

Технология программирования

Курс лекций для гр. 3057, 4057

Лекция №7

Содержание

1. Введение
2. Модели жизненного цикла ПП
3. Модели команды разработчиков
4. Управление проектами
5. Словесная коммуникация
6. Языки, модели и методы проектирования
 - <...>
 - 6.6 Спецификация программ конечными функциями
 - 6.7 Событийные конечно-автоматные модели
 - 6.8 Модели экономических информационных систем
1. Тестирование
2. CASE-системы
3. Надежность ПП, ее оценка и меры по ее повышению
4. Стандарты качества технологии программирования

На прошлой лекции

Структуризация различных видов моделей ПП:

- алгоритмической модели → структурное программирование
- функциональной модели → многослойная структура и стандартизация интерфейсов
- функциональной модели совместно с моделью данных → абстрактный тип данных и его развитие в современных компонентах

Непроцедурные модели описания поведения программ

- Процедурная модель = алгоритмическая, она описывает последовательность шагов преобразования данных
- Непроцедурные модели описывают структуру и свойства программы как процесса: состояния, реакции на воздействия, временные свойства

Реакцию на входные воздействия удобно специфицировать *конечными функциями*

Конечная функция – это функция, область задания которой -
конечное множество значений

(а область существования – не обязательно конечное множество)

Простейшая форма задания КФ – таблица с числом входов, равным мощности области задания

Так обычно представляются эмпирические зависимости, например: $K = F(V)$:

Скорость V , км/ч	5	10	25	50
Коэффициент трения K	0,0212	0,0231	0,0257	0,0276

Спецификация поведения программ логическими функциями

Частный случай КФ – логическая функция; ей соответствует таблица истинности

Например, логическая функция двух переменных $F(X, Y)$:

X: Концевой выключатель сработал	F	F	T	T
Y: Реле времени сработало	F	T	F	T
$F(X, Y)$: Сигнал «Включить лампочку»	F	F	T	F

Обобщение – многозначная логика: мощность области значений больше двух

Соответствующая таблица называется **таблицей решений** (ТР)

Аргументы называются условиями, значения функции – решениями

Пример: спецификация программы контроля параметров насоса с электроприводом

Аргументы - условия	Температура $T > 60$ C	N	Y	~	N	Y	~
	Скорость вращения < 50 об/с	N	N	Y	N	N	~
	Давление $P < 1,6$ атм	N	Y	~	Y	N	~
	Напряжение < 90 в	N	N	N	N	N	Y
Функция	Решение	O	A	A	O	W	A

Y – yes
 N – No
 ~ – Y или N
 O – норма
 W – предупреждение
 A - авария

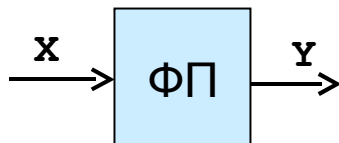
Конечный функциональный преобразователь

Достоинства табличной формы задания поведения программ:

- ✓ наглядность, понятность непрофессионалам
- ✓ таблица одновременно служит планом тестирования

- Таблицы решений удобны для проектирования программ промышленной автоматики, решающих задачи логического управления; диагностики неисправностей, ...
- Существуют текстовые нотации для ТР (языки ТР) и трансляторы с них на инструментальные языки программирования (с оптимизацией)

Все приведенные модели – варианты табличного задания дискретной кибернетической модели – конечного функционального преобразователя (ФП):



$X = \{x_1, \dots, x_n\}$ – множество входных сигналов

$Y = \{y_1, \dots, y_m\}$ – множество выходных сигналов

Выходной сигнал – функция входного: $f: X \rightarrow Y$

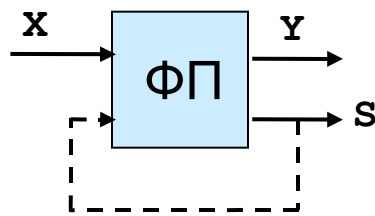
Синхронный ФП срабатывает в регулярные дискретные моменты времени – такты

Синхронный ФП традиционно используется для описания работы *комбинационных* цифровых логических схем, т.е. схем без памяти

Понятие конечного автомата

Конечный автомат (КА, FSM - Finite State Machine) – это ФП с памятью, описывающий сложное поведение:

- преобразование входа в выход зависит от текущего состояния автомата
 - состояние изменяется тоже как функция входов
- Для описания программ интересна асинхронная модель КА: входные сигналы поступают в произвольные моменты времени



КА – это шестерка:

$X = \{x_1, \dots, x_n\}$ – конечное непустое множество входных сигналов

$Y = \{y_1, \dots, y_m\}$ – конечное множество выходных сигналов

$S = \{s_1, \dots, s_k\}$ – конечное множество состояний

$s^0 \in S$ – начальное состояние

$F_y: X \times S \rightarrow Y$ – функция выходов

$F_s: X \times S \rightarrow S$ – функция переходов

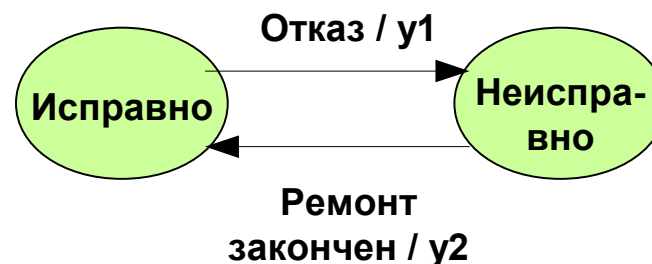
- Состояние – это:
- Стадия в жизненном цикле объекта, во время которой он выполняет определенные действия или ожидает какого-либо события
 - Фундаментальное понятие дискретных динамических систем
- Переход в новое состояние происходит мгновенно

Спецификация моделей КА

Два способа задания КА

- Табличное (как для любых конечных функций) представление функций выходов и переходов
- Диаграммой состояний и переходов (State -Transition Diagram)
 - Вершины графа – состояния, дуги - переходы

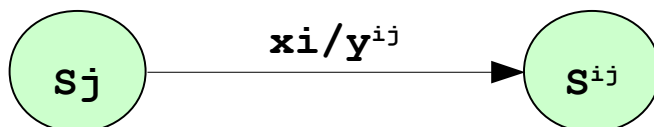
	s1: Исправно	s2: Неисправно
x1: Отказ	s2 / y1	-
x2: Ремонт закончен	-	s1 / y2



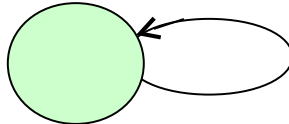
y1 – Включить ламп. «Авария»

y2 – Выключить ламп. «Авария»

Ячейке таблицы в i строке и j столбце соответствует фрагмент графа:



Общие свойства КА моделей

- КА описывает бесконечное множество траекторий поведения (путей в графе диаграммы состояний) – возможных историй процесса
 - КА = дискретный процесс – последовательность смены состояний
 - Граф диаграммы состояний – односвязный: к любой вершине есть путь
 - Степень вершин обычно $\ll n$
 - таблица сильно разрежена
 - Возможен переход из состояния в самого себя, он называется петлей:
- 
- Выходные сигналы y_{ij} могут отсутствовать
 - напр., в автоматах лексического разбора
 - Если с каждым входным сигналом x_i связывать событие, его вызвавшее, то модель КА становится событийной моделью поведения объекта
 - Два вида событий
 - внешние – в аппаратуре или в других программных процессах
 - внутренние – завершение деятельности (программной или аппаратной) в данном состоянии или окончание таймаута (выдержки времени)
 - Привязка событий к абсолютному времени обычно не специфицируется

Виды ПП, для спецификации которых удобны КА модели

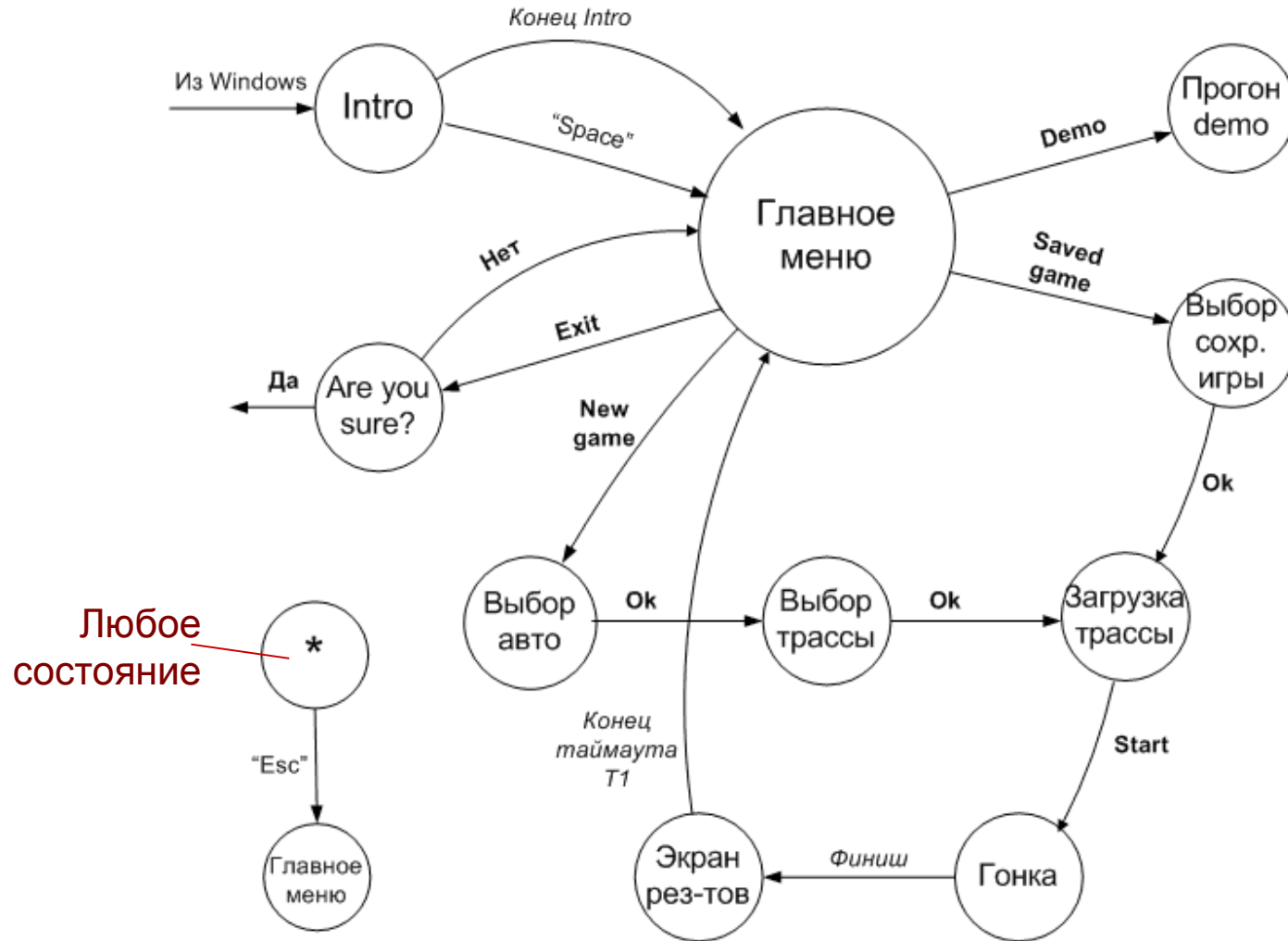
- Промышленная и бытовая автоматика
- Диалоговые программы
- Сетевые протоколы
- Управление в системах связи и передачи данных
 - напр., цифровая телефония
- Встроенные системы управления
- Поведение компьютерных персонажей в играх
- Программы обработки текстов
 - напр., лексический разбор
- Распознавание речи
- и многие другие

–Реагирующие,
или реактивные
(reactive)
системы

Два полярных вида программ

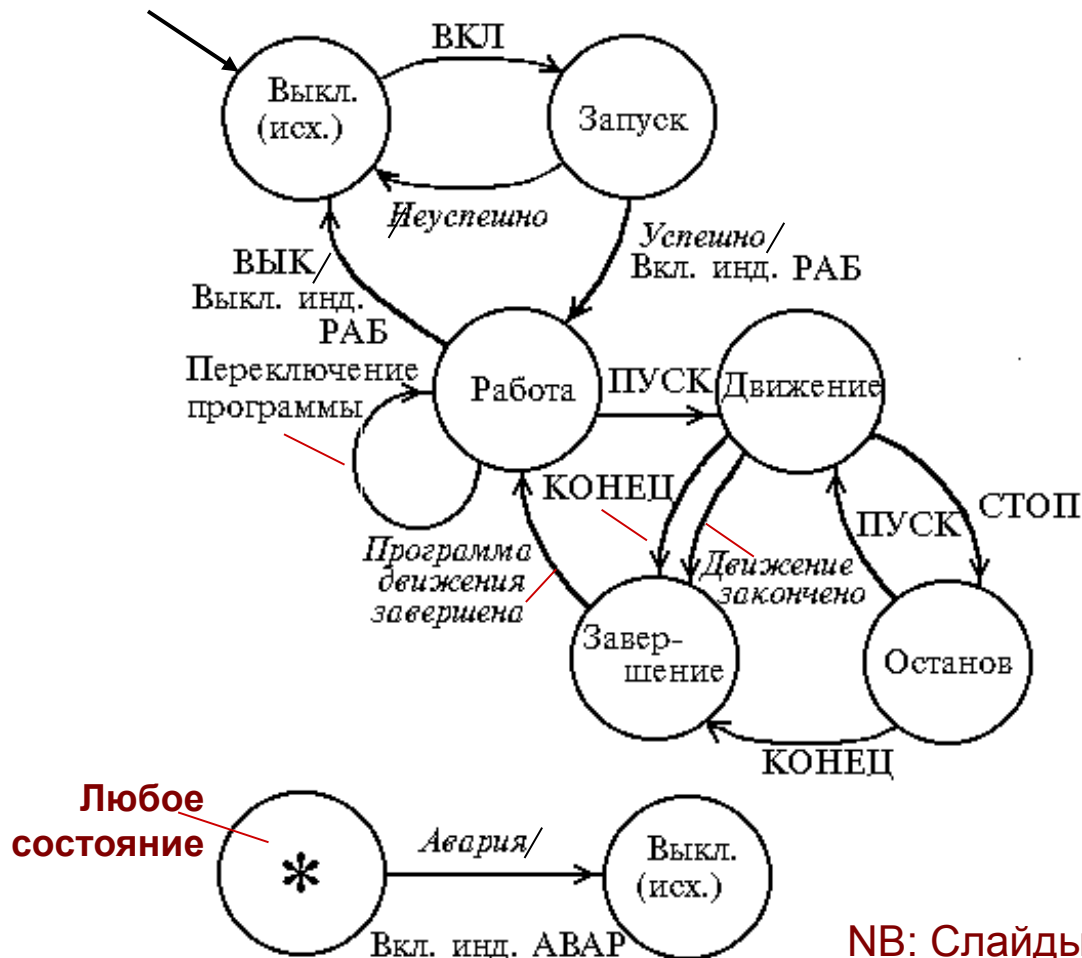
	Трансформационные программы	Реагирующие системы
Примеры	Решение системы уравнений Преобразование текста	Операционная система Система управления роботом
Цель	Вычисление выходных значений как функции входных	Последовательность реакций на внешние события
Процесс	Конечный	Бесконечный

Спецификация диалоговой программы



Спецификация поведения робота-манипулятора

в графической нотации системы DARTS (Design Approach Real Time Systems)



Интерфейс человека-оператора

1. Кнопки пульта:
ВКЛ, ВЫК, ПУСК, СТОП, КОНЕЦ
2. Переключение программ движения робота (напр., «Поднять груз»)
3. Лампочки индикации РАБ (работа) и АВАР (авария)

Курсивом записаны внутренние входные сигналы

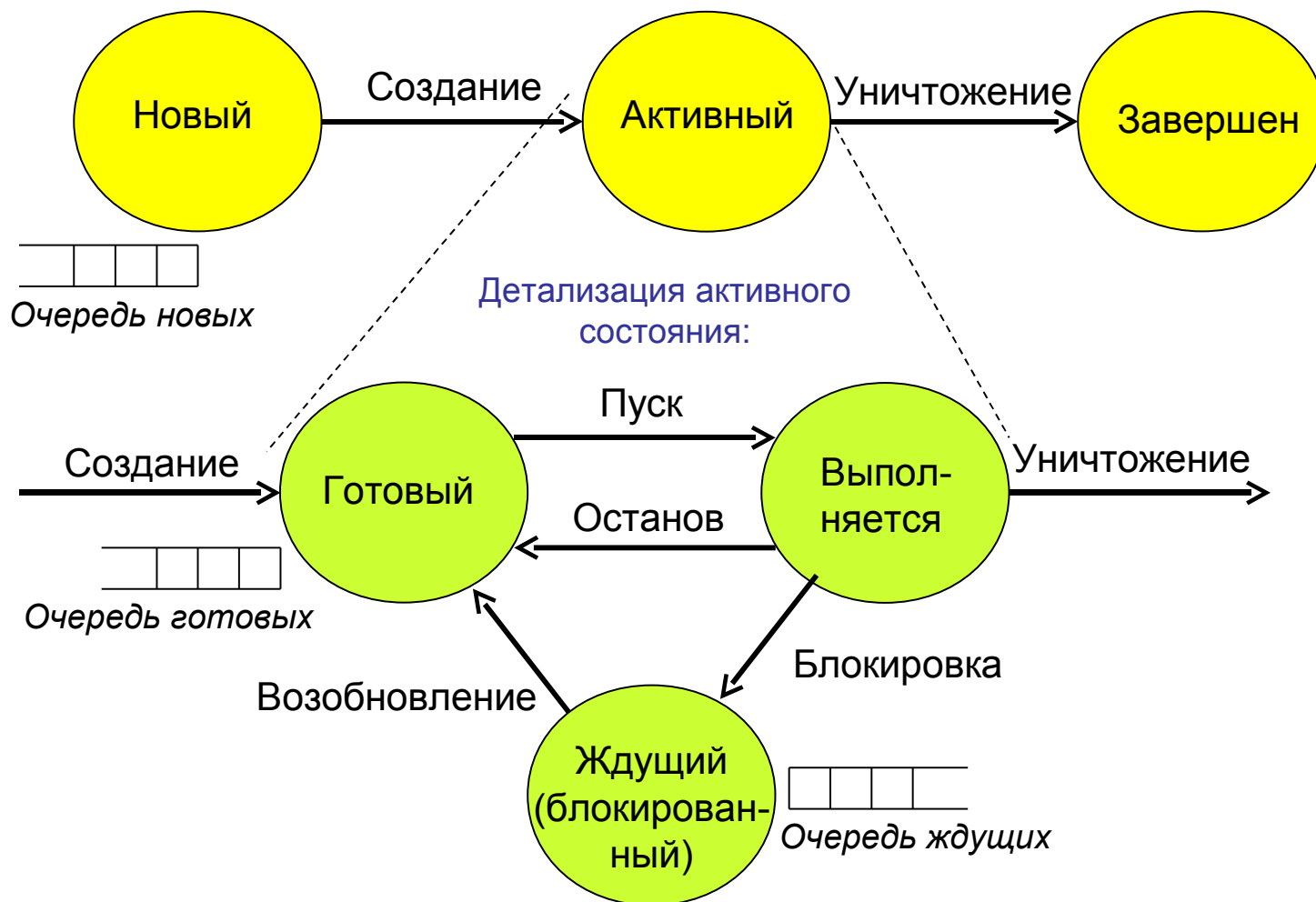
Выходные сигналы – Вкл. Инд. ...
(включение индикации ...)

Имя состояния:

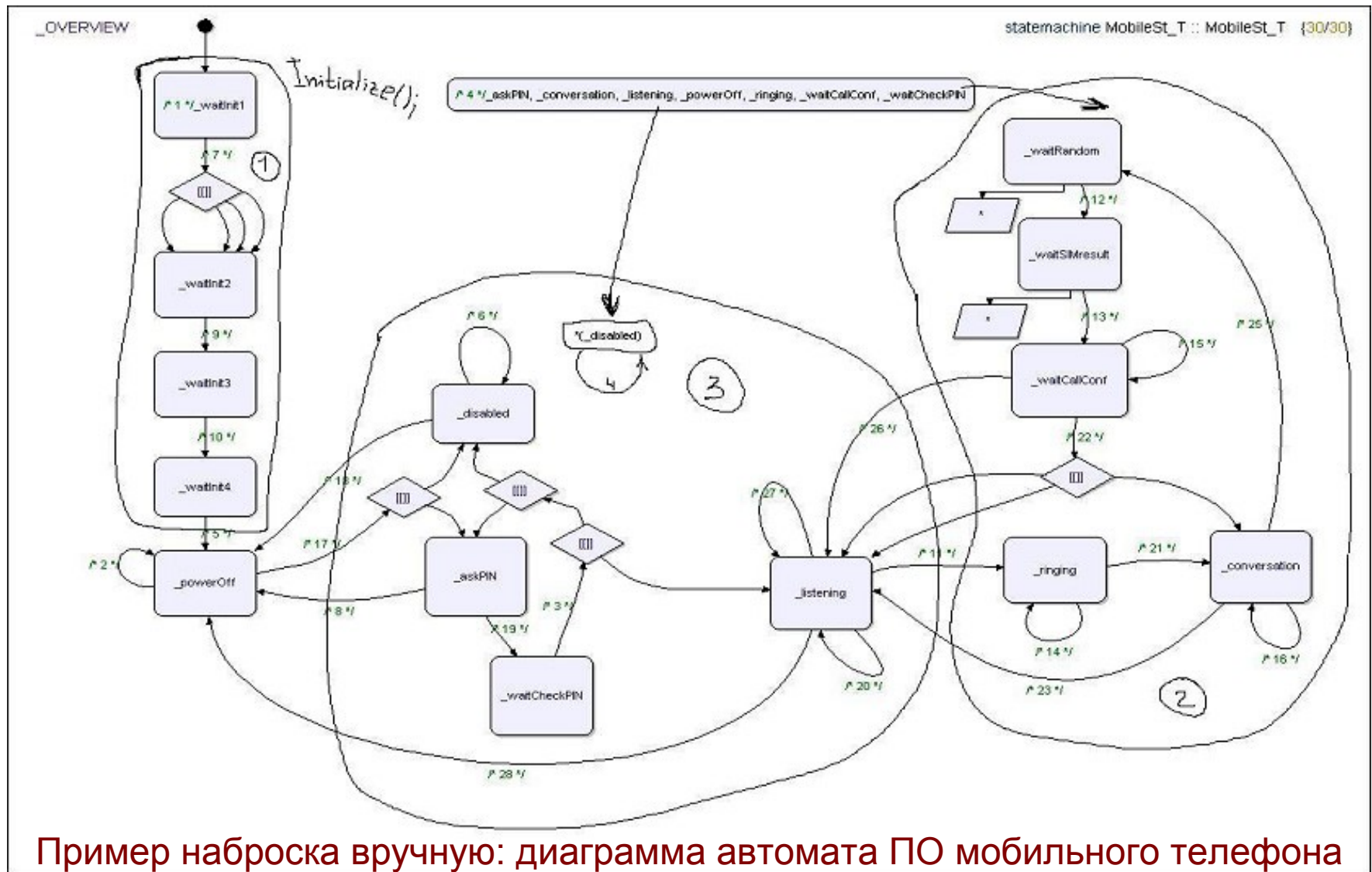
- Имя выполняемой подпрограммы
- Выкл. или Останов - состояния ожидания

NB: Слайды 11 – 14, 17 и 30 – только иллюстрации, не для опроса на экзамене 12

Диаграмма состояний процессов ОС

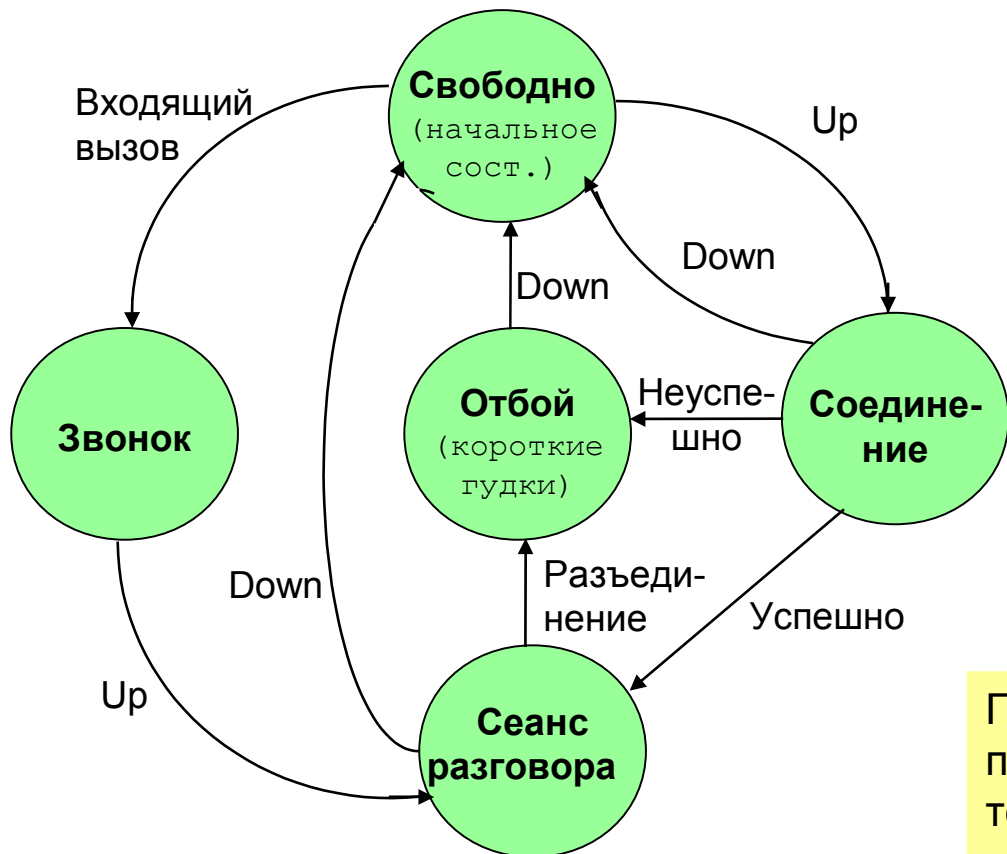


Эскиз диаграммы состояний



Модель работы телефона

Диаграмма состояний «телефонного процесса» абонента:



Up - трубка поднята с рычага

Down - трубка опущена на рычаг

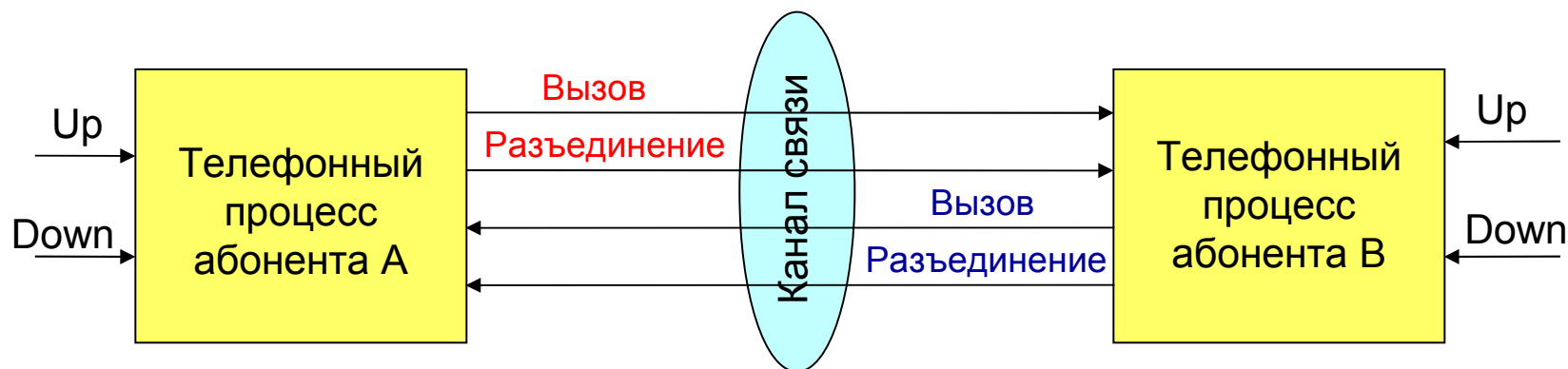
Соединение успешно, если на АТС найден путь, другой абонент свободен и снял трубку

Разъединение – по инициативе другого абонента

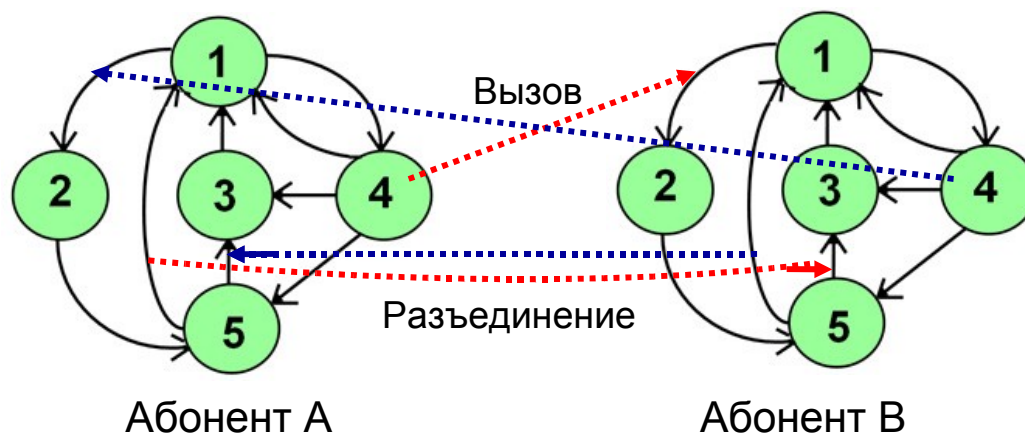
Действия в состояниях и на переходах не специфицированы

Процесс – распределенный: он происходит частично в телефонном аппарате, частично в аппаратуре АТС

Взаимодействующие автоматы



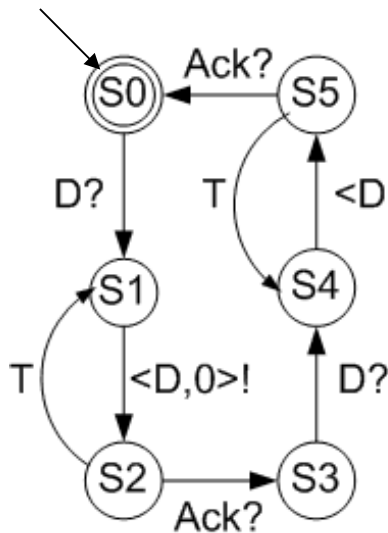
Автоматы обмениваются сигналами двух видов: Вызов и Разъединение



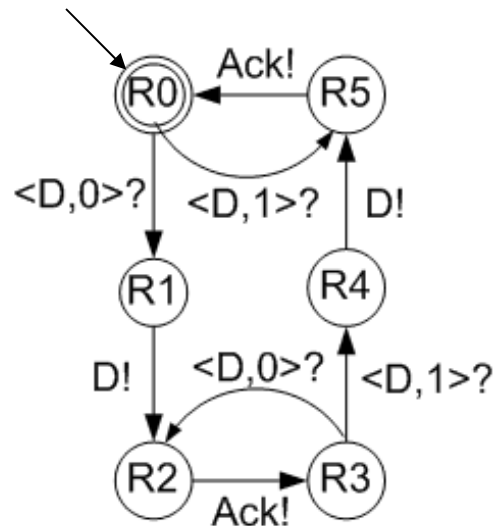
Спецификация сетевого протокола

Два взаимодействующих автомата:

Передатчик:



Приемник:



D! / D? – ввод/вывод данных пользователя

<D, №>! / <D, №>? – передача/прием сообщения с номером № из канала

Ack! / Ack? – передача /прием сигнала подтверждения приема

T – переход по таймауту

Протокол передачи сообщений PAR (Positive Acknowledge with Retransmission)

- Канал может исказить и терять сообщения
- Использование корректирующего кода позволяет обнаруживать искаженные сообщения, и они отбрасываются
- Для обнаружения потери сообщений они нумеруются по модулю 2

➤ Глобальное состояние системы – двухкомпонентный вектор $\langle S_i, R_j \rangle$

➤ Последовательность состояний $\langle S_0, R_0 \rangle, \langle S_1, R_0 \rangle, \langle S_1, R_1 \rangle, \dots$ -

12.04.10 траектория поведения системы из двух автоматов

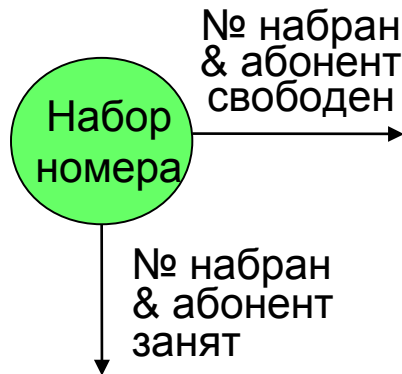
Расширения канонической модели КА

- С состояниями и/или с переходами связываются действия
 - ✓ Вершины/дуги диаграммы состояний помечаются именами действий
 - ✓ Если с состоянием не связано никаких действий, то это состояние ожидания следующего события (входа)
- С переходом связывается сторожевое условие (предикат), так что переход, заданный функцией переходов, осуществляется только при выполнении условия (истинности предиката)
- В диаграммах с большим числом состояний группу взаимосвязанных состояний можно объединить в *макростояние*: структуризация, улучшение обозримости модели
 - ✓ В UML это называется составным состоянием

Графические языки описания расширенных КА:

- SDL – Specification & Description Language (1976)
- Statecharts (D.Harel, 1972 → UML, 1995)
- Asml – Abstract State Machine Language (Microsoft, 2002)

Сторожевые условия перехода (предикаты)



Здесь нотация: <вход> & <предикат>

Функции переходов / выходов расширенного КА:

$f_s : X \times S \times P \rightarrow S$; $f_y : X \times S \times P \rightarrow Y$,

где P - множество предикатов – условий перехода

Шаблон табличного представления этих функций:

Текущее состояние S	Вход X	Предикат P	Выход Y	Новое состояние S
Набор номера	Номер набран	Абонент свободен	Длинные гудки	Вызов абонента
Набор номера	Номер набран	Абонент занят	Короткие гудки	Отбой

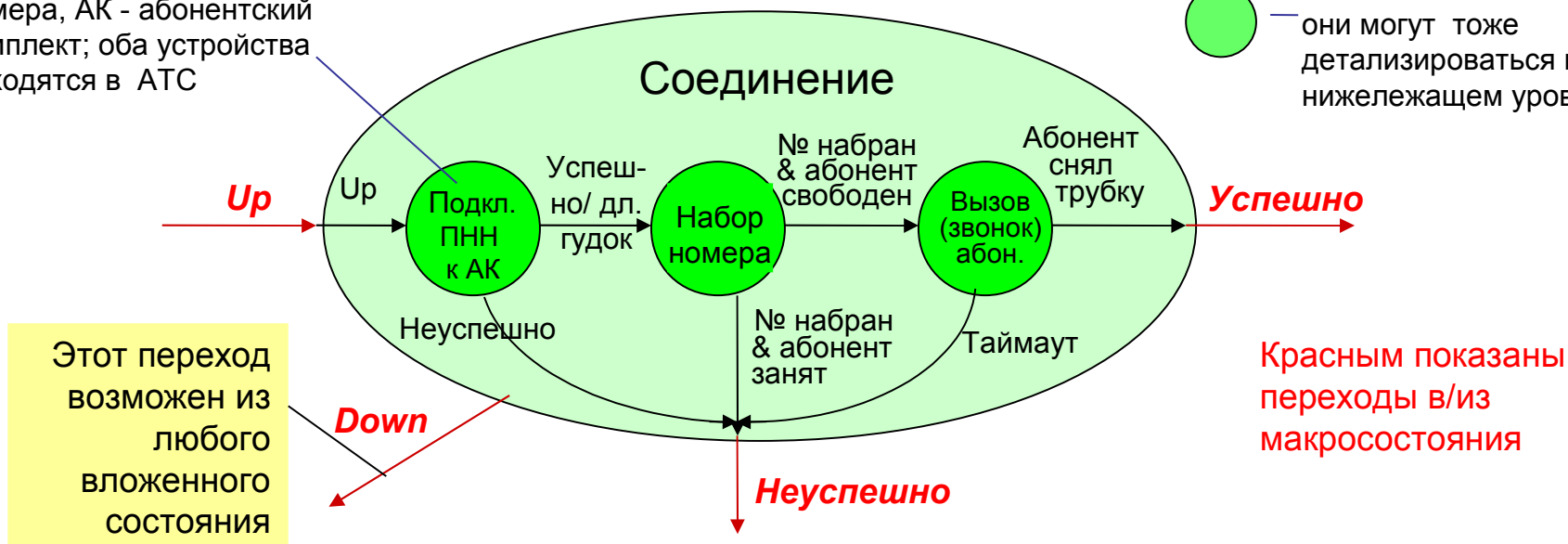
- Переход возможен, если только связанный с ним предикат = истина
- Значение предиката вычисляется в момент появления входного сигнала, связанного с данным переходом

Декомпозиция макросостояния "Соединение"

(см. исходную диаграмму состояний на слайде 15)

ПНН - приемник набора номера, АК - абонентский комплект; оба устройства находятся в АТС

Вложенные состояния; они могут тоже детализироваться на нижележащем уровне

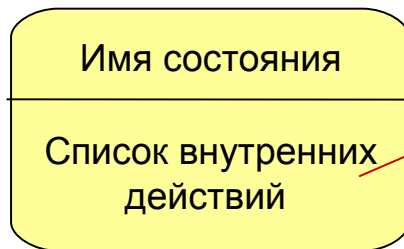


- Входная дуга перехода **в** макросостояния всегда должна вести в единственное его вложенное состояние
- Выходная дуга перехода **из** макросостояния может исходить из одного или нескольких вложенных состояний

Модель КА в языке UML. Состояния

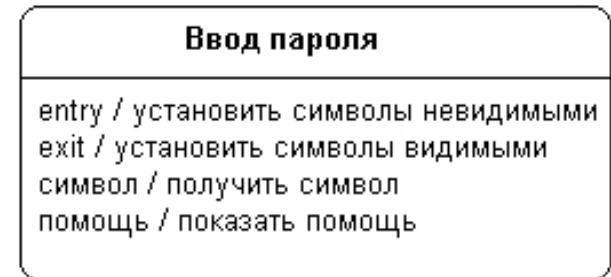
- Диаграммы UML для моделирования поведения: д. состояний, деятельности, последовательности и кооперации
- В отличие от других д., диаграмма состояний описывает процесс изменения состояний только одного экземпляра определенного класса – автомата (State machine)

← ● - начальное состояние → ● - конечное состояние



Изображение состояния

Набор строк вида:
<метка действия>/<действие>
(может быть пустым)



Пример состояния

Предопределенные метки внутренних действий:

- **entry** – входное действие, выполняется в момент входа в данное состояние
- **exit** - выходное действие, выполняется в момент выхода из данного состояния
 - действие – атомарное (непрерываемое) и «мгновенное»
- **do** - деятельность ("do activity") – завершается, когда закончится заданное вычисление или произойдет переход в другое состояние

В остальных случаях метка действия идентифицирует событие, которое запускает соответствующее действие; эти события называются внутренними переходами

Модель КА в языке UML. Переходы

Переход осуществляется при наступлении одного из двух событий:

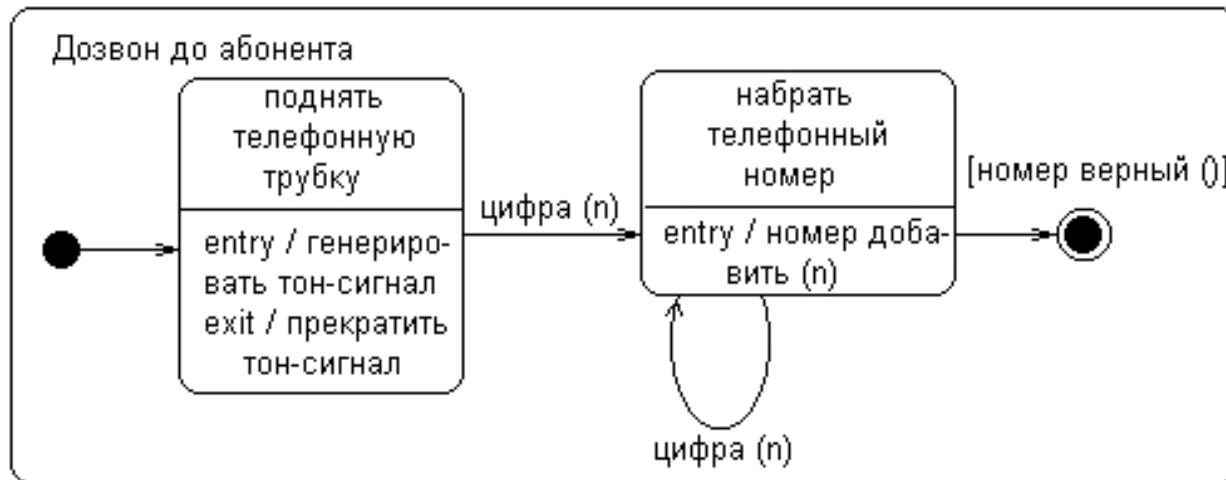
- окончания выполнения деятельности (do activity)
- приема входного сигнала

Каждый переход может быть помечен строкой текста:

<событие>[<сторожевое условие>] <действие>

Событие: <имя события>, или

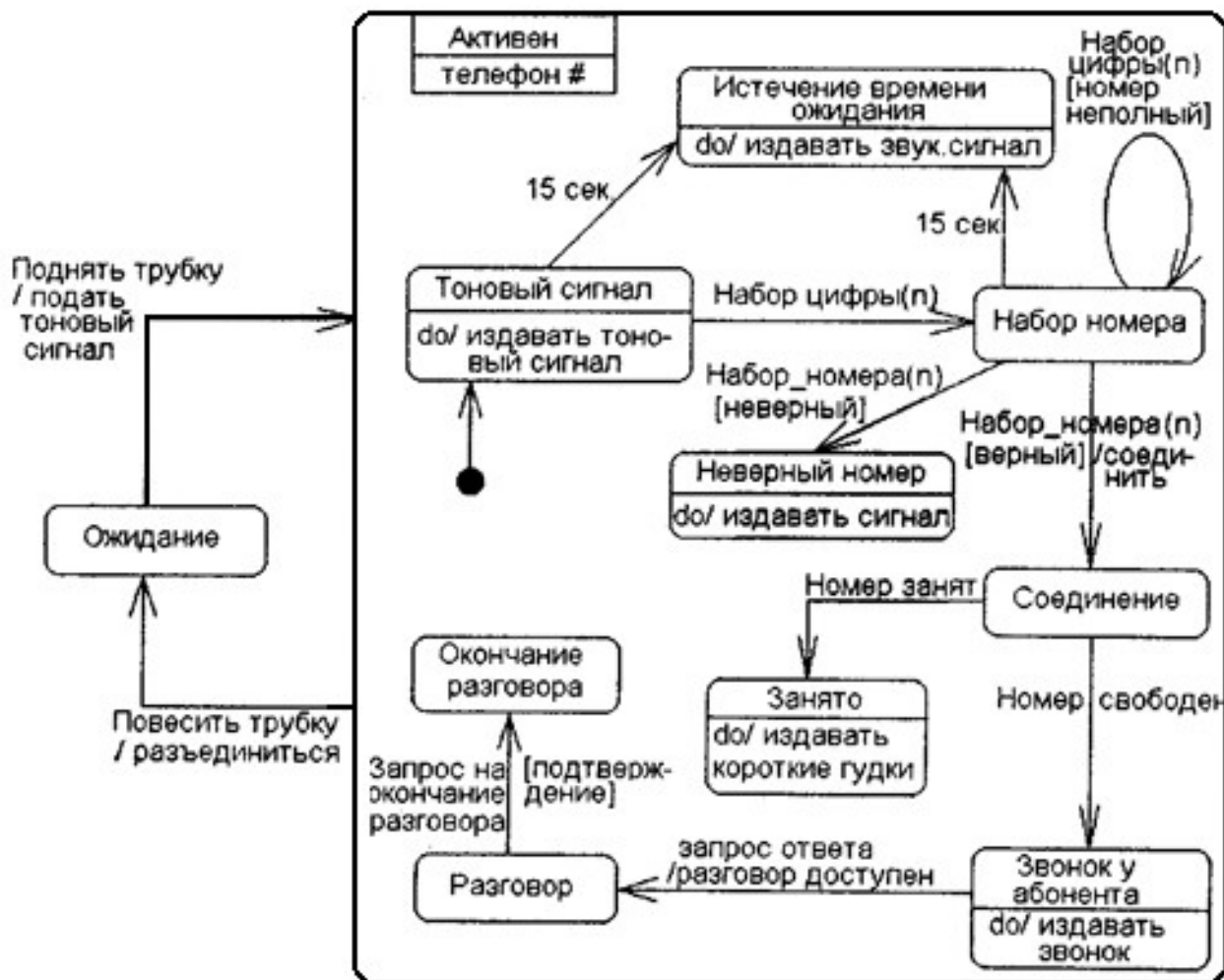
<имя события> (<список параметров, разделенных запятыми>)



Пример: *последовательное* составное состояние с одним вложенным автоматом

(возможны *параллельные* составные состояния с несколькими вложенными автоматами, работающими параллельно)

UML-модель работы телефона



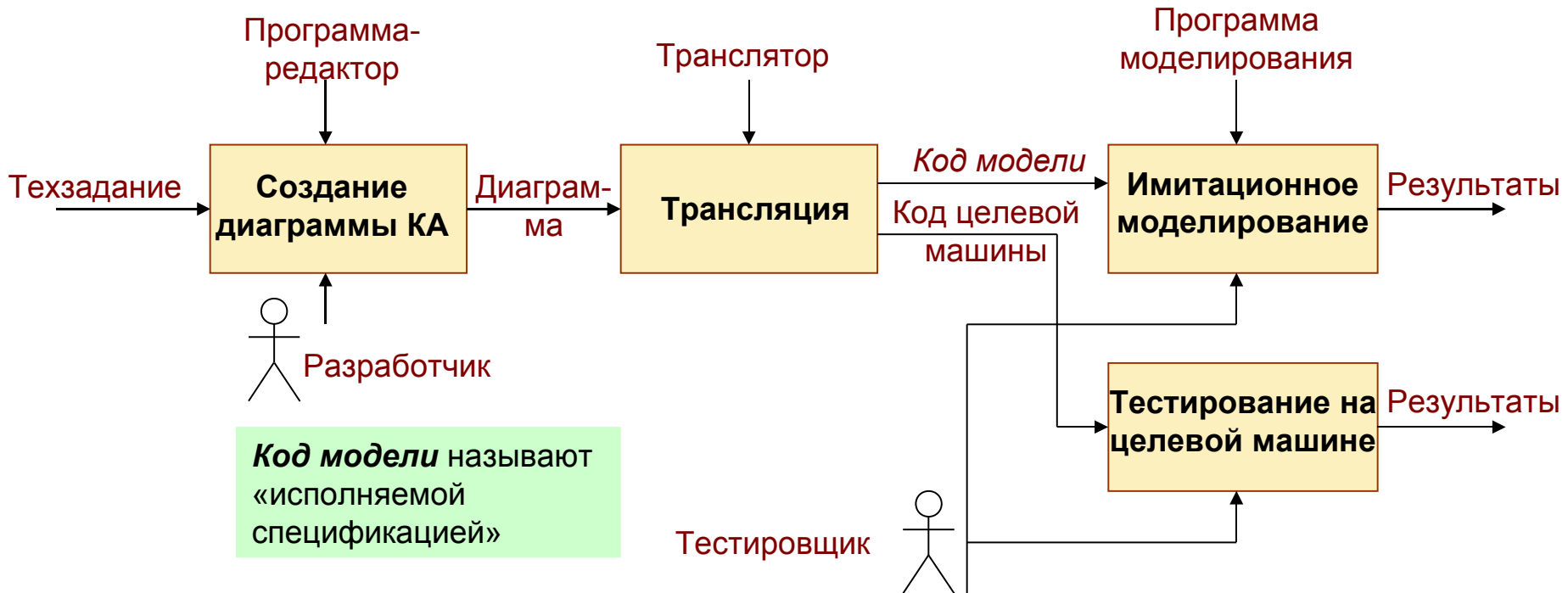
Программная реализация КА

Общая схема (одна из возможных):

```
state := 1;           // 1 - начальное состояние
while state ≠ k { // k - финальное состояние
    input = get();    // ожидание входного символа
    switch (state){
        case 1:
            switch (input) {
                case x1:
                    state = ... ; // переход в новое состояние
                    A = f(...);   // действие в новом состоянии

                    break;
                case x2:
                    ...
            }
        case 2:
            ...
    }
}
```

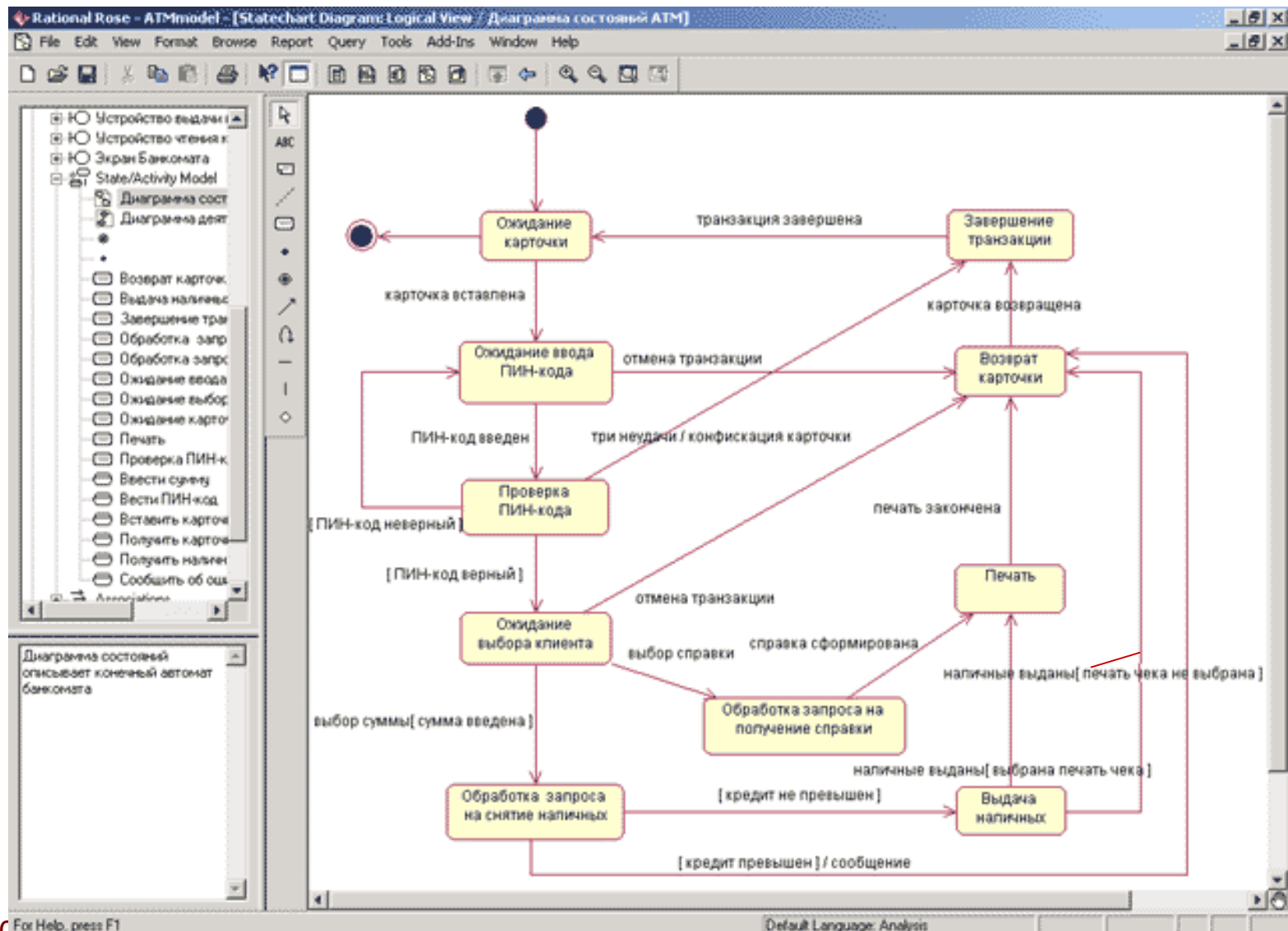

Инструментальные системы автоматного программирования



Инструментальные системы:

- Statemate
 - Rational Rose (IBM)
- } Визуальный формализм Statecharts Д.Харела
- Switch-технология
 - UniMod
- } «Автоматное программирование» А.Шальто
- Stateflow в составе Simulink / MatLab

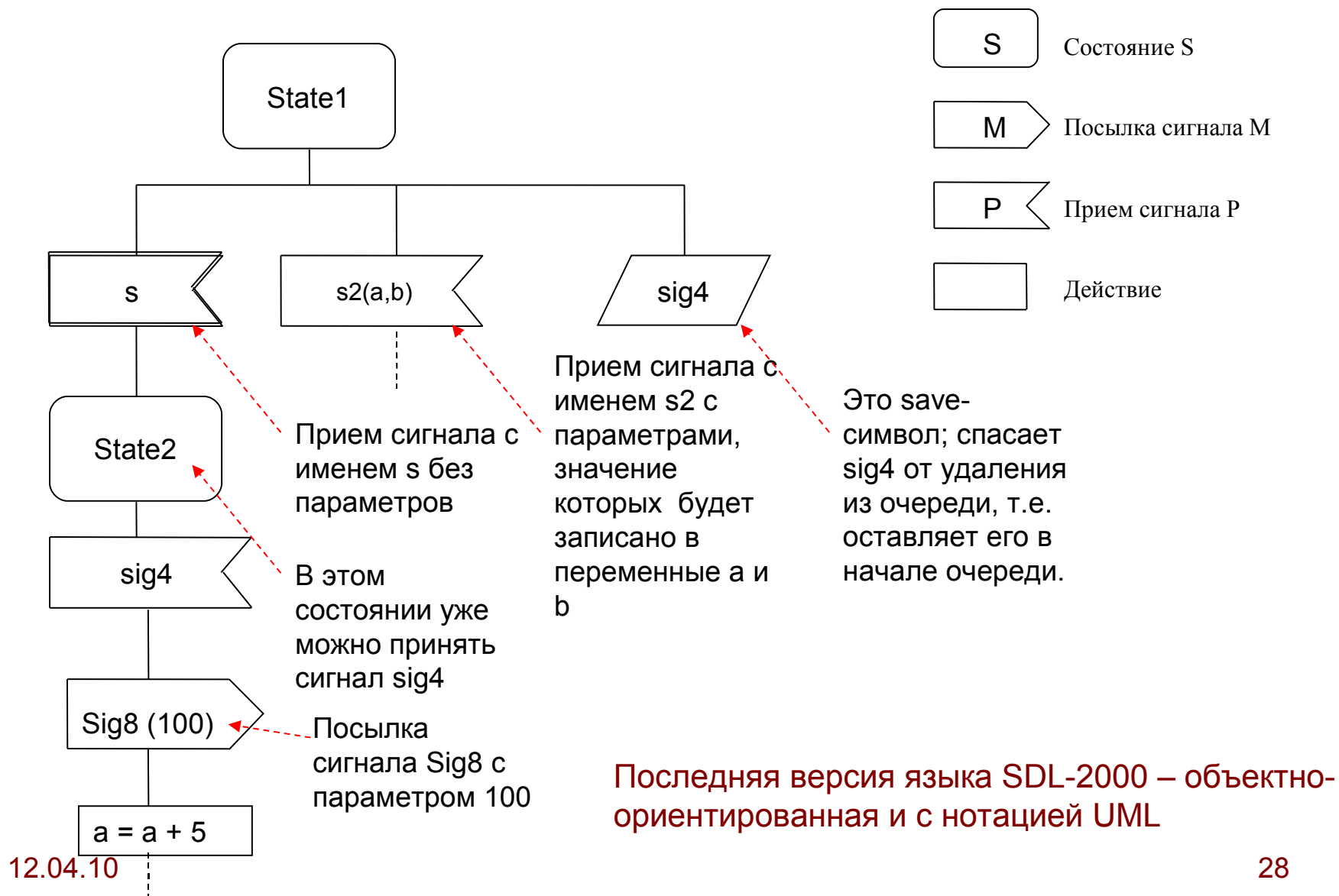
Модель банкомата в Rational Rose



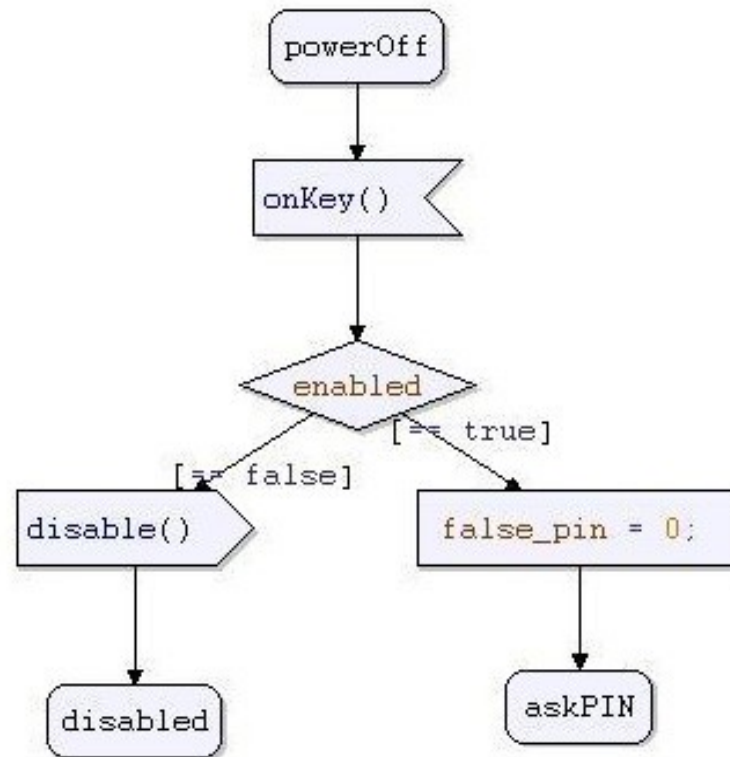
Язык SDL (Specification & Description Language)

- Применяется с 1976 для проектирования телекоммуникационных систем; конкурирует с UML в этой предметной области
- Основан на модели взаимодействующих конечных автоматов
 - ✓ Взаимодействие — обмен асинхронными сигналами: <имя>(<параметры>)
- Имеет две нотации: графическую и текстовую и трансляторы на языки программирования
- Компоненты функциональной структуры: система, блок, процесс, процедура — вложенные друг в друга
 - ✓ Процесс описывается как расширенный КА
- В состояниях нет действий: любое состояние — это ожидание прихода сигнала
 - ✓ Все действия происходят на переходах: там описаны ветвления, арифметические выражения, посылка сигналов, вызов процедур и т. п.
- Составных состояний нет
- Невостребованные сигналы (т. е. для которых не описан переход) теряются, только если нет явного указания save для их сохранения
 - ✓ согласно терминологии ОС, сигналы, сохраняемые в очередях (буферах), следует называть сообщениями

Элементы графического языка SDL-88

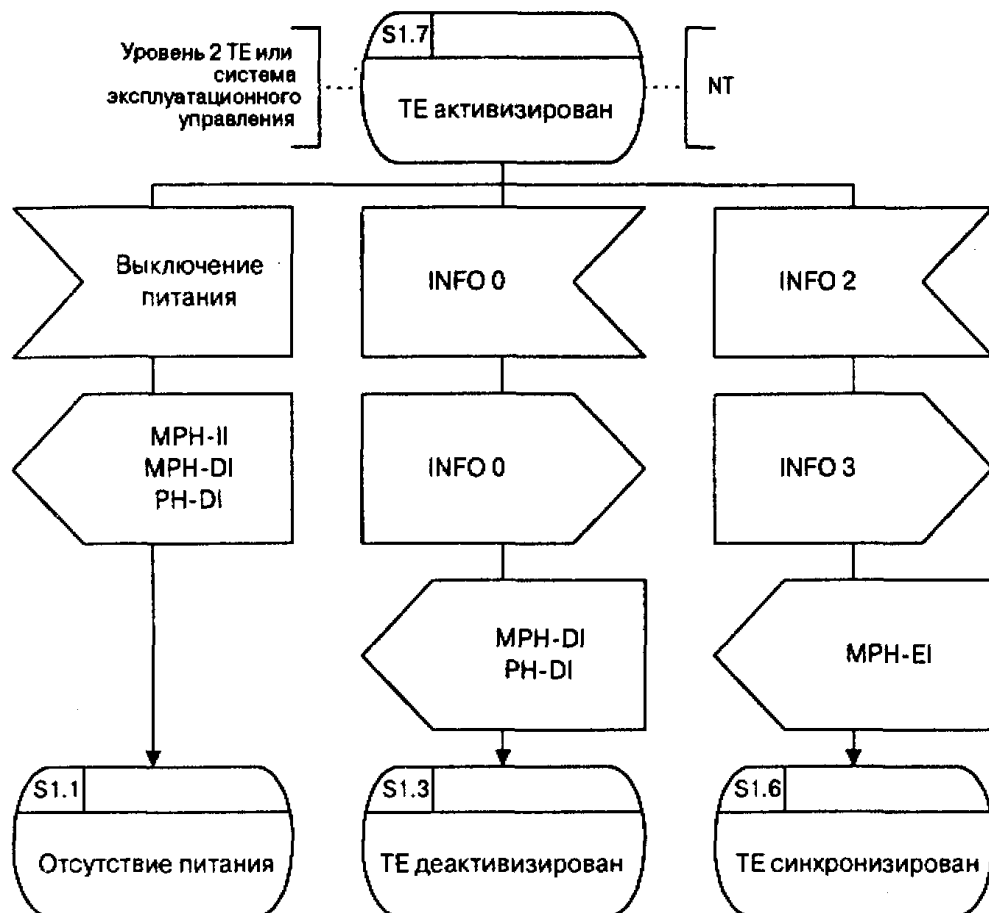


Пример модели SDL (1)



Реакция мобильного телефона на нажатие кнопки On

Пример модели SDL (2)



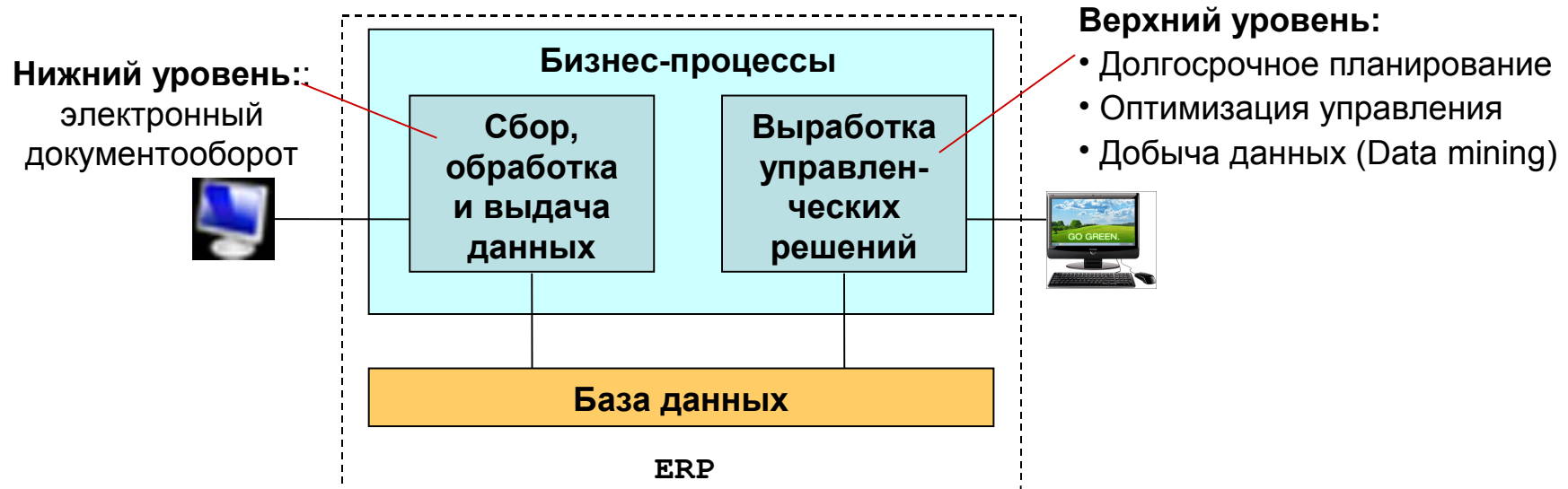
Фрагмент некоторого сетевого протокола

Резюме по моделям КА

- Диаграммы состояний КА – широко распространенный способ *непроцедурного* описания поведения систем разного назначения в форме событийной модели
- Диаграммы КА наглядны, понятны непрограммистам и служат основой для моделирования и тестирования программно-аппаратных систем
- Автоматическое преобразование диаграмм состояний в код применяется для генерации исполняемых спецификаций (при проектировании) или кода для целевой машины (при реализации)

Модели экономических информационных систем

- Англоязычный термин: ERP (Enterprise Resource Planning) или ICAM (Integrated Computer Aided Manufacturing)
- Здесь объекты управления – организационные структуры: люди + производственное оборудование



Виды моделей описания бизнес-процессов

- **Структурный анализ:** единое представление материальных и информационных процессов в их взаимосвязи с организационной и технической структурой предприятия.
- **Диаграммы потоков данных** - DFD (Data Flow Diagram) или **документов** – DocFlow
- **Диаграммы потоков работ** - WFD (Work Flow Diagram)

Структурный анализ

Основа – универсальный графический язык диаграмм SADT (Structured Analysis and Design Technique) и его развитие – IDEF (Integrated DEfinition Function modeling) с базовым языком IDEF0

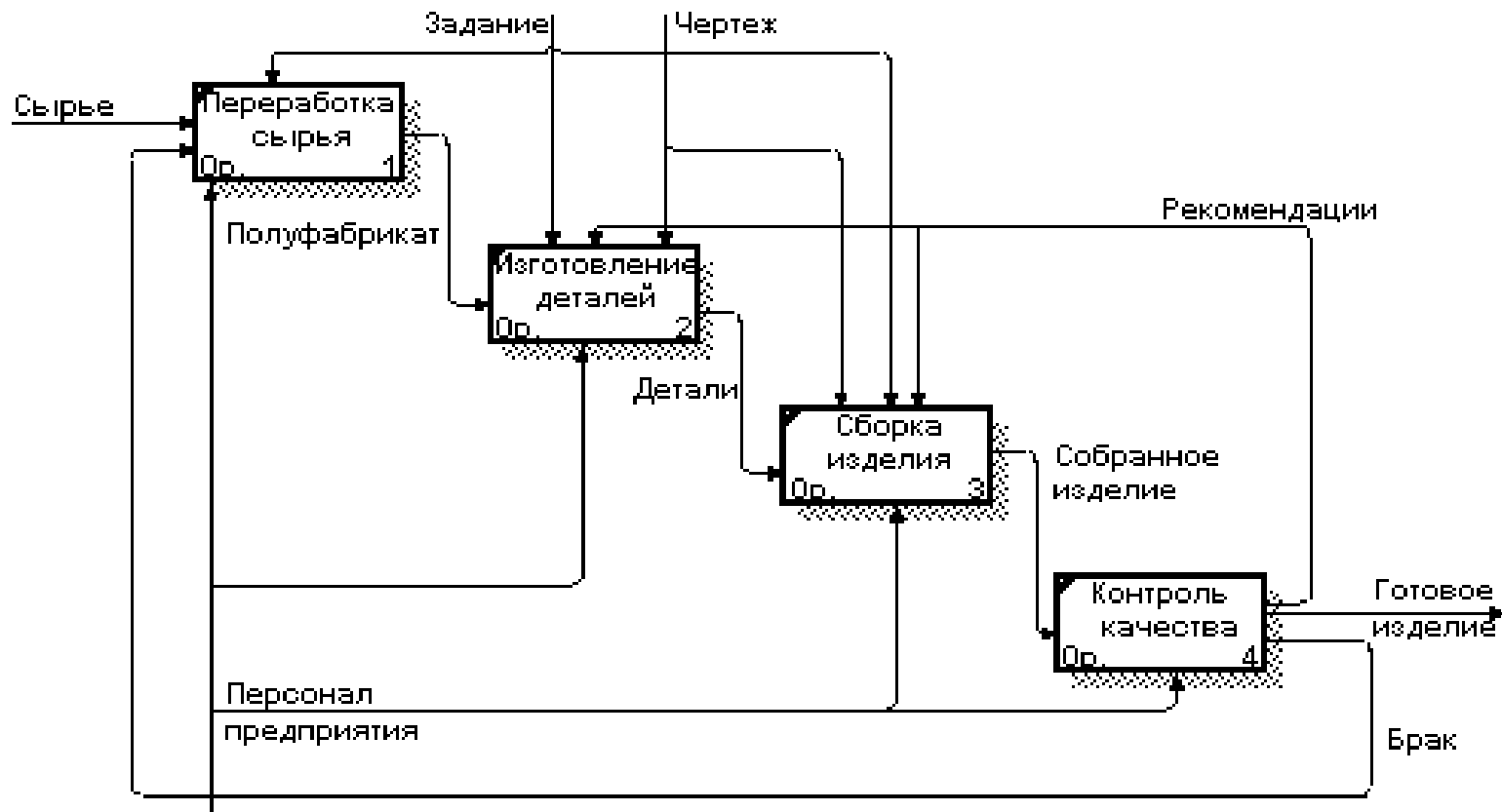
Основные понятия IDEF0

- **Операция** – элементарное (неделимое) действие на одном рабочем месте
- **Функция** – это совокупность операций – некоторый процесс, преобразователь входных объектов в выходные
- Последовательность взаимосвязанных по входам и выходам функций образует **бизнес-процесс**, который может порождать объекты любой природы (материальные, денежные, информационные)
- Материальные процессы и информационные бизнес-процессы неразрывны
 - напр., отгрузка готовой продукции осуществляется на основе документа "Заказ", который, порождает документ "Накладная", сопровождающий партию отгруженного товара
- **Бизнес-модель** предприятия – структурированное графическое описание сети процессов и подпроцессов, связанных с данными, документами, организационными единицами и прочими объектами, отражающими существующую или планируемую деятельность предприятия
 - Список основных видов бизнес-процессов предприятия обычно насчитывает 15–20 видов и соответствует перечню подсистем (управление кадрами, CRS,

Функциональный блок IDEF0



Пример диаграммы IDEF0 (1)



Бизнес-процесс изготовления изделий

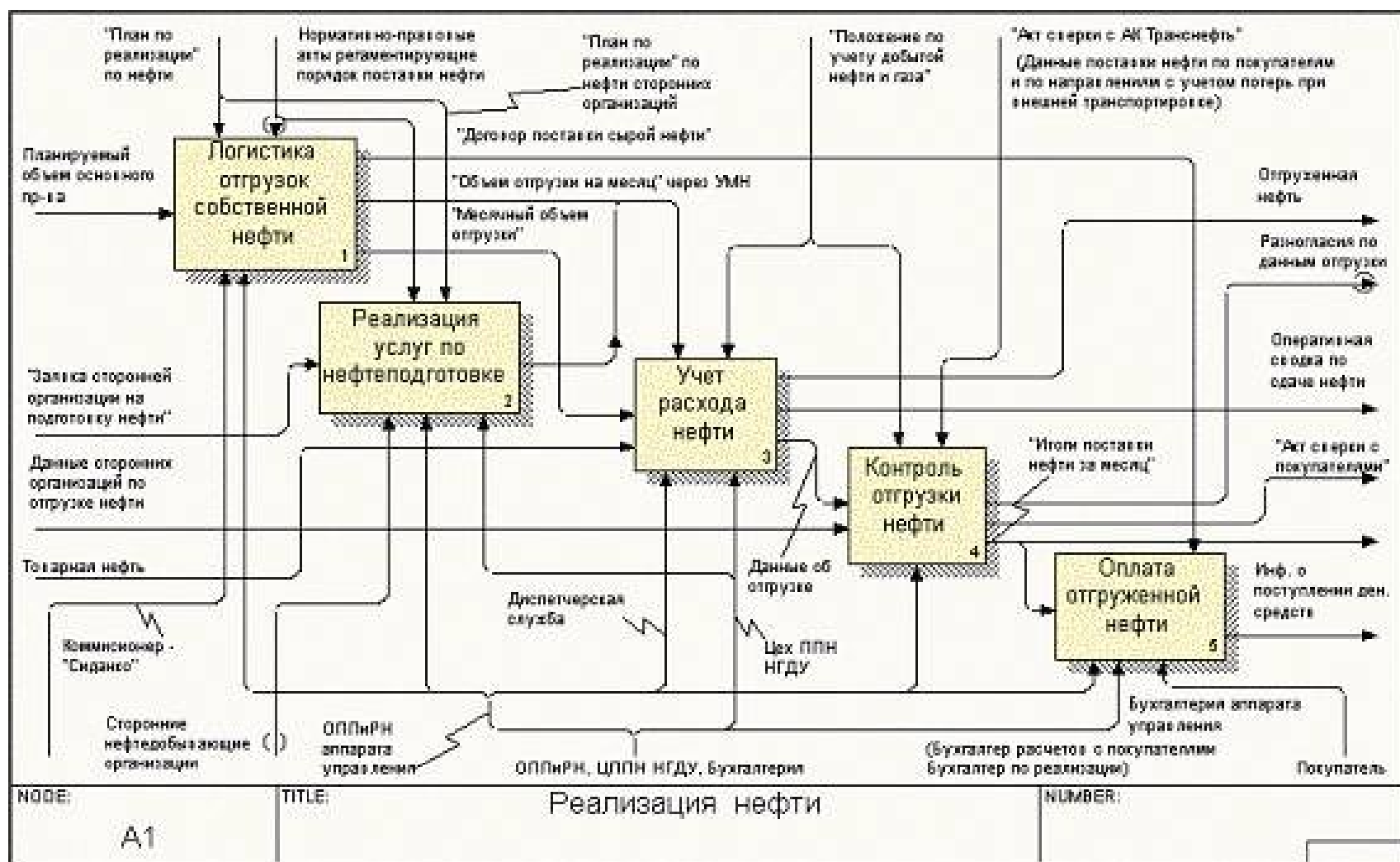
Еще один пример диаграммы IDEF0 – на слайде 25

Пример диаграммы IDEF0 (2)



Контекстная диаграмма – описывает функционирование организации в целом
12.04.10

Пример диаграммы IDEF0 (3)



Основные элементы DFD

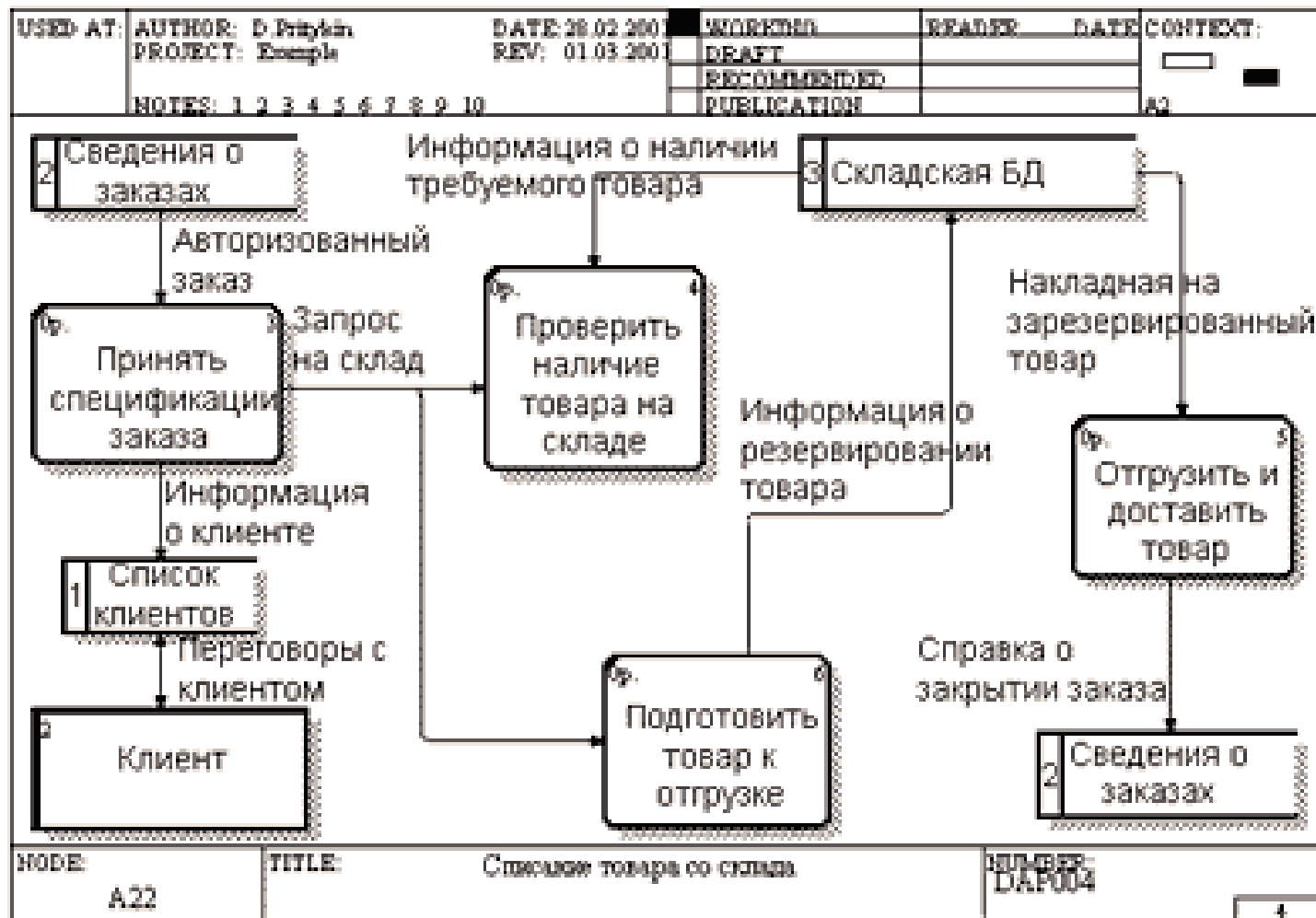


(Это нотация, используемая в BPWin)

Отличия от процессной модели IDF0:

- Отображается хранение данных
- Материальные потоки не моделируются
- Стрелки могут быть изотропны и изображать только движение данных, но не управление или механизм

Пример DFD (1)



Бизнес-процесс подготовки и отгрузки товара со склада по заказу клиента

Пример DFD (2)



Бизнес-процесс регистрации входящих документов и писем

12.04.10

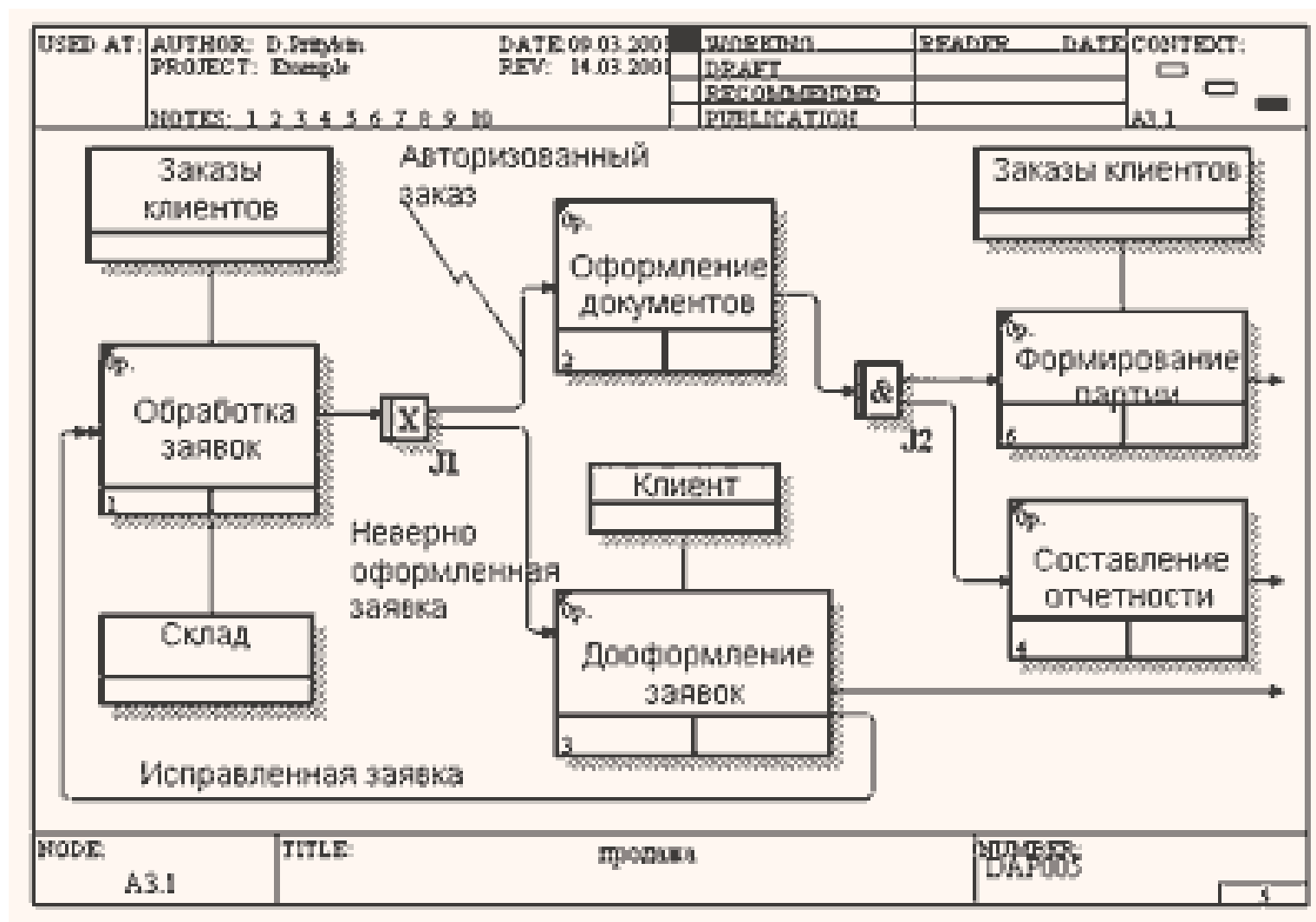
(здесь направление стрелок имеет ту же семантику, как в IDEF0)

Модели потока работ (WFD)

- используются для описания последовательности работ в их логической и временной взаимосвязи

WFD-нотация IDEF3 близка к DFD; дополнительно отображаются связи между работами: отношение следования, параллельное выполнение, разветвление, синхронизация

Пример диаграммы потока работ



Последовательность проектирования информационных систем

1. Сначала создается модель существующей неавтоматизированной системы AS-IS (Как есть) на основе:
 - ✓ изучения документации (должностных инструкций, положений о предприятии, и т.п.)
 - ✓ анкетирования и опроса служащих
2. Модель AS-IS анализируется на предмет выявления дублированных, ненужных, неэффективных и неуправляемых работ и других недостатков в организации деятельности предприятия
3. Исправление недостатков, перенаправление информационных и материальных потоков приводит к созданию модели TO-BE (Как будет) - модели идеальной организации бизнес-процессов
 - ✓ это называют реинжинирингом бизнеса
4. На основе модели TO-BE и модели базы данных разрабатывается спецификация системы – техническое задание для программистов

Это работа системных аналитиков и архитекторов ПО

Инструменты бизнес-моделирования

1. BPWin (Business Processes for Windows)

Поддерживает три вида моделей: IDEF0, DFD и IDEF3

- Редактирование с синтаксическим контролем
 - ✓ в том числе контроль целостности и соответствия имен объектов в трех моделях
- Поддержка иерархической структуры моделей
- Вычисление сводных характеристик (стоимостных, временных)
- Сопряжение моделей с инфологической моделью Entity-Relation в инструменте ERWin

2. IBM Rational

UML-диаграммы:

- ❖ Универсальны, не ориентированы на бизнес-процессы
- ❖ Менее понятны администраторам
- ❖ Используются на более поздних этапах проектирования

3. Языки проектирования в составе ПП ERP: SAP, MS Dynamics, C1

Резюме по моделям экономических информационных систем

- Модели описывают бизнес-процессы, в которых участвуют людские, информационные, материальные и финансовые ресурсы и потоки
- Графические диаграммы служат для наглядного представления моделей; есть несколько инструментов их редактирования и полуавтоматического анализа

Дополнительные вопросы

1. Предположим, что транслятор с языка таблиц решений генерирует последовательность операторов IF. Какие автоматические оптимизации при этом можно осуществить ?
2. Можно ли выключить систему управления роботом (диаграмма на слайде 12) во время ее запуска? Какую последовательность команд нужно применить для ее выключения в состоянии «Движение» ?
3. Какие переходы на диаграмме слайда 15 вызваны внутренними событиями, а какие – внешними?
4. Опишите в виде диаграммы состояний КА процесс управления движением кабины лифта между этажами. Входные сигналы – кнопки вызова на этажах и номеров этажей в кабине, а также датчики прохождения кабиной очередного этажа. Выходные сигналы - команды подъемного механизма: Вверх, Вниз и Стоп.
5. Детализируйте макросостояние «Набор номера» телефонного абонента: что происходит во время набора 7 цифр для местного номера или 11 для междугороднего.
6. Опишите в виде диаграммы IDEF0 технологический процесс подготовки программного модуля к выполнению, начиная с написания исходного кода и заканчивая загрузкой исполняемого кода в память