CASE-средства проектирования

Лекция 7

Понятия инженерии, CASE, проектирования.

Овчинников П.Е.

МГТУ «СТАНКИН»,

ст.преподаватель кафедры ИС

29.10.2020

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

ГОСТ Р 57193-2016

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.

CAx

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 аналитика, разработчика программного обеспечения 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

CASE (Википедия)

Терминология: CASE

В состав 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

CASE (Википедия)

Терминология: CASE

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**

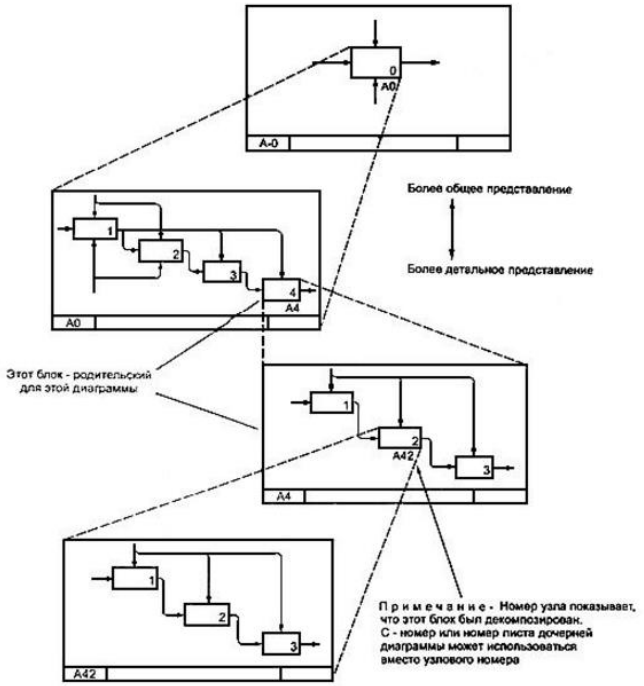
CASE (Википедия)

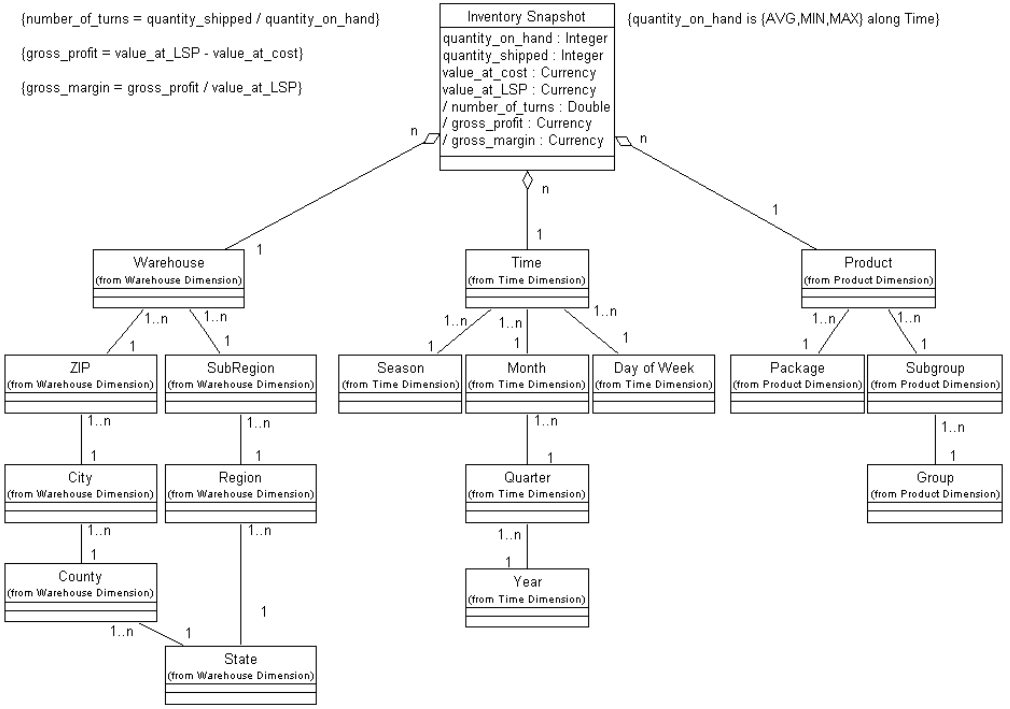
Терминология: CASE

Основной целью CASE-технологии является разграничение **процесса проектирования** программных продуктов **от процесса кодирования** и последующих этапов разработки, максимальная автоматизация процесса разработки.

Для выполнения поставленной цели CASE-технологии используют два принципиально разных подхода к проектированию:

структурно-функциональный и **объектно-ориентированный**.





[CASE (Википедия)]

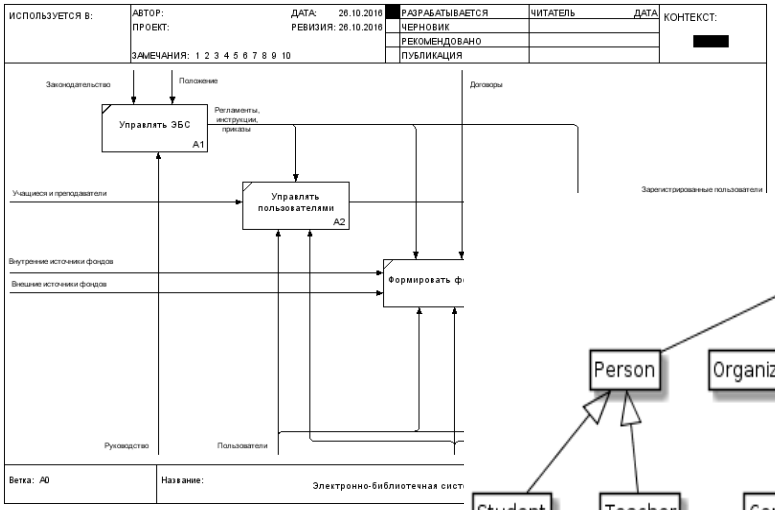
Терминология: CASE

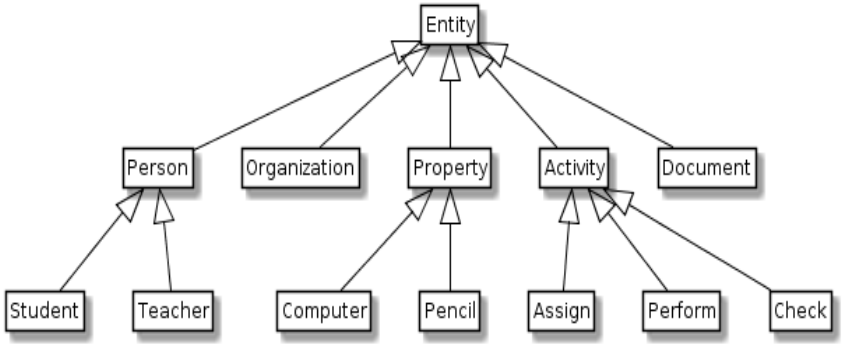
Структурно-функциональный подход предполагает **декомпозицию** (разделение) поставленной задачи **на функции**, которые необходимо автоматизировать.

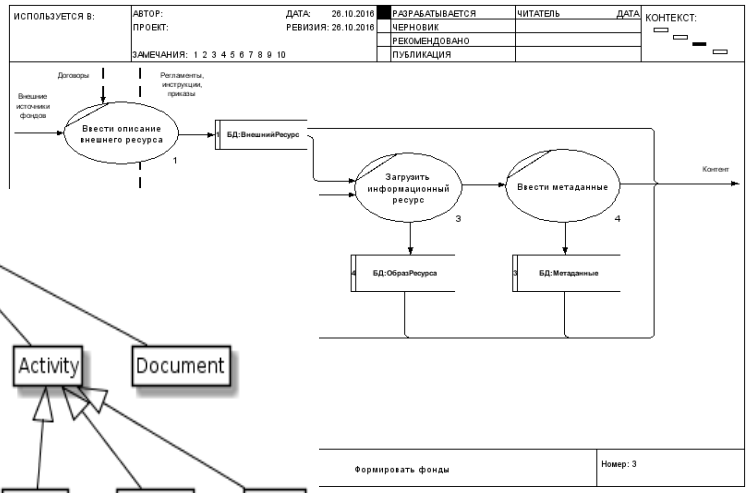
В свою очередь, функции также разбиваются на подфункции, задачи, процедуры. В результате получается упорядоченная иерархия функций и передаваемой информацией между функциями.

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)



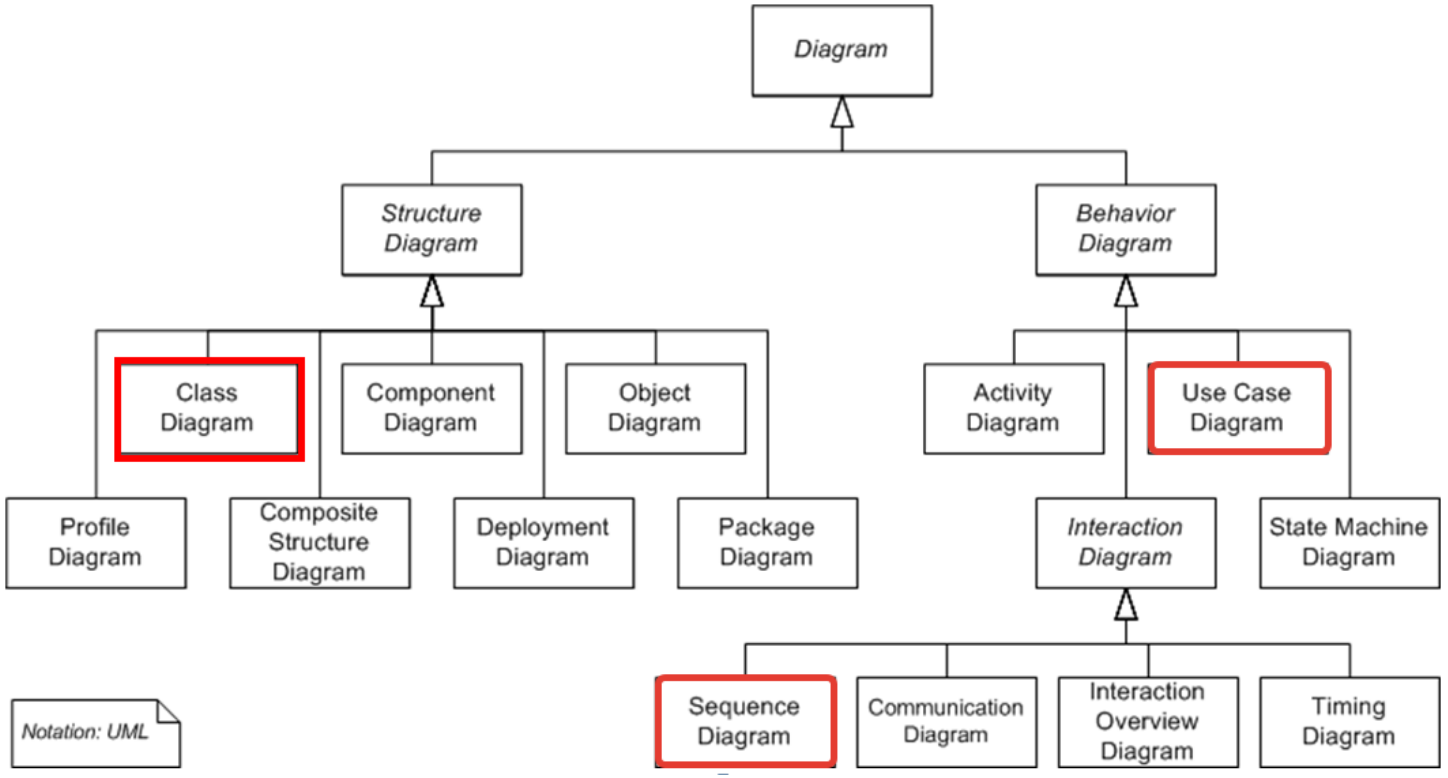




Терминология: CASE

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.



http://www.intuit.ru/studies/courses/1007/229/lecture/5950

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, которые 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

ГОСТ Р ИСО/МЭК 14764-2002

Терминология: обратная разработка

Проекти́рование (англ. *design*):
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

Обра́тная разрабо́тка (**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.**

Терминология: проектирование

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.

ISO/IEC/IEEE 24765:2017

Объект проектирования: система

Модель

искусственный объект, представляющий собой отображение (образ) системы и ее компонентов

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

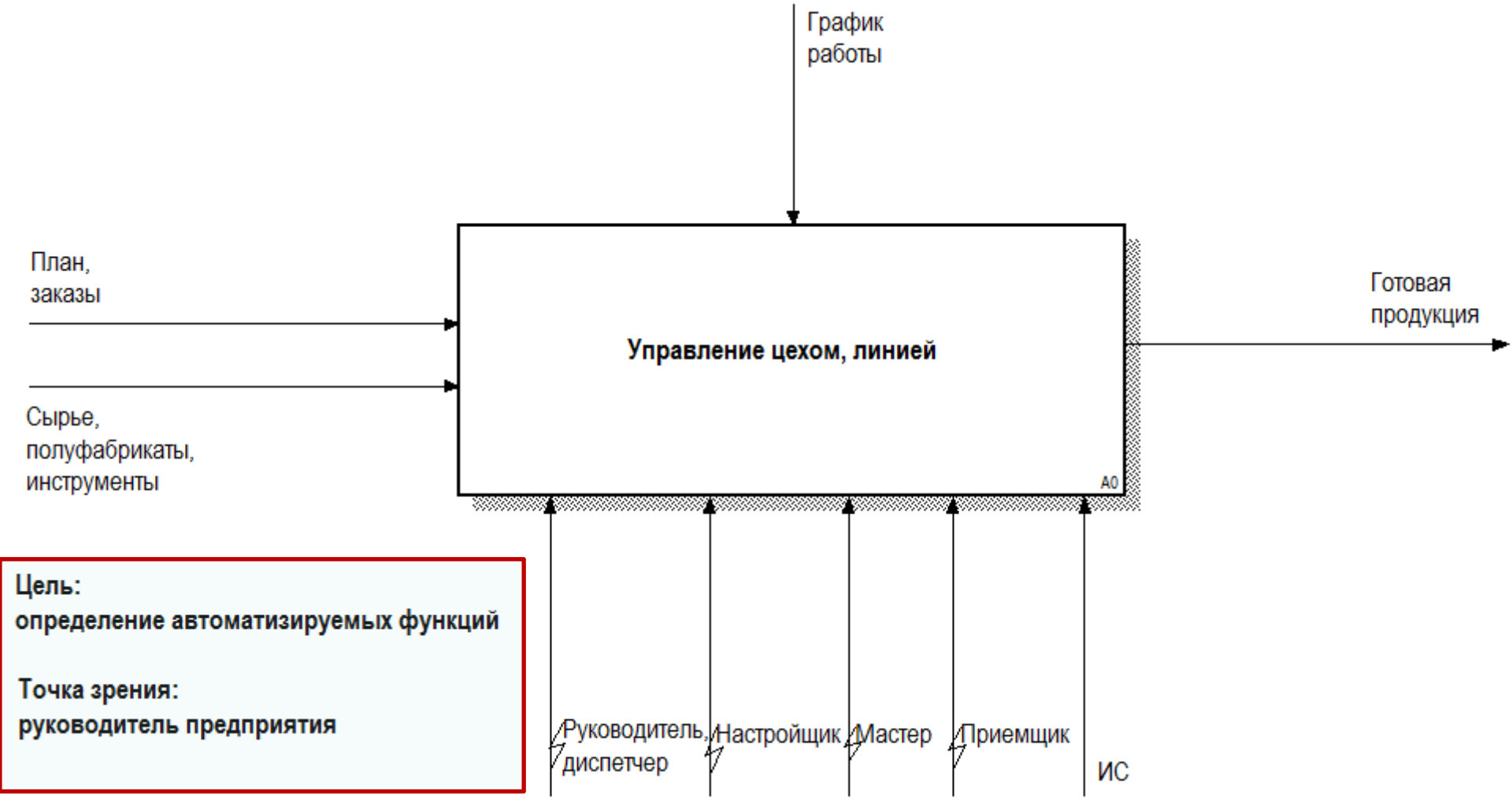
Частями (**элементами**) системы могут быть любые **комбинации** разнообразных **сущностей**, включающие **людей**, **информацию, программное обеспечение, оборудование, изделия, сырье или энергию** (энергоносители)

Структурно-функциональная модель

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

Р 50.1.028-2001

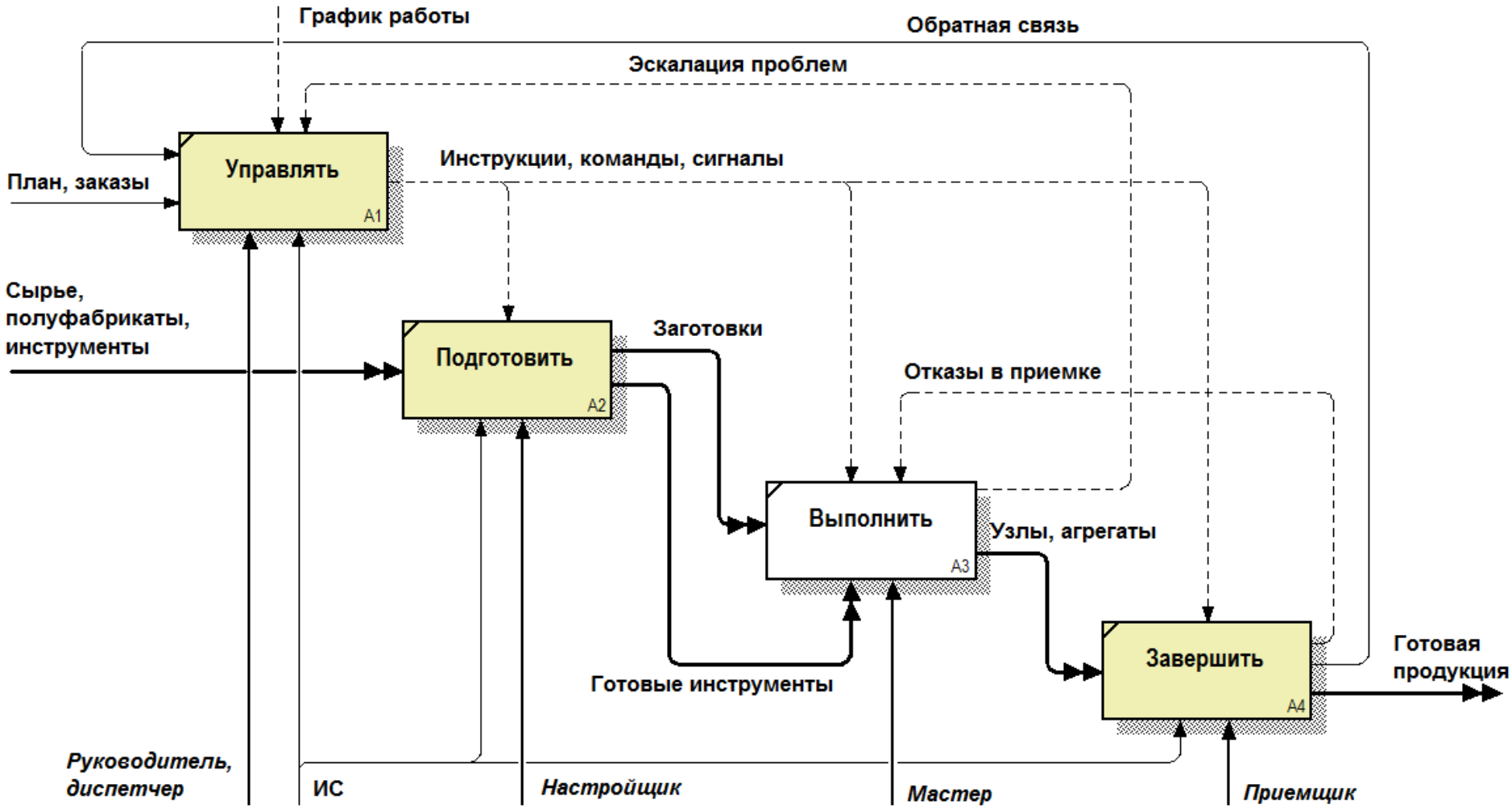
Цель моделирования и точка зрения



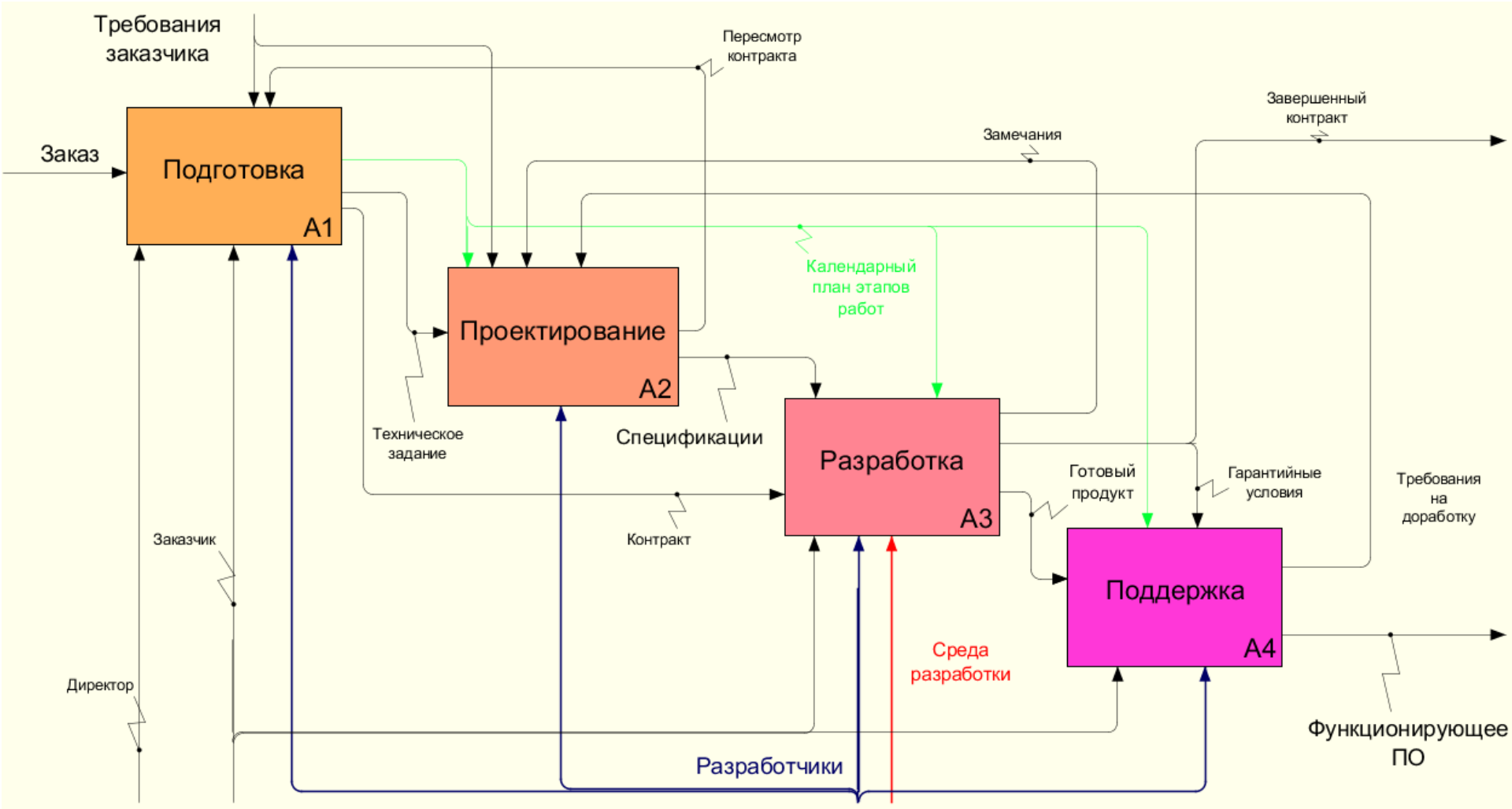
Цель:
определение автоматизируемых функций

Точка зрения:
руководитель предприятия

Определение автоматизируемых функций

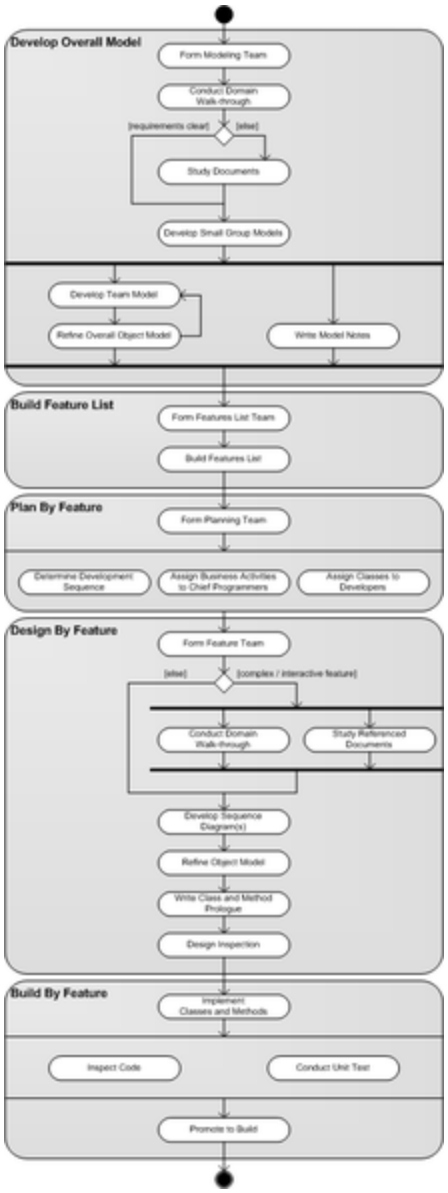


Модели «как есть» и «как будет»: FDD



Модели «как есть» и «как будет»: FDD

Feature driven development (**FDD**, *разработка, управляемая функциональностью*) — итеративная методология разработки программного обеспечения, одна из *гибких 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.

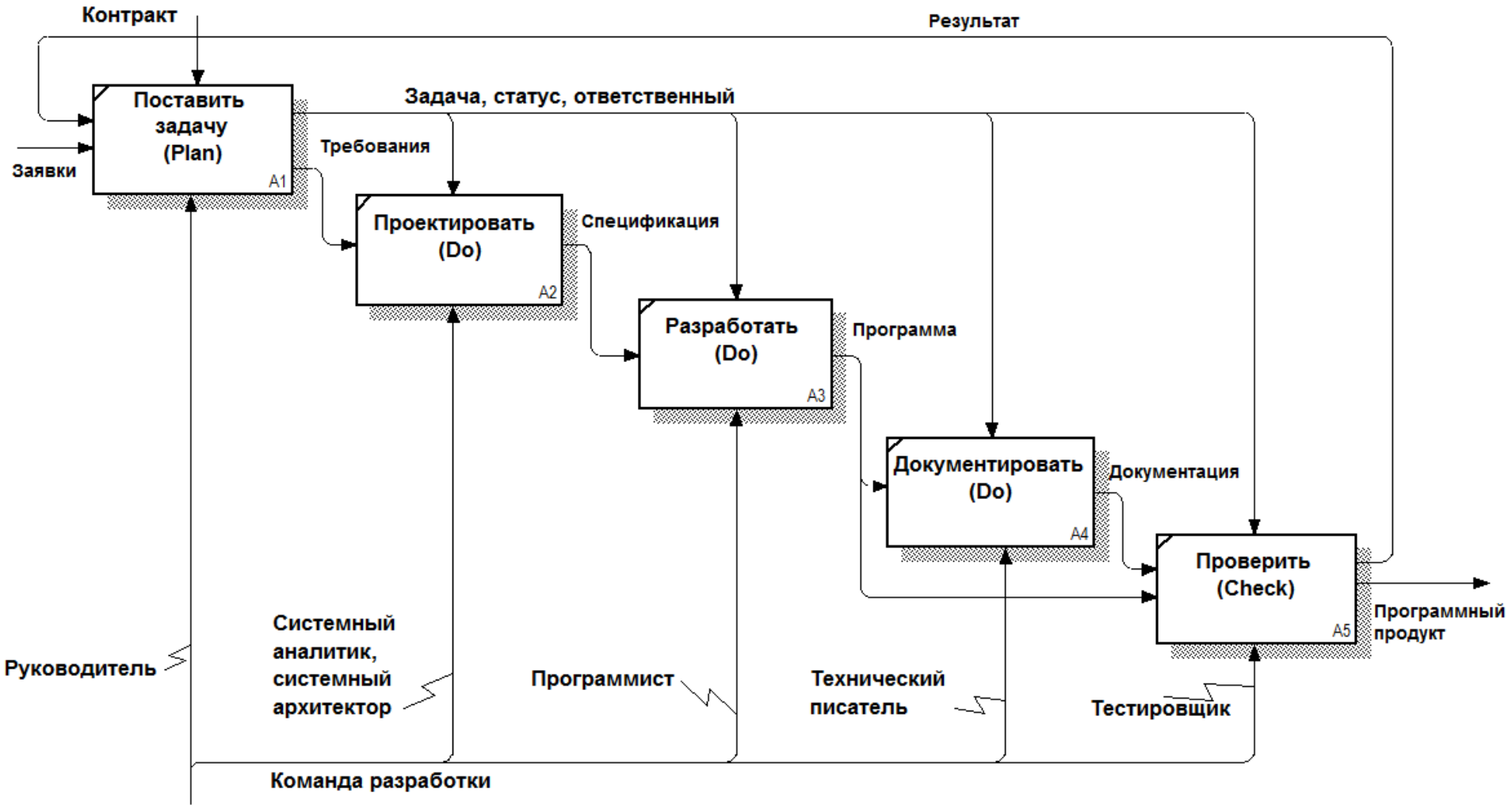


FDD включает в себя пять базовых видов деятельности:

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

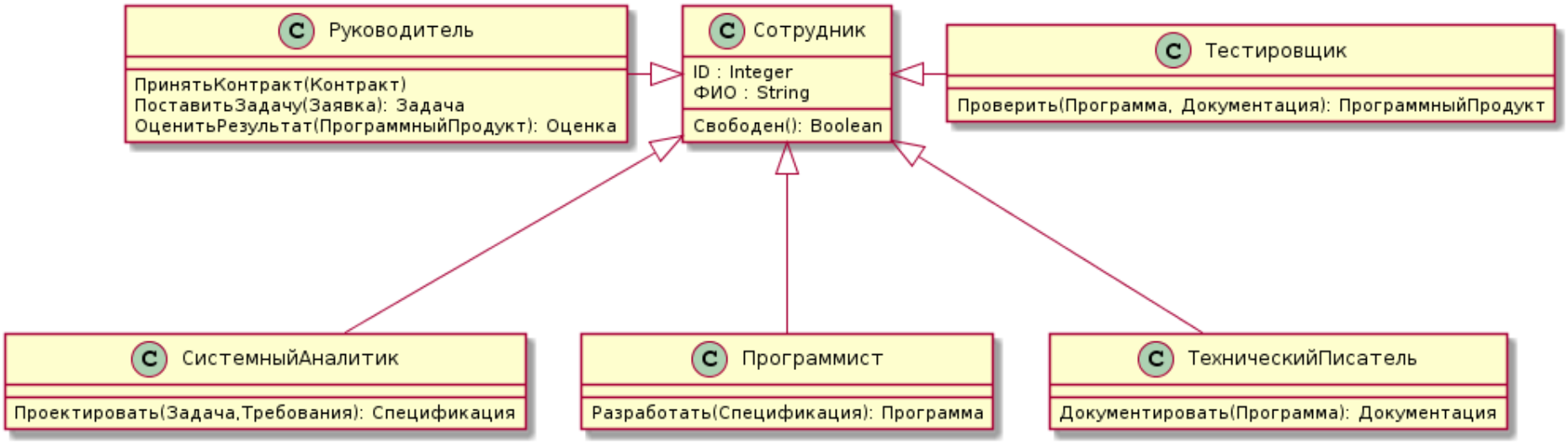
FDD и цикл Деминга PDC(A)



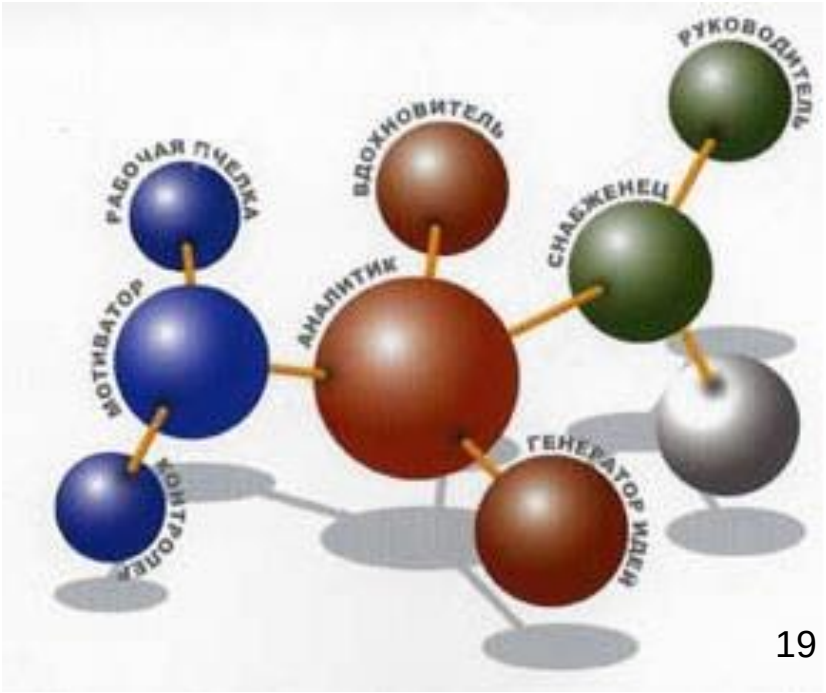
PDCA

Введение в предпринимательство для ИТ-проектов

Как есть: модель FDD и команда



Аналитик
Вдохновитель
Генератор идей
Контролер
Мотиватор
Рабочая пчелка
Руководитель
Снабженец



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

Модели «как есть» и «как будет»: TDD

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:

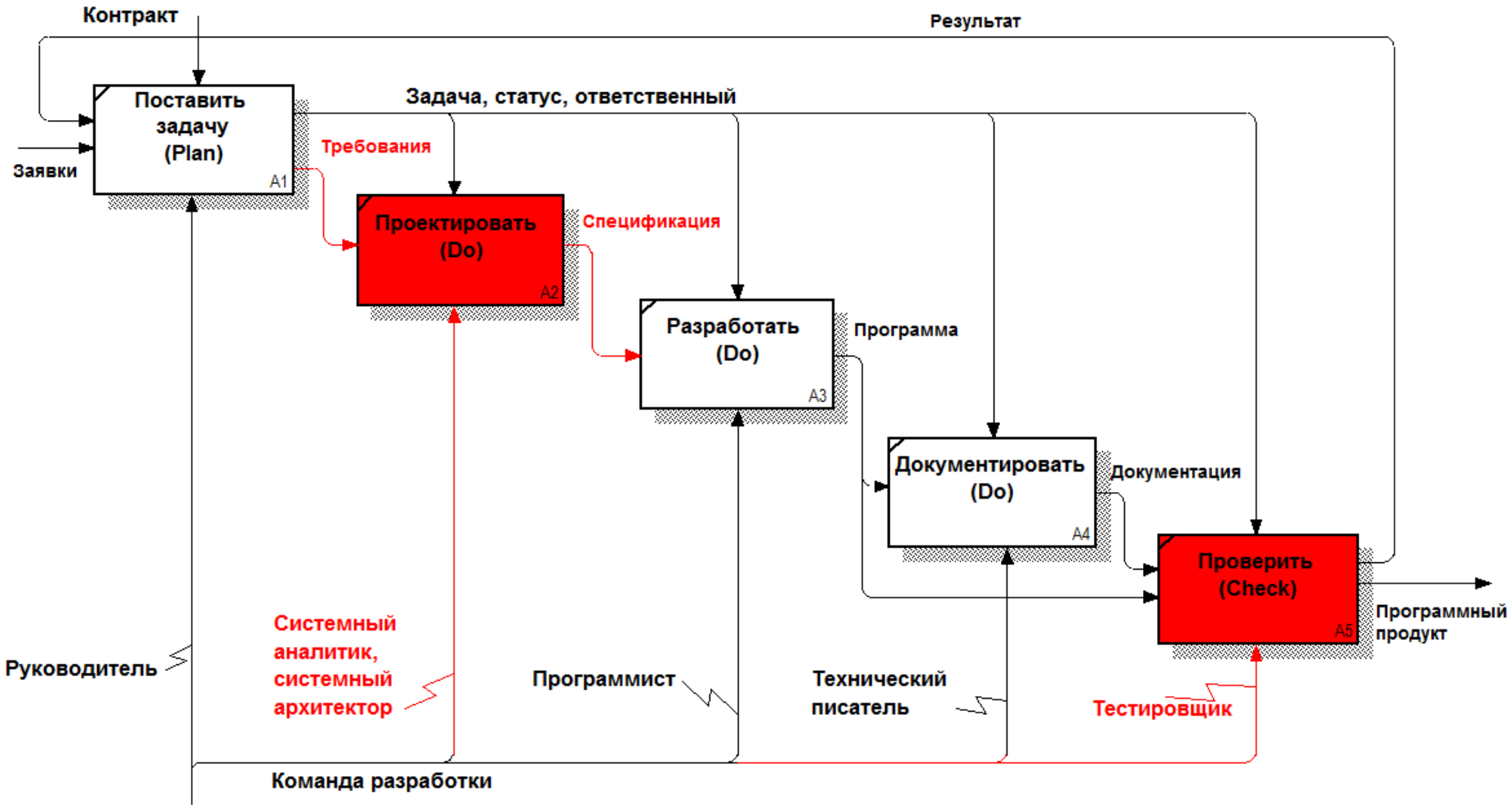
1. Быстро **добавить тест**
2. **Выполнить** все **тесты** и увидеть, что новый тест **"падает"**
3. Выполнить **небольшое изменение** системы
4. Убедиться, что все **тесты проходят**
5. Выполнить **рефакторинг**, удаляя дублирование

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).

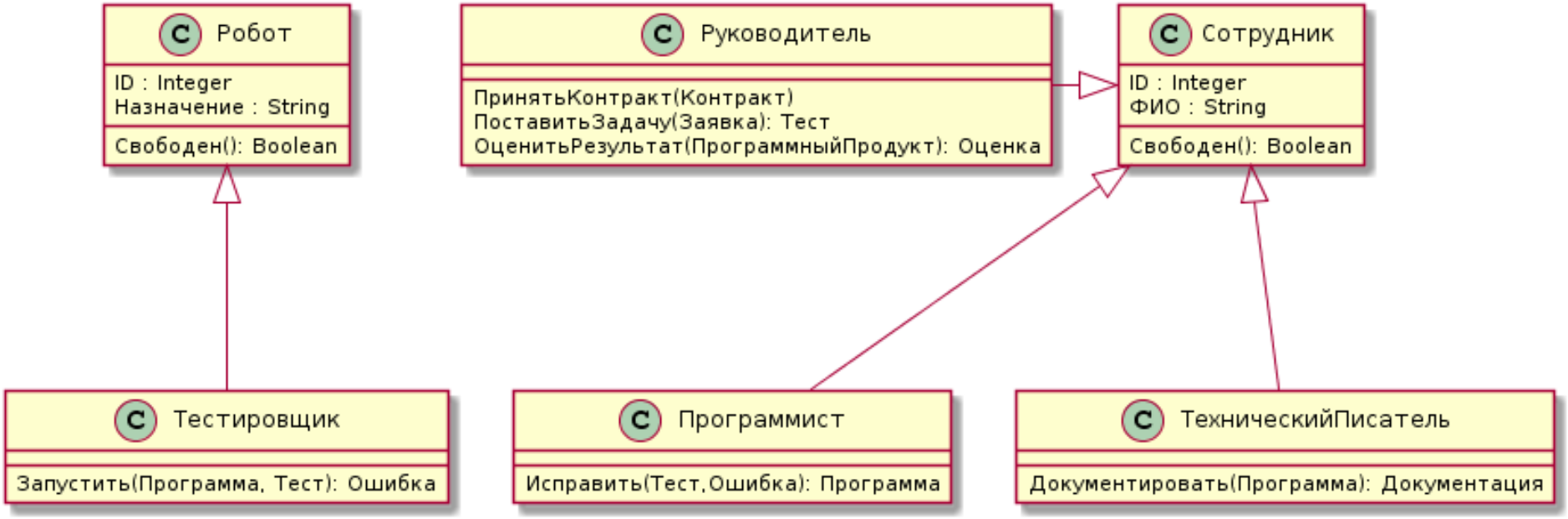
Рефакторинг — процесс изменения **внутренней структуры** программы, не затрагивающий её **внешнего поведения** и имеющий целью:

- облегчить **понимание** её работы,
- устранить **дублирование** кода,
- облегчить **внесение изменений** в ближайшем будущем.

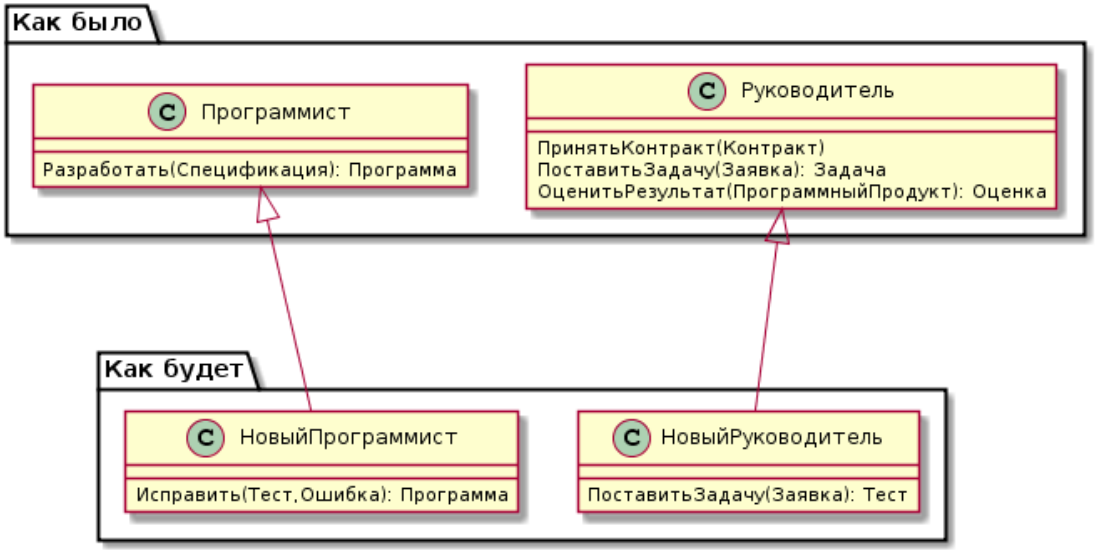
Как будет: модель TDD и команда



Как будет: модель TDD и команда

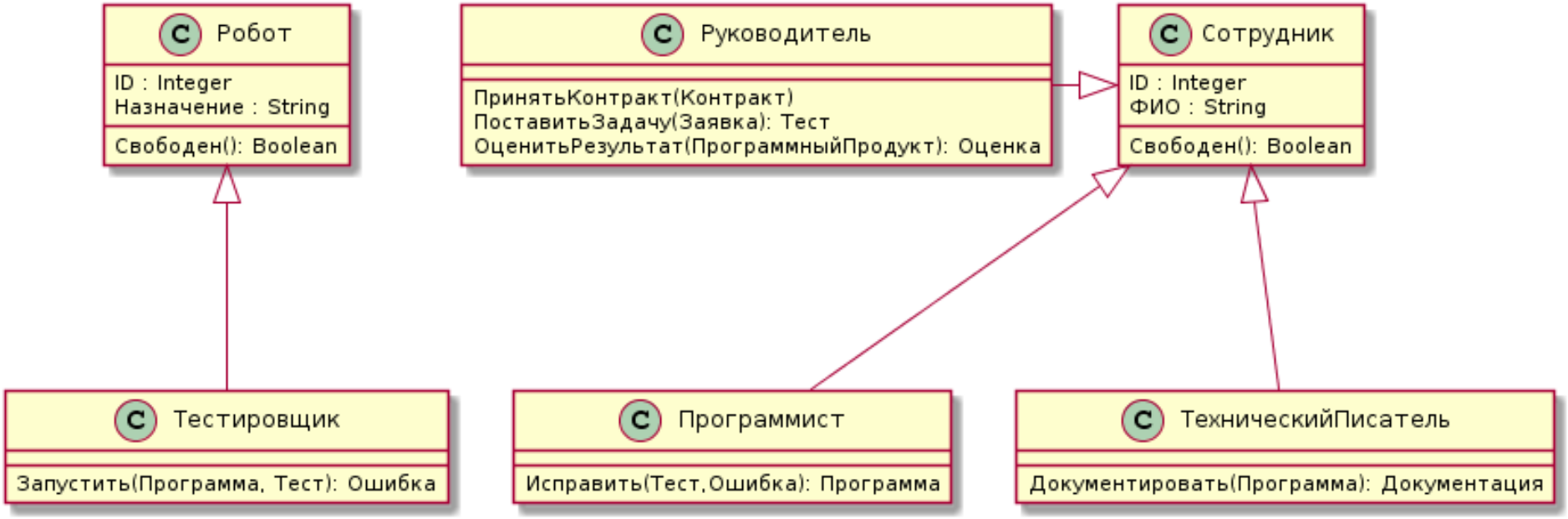


Путем сравнения моделей «как есть» (As Is) и «как должно быть» (To Be) составляется модель «что сделать» (To Do).

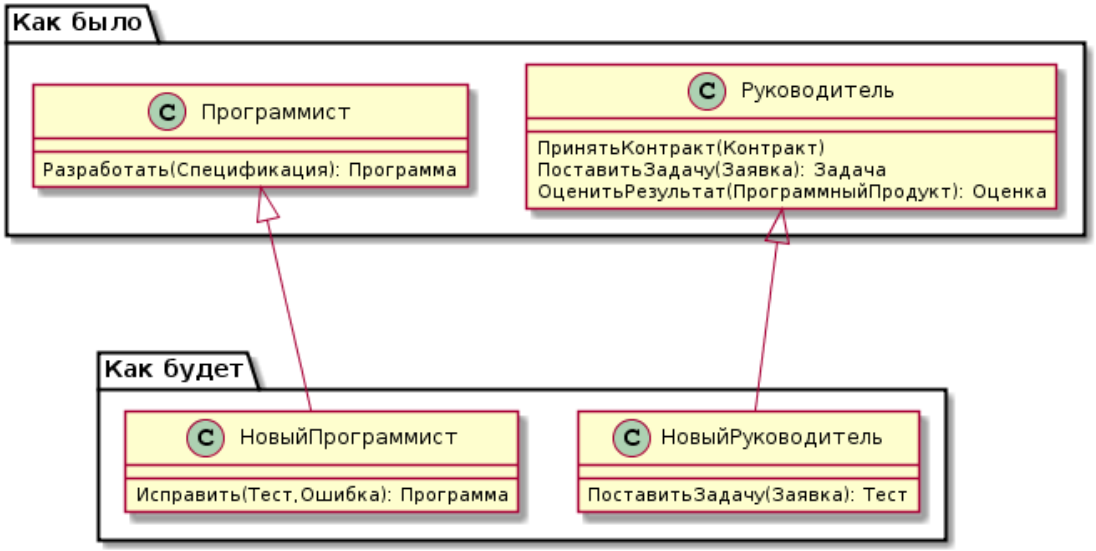


Формирование команды и управление
Роли в команде (теория М.Белбина)

Как будет: модель TDD и команда

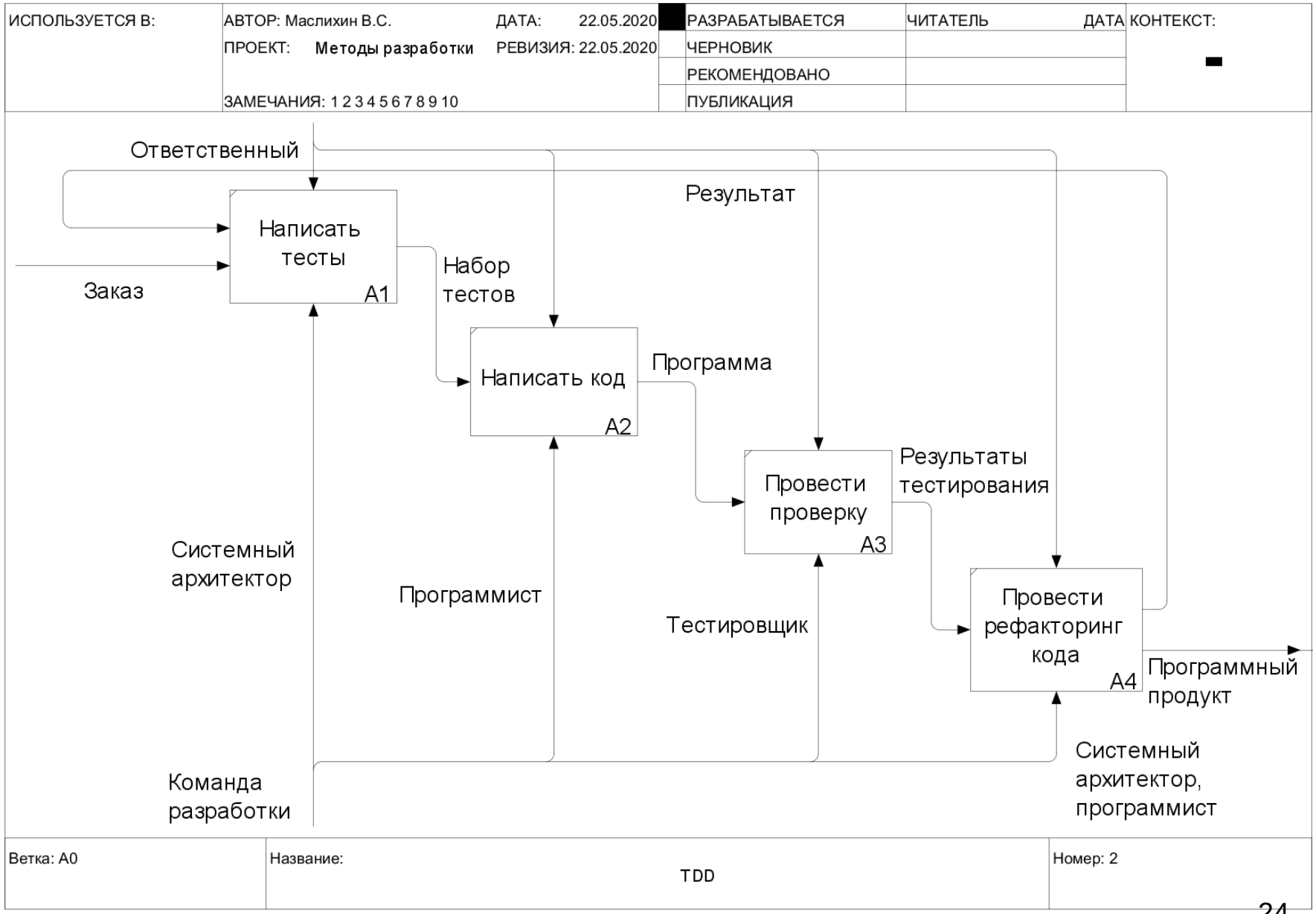


Путем сравнения моделей «как есть» (As Is) и «как должно быть» (To Be) составляется модель «что сделать» (To Do).



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

Как будет: промежуточная модель TDD



Модели TDD, TFD, BDD (реферат)

Модели «как есть» и «как будет»: MDD

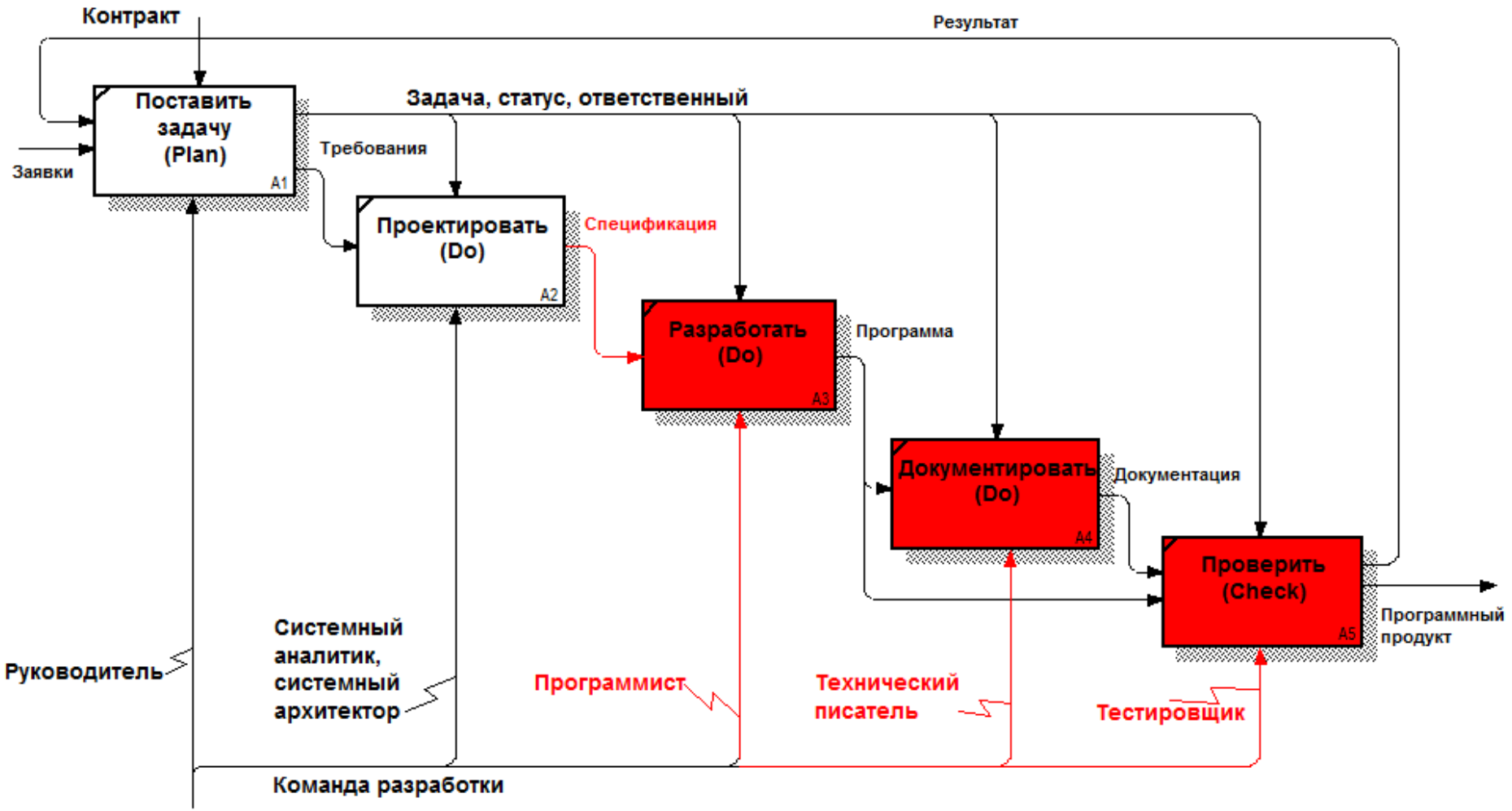
Разработка, управляемая моделями (*model-driven development,* **MDD***, Model-driven engineering*, **MDE**) — это стиль разработки программного обеспечения, когда **модели становятся основными артефактами разработки**, из которых генерируется код и другие артефакты.

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 **метамодель**. Нотация представляет собой совокупность графических элементов, которые применяются в модели и могут быть интерпретированы людьми. Метамодель описывает используемые в модели понятия и фиксирует информацию в виде метаданных, которые могут быть обработаны инструментальными средствами.

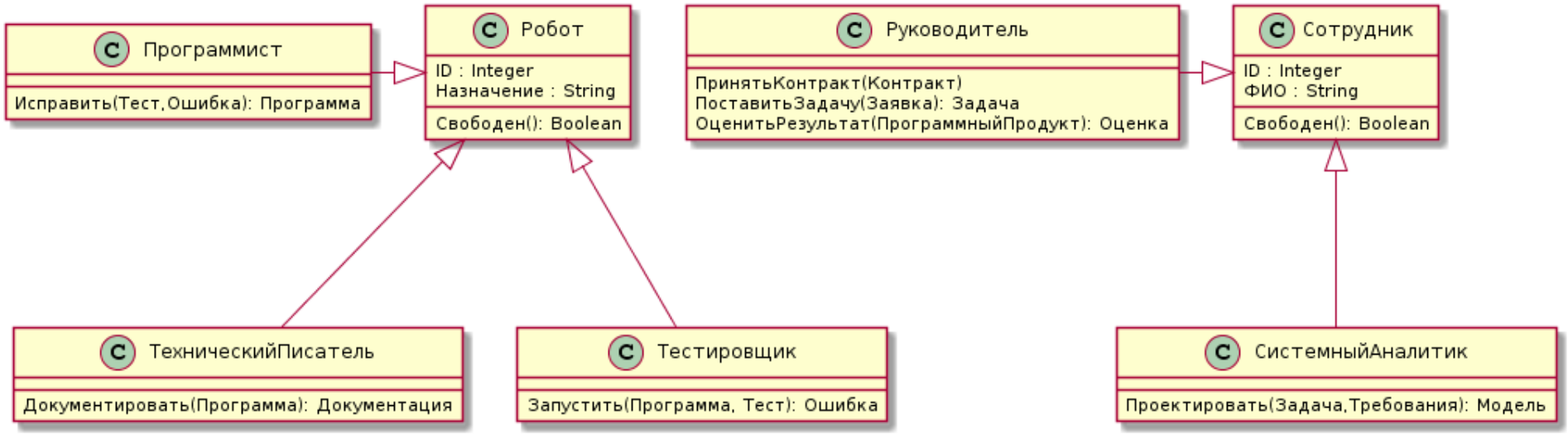
Наиболее известными современными MDE-инициативами являются:

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

Как будет: модель MDD и команда



Как будет: модель MDD и команда



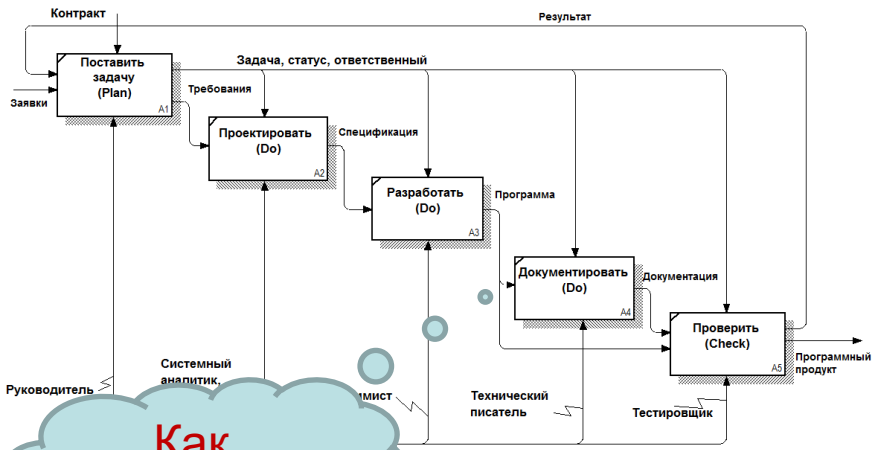
RAD (от англ. *rapid application development* — быстрая разработка приложений) — концепция создания **средств разработки** программных продуктов, уделяющая особое внимание быстроте и удобству программирования, созданию технологического процесса, позволяющего программисту максимально быстро создавать компьютерные программы.

Практическое определение: RAD — это жизненный цикл процесса проектирования, созданный для достижения более высокой **скорости разработки** и качества программного обеспечения, 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.

RAD

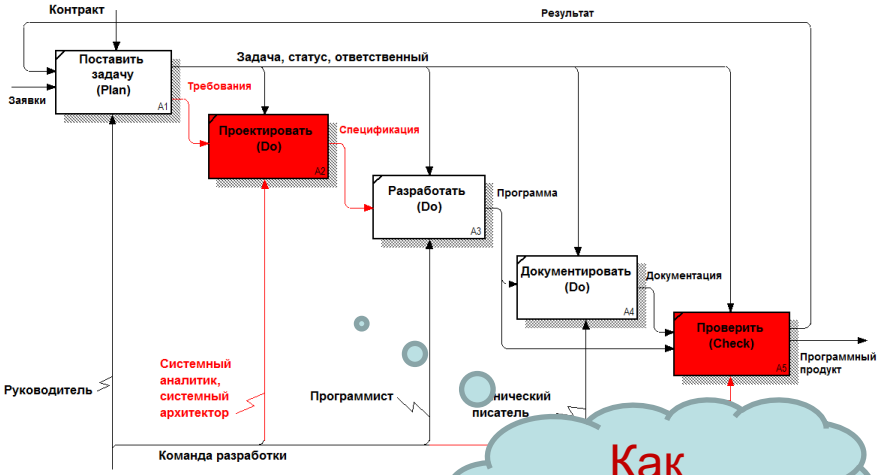
Проблематика цикла PDC(A): через A (act) могут изменяться процессы

AsIs (как есть)



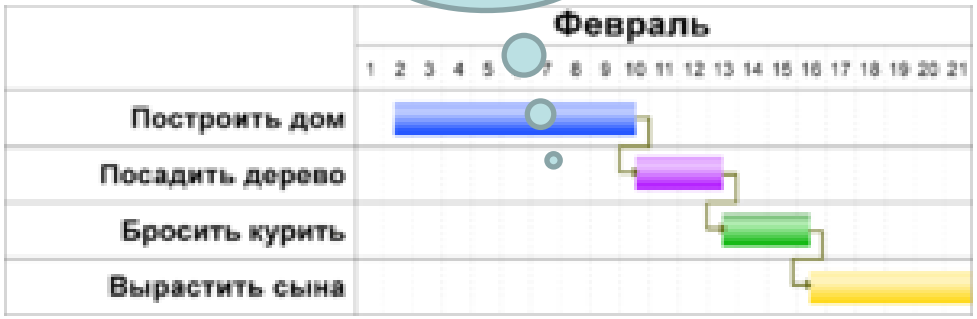
Как узнать?

ToBe (как будет)



Как выбрать?

ToDo (что сделать?) = ИННОВАЦИЯ



Диаграммы Гантта

Как узнать: изучение процессов

Фотография рабочего дня — метод изучения рабочего времени наблюдением и измерением всех без исключения затрат на протяжении рабочего дня или отдельной его части

Учёт рабочего времени с помощью *фотографии рабочего дня* проводится с целью выявления резервов повышения производительности труда. Задачи проведения наблюдения:

- **фиксировать** фактический **баланс рабочего времени**, фактическую выработку продукции и темпов её выпуска на протяжении рабочей смены
- **выявить потери** рабочего времени, установить их причины и разработать мероприятия по совершенствованию организации труда и за счёт устранения потерь и нерациональных затрат времени
- **получить исходные данные** для разработки **нормативов** подготовительно-заключительного времени, времени на отдых и личные надобности, нормативов обслуживания
- **определить причины** невыполнения норм рабочими, изучить лучший опыт, определить возможности совмещения профессий и многостаночного обслуживания
- **выявить устаревшие и ошибочные нормы**, провести анализ использования рабочего времени передовыми рабочими (лучшим опытом)
- определить **рациональный состав бригады** и **формы разделения труда** при бригадном методе организации труда
- получить исходные материалы с целью установления **наиболее рациональной организации** рабочих мест и их обслуживания

Фотография рабочего дня

Как выбрать: имитационное моделирование

Имитационное моделирование (англ. *simulation modeling*) — метод исследования, при котором изучаемая **система заменяется** моделью, с достаточной точностью описывающей реальную систему (построенная модель описывает процессы так, как они проходили бы в действительности), с которой проводятся эксперименты с целью получения информации об этой системе.

Такую модель можно «проиграть» во времени, как для одного испытания, так и заданного их множества, при этом результаты будут определяться **случайным характером процессов**. Экспериментирование с моделью называют имитацией (имитация — это постижение сути явления, не прибегая к экспериментам на реальном объекте).

Имитационное моделирование — это частный случай математического моделирования.

Имитационная модель — логико-математическое **описание объекта**, которое может быть использовано для экспериментирования на компьютере в целях проектирования, анализа и оценки функционирования объекта.

Расчёт себестоимости по видам деятельности (функционально-стоимостной анализ, ФСА; англ. *Activity-based costing, ABC*) — специальная **модель описания затрат,** которая **идентифицирует работы** фирмы и **назначает затраты** каждой такой работы в соответствии с настоящей стоимостью каждой отдельно взятой работы.

Эта модель также переводит накладные расходы в прямые затраты, в отличие от обычной модели описания затрат.

Как выбрать: деловая игра

Деловая игра — метод **имитации принятия решений** руководящих работников или специалистов в различных производственных ситуациях, осуществляемый **по заданным правилам** группой людей или человеком с компьютером в диалоговом режиме, при наличии конфликтных ситуаций или информационной неопределённости

Психолого-педагогические принципы организации деловой игры:

*принцип **имитационного моделирования***

конкретных условий и динамики производства. Моделирование реальных условий профессиональной деятельности специалиста во всем многообразии служебных, социальных и личностных связей является основой методов интерактивного обучения;

*принцип **игрового моделирования***

содержания и форм профессиональной деятельности. Реализация этого принципа является необходимым условием учебной игры, поскольку несет в себе обучающие функции;

*принцип **совместной деятельности***

В деловой игре этот принцип требует реализации посредством вовлечения в познавательную деятельность нескольких участников. Он требует от разработчика выбора и характеристики ролей, определения их полномочий, интересов и средств деятельности. При этом выявляются и моделируются наиболее характерные виды профессионального взаимодействия «должностных» лиц;

Как выбрать: деловая игра

Деловая игра — метод **имитации принятия решений** руководящих работников или специалистов в различных производственных ситуациях, осуществляемый **по заданным правилам** группой людей или человеком с компьютером в диалоговом режиме, при наличии конфликтных ситуаций или информационной неопределённости

Психолого-педагогические принципы организации деловой игры:

*принцип **диалогического общения***

В этом принципе заложено необходимое условие достижения учебных целей. Только диалог, дискуссия с максимальным участием всех играющих способна породить поистине творческую работу. Всестороннее коллективное обсуждение учебного материала обучающимися позволяет добиться комплексного представления ими профессионально значимых процессов и деятельности.

*принцип **двуплановости***

Принцип двуплановости отражает процесс развития реальных личностных характеристик специалиста в «мнимых», игровых условиях. Разработчик ставит перед обучающимся двоякого рода цели, отражающие реальный и игровой контексты в учебной деятельности.

*принцип **проблемности***

содержания имитационной модели и процесса её развёртывания в игровой деятельности.

Деловая игра: роли менеджмента

Фамилия	Имя	Роль (основная)	Роль (дополнительная)
		АД (Мастер)	БА (Тестировщик)
		БА (Тестировщик)	ПП (Программист)
		КО (Тех.писатель)	НИ (Архитектор)
		НИ (Архитектор)	КО (Тех.писатель)
		ПП (Программист)	ВН (Дизайнер)
		РП (Владелец продукта)	АД (Мастер)
		СП (Аналитик)	ВН (Дизайнер)

Основная роль	Ответственность (компетенция, зона принятия решений)
РП (Владелец продукта)	Бизнес-результат, решение проблем, обеспечение ресурсами
АД (Мастер)	Диспетчирование и контроль задач, выявление проблем
СП (Аналитик)	Сбор и управление всеми требованиями в проекте
ВН (Дизайнер)	Удобство использования, привлекательность продукта
БА (Тестировщик)	Выявление бизнес-проблем, способы тестирования
НИ (Архитектор)	Структура продукта, инструменты разработки и поставки
ПП (Программист)	Стиль и способы разработки, используемые фреймворки
КО (Тех.писатель)	Документирование проекта и продукта

Деловая игра: пособие по выбору ролей
Командные роли сотрудников (реферат)

Деловая игра: роли как сильные стороны

Сильные стороны

Слабые стороны

РЕЗУЛЬТАТ ПО ШКАЛАМ

- **Оценщик - 24.3% (17 баллов)**
- **Коллективист - 14.3% (10 баллов)**
- **Разведчик - 14.3% (10 баллов)**
- Доводчик - 11.4% (8 баллов)
- Председатель - 11.4% (8 баллов)
- Исполнитель - 11.4% (8 баллов)
- Мыслитель - 8.6% (6 баллов)
- Формирователь - 4.3% (3 баллов)

	Возможности (Opportunities)				Угрозы (Threats)			
Сильные стороны (Strengths)						10%		
	90%							
			50%					
								30%
Слабые стороны (Weaknesses)		70%						
				20%				
							40%	
					15%			

Тест: SWOT-анализ (вопросы 19-20)

Деловая игра: управление личными рисками

Источник рисков – сомнение в собственной способности выполнить какие-то из требуемых действий выбранных ролей:

Участник	Стадия	Действие (activity)
БА (Тестировщик)	1 старт	Разбивает задачу истории на подзадачи - тесты
БА (Тестировщик)	2 контроль	Разрабатывает процедуры - тесты и тестовые наборы данных
БА (Тестировщик)	3 финиш	Разрабатывает и отлаживает процедуры - тесты и тестовые наборы данных

Для каждого действия нужно определить, имеется ли личный риск (потенциальная угроза для проекта) какой-то **одной** из типовых **категорий**:

НАПРИМЕР

1. Дефицит специалистов □
2. Нереалистичные сроки и бюджет □
3. Реализация несоответствующей функциональности □
4. Разработка неправильного пользовательского интерфейса □
5. «Золотая сервировка», перфекционизм, ненужная оптимизация и оттачивание деталей □
6. Непрекращающийся поток изменений □
7. Нехватка информации о внешних компонентах, определяющих окружение системы или вовлечённых в интеграцию □
8. Недостатки в работах, выполняемых внешними (по отношению к проекту) ресурсами □
9. Недостаточная производительность получаемой системы □
10. Разрыв между квалификацией специалистов и требованиями проекта □

Тест: SWOT-анализ (вопросы 19-20)

Деловая игра: управление личными рисками

Если риск имеется и существенен, требуется выбрать **одну** наиболее приемлемую **стратегию управления** этим риском (угрозой)

После определения стратегии требуется определить **основное мероприятие** управления каждым риском в выбранной стратегии

Результирующая формулировка для действия «Разрабатывает процедуры – тесты» и категории «Разрыв между квалификацией специалистов и требованиями проекта» может выглядеть так:

- **Уклонение (Avoidance)**

  разработка теста невозможна, действие нужно передать другому сотруднику или роли

- **Снижение (Mitigation)**

  могут быть нарушения сроков в разработке, требуется дополнительное обучение

- **Передача (Transference)**

  разработка возможна, если будут определены средства тестирования (НИ), разработан (БА) и документирован (КО) пример

- **Принятие (Acceptance)**

  разработка не предусматривается, в крайнем случае потребуется консультация специалистов или обучение

Тест: SWOT-анализ (вопросы 19-20)