Структурный подход проектирования ПП

В основе лежит декомпозиция программы по функциональному принципу.

Выделяют два варианта проектирования:

- Обрабатывающие компоненты считают основными и их проектируют в первую очередь, а компоненты данных проектируют попутно.

- Компоненты данных являются основными и сначала их проектируют, а затем проектируют обрабатывающие компоненты.

Цель проектирования

Цель любого проектирования –

разработать *формальную модель*.

Группы формальных моделей:

- **Модели, не зависящие от подхода к разработке**
  (диаграммы переходов состояний, математические модели предметной области, структурные схемы);
- **Модели для структурного подхода**
  (функциональные диаграммы, диаграммы потоков данных, диаграммы отношений компонентов данных);
- **Модели для объектно-ориентированного подхода**
  (диаграммы вариантов использования, контекстные диаграммы классов. диаграммы последовательности).

Рекомендации:

- Целесообразно представить программный продукт:
  - с разных сторон;
  - с помощью ряда моделей (диаграмм и др.);
  - с описанием компонентов.

- Основные диаграммы необходимо согласовать с заказчиком.

Обычно проектирование начинают
с определения структуры ПП.
Показать структуру можно с помощью нескольких схем (диаграмм).
Рассмотрим основные.

Структурная схема

Отражает:

- ***состав (компоненты)*** *разрабатываемого программного продукта;*
- ***взаимодействие по управлению*** частей разрабатываемого продукта.

Структурными компонентами могут быть:

программы
подсистемы
базы данных
библиотеки ресурсов и т. п.

Структурные компоненты делятся на три вида:

- **Интерфейсные**
  (обеспечивают взаимодействие системы с внешними сущностями);

- **Обрабатывающие**
  (обеспечивают функционал системы);

- **Компоненты данных**
  (обычных, управляющих, служебных, мета-данных и др.).

Пример структурной схемы

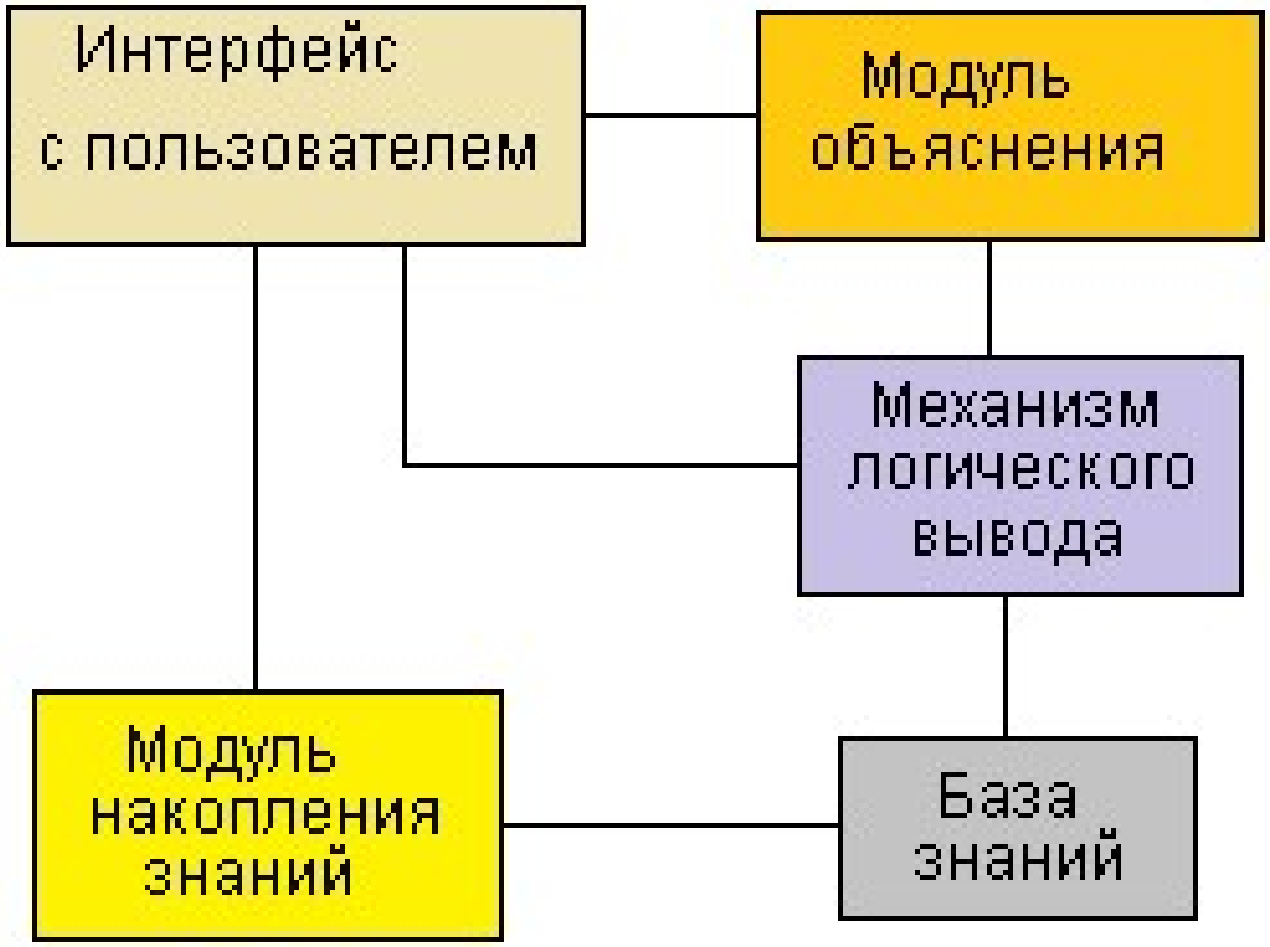


Рисунок - Структурная схема экспертной системы.

Достоинства структурной схемы:

- ✓ структурная схема универсальна;
- ✓ позволяет показать все части программного продукта;
- ✓ части могут разрабатываться по различным технологиям.

Пример, на структурной схеме можно показать:

- ✓Структуру информационной системы в целом;
- ✓Структуру подсистемы (детально и отдельно);
- ✓Структуру информационной системы, где компоненты, не подлежащие проработке, отображаются не детально, а подлежащие проработке детально (с целью показать место разрабатываемой части в структуре информационной системы).

Функциональная схема

Функциональная схема более информативна.

Показывает:

- взаимодействие компонентов;
- информационные потоки;
- состав данных в потоках;
- используемые файлы и устройства.

Специальные обозначения установлены стандартом ГОСТ 19.701-90 .

Пример функциональной схемы

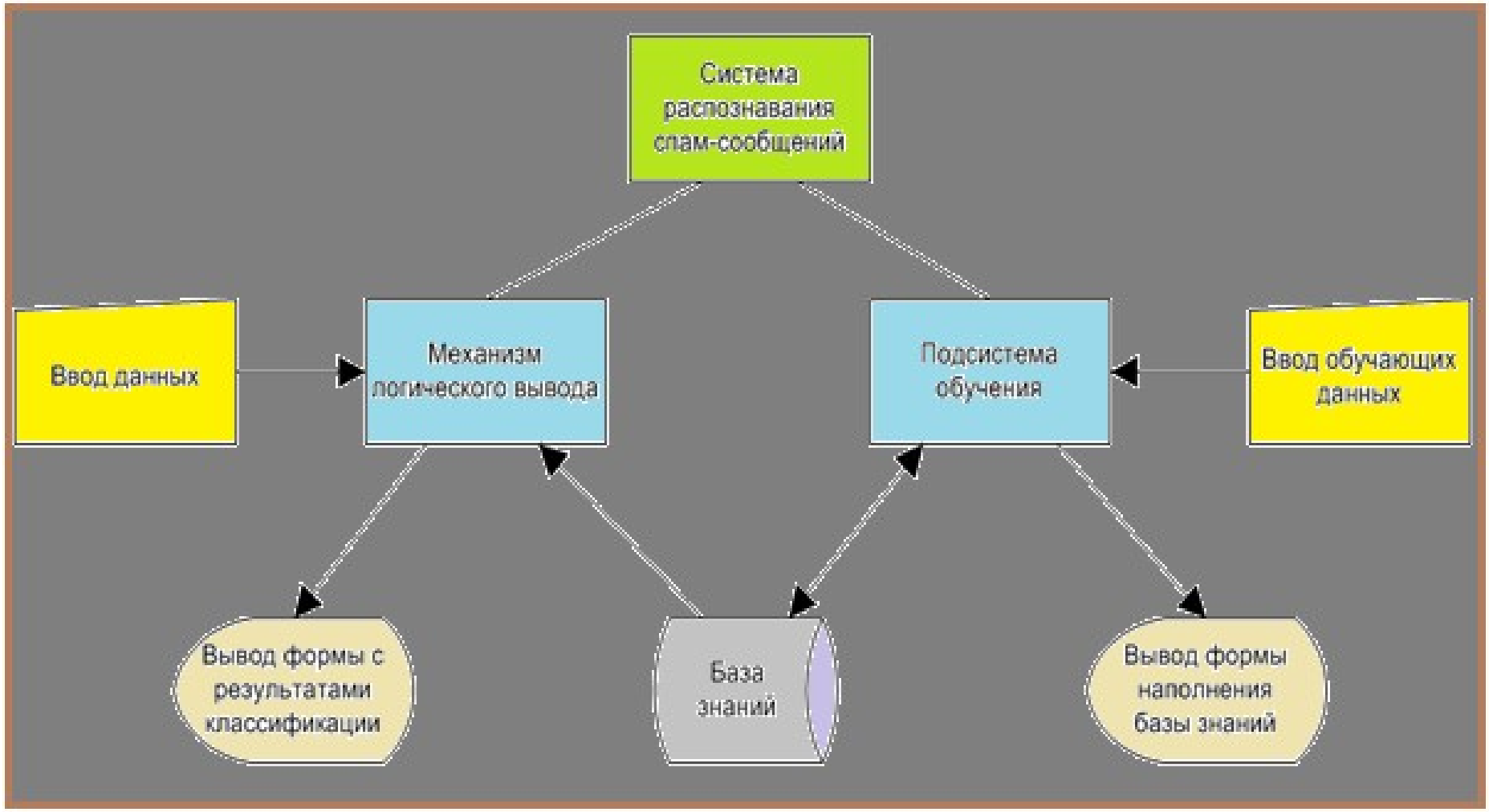


Рисунок - Функциональная схема системы СРС-С.

После определения структуры необходимо описать компоненты.

Ниже приведено описание компонентов функциональной схемы системы СРС-С.

Механизм логического вывода (МЛВ) осуществляет следующие действия:

- принимает сообщение из модуля ввода данных;
- обращается к базе знаний за необходимой информацией;
- проводит классификацию и отображает пользователю результат классификации.

Подсистема обучения позволяет внести изменения в базу знаний, т.е. пересчитать вероятности и другие параметры в соответствии с результатами классификации.

Диаграммы переходов состояний

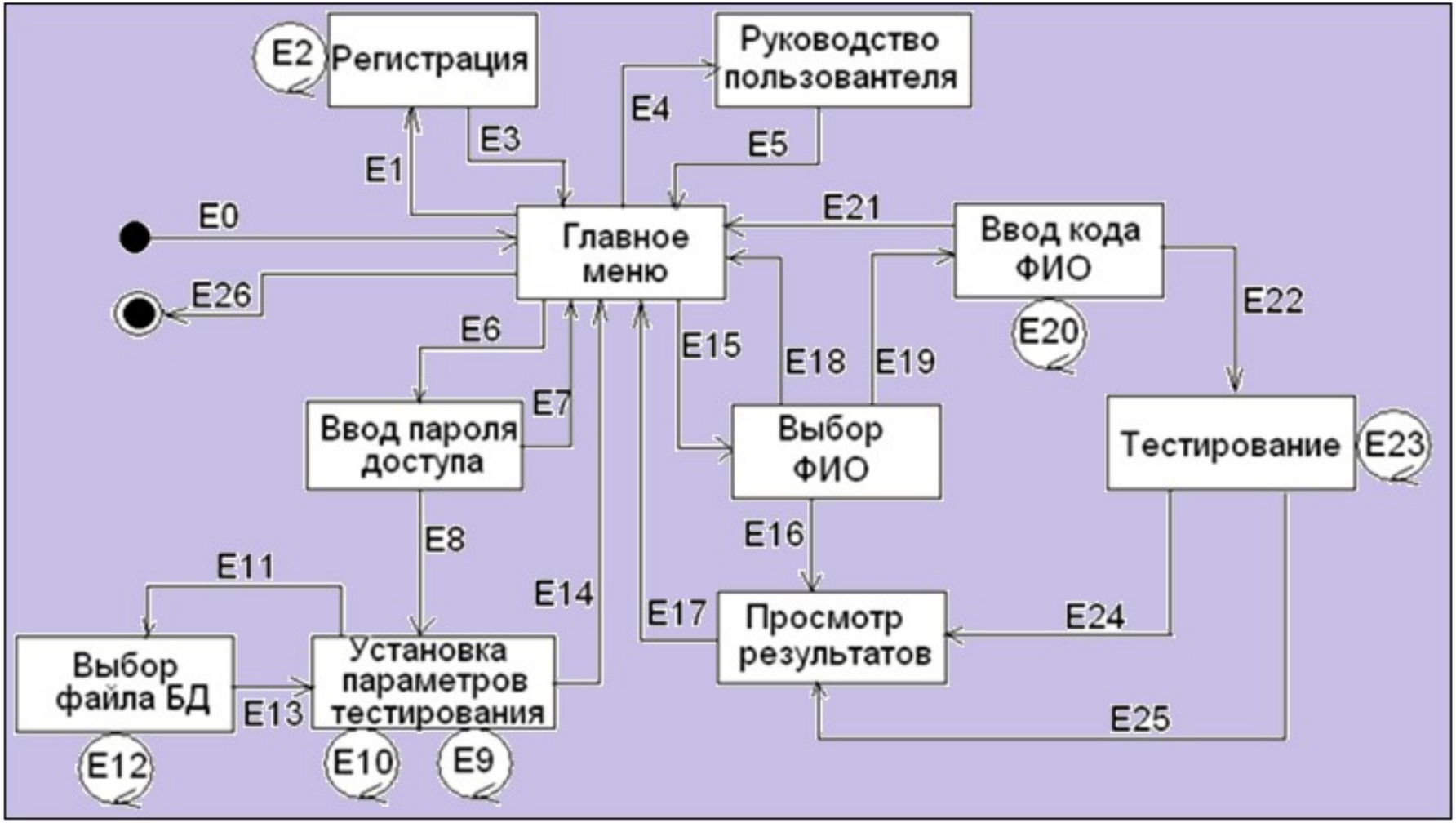
Используются для моделирования поведения ПП.

Это графическая форма представления **конечного автомата**.

Для построения диаграммы необходимо определить:

- основные состояния;
- управляющие воздействия (или условия перехода);
- выполняемые действия;
- возможные варианты переходов из одного состояния в другое.

Пример:


E2
Регистрация
Руководство пользователя
E4
E3
E5
E1
E0
Главное меню
E21
Ввод кода ФИО
E26
E22
E6
E15
E18
E19
E20
E7
Ввод пароля доступа
Выбор ФИО
Тестирование
E23
E8
E16
E11
E14
E17
Просмотр результатов
E24
Выбор файла БД
Установка параметров тестирования
E13
E25
E12
E10
E9

Используются следующие условные обозначения:

Описание событий:

E0 - ресурсов достаточно/запуск исполняемого модуля
E1 - база данных подключена/выбран пункт меню "Регистрация"
E2 - ввод ФИО и кода ФИО
E3 - введены правильно ФИО и код/нажата кнопка "Зарегистрировать"
или нажата кнопка "Отменить"
E4 - существует файл Help.txt/выбран пункт меню "Справка"
.....
E24 - последний вопрос/ нажата кнопка "Закончить"
E25 - закончилось время тестирования

Математические модели предметной области

Математические модели используют:

- ✓ Если алгоритм решения задачи не очевиден;
- ✓ Если алгоритм решения задачи формализуем.

Существует много моделей и методов решения, например, для задач **аналитической геометрии**, моделирования **дискретных систем** и т.д.

Основная проблема – *обоснование применимости* математической модели для решения конкретной задачи.

Если методов решения существует **несколько**, тогда для выбора может потребоваться **специальное исследование**.

При выборе метода учитывают:

- особенности данных конкретной задачи, связанные с предметной областью (погрешность, возможные особые случаи и т.п.);
- требования к результатам (допустимую погрешность);
- характеристики метода (точный или приближенный, погрешности результатов, вычислительную и емкостную сложности, сложность реализации и т.п.).

Модели для структурного подхода

В данных моделях часто используется **метод пошаговой детализации** (см. ранее).

Основное метода пошаговой детализации:

- ✓ предполагает разложение функции на подфункции;
- ✓ сначала детализируют управляющие процессы (данные попутно);
- ✓ получают структурную схему, отражающую результаты процесса декомпозиции.

Рекомендации:

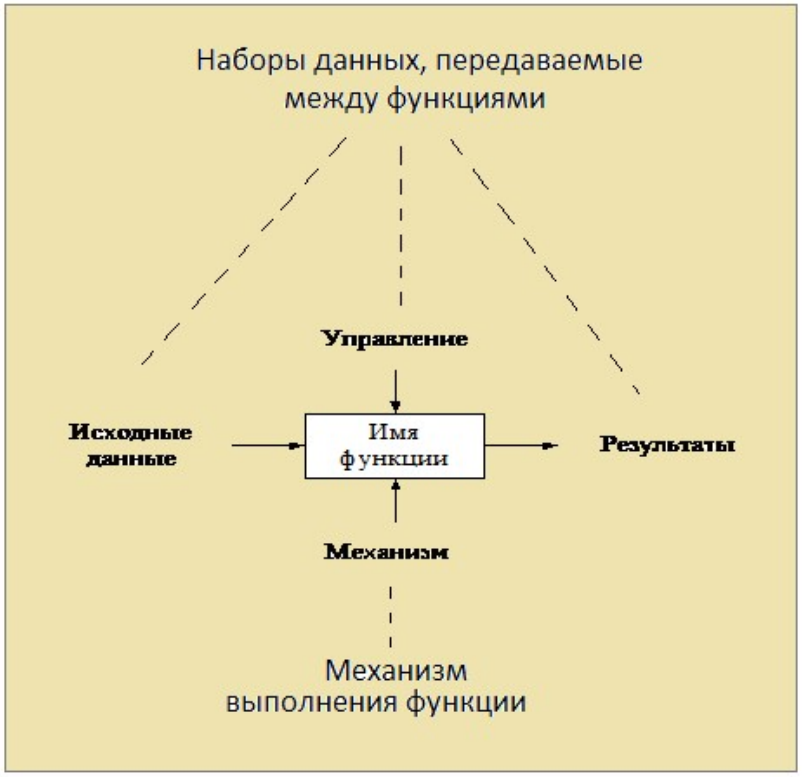
- Не проектировать слишком специализированные модули (увеличивается их количество);
- Не проектировать слишком универсальные модули (увеличивается сложность);
- Избегать дублирования действий в различных модулях;
- Группировать сообщения об ошибках в один модуль (будет легче согласовать формулировки, избегать дублирования сообщений и переводить на другой язык).

Функциональные диаграммы

Основное:

- это схематическое **отображение** взаимосвязей нескольких **функций**;
- используют, если нет **сложных** структур **данных**.

Для каждой функции должны быть определены:

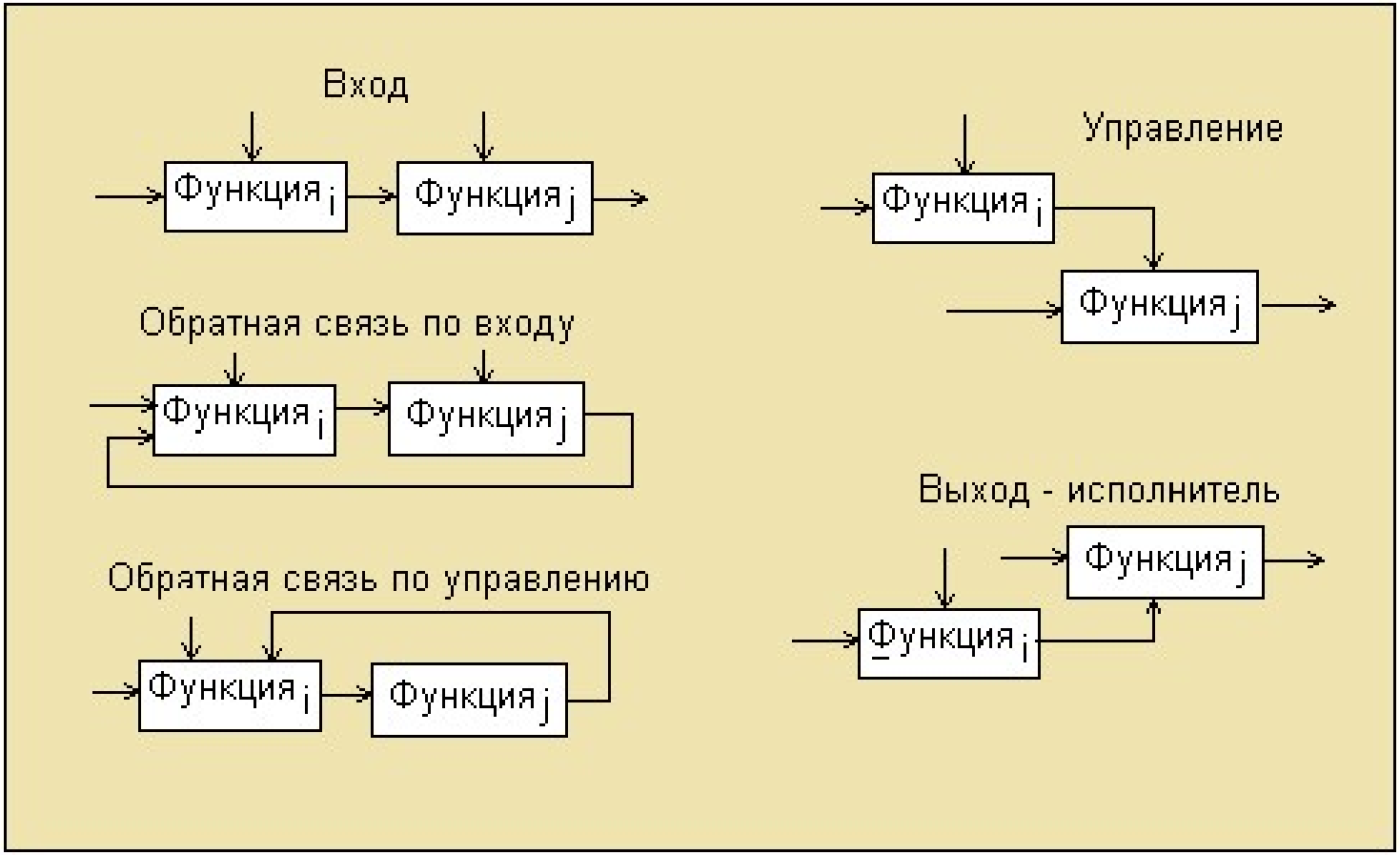


*механизмы используются при разработке сложных систем.

Функциональные диаграммы SADT

(Structured Analysis and Design Technique— методология структурного анализа и проектирования)

Различают пять типов влияний блоков друг на друга:



Правила (рекомендации):

- Построение модели начинают **с единственного блока** самого высокого уровня (затем он детализируется).

- Рекомендуется каждую функцию представлять **не более чем** 3–7-ю блоками.

- Блоки размещают по «ступенчатой» схеме в соответствии с последовательностью их работы или *доминированием*.

- ***Каждая подфункция может использовать или продуцировать только те элементы данных, которые использованы или продуцируются родительской функцией.***

- Стрелки нумеруют, используя символы и числа. Символ обозначает тип связи:
  I – входные; C – управляющие; M – механизмы; R – результаты.
  Число – номер связи по соответствующей стороне родительского блока, считая сверху вниз и слева направо.

- Правила нумерации:
  первый уровень – A0,
  следующий – A1, A2 и т.п.,
  следующий – A11, A12, A13 и т.д.
  (где первые цифры – номер родительского блока, а последняя – номер конкретного субблока).

- Детализацию завершают при получении функций, назначение которых хорошо понятно, как заказчику, так и разработчику.

- В процессе построения диаграмм составляют словарь данных, в котором определяют структуры и элементы данных.

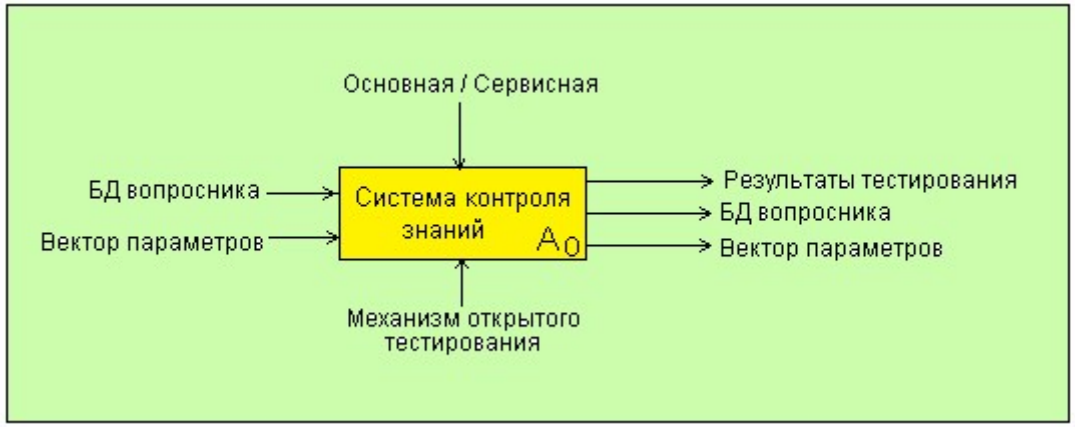
- Сами функции описывают с помощью естественного языка или псевдокодов.

В результате получают спецификацию, *которая состоит из:*

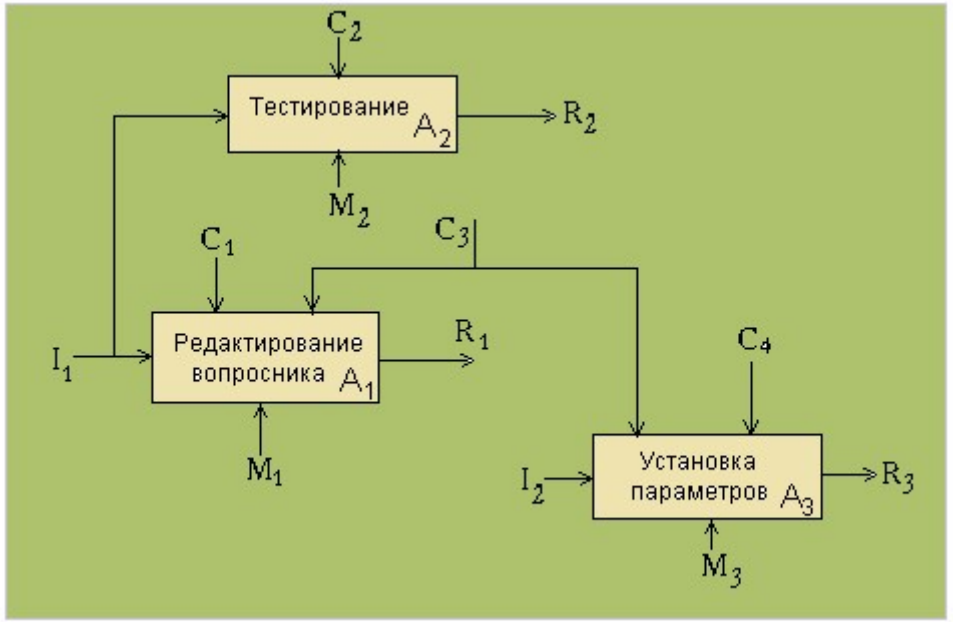
- ***иерархии функциональных диаграмм;***
- ***описаний функций нижнего уровня;***
- ***словаря***.

Пример функциональной диаграммы

(для системы контроля знаний)



а – диаграмма верхнего уровня



б – уточняющая диаграмма

Описание диаграммы:

I1 - БД вопросника
I2 - вектор параметров

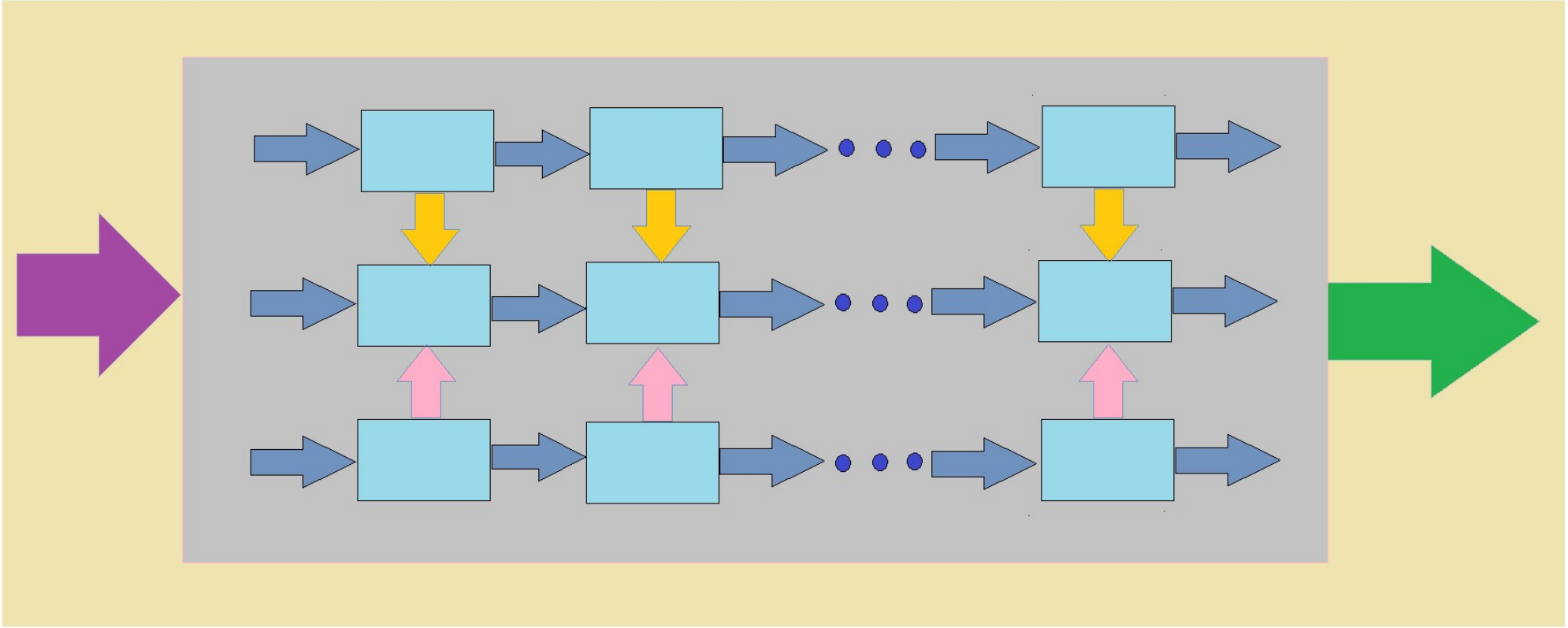
R1 - БД вопросника;
R2 - результаты тестирования
R3 - вектор параметров

C1– режим редактирования
C2 – режим тестирования
C3 – пароль
C4 – режим установки параметров

M1 - шаблон-таблица
M2 - случайный выбор N вопросов и M вариантов ответов
M3 - шаблон-вектор

Диаграммы потоков данных

Описывают асинхронный процесс преобразования информации от ввода в систему до выдачи пользователю.



- Уточнение процессов приводит к разработке определенного количества уровней.
- После достижения элементарных процессов уточнение завершается.

Основные понятия:

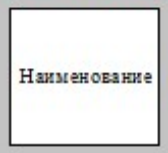
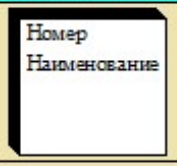
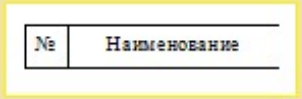
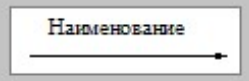
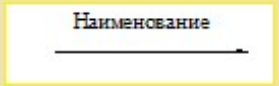
Внешняя сущность – объект, выступающий в системе в качестве источников или приемников информации (поставщики, клиенты, банк и т.п.).

Процесс – преобразование входных потоков данных в выходные в соответствии алгоритмом. На верхних уровнях иерархии используют понятия «система» и «подсистема».

Хранилище данных – абстрактное устройство для хранения информации (БД, файл, таблица в ОП, картотека на бумаге и т.п.).

Поток данных – процесс передачи некоторой информации от источника к приемнику.

Традиционно используют два вида нотаций:

Понятие	Нотация Йордана	Нотация Гейна-Сарсона
Внешняя сущность		
Система, подсистема или процесс		
Накопитель данных		
Поток		

Контекстная диаграмма

- ✓ Это диаграмма **верхнего уровня**.
- ✓ Показывает **взаимодействие** системы с приемниками и источниками информации (система при этом не детализируется)

Другими словами: описывает интерфейс системы с внешним миром.

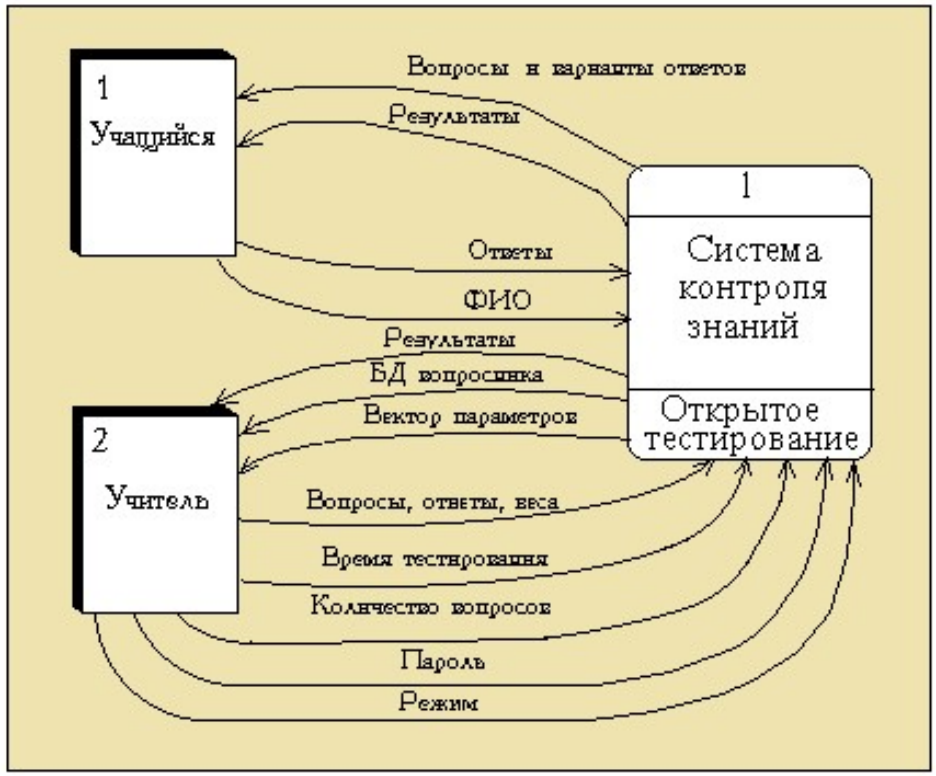


Рисунок - *Контекстная диаграмма системы СКЗ*

Правила и рекомендации:

- ✓ Если внешних сущностей более 10, то строят *иерархии* контекстных диаграмм.
- ✓ Каждую подсистему контекстной диаграммы детализируют при помощи диаграмм потоков данных.
- ✓ При детализации соблюдают правило балансировки (*детализируются компоненты, которые связаны потоками данных*).

Детализацию завершают, если:

- процесс взаимодействует с 2-3 потоками данных;
- возможно описание процесса последовательным алгоритмом;
- процесс выполняет единственную логическую функцию преобразования входной информации в выходную.

Далее:

- *на процессы* нижнего уровня составляют *спецификации;*

- спецификации должны содержать описание логики (функций) процесса;

- описание логики может выполняться:

  - ✓ на естественном языке;
  - ✓ с применением псевдокодов;
  - ✓ таблиц и деревьев решений;
  - ✓ в виде схем алгоритмов и др.

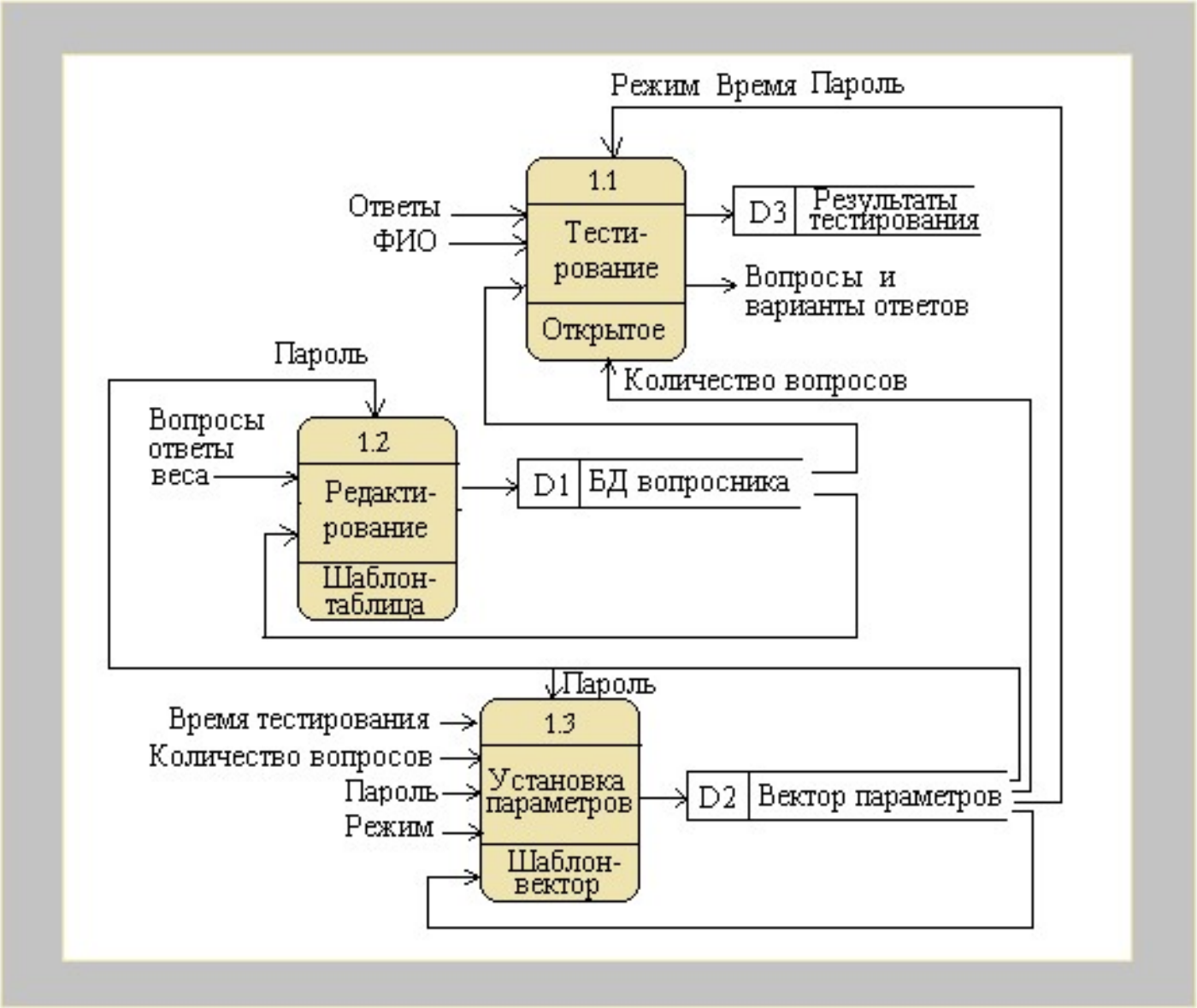
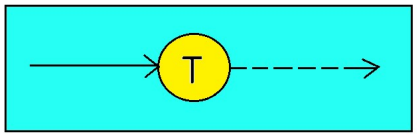


Рисунок - Детализирующая диаграмма потоков данных системы СКЗ

На диаграмме потоков данных можно:

- показать управляющие процессы (пунктирной линией);
- показать тип потока:
  - T- поток (Trigger Flow) – может **только «включать»** процесс;
  - A- поток (Activator Flow) – может как **«включать»**, так и **«выключать»** процесс (если процесс включен, то следующий сигнал его выключит);
  - E/D- поток (Enable/Disable Flow) – может включать процесс сигналом **по одной** (E) линии и выключать – сигналом **по другой** (D) линии.
- Изменять тип потока данных (управляющий или обычный)



В систему контроля знаний **может быть добавлен управляющий процесс-Меню**, от которого могут идти управляющие потоки, запускающие вышеописанные процессы.