

Лекция 4

«CASE-средства проектирования ИС»

Овчинников П.Е.
МГТУ «СТАНКИН»,
ст.преподаватель кафедры ИС

Терминология: инженерия

Инженерное дело (от [фр.](#) *ingénierie*; [син.](#) инженерия, инженерная деятельность, инженерно-техническая деятельность; инжиниринг от [англ.](#) *engineering* ← от [лат.](#) *ingenium* — «искусность» и [лат.](#) *ingeniare* — «изловчиться, разработать» — «изобретательность», «выдумка», «знания», «искусный») — область [технической деятельности](#), включающая в себя целый ряд специализированных областей и дисциплин, направленная на практическое приложение и применение [научных](#), экономических, социальных и практических знаний с целью обращения [природных ресурсов](#) на пользу [человека](#)

ГОСТ Р 57193-2016 Системная и программная инженерия. Процессы жизненного цикла систем

системная инженерия (systems engineering): междисциплинарный подход, управляющий полным техническим и организаторским усилием, требуемым для преобразования ряда потребностей заинтересованных сторон, ожиданий и ограничений в решение и для поддержки этого решения в течение его жизни

ISO/IEC/IEEE 24765:2017 Systems and software engineering -- Vocabulary

программная инженерия ([англ.](#) *software engineering*): приложение систематического, дисциплинированного, измеримого подхода к разработке, функционированию и сопровождению [программного обеспечения](#), а также исследованию этих подходов; то есть, приложение дисциплины [инженерии](#) к программному обеспечению

Терминология: компьютерный

computer aided – автоматизированный, компьютерный

Computer-aided design (CAD) is the use of [computer systems](#) (or [workstations](#)) to aid in the creation, modification, analysis, or optimization of a [design](#)

Computer-aided engineering (CAE) is the broad usage of [computer software](#) to aid in [engineering](#) analysis tasks. It includes [finite element analysis](#) (FEA), [computational fluid dynamics](#) (CFD), [multibody dynamics](#) (MBD), and [optimization](#)

Computer-aided manufacturing (CAM) is the use of software to control [machine tools](#) and related ones in the [manufacturing](#) of workpieces. CAM may also refer to the use of a computer to assist in all operations of a manufacturing plant, including planning, management, transportation and storage

Computer-aided process planning (CAPP) is the use of computer technology to aid in the process planning of a part or product, in manufacturing. CAPP is the link between CAD and CAM in that it provides for the planning of the process to be used in producing a designed part

Computer-aided software engineering (CASE) is the domain of software tools used to design and implement applications. CASE tools are similar to and were partly inspired by [computer-aided design](#) (CAD) tools used for designing hardware products. CASE tools are used for developing high-quality, defect-free, and maintainable software.

Терминология: CASE

CASE ([англ. computer-aided software engineering](#)) — набор инструментов и методов программной инженерии для проектирования программного обеспечения, который помогает обеспечить высокое качество программ, отсутствие ошибок и простоту в обслуживании программных продуктов. Также под CASE понимают совокупность методов и средств проектирования информационных систем с использованием CASE-инструментов

Средства автоматизации разработки программ (CASE-средства) — инструменты автоматизации процессов проектирования и [разработки программного обеспечения](#) для [системного анализа](#), разработчика ПО и [программиста](#).

Первоначально под CASE-средствами понимались только инструменты для упрощения наиболее трудоёмких процессов анализа и [проектирования](#), но с приходом стандарта ISO/IEC 14102 CASE-средства стали определять, как программные средства для поддержки процессов [жизненного цикла ПО](#)

Терминология: CASE

Основной целью CASE-технологии является разграничение процесса проектирования программных продуктов от процесса кодирования и последующих этапов разработки, максимально автоматизировать процесс разработки.

Для выполнения поставленной цели CASE-технологии используют два принципиально разных подхода к проектированию: **структурный** и **объектно-ориентированный**.

Структурный подход предполагает **декомпозицию** (разделение) поставленной задачи на функции, которые необходимо автоматизировать. В свою очередь, функции также разбиваются на подфункции, задачи, процедуры. В результате получается упорядоченная иерархия функций и передаваемой информацией между функциями.

Структурный подход подразумевает использование определенных общепринятых методологий при моделировании различных информационных систем:

[SADT](#) (structured analysis and design technique);

[DFD](#) (data flow diagrams);

[ERD](#) (entity-relationship diagrams).

Основным инструментом **объектно-ориентированного** подхода является язык [UML](#) — унифицированный язык моделирования, который предназначен для визуализации и документирования объектно-ориентированных систем с ориентацией их на разработку программного обеспечения.

Данный язык включает в себя систему различных диаграмм, на основании которых может быть построено представление о проектируемой системе.

Терминология: CASE

В функции CASE входят средства анализа, проектирования и программирования программных средств, проектирования интерфейсов, документирования и производства структурированного кода на каком-либо языке программирования.

Классификация по типам CASE-инструментов отражает функциональную ориентацию средств на те или иные процессы жизненного цикла разработки программного обеспечения:

- средства построения и анализа модели предметной области
- средства [проектирования баз данных](#)
- средства разработки приложений
- средства [реинжиниринга процессов](#)
- средства планирования и управления проектом
- средства тестирования
- средства документирования

Классификация по категориям CASE-инструментов определяет степень интегрированности (отдельные локальные средства, решающие небольшие автономные задачи, набор частично интегрированных средств, охватывающих большинство этапов жизненного цикла и полностью интегрированных средств, охватывающий весь жизненный цикл информационной системы и связанных общим репозиторием). Типичными CASE-инструментами являются:

- инструменты управления конфигурацией
- инструменты моделирования данных
- инструменты анализа и проектирования
- инструменты преобразования моделей
- инструменты редактирования программного кода
- инструменты [рефакторинга](#) кода
- генераторы кода

Терминология: CASE

ГОСТ Р ИСО/МЭК 14764-2002. Информационная технология. Сопровождение программных средств

Процесс сопровождения необходим вследствие подверженности программных продуктов изменениям на протяжении их жизненного цикла. Если программный продукт разработан с использованием инструментальных средств автоматизации программной инженерии (CASE), его сопровождение все равно необходимо. Использование инструментальных средств CASE упрощает сопровождение, но не устраняет потребность в нем.

Потенциальными средствами, определяющими стоимость сопровождения программных средств, являются инструментальные CASE-средства. Данный инструментарий обеспечивает проведение работ по сопровождению. CASE представляют собой взаимосвязанный набор инструментальных средств, обеспечивающих все аспекты разработки и сопровождения программных средств (ИСО/МЭК ТО 14471). Взаимосвязанный набор CASE-средств должен быть скомпонован в виде среды программной инженерии (СПИ), представляющей собой методы, политики, руководства и стандарты, обеспечивающие проведение работ по сопровождению программных средств.

ISO ISO/IEC 14102-2008 (prev.:1995) Information technology - Guideline for the evaluation and selection of CASE tools - Second Edition

ISO/IEC TR 14471:2007 (prev.:1999) Information technology -- Software engineering -- Guidelines for the adoption of CASE tools

Терминология: проектирование

Проектирование:

процесс определения [архитектуры](#), [компонентов](#), [интерфейсов](#) и других характеристик [системы](#) или её части (ISO 24765). Результатом проектирования является **прое́кт** — целостная совокупность [моделей](#), свойств или характеристик, описанных в форме, пригодной для реализации системы

Обратная разработка (обратное проектирование, обратный инжиниринг, реверс-инжиниринг; [англ.](#) *reverse engineering*):

исследование некоторого готового устройства или программы, а также [документации](#) на него с целью понять принцип его работы; например, чтобы обнаружить [недокументированные возможности](#) (в том числе [программные закладки](#)), сделать изменение или воспроизвести устройство, программу или иной объект с аналогичными функциями, но без прямого копирования.

ISO/IEC/IEEE 24765:2017 Systems and software engineering – Vocabulary design

1. the process of defining the architecture, components, interfaces, and other characteristics of a system or component.
2. the result of the process in (1).
3. the process of defining the software architecture, components, modules, interfaces, and data for a software system to satisfy specified requirements
4. the process of conceiving, inventing, or contriving a scheme for turning a computer program specification into an operational program
5. activity that links requirements analysis to coding and debugging
6. stage of documentation development that is concerned with determining what documentation will be provided in a product and what the nature of the documentation will be.

Объект проектирования: система

Р 50.1.028-2001 Информационные технологии поддержки жизненного цикла продукции. Методология функционального моделирования

4 Концепция IDEF0

Методология IDEF0 основана на следующих концептуальных положениях.

4.1 Модель - искусственный объект, представляющий собой отображение (образ) системы и ее компонентов.

Считается, что **М моделирует А**, если **М отвечает на вопросы относительно А**.

Здесь **М** - модель, **А** - моделируемый объект (оригинал).

Модель разрабатывают для понимания, анализа и принятия решений о реконструкции (реинжиниринге) или замене существующей, либо проектировании новой системы.

Система представляет собой совокупность взаимосвязанных и взаимодействующих частей, выполняющих некоторую полезную работу.

Частями (элементами) системы могут быть любые комбинации разнообразных сущностей, включающие людей, информацию, программное обеспечение, оборудование, изделия, сырье или энергию (энергоносители).

Модель описывает, что происходит в системе, как ею управляют, что она преобразует, какие средства использует для выполнения своих функций и что производит.

ООП: Инженерия знаний

ГОСТ 7.0-99 СИБИД. Информационно-библиотечная деятельность, библиография. Термины и определения

Информация - сведения, воспринимаемые человеком и (или) специальными устройствами как отражение фактов материального или духовного мира в процессе коммуникации

ГОСТ 34.321-96 Информационные технологии. Система стандартов по базам данных. Эталонная модель управления данными

Данные - информация, представленная в формализованном виде, пригодном для передачи, интерпретации или обработки с участием человека или автоматическими средствами

Знания - структурированная информация, связанная причинно-следственными отношениями и образующая системы. **Научные** знания могут быть:

- **эмпирическими** (на основе опыта или наблюдения)
- **теоретическими** (на основе анализа абстрактных моделей).

Научные знания в любом случае должны быть обоснованными на эмпирической или теоретической доказательной основе. Теоретические знания — абстракции, аналогии, схемы, отображающие структуру и природу процессов изменения объектов, протекающих в предметной области. Эти знания объясняют явления и могут использоваться для [прогнозирования поведения объектов](#).

ООП: Инженерия знаний

Фрейм (англ. frame — «каркас» или «рамка») — способ представления знаний в искусственном интеллекте, представляющий собой схему действий в реальной ситуации

Первоначально термин «фрейм» ввёл Марвин Минский в 70-е годы XX века для обозначения структуры знаний для восприятия пространственных сцен.

Фрейм — это модель абстрактного образа, минимально возможное описание сущности какого-либо объекта, явления, события, ситуации, процесса.

Фреймы используются в системах искусственного интеллекта (например, в экспертных системах — компьютерных системах, способных заменить экспертов в конкретной предметной области) как одна из распространенных форм представления знаний.

О системах программирования, основанных на фреймах, говорят, что они являются объектно-ориентированными. Каждый фрейм соответствует некоторому объекту предметной области, а слоты содержат описывающие этот объект данные, то есть в слотах находятся значения признаков объектов. Фрейм может быть представлен в виде списка свойств, а если использовать средства базы данных, то в виде записи.

Инженерия знаний

Пример: структура фрейма «Солнечная система»

Имя слота	Значение слота	Фасет
Системы.Наименование	Солнечная	
Системы(Солнечная).Элементы.Планеты.Наименование	Земля	
Системы(Солнечная).Элементы.Планеты(Земля).Масса	5,98·10 ²⁴ кг	0.01 Mc < m < 0.08 Mc (Mc –масса Солнца)
Системы(Солнечная).Элементы.Планеты(Земля).ОрбитаМакс	152 098 238 км	
Системы(Солнечная).Элементы.Планеты(Земля).ОрбитаМин	147 098 290 км	
Системы(Солнечная).Взаимосвязи.Наименование	Законы Кеплера	
Система(Солнечная).Гелиопауза	100 а.е. от Солнца (1а.е. = 149,6 млн. км.)	
Системы(Солнечная).Граница	Система (Солнечная). Гелиопауза	
Системы(Солнечная).Цель	Не установлена	

Фрейм

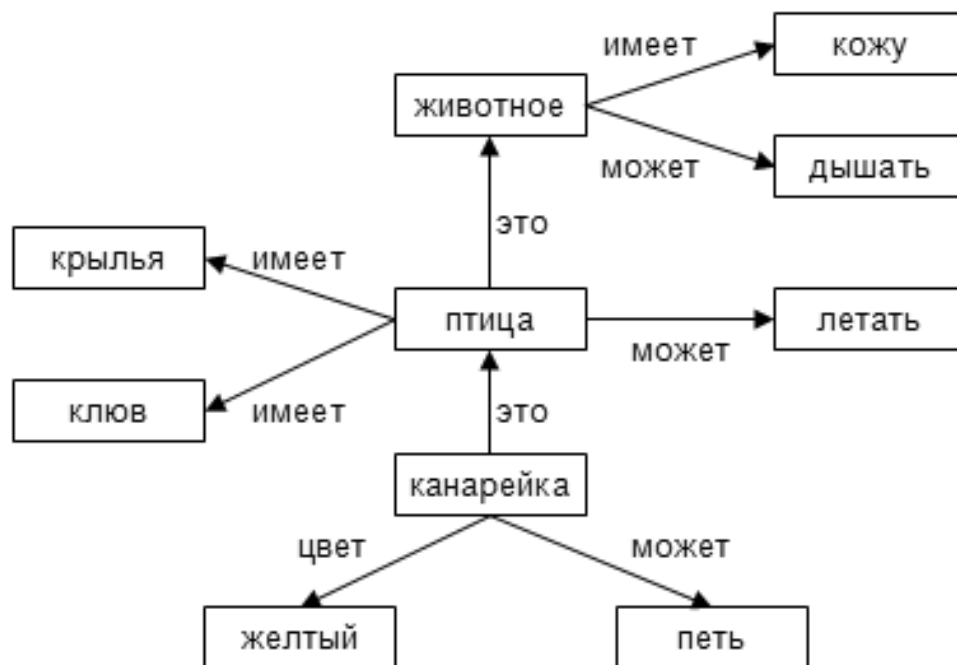
Минский М. Фреймы для представления знаний. М.: Мир, 1979.

Инженерия знаний

Семантическая сеть — информационная модель предметной области, имеющая вид ориентированного графа, вершины которого соответствуют объектам предметной области, а дуги (рёбра) задают отношения между ними. Объектами могут быть понятия, события, свойства, процессы. Таким образом, семантическая сеть является одним из способов представления знаний.

Семантическую сеть может образовывать система связанных фреймов, при этом различают:

- фреймы-образцы
- фреймы-экземпляры
- фреймы-структуры
- фреймы-роли
- фреймы-сценарии
- фреймы-ситуации



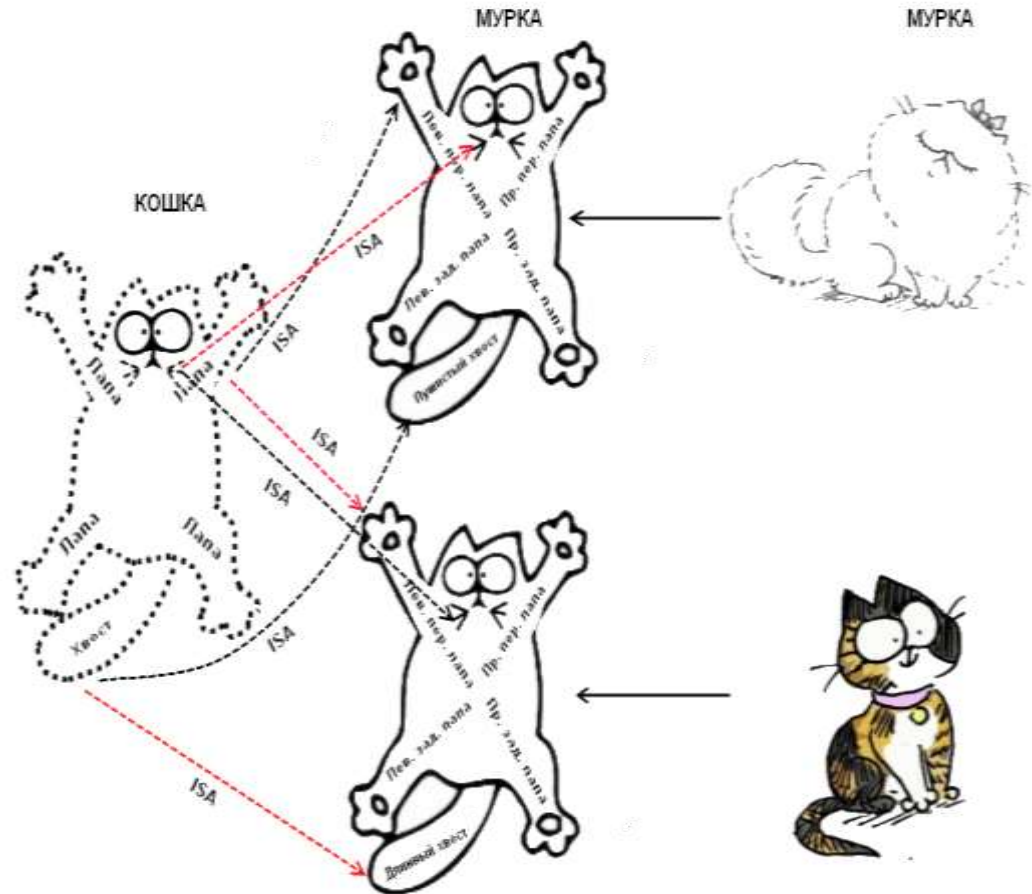
Семантическая сеть

Минский М. Фреймы для представления знаний. М.: Мир, 1979.

Инженерия знаний

Базовыми видами отношений для представления фреймов в виде семантической сети являются:

- отношение **АКО** (англ. a kind of), являющееся отношением классификации (общее-частное) и позволяющее строить иерархические связи между объектами, реализующие основные принципы наследования свойств объектов
- отношение **HasPart** (англ. has part), являющееся отношением вхождения (целое-часть) и позволяющее декомпозировать сложные объекты на их составляющие
- отношение **ISA** (англ. is a), являющееся отношением между фреймом-образцом и фреймом-экземпляром.

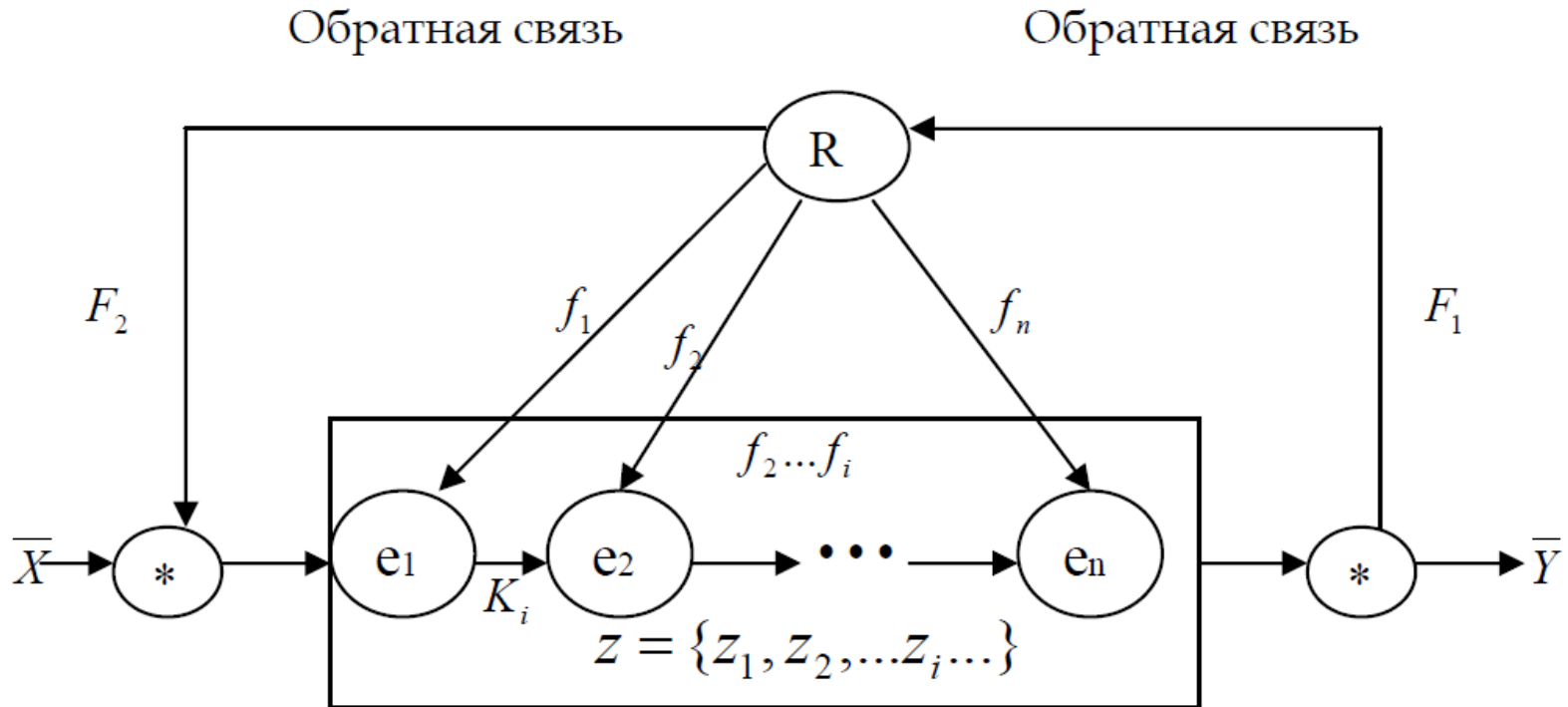


[Семантическая сеть](#)

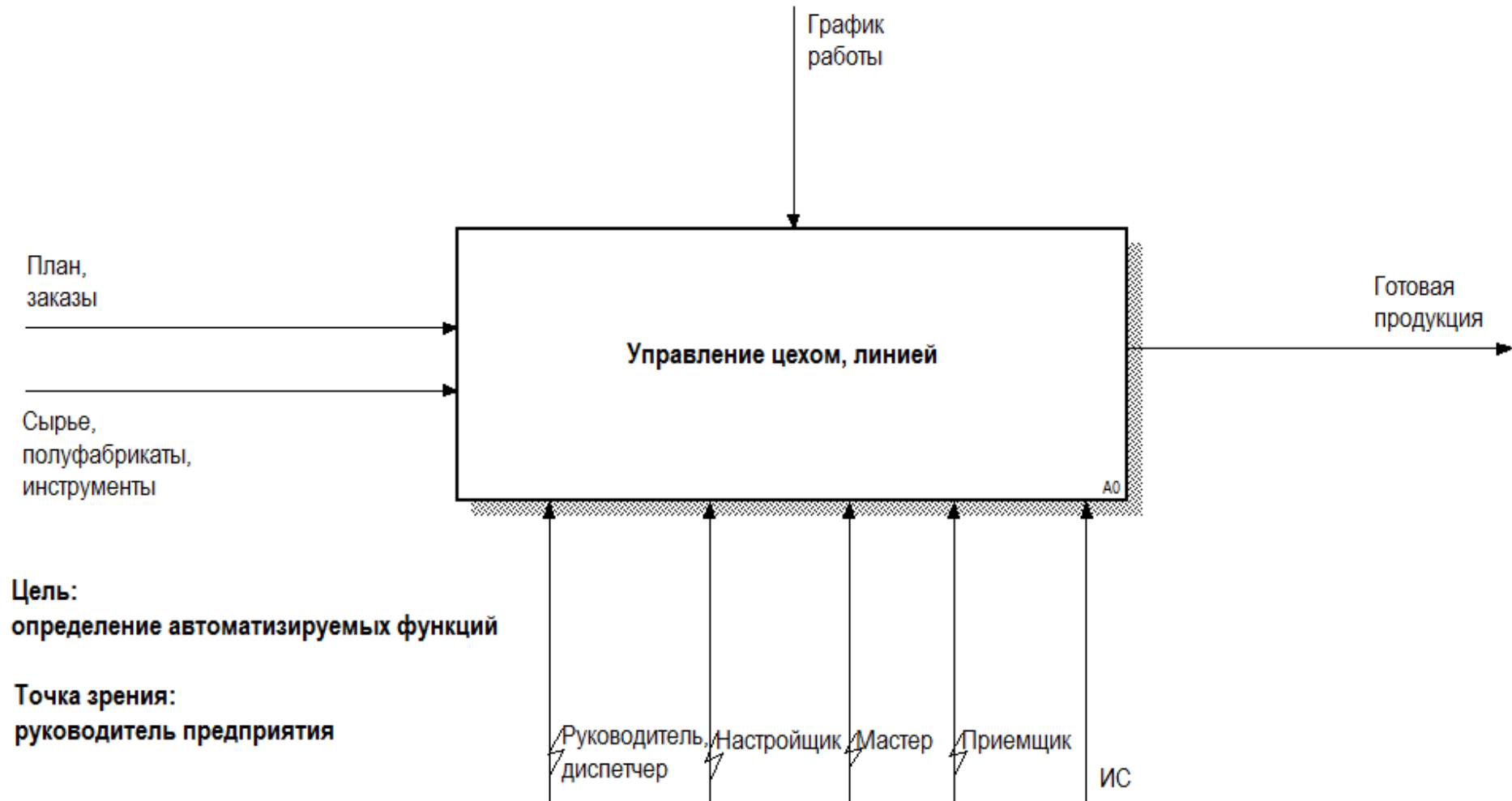
Минский М. Фреймы для представления знаний. М.: Мир, 1979.

УПРАВЛЕНИЕ В СИСТЕМАХ

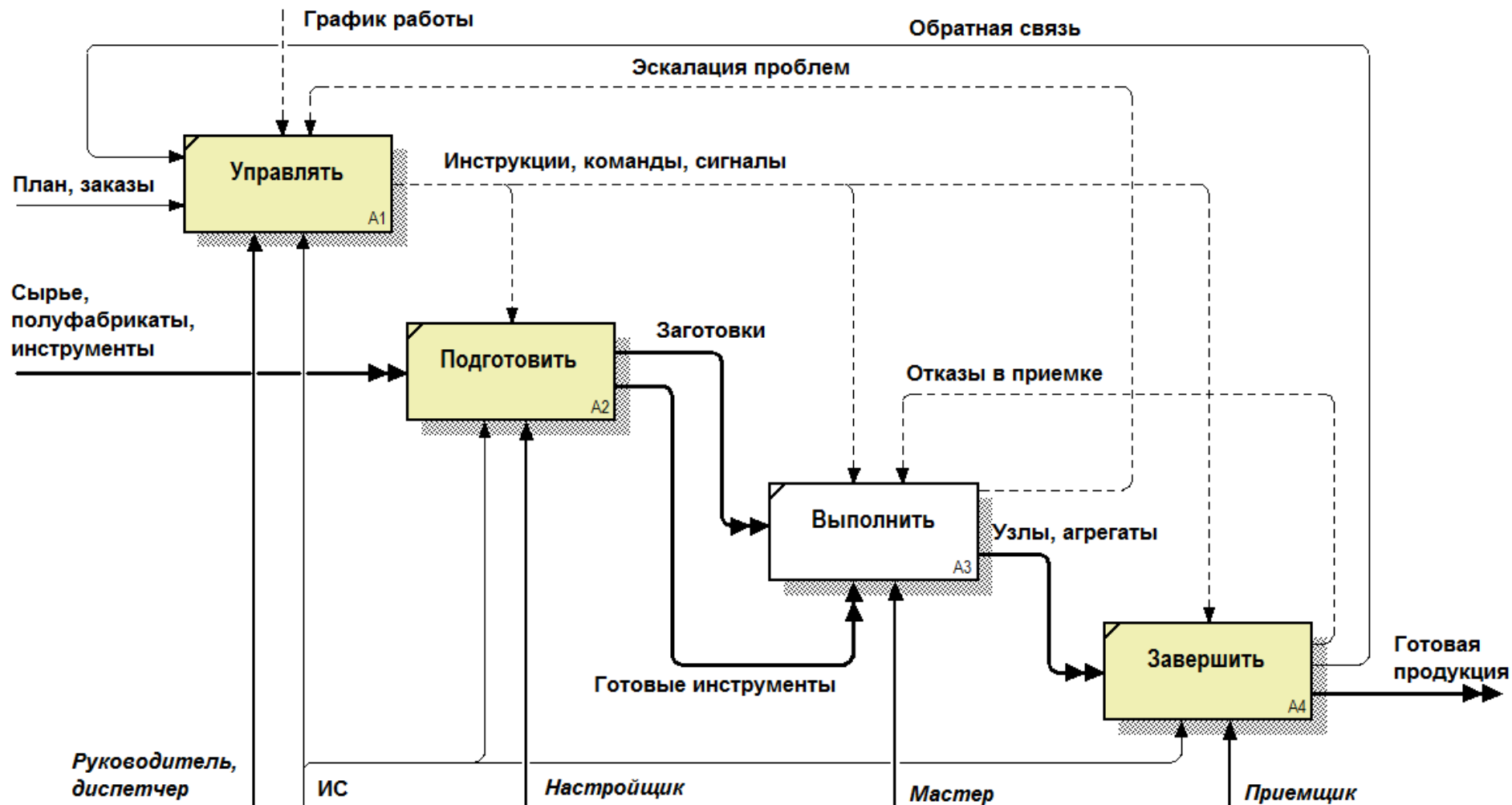
Система Σ – это конечная совокупность элементов (E) и некоторого регулирующего устройства (R), которое устанавливает связи между элементами (e_i) по преобразованию и управлению, управляет этими связями, создавая неделимую единицу функционирования. Топологически система представлена на рис. 1.



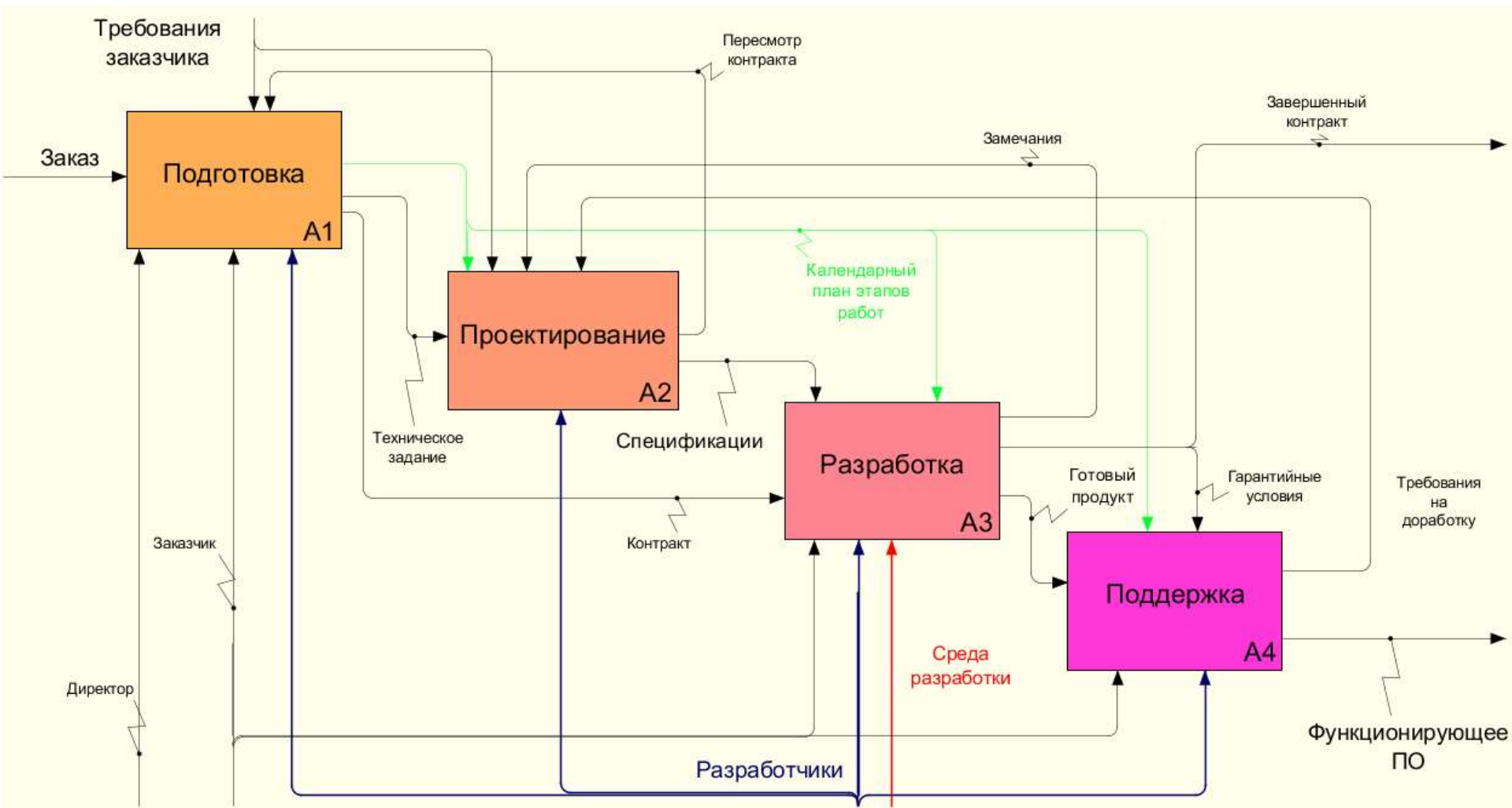
ЦЕЛЬ МОДЕЛИРОВАНИЯ И ТОЧКА ЗРЕНИЯ



Определение автоматизируемых функций



Модели «как есть» и «как будет»



Модели «как есть» и «как будет»

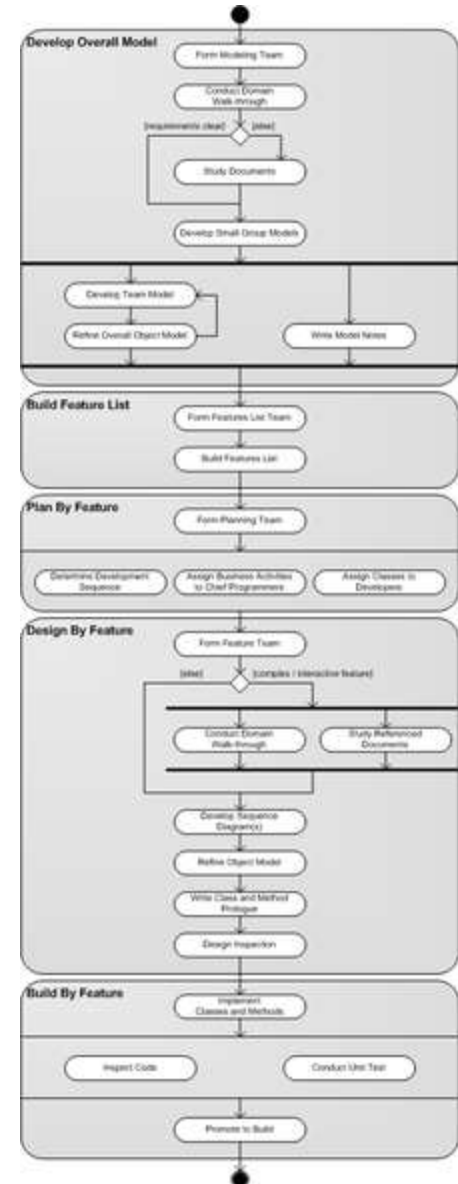
Feature driven development (FDD, разработка, управляемая функциональностью) — итеративная методология разработки программного обеспечения, одна из глубких методологий разработки (agile). FDD представляет собой попытку объединить наиболее признанные в индустрии разработки программного обеспечения методики, принимающие за основу важную для заказчика функциональность (свойства) разрабатываемого программного обеспечения. Основной целью данной методологии является разработка реального, работающего программного обеспечения систематически, в поставленные сроки.

FDD включает в себя пять базовых видов деятельности:

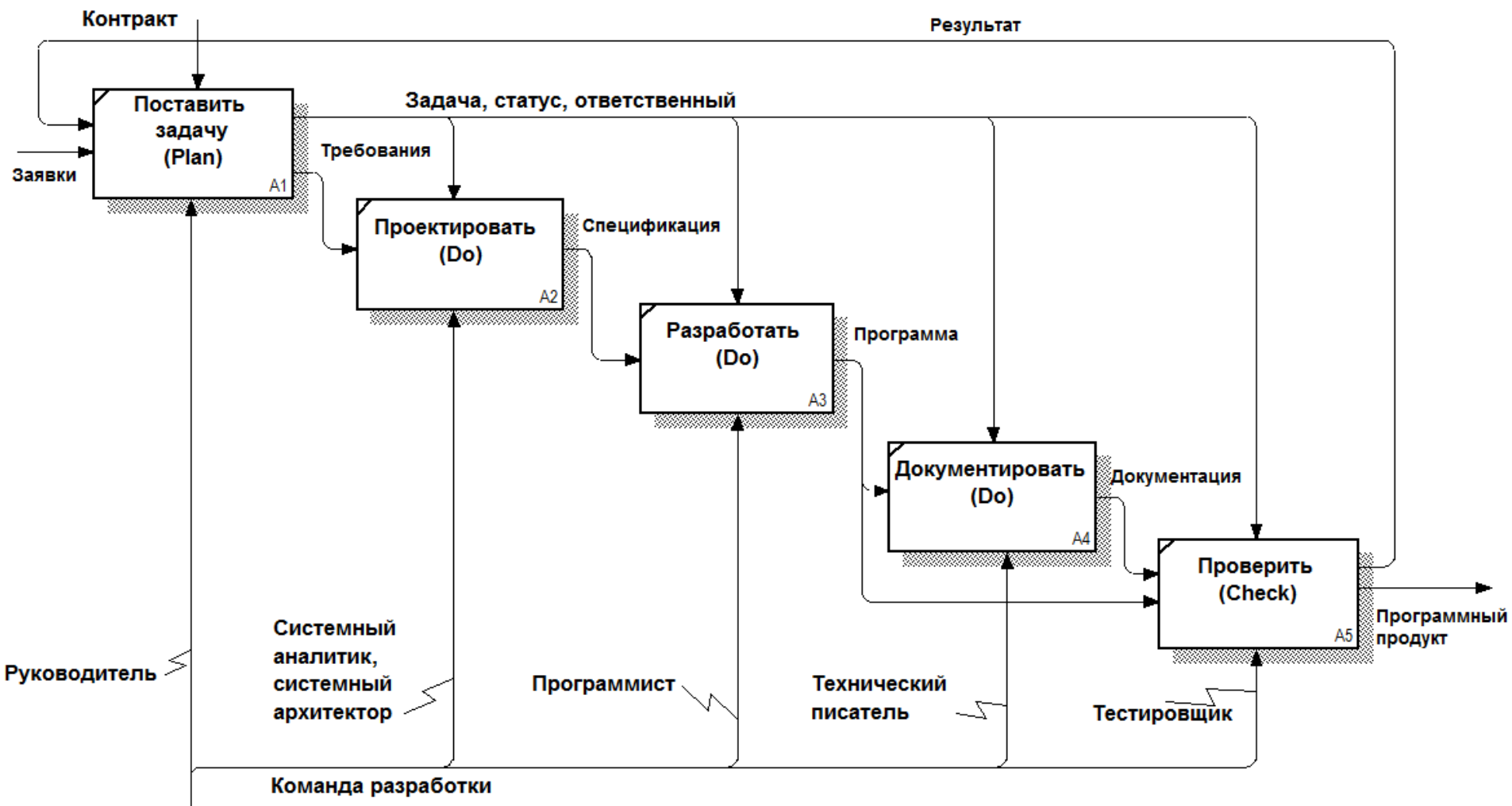
1. разработка общей модели
2. составление списка необходимых функций системы
3. планирование работы над каждой функцией
4. проектирование функции
5. реализация функции

FDD

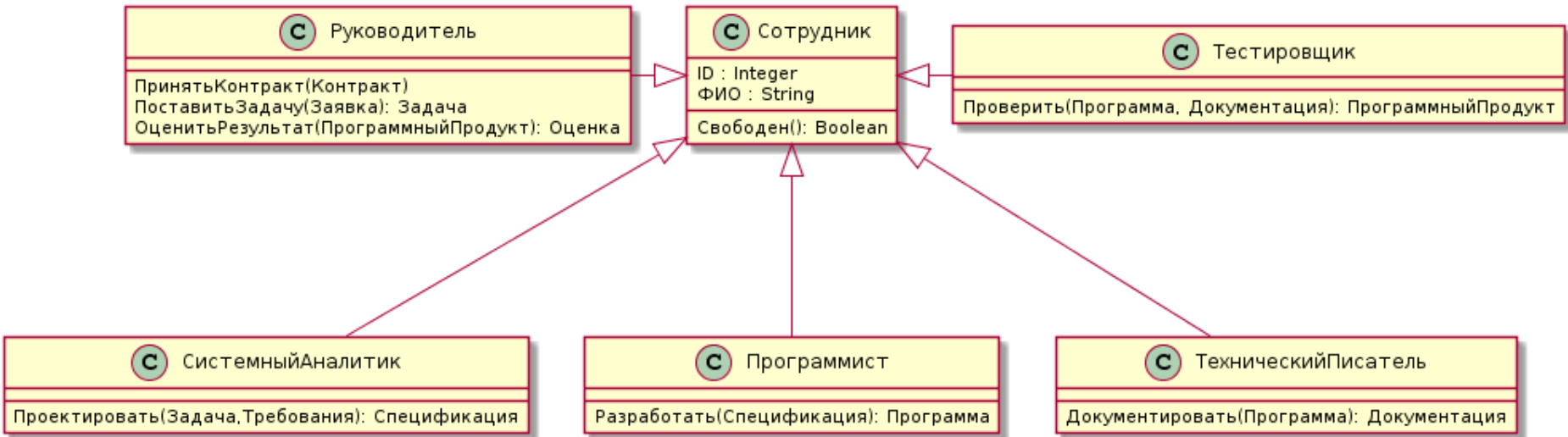
Анализ и оценка методов разработки программного обеспечения (Agile)
Анализ требований к автоматизированным информационным системам



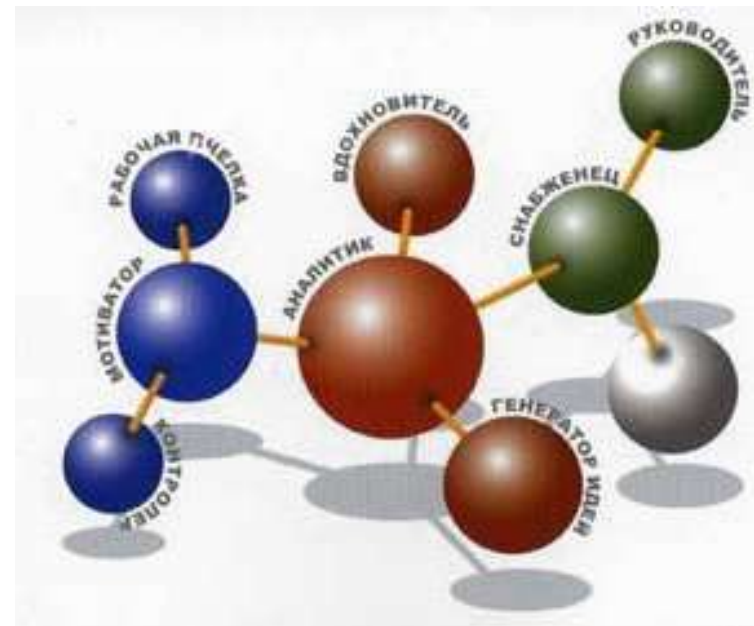
Цикл Деминга (PDCA)



Модель FDD и команда



Аналитик
Вдохновитель
Генератор идей
Контролер
Мотиватор
Рабочая пчелка
Руководитель



Формирование команды и управление проектом
Роли в команде (теория М.Белбина)

Модели «как есть» и «как будет»

Разработка через тестирование (*test-driven development*, **TDD**) — техника разработки программного обеспечения, которая основывается на повторении очень коротких циклов разработки: сначала пишется тест, покрывающий желаемое изменение, затем пишется код, который позволит пройти тест, и под конец проводится рефакторинг нового кода к соответствующим стандартам.

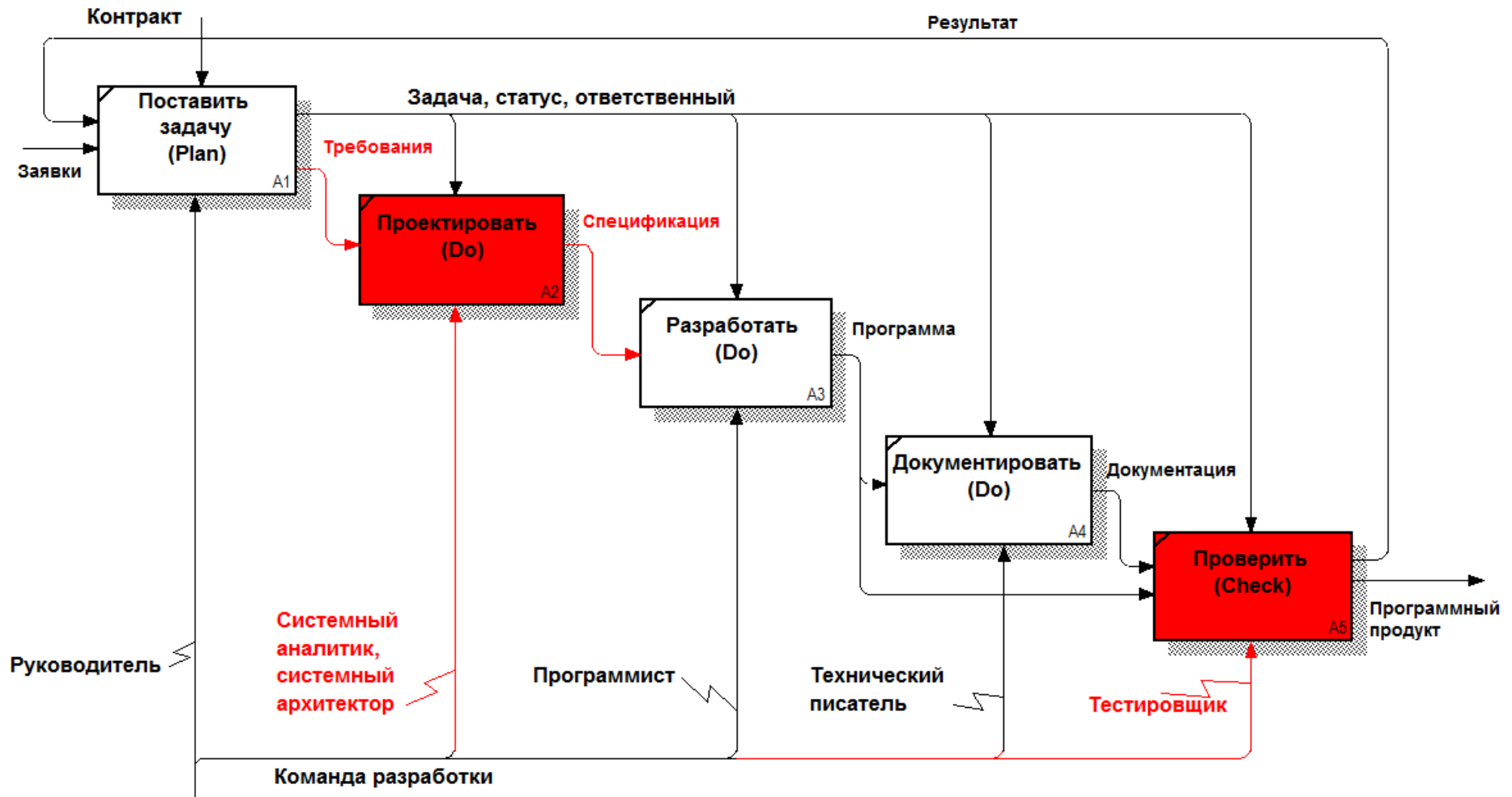
TDD цикл включает в себя пять основных шагов:

1. Быстро добавить тест
2. Выполнить все тесты и увидеть, что новый тест "падает"
3. Выполнить небольшое изменение системы
4. Убедиться, что все тесты проходят
5. Выполнить **рефакторинг**, удаляя дублирование

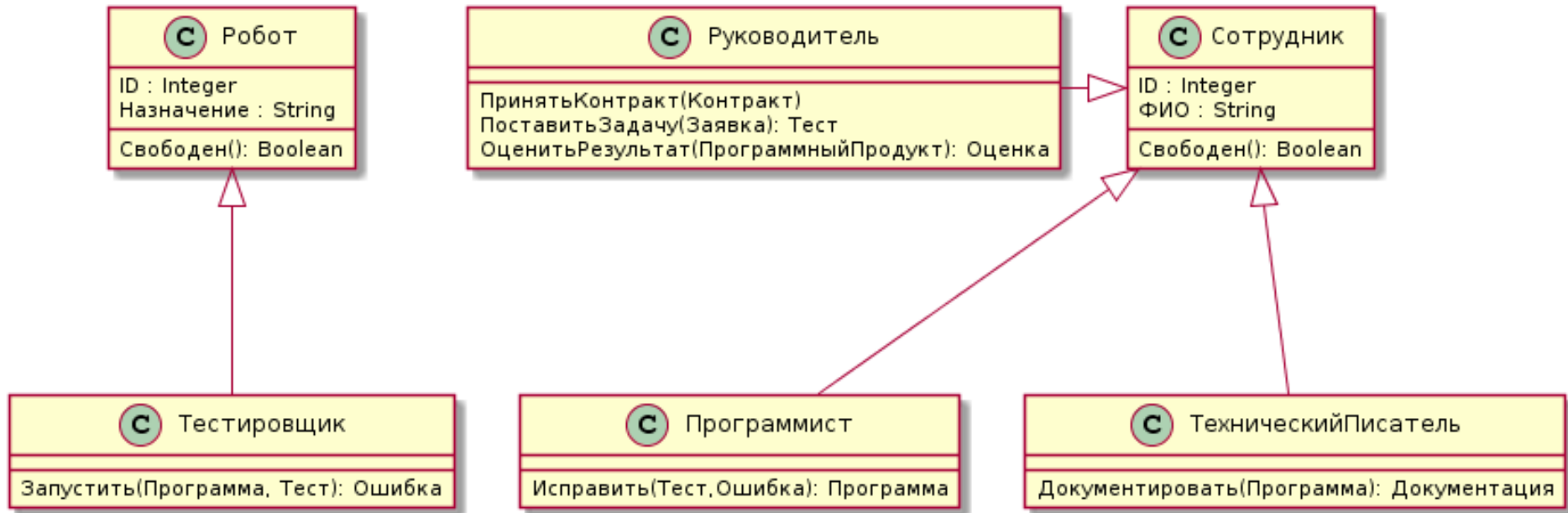
В модели TDD тест всегда пишется прежде чем создается соответствующий программный элемент. Если далее не выполнять шаги 2, 4, 5 то получится модель **TFD** (разработка "вначале тест«, *test first development*).

Рефакторинг — процесс изменения внутренней структуры программы, не затрагивающий её внешнего поведения и имеющий целью облегчить понимание её работы, устранить дублирование кода, облегчить внесение изменений в ближайшем будущем.

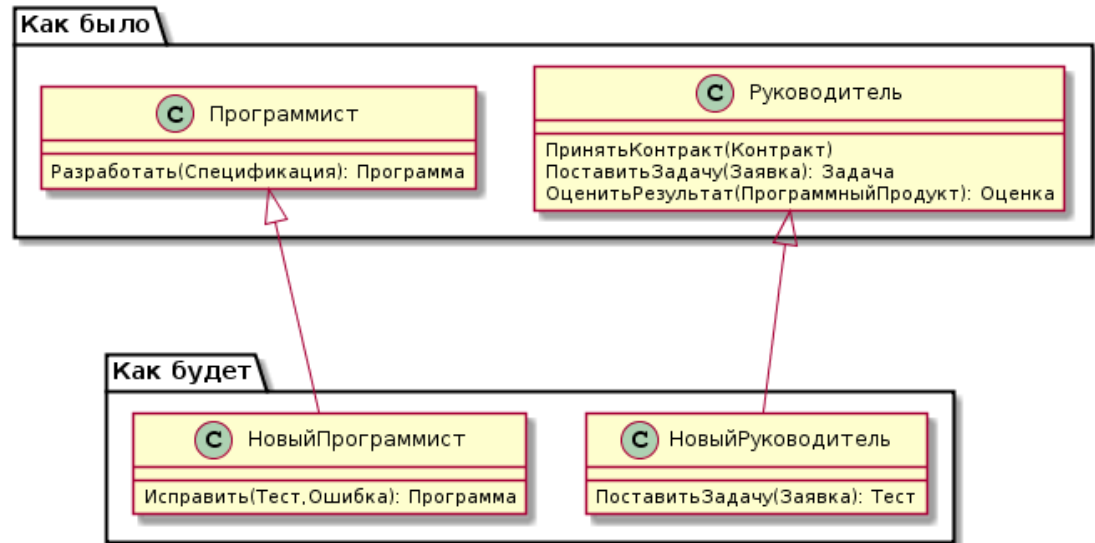
Модель TDD и команда



Модель TDD и команда



Путем сравнения моделей «как есть» (As Is) и «как должно быть» (To Be) составляется модель «что сделать» (To Do).



Формирование команды и управление
Роли в команде (теория М.Белбина)

Модели «как есть» и «как будет»

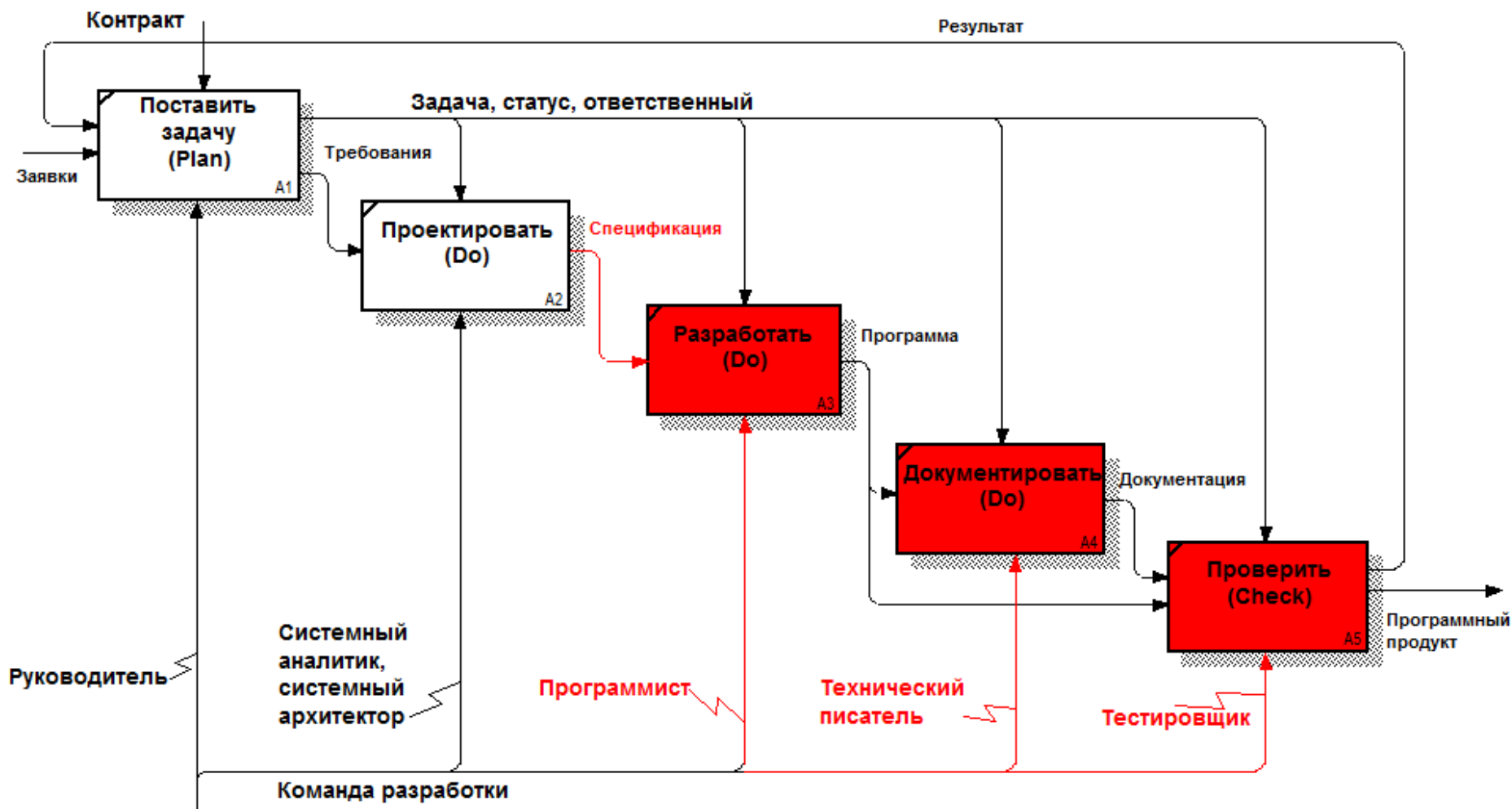
Разработка, управляемая моделями (*model-driven development*, **MDD**, *Model-driven engineering*, **MDE**) — это стиль разработки программного обеспечения, когда модели становятся основными артефактами разработки, из которых генерируется код и другие артефакты.

Модель — это абстрактное описание программного обеспечения, которое скрывает информацию о некоторых аспектах с целью представления упрощенного описания остальных. Модель может быть исходным артефактом в разработке, если она фиксирует информацию в форме, пригодной для интерпретаций людьми и обработки инструментальными средствами. Модель определяет **нотацию** и **метамоделю**. Нотация представляет собой совокупность графических элементов, которые применяются в модели и могут быть интерпретированы людьми. Метамоделю описывает используемые в модели понятия и фиксирует информацию в виде метаданных, которые могут быть обработаны инструментальными средствами.

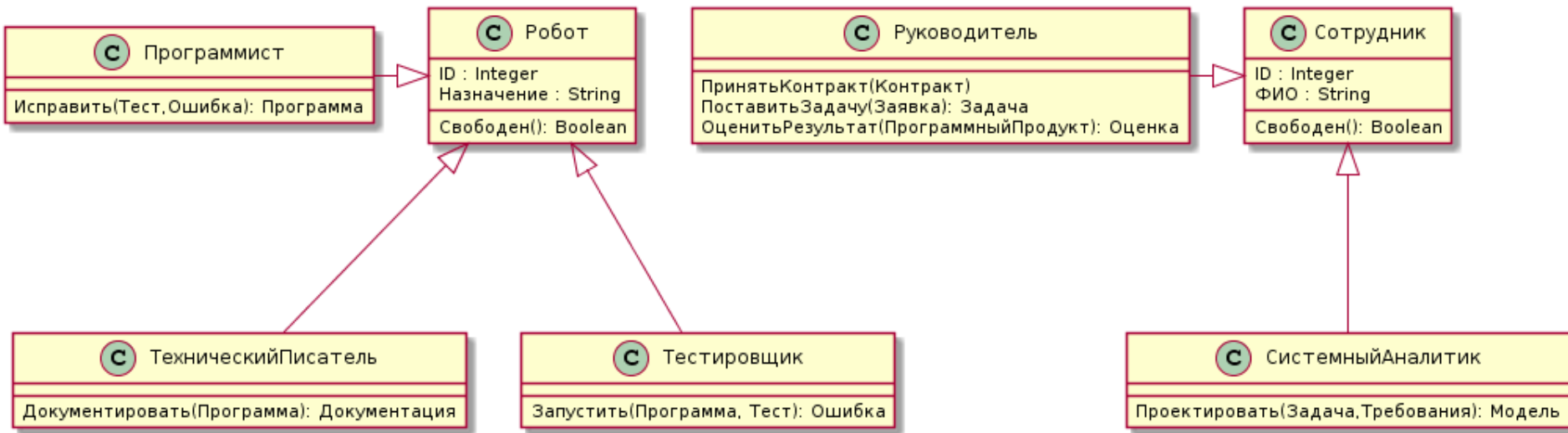
Наиболее известными современными MDE-инициативами являются:

1. разработка Object Management Group (OMG) под названием model-driven architecture (MDA)
2. экосистема Eclipse для инструментов моделирования и программирования (Eclipse Modeling Framework)

Модель MDD и команда



Модель MDD и команда



RAD (от [англ.](#) *rapid application development* — быстрая разработка приложений) — концепция создания средств разработки [программных продуктов](#), уделяющая особое внимание скорости и удобству [программирования](#), созданию технологического процесса, позволяющего программисту максимально быстро создавать [компьютерные программы](#). Практическое определение: RAD — это [жизненный цикл](#) процесса проектирования, созданный для достижения более высокой скорости разработки и качества ПО, чем это возможно при традиционном подходе к проектированию. С конца [XX века](#) RAD получила широкое распространение и одобрение. Концепцию RAD также часто связывают с концепцией [визуального программирования](#).

Объект проектирования: продукт



Internal Logical Files (ILF) – понятная для пользователя группа логически-связанных данных, которые находятся полностью внутри приложения и обслуживаются через внешние входы системы.

External Interface Files (EIF) – понятная для пользователя группа логически-связанных данных, которые используются только для целей ссылки. Сюда относятся так же данные, которые использует наше приложение, но которые управляются другим приложением.

External Inputs (EI) - Элементарный процесс, в котором данные вводятся в систему с наружи. Эти данные могут поступать от экранов ввода данных, электронных устройств или от другого приложения.

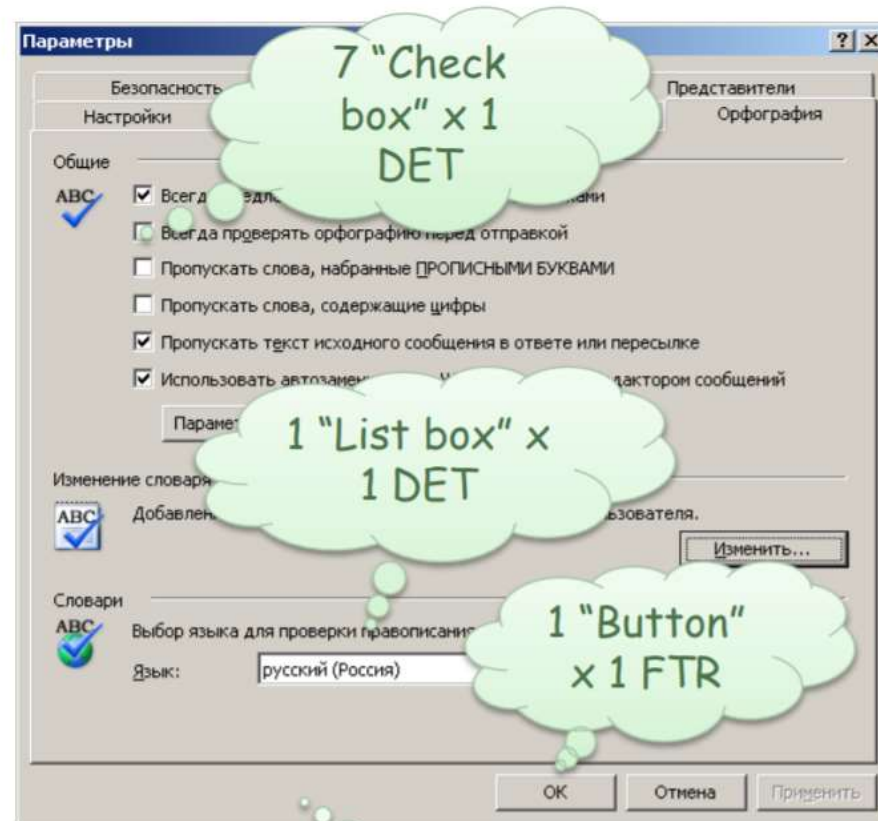
External Outputs (EO) - Элементарный процесс, в котором данные выводятся из системы. Данные создают отчеты или внешние файлы, которые пересылаются другим приложениям. Они могут создаваться одним или несколькими внутренними логическими файлами (ILF) и внешним файлом интерфейса (EIF).

External Inquiry (EQ) - Элементарный процесс, в котором данные выводятся из системы в результате выполнения комплексного запроса и процесса обработки внутренних логических файлов (ILF) и внешних интерфейсных файлов (EIF).

Объект проектирования: интерфейс

Сложность транзакций EI & EQ	Количество UFP
Low	3
Average	4
High	6

Сложность транзакций EO	Количество UFP
Low	4
Average	5
High	7



(1 FTR, 8 DET) -> Low -> 3 UFP

Объект проектирования: интерфейс

Интерфейс (англ. interface — сопряжение, поверхность раздела, перегородка) — совокупность возможностей, способов и методов взаимодействия двух систем, устройств или программ для обмена информацией между ними, определенная их характеристиками, характеристиками соединения сигналов обмена и т.п.

ГОСТ Р ИСО 9241-210-2016 Эргономика взаимодействия человек-система. Часть 210. Человеко-ориентированное проектирование интерактивных систем

человеко-ориентированное проектирование (human-centred design): Способ проектирования и разработки систем с применением при проектировании принципов эргономики для повышения пригодности использования интерактивных систем

интерактивная система (interactive system): Система компонентов аппаратного и программного обеспечения, которая получает информацию, вводимую пользователем, и передает ему свой ответ, помогая в работе или выполнении задачи.

пользовательский интерфейс (интерфейс пользователя) (user interface): Все компоненты интерактивной системы (программное обеспечение или аппаратное обеспечение), которые предоставляют пользователю информацию и являются инструментами управления для выполнения определенных задач

пригодность использования (usability): Свойство системы, продукции или услуги, при наличии которого установленный пользователь может применить продукцию в определенных условиях использования для достижения установленных целей с необходимой результативностью, эффективностью и удовлетворенностью.

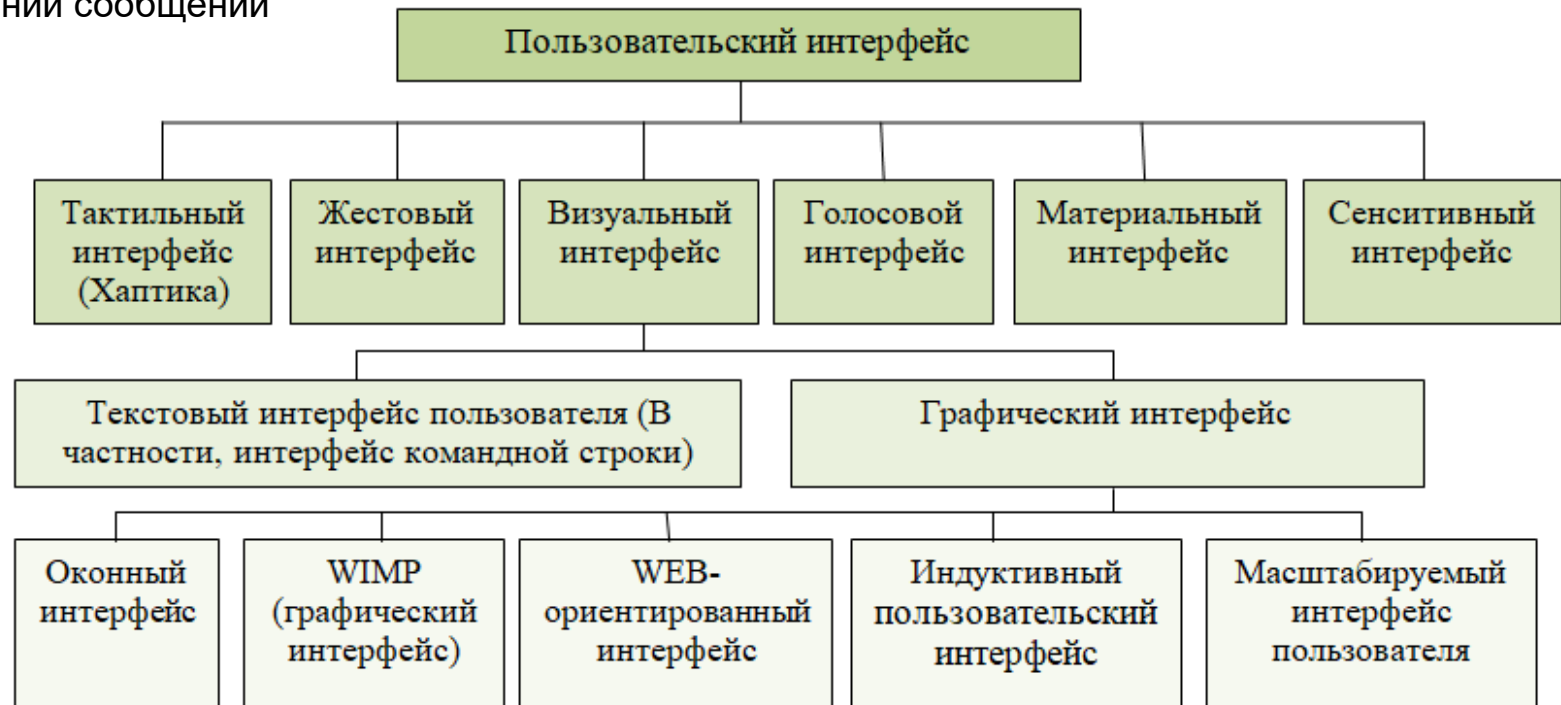
Объект проектирования: интерфейс

ГОСТ Р 43.0.2-2006 Информационное обеспечение техники и операторской деятельности. Термины и определения

оператор: Человек, занимающийся какой-либо деятельностью с использованием технических(ого) устройств(а)

ГОСТ Р 43.0.3-2009 Информационное обеспечение техники и операторской деятельности. Ноон-технология в технической деятельности. Общие положения

ноон-технология: Технология создания информации в виде, соответствующем психофизиологии человека (с использованием результатов исследований, полученных в ноонике), для реализации оптимизированных информационно-обменных процессов в СЧИ при создании, хранении, передаче, применении сообщений



Объект проектирования: интерфейс

Тактильный интерфейс

Средства отображения и позиционирования функционально объединены и пространственно совмещены. Тактильный интерфейс описывает управление посредством касания сенсорного экрана с изображением стилусом, пальцами, другими частями тела.



Объект проектирования: интерфейс

Жестовый интерфейс

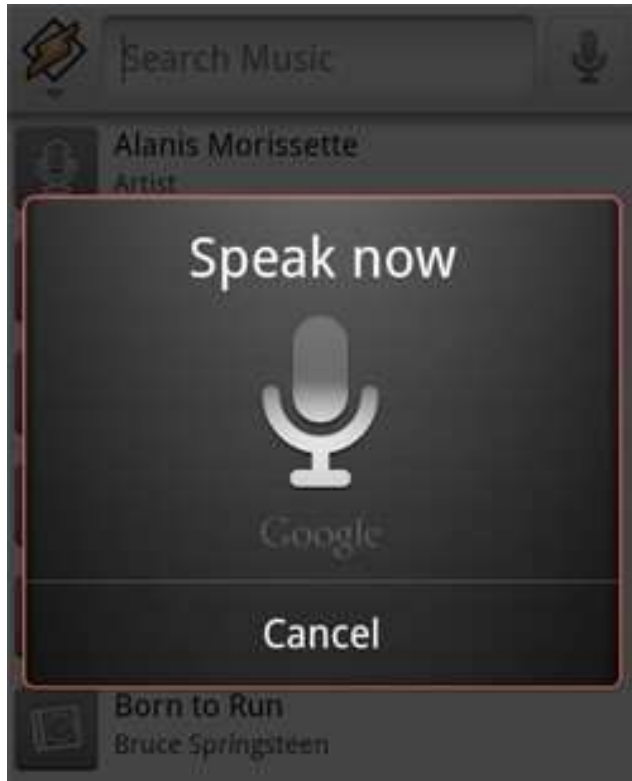
Позволяет эмулировать клавиатурные команды (либо сочетания клавиш, касания) при помощи жестов.



Объект проектирования: интерфейс

Голосовой интерфейс

Вместо меню используются слышимые инструкции, которые были ранее сохранены или созданы синтезатором голоса в реальном времени.



Объект проектирования: интерфейс

Материальный интерфейс

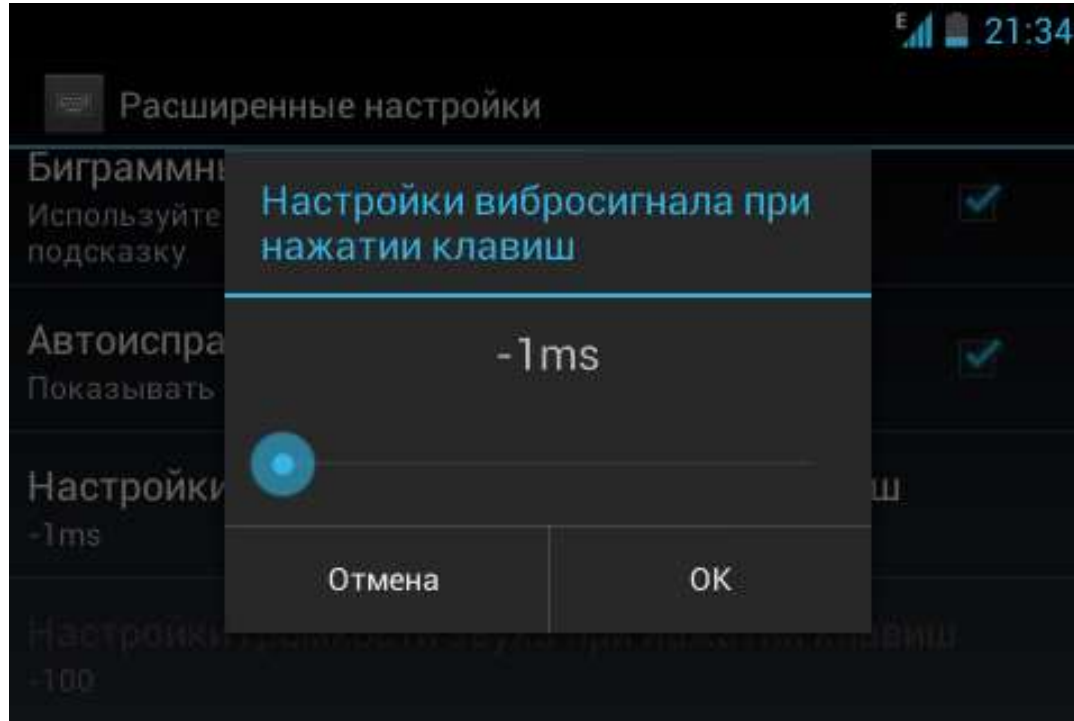
Взаимодействие человека с электронными устройствами происходит при помощи материальных предметов и конструкций.



Объект проектирования: интерфейс

Сенситивный интерфейс

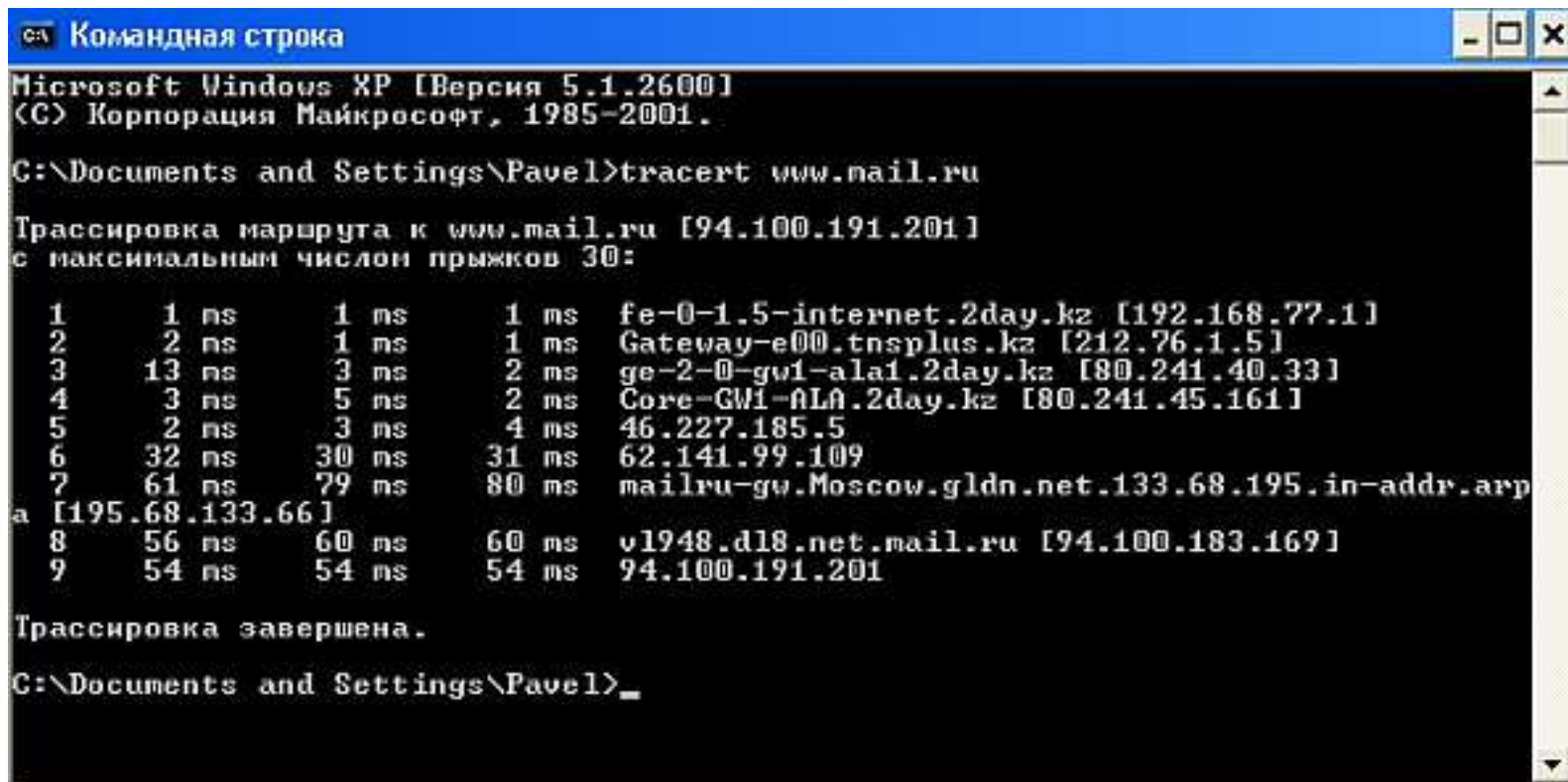
В ответ на взаимодействие человека с машиной происходит воздействие на органы чувств.



Объект проектирования: интерфейс

Текстовый интерфейс пользователя

Инструкции компьютеру даются путём ввода с клавиатуры текстовых строк (команд). Система как ответ на действия оператора тоже выдаёт или сообщения, или результат выполнения введенной команды, опять же в текстовом виде. Данный интерфейс также используется для взаимодействия с другими прикладными программами, включая удаленно расположенные.



```
C:\Documents and Settings\Pavel>tracert www.mail.ru

Трассировка маршрута к www.mail.ru [94.100.191.201]
с максимальным числом прыжков 30:

  1      1 ms      1 ms      1 ms  fe-0-1.5-internet.2day.kz [192.168.77.1]
  2      2 ms      1 ms      1 ms  Gateway-e00.tnsplus.kz [212.76.1.5]
  3     13 ms      3 ms      2 ms  ge-2-0-gw1-ala1.2day.kz [80.241.40.33]
  4      3 ms      5 ms      2 ms  Core-GW1-ALA.2day.kz [80.241.45.161]
  5      2 ms      3 ms      4 ms  46.227.185.5
  6     32 ms     30 ms     31 ms  62.141.99.109
  7     61 ms     79 ms     80 ms  mailru-gw.Moscow.gldn.net.133.68.195.in-addr.arp
a [195.68.133.66]
  8     56 ms     60 ms     60 ms  v1948.d18.net.mail.ru [94.100.183.169]
  9     54 ms     54 ms     54 ms  94.100.191.201

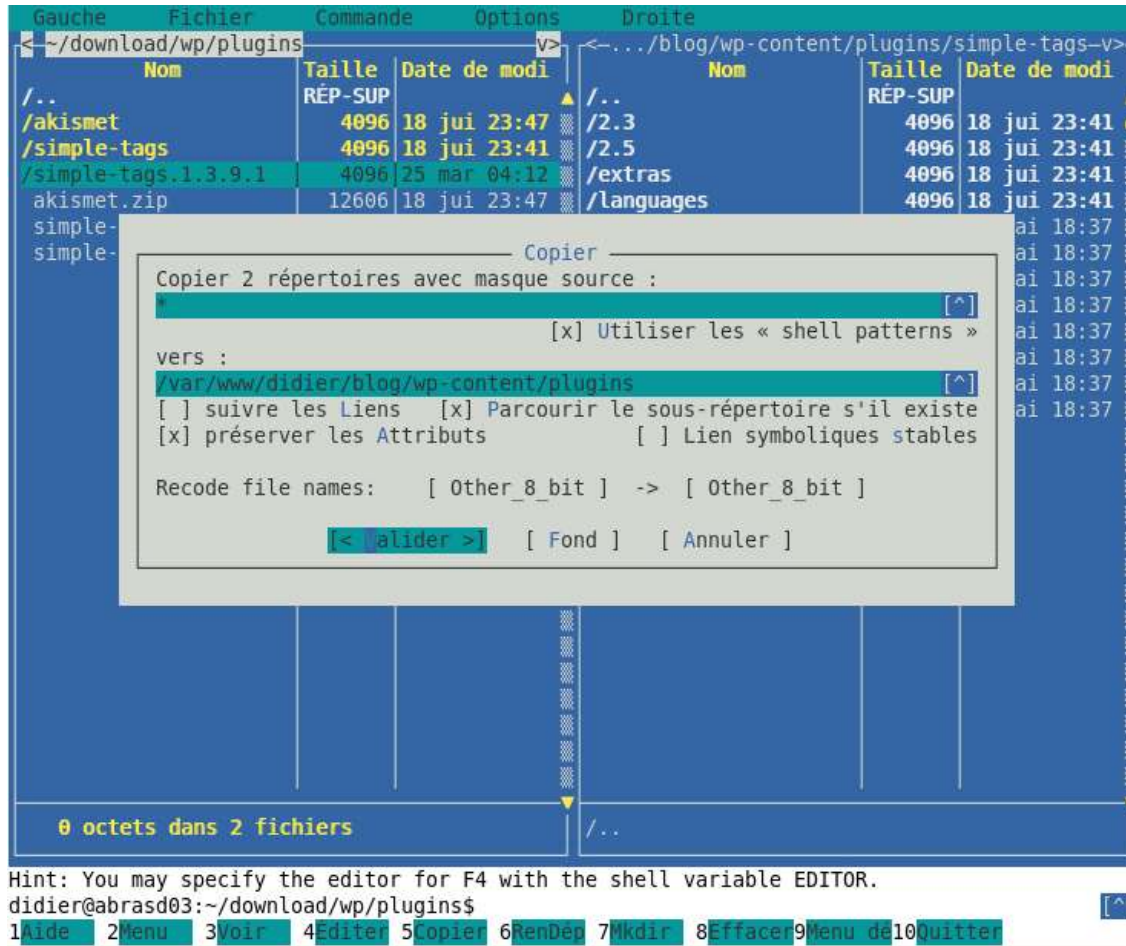
Трассировка завершена.

C:\Documents and Settings\Pavel>
```

Objet проектирования: интерфейс

Оконный интерфейс

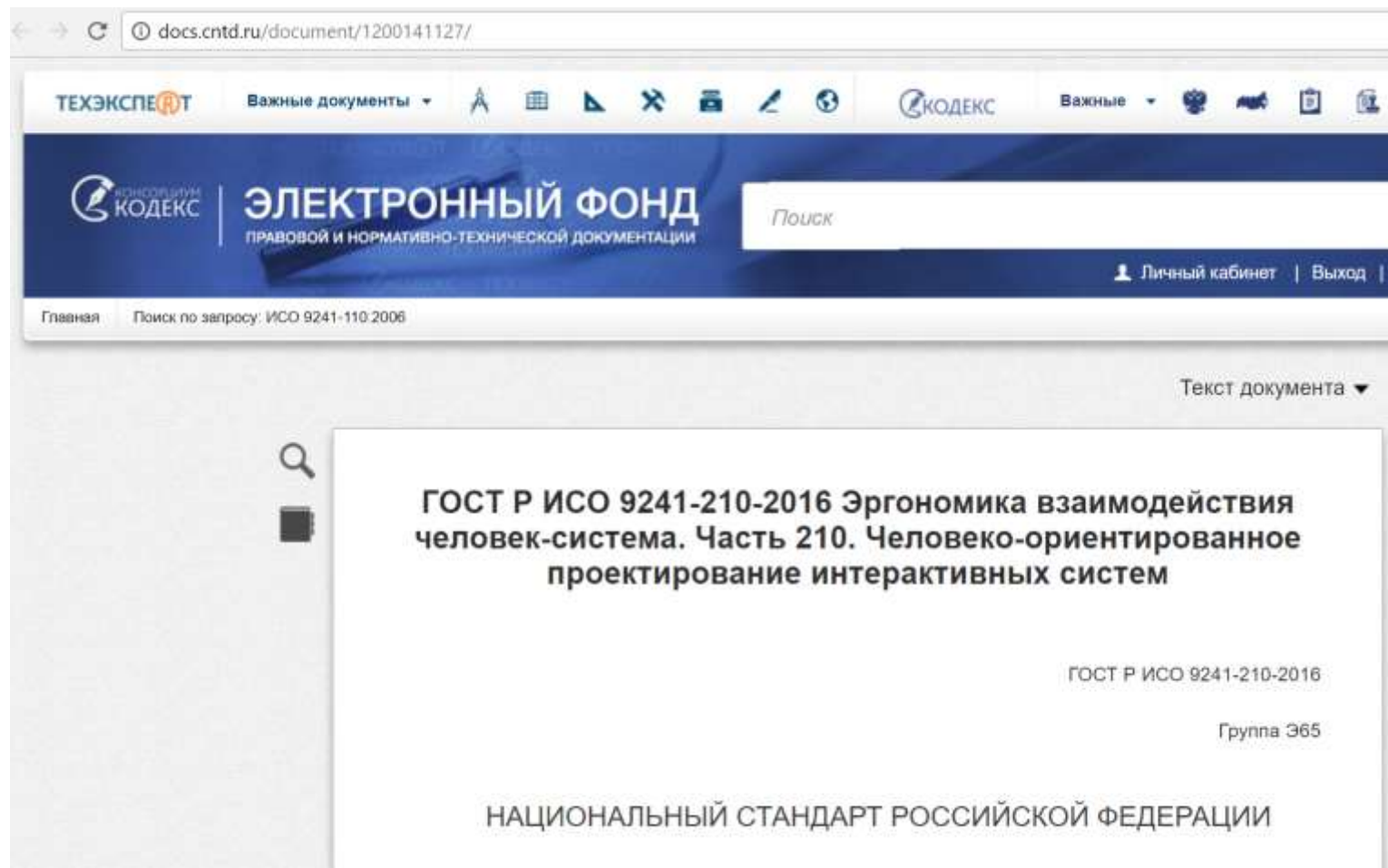
Каждая интегральная часть располагается в окне — собственном субэкранном пространстве, находящемся в произвольном месте «над» основным экраном.



Объект проектирования: интерфейс

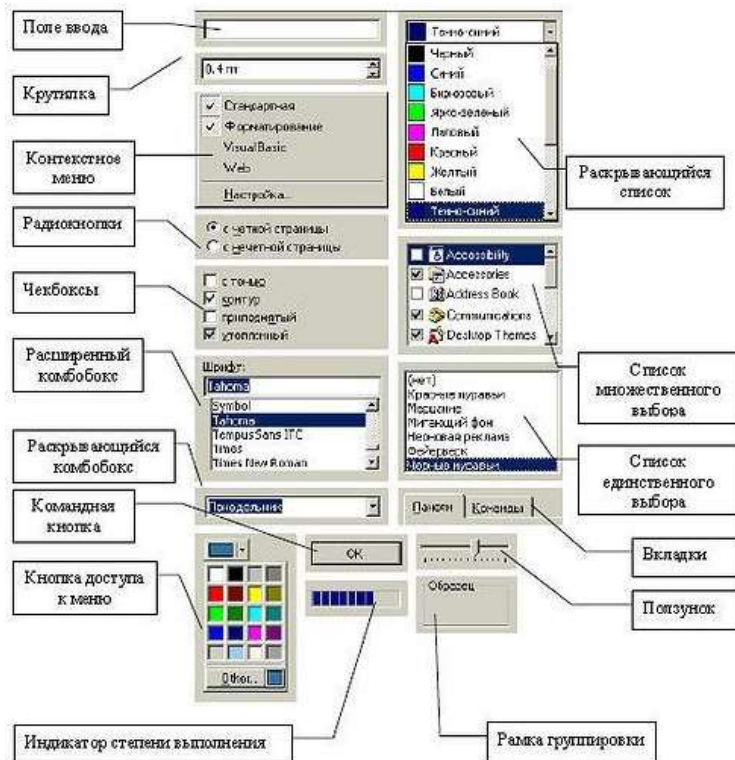
WEB-ориентированный интерфейс

Совокупность средств, при помощи которых пользователь взаимодействует с [веб-сайтом](#) или любым другим [приложением](#) через [браузер](#).



Объект проектирования: интерфейс

Окулография (отслеживание [глаз](#), [трекинг](#) глаз; [айтрекинг](#)) — определение [координат взгляда](#) («точки пересечения [оптической оси](#) [глазного яблока](#) и [плоскости](#) наблюдаемого объекта или экрана, на котором предъявляется некоторый зрительный раздражитель»)



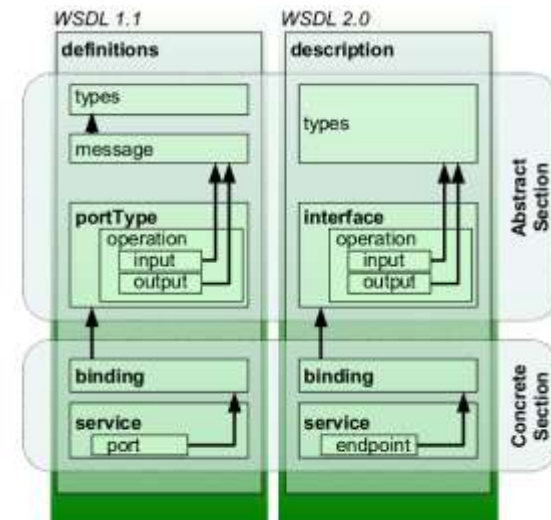
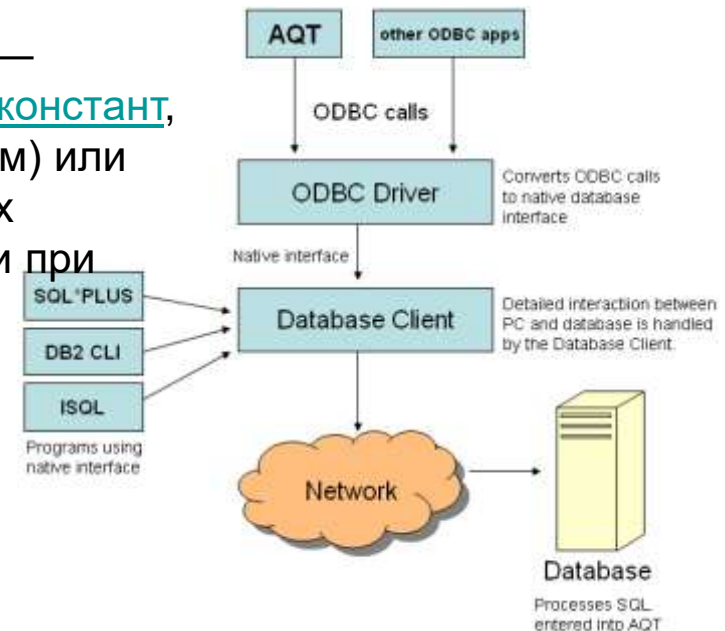
Объект проектирования: интерфейс

API (программный интерфейс приложения, интерфейс прикладного программирования)

([англ.](#) *application programming interface*, *API* [эй-пи-ай]) — набор готовых [классов](#), [процедур](#), [функций](#), [структур](#) и [констант](#), предоставляемых приложением (библиотекой, сервисом) или операционной системой для использования во внешних программных продуктах. Используется программистами при написании всевозможных приложений.

ODBC ([англ.](#) *Open Database Connectivity*) — это программный интерфейс ([API](#)) доступа к [базам данных](#), разработанный компанией [Microsoft](#) в сотрудничестве с [Simba Technologies](#) на основе спецификаций [Call Level Interface](#) (CLI), который разрабатывался организациями [SQL Access Group](#), [X/Open](#) и [Microsoft](#). Стандарт CLI призван унифицировать программное взаимодействие с [СУБД](#), сделать его независимым от поставщика СУБД и программно-аппаратной платформы

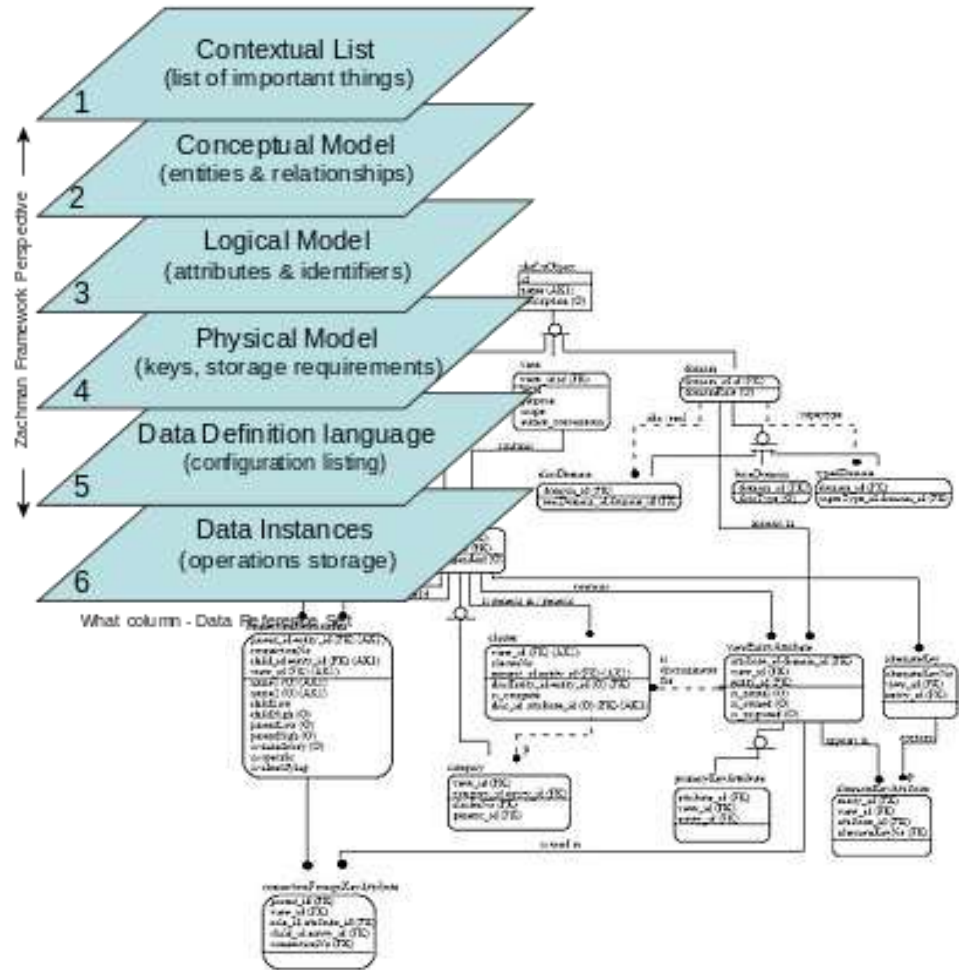
WSDL ([англ.](#) *Web Services Description Language*) — язык описания [веб-сервисов](#) и доступа к ним, основанный на языке [XML](#).



Объект проектирования: данные

IDEF1 (*integration definition for information modeling*) — одна из методологий семейства [IDEF](#). Применяется для построения информационной модели, которая представляет структуру информации, необходимой для поддержки функций производственной системы или среды.

Модель Захмана ([англ. Zachman Framework](#), некорректная передача фамилии — «Захман») — [онтология](#) предприятия, представляющая собой подход к описанию [архитектуры предприятия](#). Онтология является двумерной классификационной схемой, клетки которой — пересечения элементов двух исторических классификаций. Первая классификация включает набор простейших вопросов: «что», «как», «когда», «кто», «где» и «почему». Вторая классификация проистекает из философской концепции овеществления, то есть претворения абстрактных идей в жизнь. В модели Захмана использованы следующие овеществляющие трансформации: «идентификация», «определение», «представление», «спецификация», «конфигурация» и «конкретизация»



Объект проектирования: данные

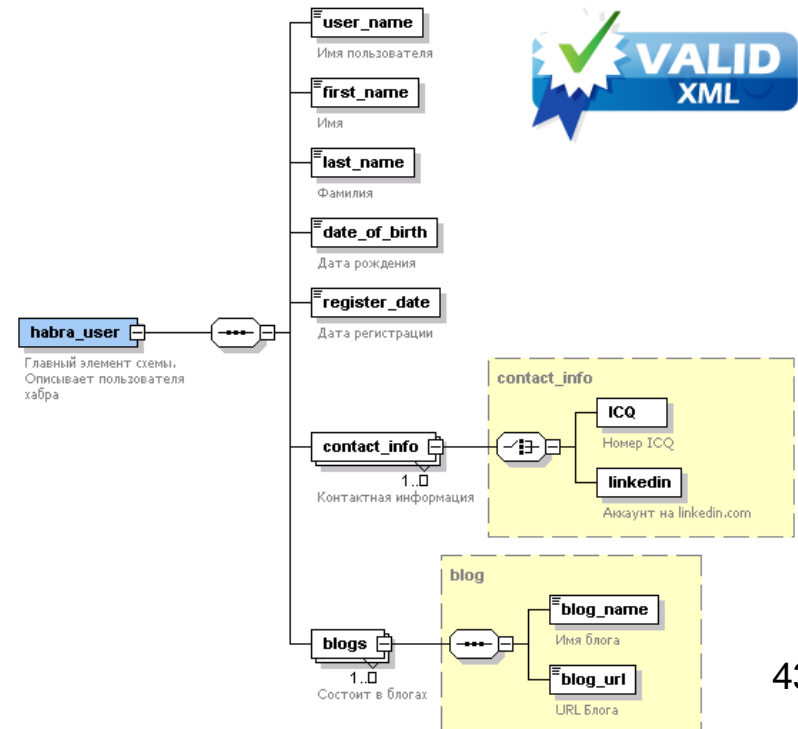
XML Schema — язык описания структуры [XML](#)-документа. Спецификация XML Schema является рекомендацией [W3C](#).

Как большинство языков описания XML, XML Schema была задумана для определения правил, которым должен подчиняться документ. Но, в отличие от других языков, XML Schema была разработана так, чтобы её можно было использовать в создании программного обеспечения для обработки документов XML.

После проверки документа на соответствие XML Schema читающая программа может создать модель данных документа, которая включает:

- словарь (названия элементов и атрибутов);
- модель содержания (отношения между элементами и атрибутами и их структура);
- типы данных.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="country">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="country_name" type="xs:string"/>
        <xs:element name="population" type="xs:decimal"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```



Объект проектирования: метамодель

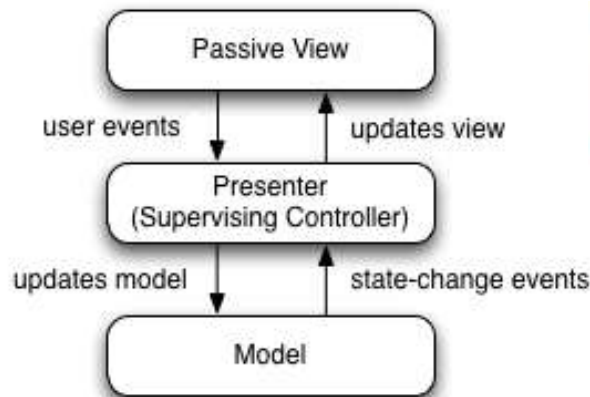
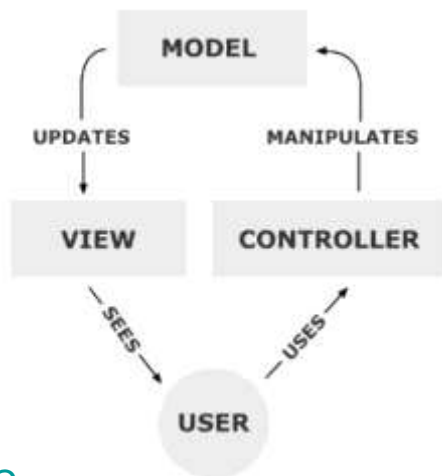
Фрэймворк (иногда *фреймвóрк*; [англицизм](#), [неологизм](#) от *framework* — каркас, структура) — [программная платформа](#), определяющая структуру программной системы; [программное обеспечение](#), облегчающее разработку и объединение разных компонентов большого программного проекта.

Model-View-Controller (MVC), «Модель-Представление-Контроллер», «Модель-Вид-Контроллер») — схема разделения данных приложения, [пользовательского интерфейса](#) и управляющей логики на три отдельных компонента: модель, представление и контроллер — таким образом, что модификация каждого компонента может осуществляться независимо

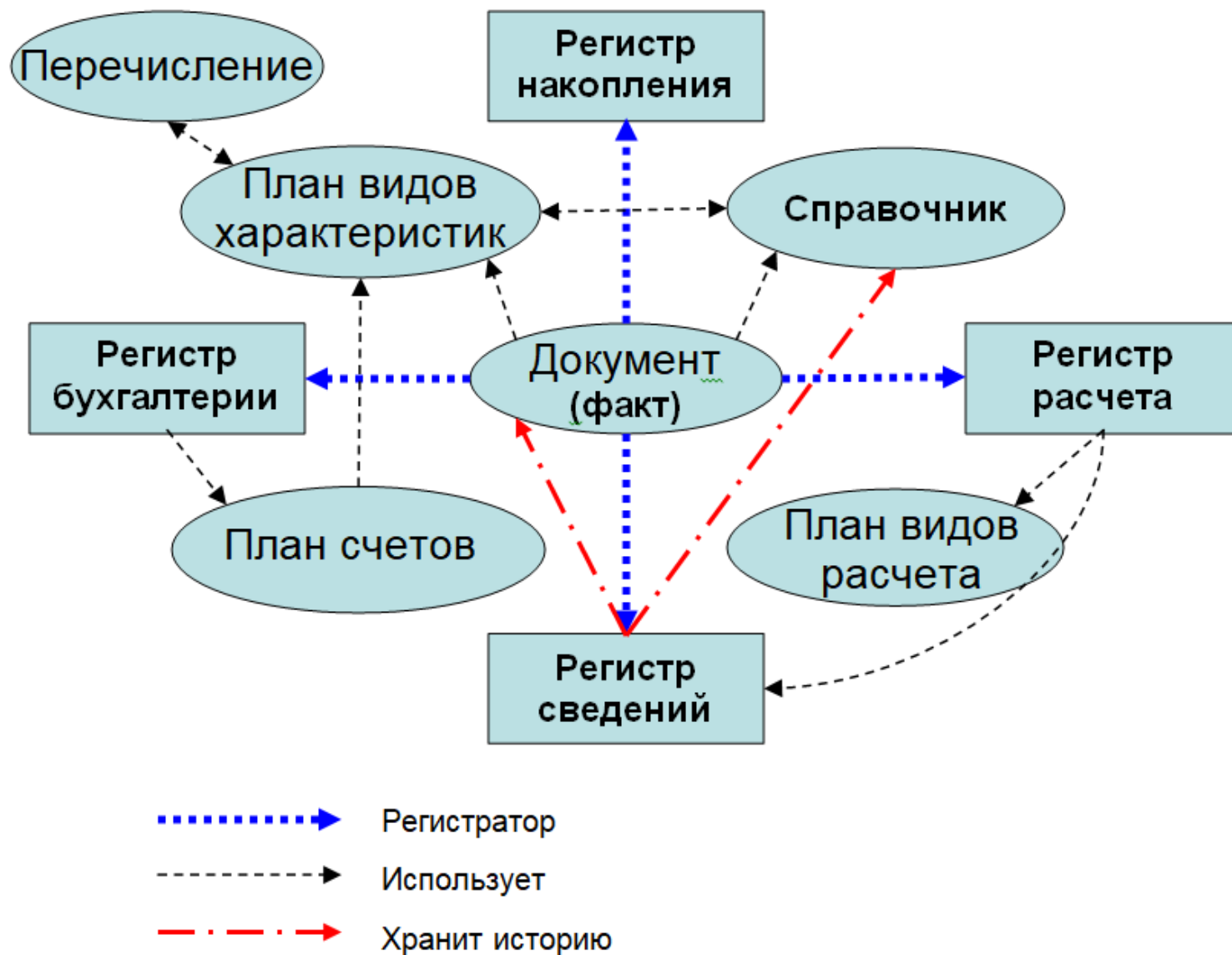
Модель (*Model*) предоставляет данные и реагирует на команды контроллера, изменяя свое состояние^[1].

Представление (*View*) отвечает за отображение данных модели пользователю, реагируя на изменения модели

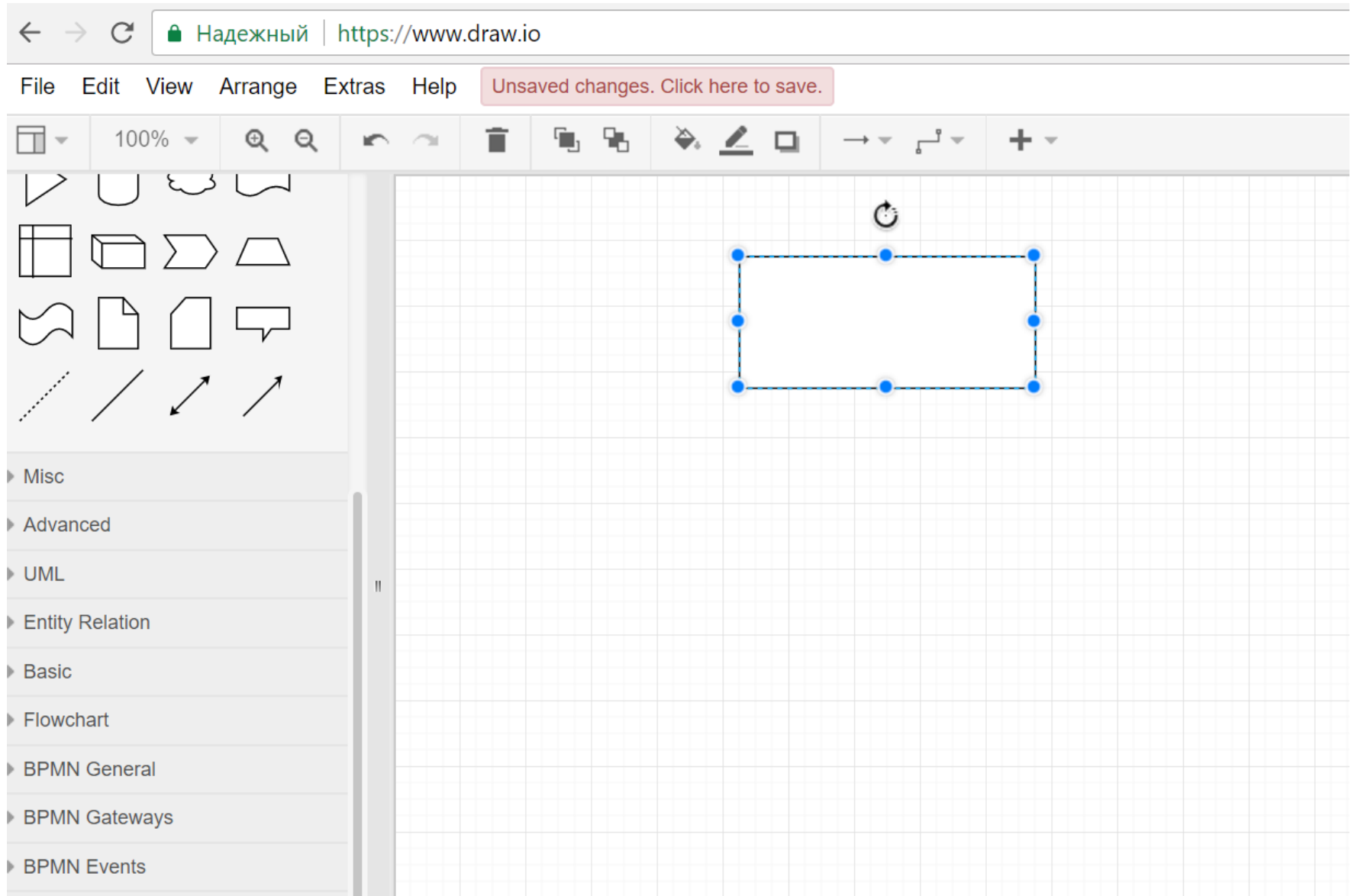
Контроллер (*Controller*) интерпретирует действия пользователя, оповещая модель о необходимости изменений



Объект проектирования: метамодель



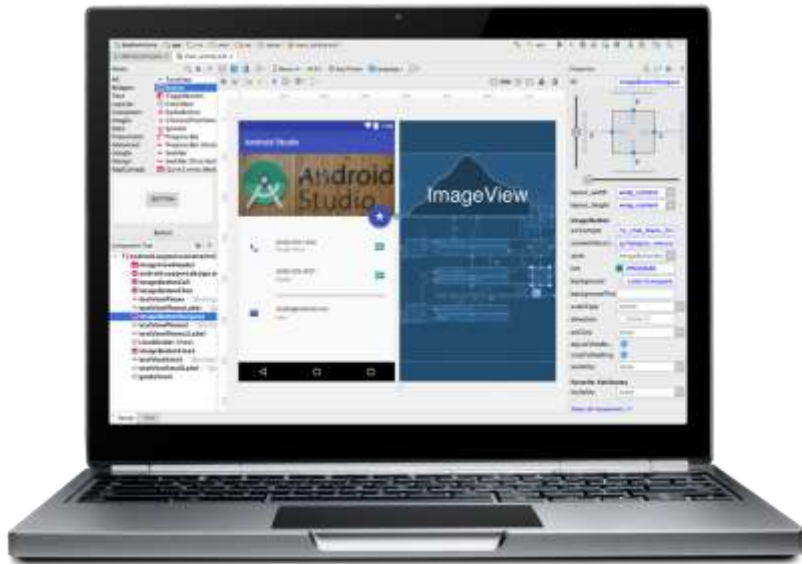
Инструменты разработки схем



Симуляторы и эмуляторы

Симулятор — имитатор (обычно механический или компьютерный), задача которого состоит в имитации управления каким-либо процессом, аппаратом или транспортным средством.

Эмуляция ([англ. emulation](#)) в [вычислительной технике](#) — комплекс программных, аппаратных средств или их сочетание, предназначенное для копирования (или *эмулирования*) функций одной вычислительной системы (*гостя*) на другой, отличной от первой, вычислительной системе (*хосте*) таким образом, чтобы эмулированное поведение как можно ближе соответствовало поведению оригинальной системы (*гостя*).



Инструменты DevOps

DevOps — это командная работа (между сотрудниками, занимающимися разработкой, операциями и тестированием), нет единого инструмента «DevOps»: это скорее набор (или «инструментальная цепочка DevOps»), состоящий из нескольких инструментов. Как правило, инструменты DevOps вписываются в одну или несколько из этих категорий, что отражает ключевые аспекты разработки и доставки программного обеспечения:

Code — разработка и анализ кода, инструменты контроля версий, слияние кода

Build — инструменты непрерывной интеграции, статус сборки

Test — инструменты непрерывного тестирования, которые обеспечивают обратную связь по бизнес-рискам

Package — репозиторий артефактов, предварительная установка приложения

Release — управление изменениями, официальное утверждение выпуска, автоматизация выпуска

Configure — Конфигурация и управление инфраструктурой, Инфраструктура как инструменты кода

Monitor — мониторинг производительности приложений, опыт работы с конечным пользователем

Непрерывная интеграция (CI, [англ. Continuous Integration](#)) — это практика [разработки программного обеспечения](#), которая заключается в слиянии рабочих копий в общую основную ветвь разработки несколько раз в день и выполнении частых автоматизированных сборок проекта для скорейшего выявления и решения интеграционных проблем.