** FFT, TFT, FTT, TTF

**Multiple CC:** FFF, FFT, FTF, FTT,  
TFF, TFT, TTF, TTT

# Black-box техники

# Эквивалентное разбиение

**Класс эквивалентности** – это набор значений переменной, который считается эквивалентным.

Тестовые сценарии эквивалентны, если они тестируют одно и то же (если один из них [не]выявляет ошибку, то и остальные [не]выявят ее)

# Эквивалентное разбиение

Пример 7:

В зависимости от размера заказа, клиенту предоставляется скидка:

До 100 ед. : 0%

От 100 ед. до 1000 ед. : - 7%

1000 ед. и больше : - 10%

# Эквивалентное разбиение

Пример 7:

Классы эквивалентности	
Валидные классы (valid)	Невалидные классы (invalid)
1) $1 \leq N < 100$ 2) $100 \leq N < 999$ 3) $1000 \leq N$	1) $N = 0$ 2) $N \leq -1$ 3) Null 4) "Строка" 5) 123.4567 6) $\text{Max} < N$

# Эквивалентное разбиение

## Область применения классов эквивалентности:

- входящие/исходящие данные
- числа (-, +, 0, 199<sup>999</sup>, 0.123456)
- символы ("symbol", "СИМВОЛ", "SiMboL", "!@#%^")
- количество (записей в БД, строк, элементов multi-select)
- типы файлов (.bmp, .jpg, .png)
- длина строки (Null, 200,  $\approx \infty$ )
- размер файла (Null, 4+Gb)
- объем памяти (меньше/больше требований)
- разрешение экрана (640x480, 1920x1080, 2560x1440)
- версии ОС, библиотек (Mac OS/Win/Linux, DX9/10/11)
- браузеры (Firefox, Safari, Chrome, IE6/7/8, Opera)
- объем передаваемых данных

# Эквивалентное разбиение

## Правило “хорошего тона”

При наличии нескольких полей (переменных):

- валидные классы нескольких переменных объединяются в один тест-кейс
- невалидные классы тестируются отдельно

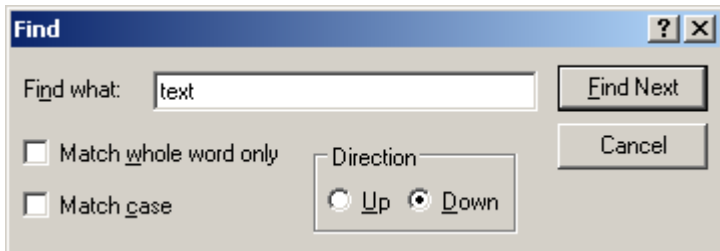


# Метод всех пар

Метод всех пар гарантирует, что все комбинации из **пар значений** любых **переменных** будут протестированы.

# Метод всех пар

## Пример 8:



### Переменные:

- 1) Find what (FW)
- 2) Match whole word (MW)
- 3) Match case (MC)
- 4) Direction (D)

### Значения:

FW = {"lower", "UPPER", "MiXeD"}

MW = {Yes, No}

MC = {Yes, No}

D = {Up, Down}

$3 \times 2 \times 2 \times 2 = 24$  теста

### Полный перебор (brute force)

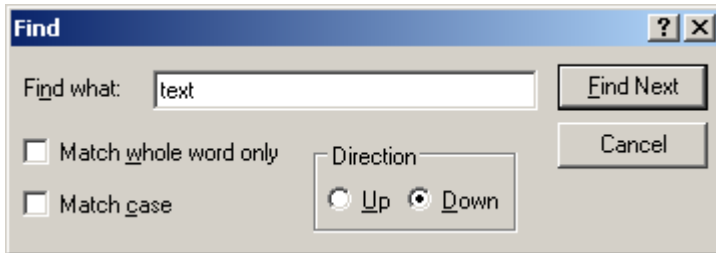
TC	FW	MW	MC	D
1	L	Y	Y	Up
2	U	Y	Y	Up
3	M	Y	Y	Up
4	L	N	Y	Up
5	U	N	Y	Up
6	M	N	Y	Up
...	...	...	...	...
24	M	N	N	Down

### Все по одному разу

TC	FW	MW	MC	D
1	L	Y	N	Up
2	U	N	Y	Down
3	M	Y	Y	Up

# Метод всех пар

## Пример 8:



### Переменные:

- 1) Find what (FW)
- 2) Match whole word (MW)
- 3) Match case (MC)
- 4) Direction (D)

### Значения:

FW = {"lower", "UPPER", "MiXeD"}

MW = {Yes, No}

MC = {Yes, No}

D = {Up, Down}

$3 \times 2 \times 2 \times 2 = 24$  теста

Метод всех пар (pairwise testing)				
TC	FW	MW	MC	D
1	L	Y	N	Down
2	L	N	Y	Up
3	M	Y	Y	Up
4	M	N	N	Down
5	U	Y	N	Down
6	U	N	Y	Up

# Метод всех пар

## Реализация

### 1. Определить переменные

Браузеры (B)

Разрешения (R)

Темы (T)

Локализация (L)

### 2. Определить значения

$B = \{IE, FF, Safari, Chrome\}$

$R = \{800 \times 600, 1280 \times 1024, 1920 \times 1080\}$

$T = \{Theme1, Theme2, Theme3\}$

$L = \{EN, RU, UA\}$

### 3. Построить таблицу

- количество столбцов = количеству переменных
- количество строк = произведению количества значений двух наиболее «весомых» переменных
- кест-кейс = строка таблицы

# Метод всех пар

## Реализация

B	R
IE	800x600
IE	1280x1024
IE	1920x1080
FF	800x600
FF	1280x1024
FF	1920x1080
Safari	800x600
Safari	1280x1024
Safari	1920x1080
Chrome	800x600
Chrome	1280x1024
Chrome	1920x1080

B	R	T
IE	800x600	Theme1
IE	1280x1024	Theme2
IE	1920x1080	Theme3
FF	800x600	Theme2
FF	1280x1024	Theme3
FF	1920x1080	Theme1
Safari	800x600	Theme3
Safari	1280x1024	Theme1
Safari	1920x1080	Theme2
Chrome	800x600	Theme1
Chrome	1280x1024	Theme2
Chrome	1920x1080	Theme3

B	R	T	L
IE	800x600	Theme1	EN
IE	1280x1024	Theme2	RU
IE	1920x1080	Theme3	UA
FF	800x600	Theme2	UA
FF	1280x1024	Theme3	EN
FF	1920x1080	Theme1	RU
Safari	800x600	Theme3	RU
Safari	1280x1024	Theme1	UA
Safari	1920x1080	Theme2	EN
Chrome	800x600	Theme1	EN
Chrome	1280x1024	Theme2	RU
Chrome	1920x1080	Theme3	UA

# Анализ граничных значений

Границы классов эквивалентности бывают:

- закрытые ( $100 \leq N < 999$ )
- открытые ( $1000 \leq N$ )

Анализ граничных значений имеет два подхода:

- **первый** – когда граница соседних эквивалентных классов проходит между ними и не принадлежит ни одному из классов, в данном случае мы имеем два граничных значения
- **второй** – когда граница принадлежит одному из классов, в этом случае необходимо проверить три значения – саму границу, значение «справа» от границы и значение «слева» от границы

# Анализ граничных значений

## Пример 9:

В зависимости от суммы на текущем счету в банке, ежемесячно насчитывается бонус:

От 100 до \$1000 (включительно) : +3%

От \$1000 до \$10 000 : +5%

\$10 000 и больше : +7%

# Анализ граничных значений

## Пример 10:



Границы классов эквивалентности		
	Валидные границы	Невалидные границы
0%	\$99.99, \$0	-\$0.01
+3%	\$100.00, \$100.01, \$999.99, \$1000.00	— — —
+5%	\$1000.01, \$9 999.99	— — —
+7%	\$10 000, \$10 000.01, \$Max	\$Max + \$0.01



## Правило “хорошего тона”

При наличии нескольких полей (переменных):

- минимальные значения валидных границ объединяются в один тест-кейс
- максимальные значения валидных границ объединяются в другой тест-кейс
- невалидные границы тестируются отдельно, как и в случае с невалидными классами

# Таблицы принятия решений

Таблицы принятия решений (decision tables/ cause-effect tables) применяются:

- в случае зависимости исходящих данных или поведения программы от комбинаций значений входящих данных
- при проверке “бизнес-правил”

# Таблицы принятия решений

## Структура таблицы:

		Правила (rules)							
		1	2	3	4	5	6	7	8
Условия (conditions)	Условие1	T	T	T	T	F	F	F	F
	Условие2	T	T	F	F	T	T	F	F
	Условие3	T	F	T	F	T	F	T	F
Действия (actions)	Действие1	X	X			X			
	Действие2	X					X		
	Действие3		X			X	X		
	Действие4			X	X			X	X

Условия – входящие данные

Действия – исходящие данные  
(expected results)

		Правила (rules)							
Условия	Значения	1	2	3	4	5	6	7	8
Условие1	T,F	T	T	T	T	F	F	F	F
Условие2	T,F	T	T	F	F	T	T	F	F
Условие3	T,F	T	F	T	F	T	F	T	F
Действия									
Действие1		X	X			X			
Действие2		X					X		
Действие3			X			X	X		
Действие4				X	X			X	X

Правила – тест-кейсы

# Таблицы принятия решений

## Шаги построения таблицы:

- 1) Определить/записать все условия
- 2) Посчитать количество возможных комбинаций условий
- 3) Заполнить комбинации
- 4) Убрать лишние комбинации
- 5) Записать действия

# Таблицы принятия решений

Пример 11:

**Ввод:**  $a, b, c$

**Вывод:** определить тип треугольника (равносторонний, равнобедренный, разносторонний, не треугольник)

**Шаг 1 (определить условия):**

**Условие 1:**  $a, b, c$  образуют треугольник?  $\{Y, N\}$

**Условие 2:**  $a=b$ ?  $\{Y, N\}$

**Условие 3:**  $a=c$ ?  $\{Y, N\}$

**Условие 4:**  $b=c$ ?  $\{Y, N\}$

# Таблицы принятия решений

Пример 11:

## Шаг 2 (посчитать комбинации):

Если все условия простые ( $\{Y, N\}$ ):

Кол-во комбинаций =  $2^{\text{кол-во условий}}$

Если условия сложные ( $\{A, B, C, D\}$ ):

Кол-во комбинаций =  $V_{C1} * V_{C2} * V_{C3} * V_{Cn}$

$$N = 2^4 = 16$$

# Таблицы принятия решений

Пример 11:

## Шаг 3-4 (заполнить комбинации):

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
C1: Треугольник?	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N
C2: a=b?	Y	Y	Y	Y	N	N	N	N	Y	Y	Y	Y	N	N	N	N
C3: a=c?	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N
C4: b=c?	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N

	1	2	3	4	5	6	7	8	9
C1: Треугольник?	Y								N
C2: a=b?	Y				N				-
C3: a=c?	Y		N		Y		N		-
C4: b=c?	Y	N	Y	N	Y	N	Y	N	-

# Таблицы принятия решений

Пример 11:

## Шаг 5 (заполнить действия):

	1	2	3	4	5	6	7	8	9
C1: Треугольник?	Y								N
C2: $a=b$ ?	Y				N				-
C3: $a=c$ ?	Y		N		Y		N		-
C4: $b=c$ ?	Y	N	Y	N	Y	N	Y	N	-
A1: Не треугольник									X
A2: Равносторонний	X								
A3: Равнобедренный				X		X	X		
A4: Разносторонний								X	
A5: Невозможно		X	X		X				



# Таблицы принятия решений

## Пример 12:

Интернет-магазину нужно разослать почту с информацией о скидках своим клиентам.

Содержание писем зависит от следующих условий:

- 1) Клиенты типа А, В получают стандартное письмо
- 2) Клиенты типа С получают специальное письмо
- 3) Клиентам, совершившим 5 и более покупок или купившим на сумму больше \$500 в письме сообщается о дополнительной скидке в 20% на следующую покупку

# Таблицы принятия решений

## Пример 12:

Условия	Значения	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Тип клиента	A,B,C,D	A	A	A	A	B	B	B	B	C	C	C	C	D	D	D	D
5 и более покупок	Y,N	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N
Сумма больше \$500	Y,N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N
Действия																	
Стандартное письмо		X	X	X	X	X	X	X	X					?	?	?	?
Специальное письмо										X	X	X	X	?	?	?	?
Сообщение о скидке		?	X	X		?	X	X		?	X	X		?	?	?	?
Не получают письмо														?	?	?	?

**D** – все остальные типы клиентов (если существуют)

**?** – если более 5 покупок **И** на сумму больше \$500

# Переход состояний

**Конечный автомат** (finite state machine) – вычислительная модель, состоящая из ограниченного числа состояний и переходов между этими состояниями, возможно сопутствующими действиями.  
[IEEE 610]

Любая система, дающая различный вывод на один и тот же ввод в зависимости от того, что случилось ранее – конечный автомат.

# Переход состояний

## Способы представления:

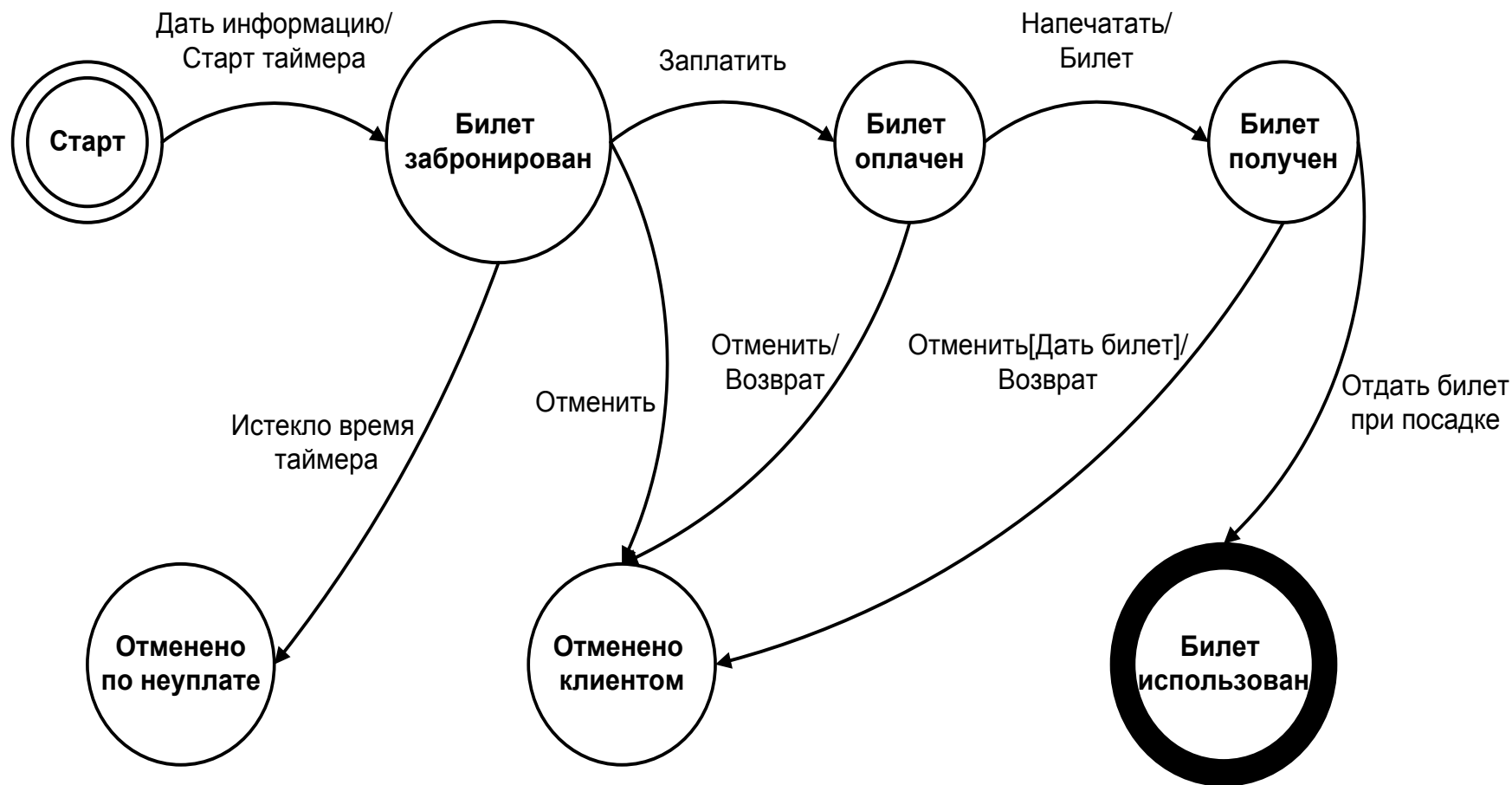
- диаграмма состояний (state-transition diagram)
- таблица переходов (state-transition table)

## Компоненты:

- состояния, в которых система может находиться
- переходы из одного состояния в другое (не все переходы разрешенные)
- события, вызывающие переход
- действия, которые являются результатом перехода

# Переход состояний

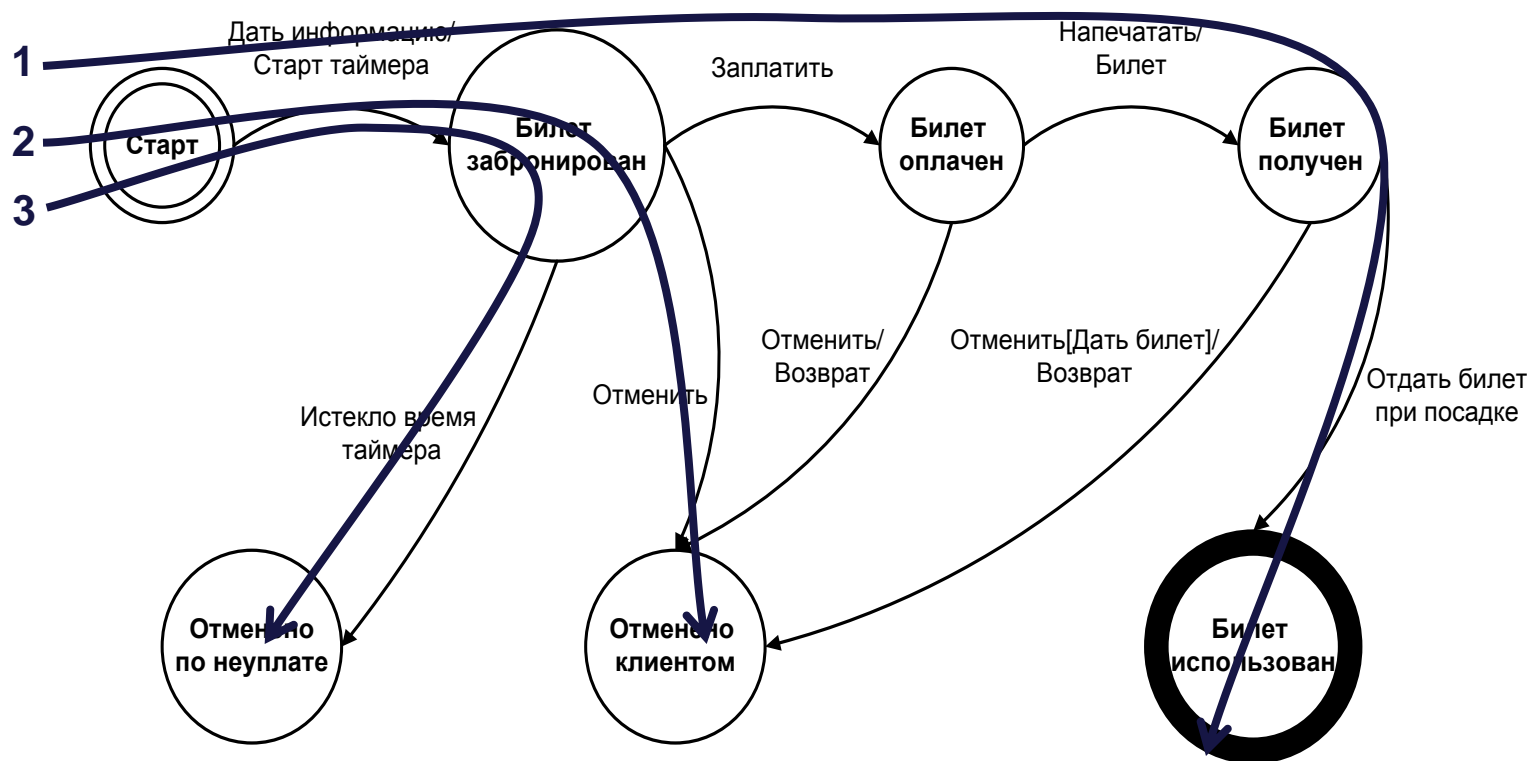
## Пример 13:



# Переход состояний

## Пример 13:

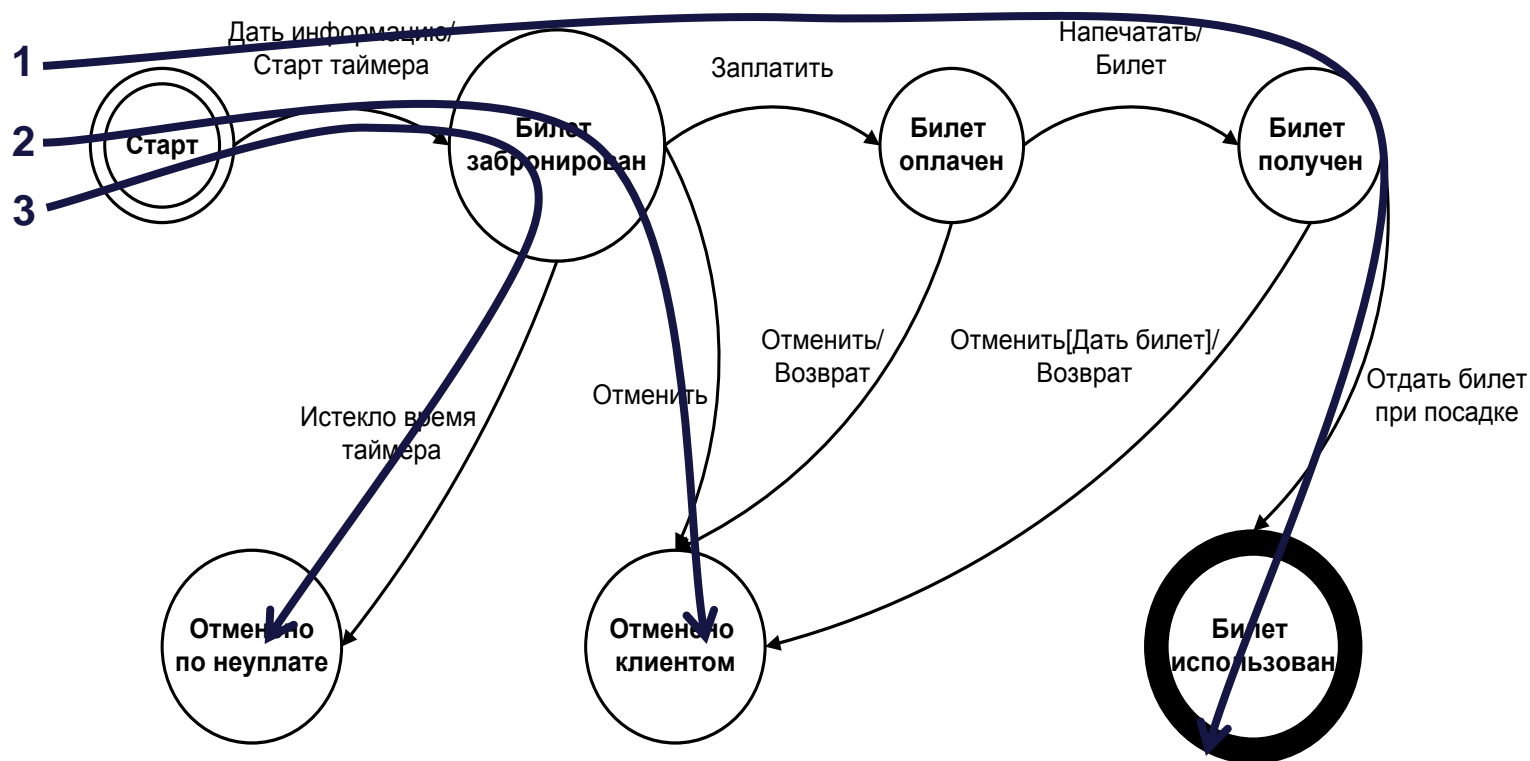
Набор тест-кейсов проходит по всем состояниям



# Переход состояний

## Пример 13:

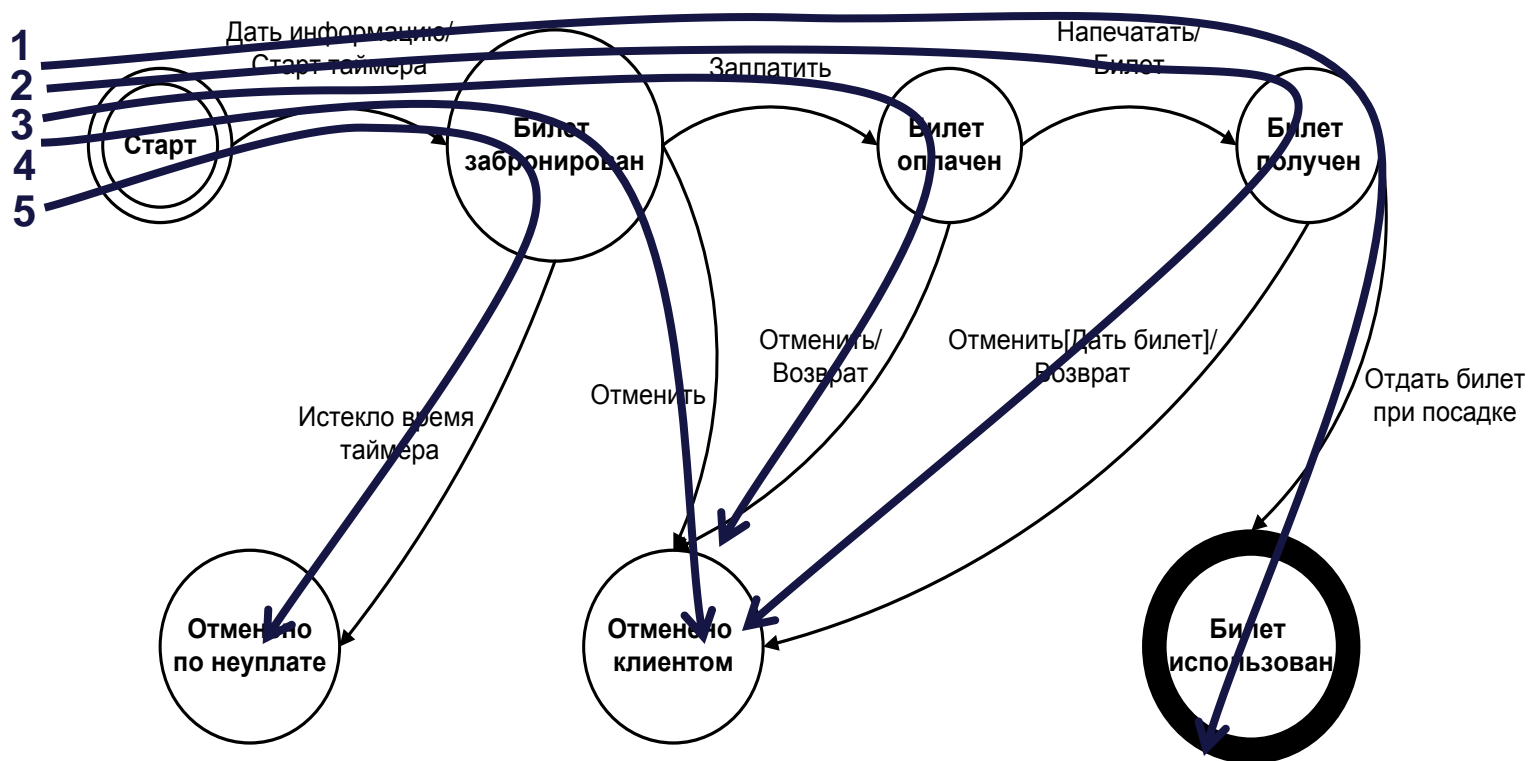
Набор тест-кейсов запускает все события



# Переход состояний

## Пример 13:

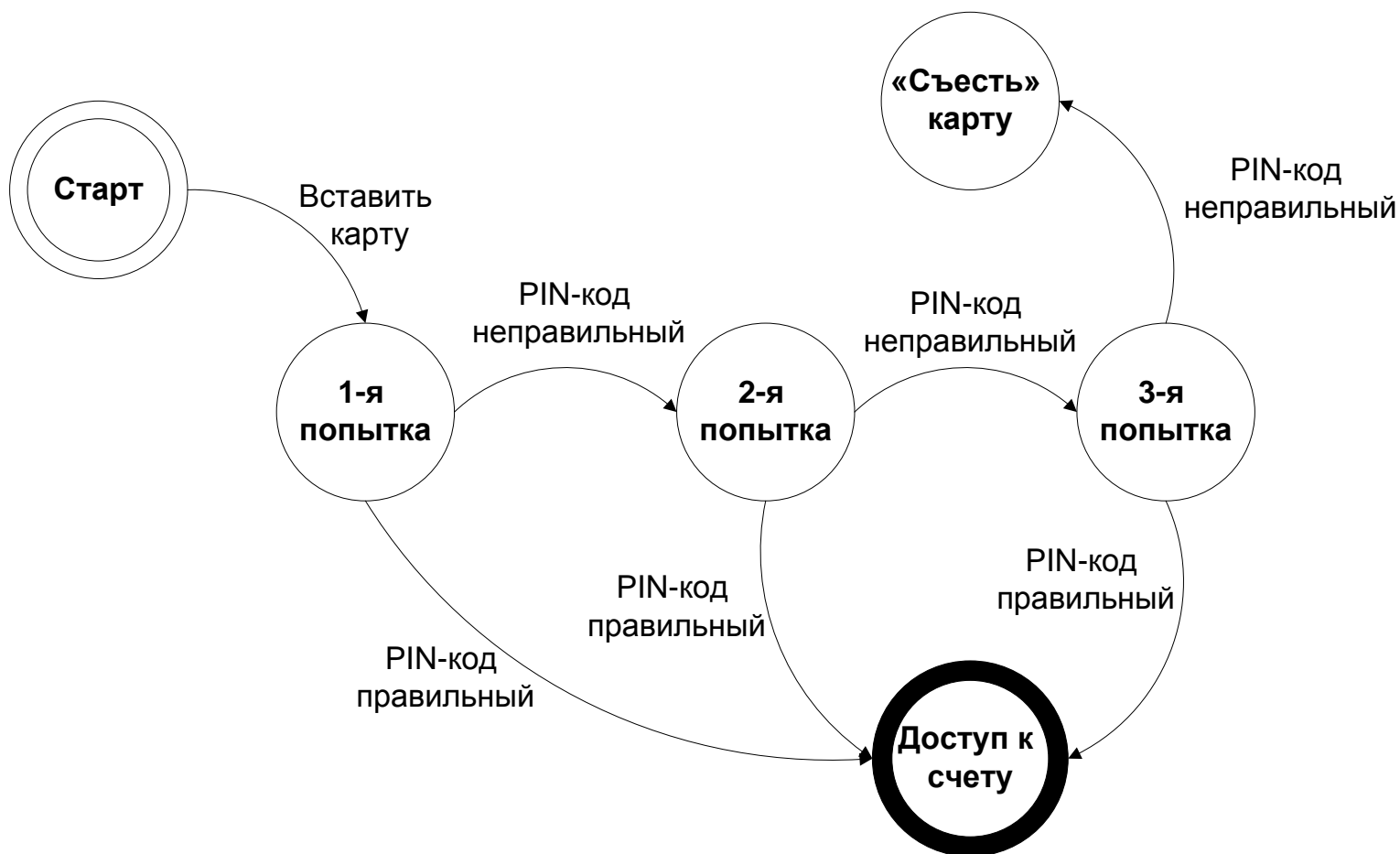
Набор тест-кейсов покрывает все переходы





# Переход состояний

## Пример 14:



# Переход состояний

Пример 14:

## Негативные тест-кейсы:

Состояния	Действия		
	Вставить карту	Правильный PIN	Неправильный PIN
S1) Старт	S2	—	—
S2) 1-я попытка	—	S5	S3
S3) 2-я попытка	—	S5	S4
S4) 3-я попытка	—	S5	S6
S5) Доступ к счету	—	?	?
S6) «Съесть» карту	S1 (новая карта)	—	—

# Use-case тестирование

**Сценарий использования** – сценарий, который описывает использование системы пользователем для достижения определенной цели.

В качестве пользователей (actors) могут быть:

- люди
- другие системы

# Use-case тестирование

## Преимущества:

- охватывают функциональность системы с точки зрения пользователя, а не с технической точки зрения
- не зависят от парадигмы программирования
- позволяют активно вовлекать пользователей в процесс сбора требований
- легко позволяют определить базовые компоненты системы, и связи между ними
- служат основой для разработки тест-кейсов для системного и приемочного тестирования

# Use-case тестирование

Атрибут	Описание	
Номер/идентификатор	Уникальный идентификатор	
Название	Название должно вкратце описывать цель	
Цель	Более детальное описание цели (если нужно)	
Масштаб	Система / подсистема / модуль /	
Уровень	Общий / основная задача / подзадача (подфункция)	
Основные исполнители	Роли исполнителей	
Предусловия	Состояние, в котором должна находиться система перед прохождением	
Критерий прохождения	Состояние системы / события после прохождения	
Критерий непрохождения	Состояние системы / события в случае непрохождения	
Триггер	Событие, после которого нужно выполнять сценарий	
Основной сценарий	Шаг	Результат
	1	
	2	
	...	
Дополнительные условия	Описание условий, при которых основной сценарий может меняться	
Вариации	Вариации, не влияющие на основной ход, но необходимые для рассмотрения	
Приоритет	Высокий / средний / низкий	
Время выполнения	Время для прохождения сценария	
Частота использования	Как часто сценарий будет использоваться	

# Use-case тестирование

Атрибут	Описание	
Номер/идентификатор	ATM123	
Название	Ввод пин-кода	
Цель	Получить доступ к счету карты	
Основные исполнители	Клиент	
Предусловия	Банкомат в режиме ожидания	
Критерий прохождения	Получение доступа к операциям	
Основной сценарий К: Клиент С: Система	Шаг	Результат
	1. Вставить карту (К)	
	2. Проверить валидность карты (С)	Запрос PIN-кода
	3. Ввести PIN (К)	
	4. Проверка PIN-кода (С)	Дать доступ к операциям
Дополнительные условия	2а) Карта не прошла валидацию	Показать сообщение, вернуть карту
	4а) PIN-код неправильный	Показать сообщение, запросить PIN повторно
	4б) 3 раза неправильный PIN-код	«Съесть карту», перейти в режим ожидания
Вариации	-	
Приоритет	Высокий	
Время выполнения	3 мин	
Частота использования	Всегда	

# Use-case тестирование

## Правило “хорошего тона”

- основной сценарий проходит в одном тест-кейсе
- для дополнительных условий создаются отдельные тест-кейсы

# Литература

- Kelly J. Hayhurst - "A Practical Tutorial on Modified Condition/Decision Coverage"
- Lee Copeland – “A Practitioner's Guide to Software Test Design”
- Dorothy Graham – “Foundations of software testing”
- IEEE SWEBOOK 2004
- IEEE 610
- Google
- Wikipedia



**Спасибо**