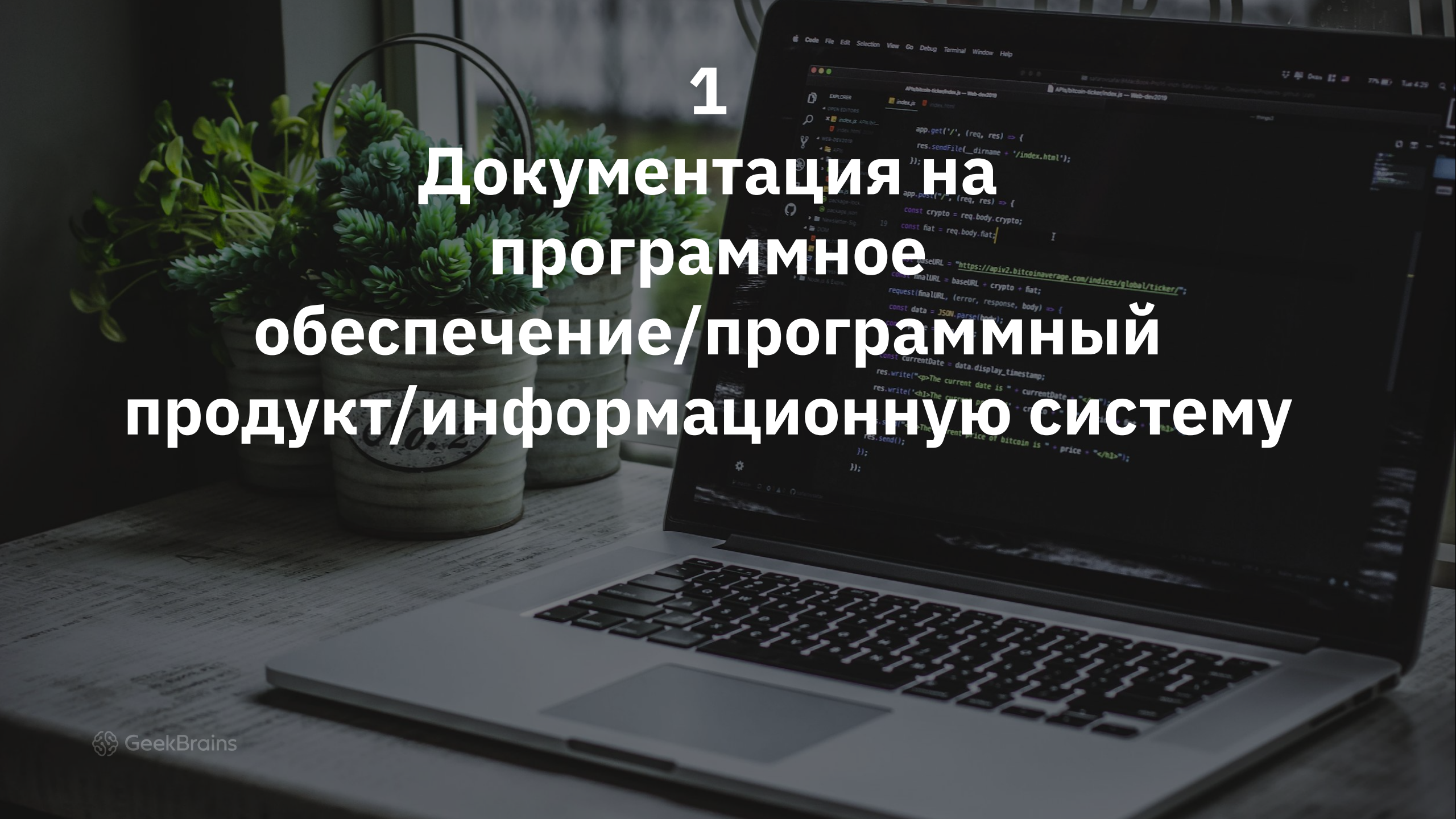
GeekBrains

Подготовка архитектурного описания и проектной документации

Архитектор информационных систем

Подготовка архитектурного описания и проектной документации

1. Общие сведения о процессе документирования
2. Стандарты жизненного цикла ПО и ИС
3. Архитектурное представление
   - Архитектурная модель 4+1 в нотации UML
   - Подход Views and Beyond
   - Представление TOGAF и ArchiMate
   - Представление архитектуры в методологии ARIS
4. Виды и содержание документов проектирования ИС
   - Software Requirement Specification — SRS
   - Software design description – SDD
5. Стандарты



1
Документация на программное обеспечение/программный продукт/информационную систему

GeekBrains

Документация - совокупность материалов (артефактов), сопровождающих разработку на всём жизненном цикле, к которым относятся документы управления и планирования, проектно-конструкторская документация, схемы, листинги программного кода, программы и методики испытаний, руководства пользователя, диалоговая (оперативная) документация и справочный текст, описывающие, как пользоваться программным продуктом.

Документ (артефакт)

- элемент документации: целевая информация, предназначенная для конкретной аудитории, размещенная на конкретном носителе (например, в книге, на диске, в краткой справочной карте) в заданном формате.

Проектная/управленческая

- для планирования, управления производством, выпуском, экономикой проекта по разработке программных продуктов (информационной системы ИС) и принятия управленческих решений, может включать в себя исходные данные и технические задания

Архитектурная/конструкторская

- для описание технических, технологических, композиционных решений, включающая описание рабочей среды, инфраструктуры, зависимостей, методов тестирования, испытаний и приёмки, а также других принципов, которые должны быть использованы при создании программного обеспечения или программных продуктов (ИС)

Техническая/технологическая

- подробная документация на принятые архитектурные решения, алгоритмы, интерфейсы, API, программный код

Пользовательская

- руководства для конечных пользователей, администраторов и другого персонала сопровождающего развитие программного продукта (ИС)

Маркетинговая

- материалы для обеспечения коммерческой деятельности по продвижению и продажам программных продуктов (ИС)

Этапы создания документации

1. Определение потребностей заинтересованных сторон. Если заинтересованные стороны и их потребности не выявлены, то возможно создание документации, которая будет бесполезна для всех участников проекта.
2. Сбор и документирование архитектурной информации в виде группы представлений и специального блока с общей информацией для всех представлений.
3. Проверка того, что созданная документация удовлетворяет требованиям заинтересованных сторон.
4. Подготовка архитектурной документации в вид, пригодный для той или иной заинтересованной стороны.

Типовые документы разработки

1. Анализы осуществимости и исходные заявки;
2. Спецификации требований;
3. Спецификации функций;
4. Проектные спецификации, включая спецификации программ и данных;
5. Планы разработки;
6. Планы сборки и тестирования программного обеспечения;
7. Планы обеспечения качества, стандарты и графики;
8. Защитная и тестовая информация.

В небольших коллективах разработчиков процессу документирования архитектуры не уделяется должного внимания. Обычно детали архитектуры передаются в устной форме. Для обсуждения архитектур используются маркерные доски.

Проблема документирования архитектур программных систем возникает с необходимостью объяснять архитектуру все большей и большей аудитории.

При документировании архитектуры часто требуют ответа следующие вопросы:

- Что именно нужно документировать и когда?
- Каким образом создавать документацию? Какой шаблон документа использовать для описания архитектуры?
- Какие нотации использовать?
- Насколько детальным должно быть описание?

2
Стандарты жизненного цикла ПО и ИС

Жизненный цикл ПО – это непрерывный процесс, который начинается с момента принятия решения о необходимости его создания и заканчивается в момент его полного изъятия из эксплуатации

ISO/IEC 12207: 2010

ГОСТ Р ИСО/МЭК 12207-2010 Информационная технология (ИТ). Системная и программная инженерия. Процессы жизненного цикла программных средств.

Стандарт на процессы и организацию жизненного цикла.
Распространяется на все виды заказного ПО.

ГОСТ 34.601-90

Распространяется на автоматизированные системы и устанавливает стадии и этапы их создания. Кроме того, в стандарте содержится описание содержания работ на каждом этапе. Стадии и этапы работы, закрепленные в стандарте, в большей степени соответствуют каскадной модели жизненного цикла.

Custom Development Method (методика Oracle)

При разработке прикладных информационных систем - технологический материал, детализированный до уровня заготовок проектных документов, рассчитанных на использование в проектах с применением Oracle. Применяется CDM для классической модели ЖЦ (предусмотрены все работы/задачи и этапы), а также для технологий "быстрой разработки" (Fast Track) или "облегченного подхода", рекомендуемых в случае малых проектов.

Rational Unified Process (RUP)

Предлагает итеративную модель разработки, включающую четыре фазы: начало, исследование, построение и внедрение. Каждая фаза может быть разбита на этапы (итерации), в результате которых выпускается версия для внутреннего или внешнего использования. Прохождение через четыре основные фазы называется циклом разработки, каждый цикл завершается генерацией версии системы. Если после этого работа над проектом не прекращается, то полученный продукт продолжает развиваться и снова минует те же фазы. Суть работы в рамках RUP - это создание и сопровождение моделей на базе UML.

Microsoft Solution Framework (MSF)

Сходна с RUP, так же включает четыре фазы: анализ, проектирование, разработка, стабилизация, является итерационной, предполагает использование объектно-ориентированного моделирования. MSF в сравнении с RUP в большей степени ориентирована на разработку бизнес- приложений.

Extreme Programming (XP)

Экстремальное программирование сформировалось в 1996 году. В основе методологии командная работа, эффективная коммуникация между заказчиком и исполнителем в течение всего проекта по разработке ИС, а разработка ведется с использованием последовательно дорабатываемых прототипов.

Agile

Диалоговый принцип разработки, при котором программы разрабатывают поэтапно, с начальной ступени проекта. Это ее основное отличие от каскадных принципов, где код становится доступным только по окончании цикла работы.

Scrum

Это фреймворк гибкой разработки ПО, который считается методологией «по умолчанию». Для многих является синонимом Agile. Процесс управления состоит из фиксированных коротких итераций — спринтов (sprints).Используя методологию Scrum, представитель заказчика плотно работает с командой разработки, расставляя приоритеты в списке требований к продукту (Product Backlog). Этот список состоит из баг-фиксов, функциональных и нефункциональных требований — из всего, что нужно сделать для реализации рабочего приложения.

Процессы жизненного цикла ISO/IEC 12207

Основные процессы

Вспомогательные процессы

Организационные процессы

Основные процессы

1. **Приобретение**
2. **Поставка**
3. **Разработка**
4. **Эксплуатация**
5. **Сопровождение**

Вспомогательные процессы

1. Документирование
2. Управление конфигурацией
3. Обеспечение качества
4. Разрешение проблем
5. Аудит
6. Аттестация
7. Совместная оценка
8. Верификация

Организационные процессы

1. **Создание инфраструктуры**
2. **Управление**
3. **Обучение**
4. **Усовершенствование**

Семь групп процессов жизненного цикла

1. Процессы соглашения - два процесса
2. Процессы организационного обеспечения проекта - пять процессов;
3. Процессы проекта - семь процессов;
4. Технические процессы - одиннадцать процессов;
5. Процессы реализации программных средств - семь процессов;
6. Процессы поддержки программных средств - восемь процессов;
7. Процессы повторного применения программных средств - три процесса.

Всего – 43 подпроцесса

Технические процессы

1. Определение требований правообладателей
2. Анализ системных требований
3. Проектирование архитектуры системы
4. Процесс реализации
5. Процесс комплексирования системы
6. Процесс квалификационного тестирования системы
7. Процесс инсталляции программных средств
8. Процесс поддержки приемки программных средств
9. Процесс функционирования программных средств
10. Процесс сопровождения программных средств
11. Процесс изъятия из обращения программных средств

Процессы реализации программных средств

1. Процесс анализа требований к программным средствам
2. Процесс проектирования архитектуры программных средств
3. Процесс детального проектирования программных средств
4. Процесс конструирования программных средств
5. Процесс комплексирования программных средств
6. Процесс квалификационного тестирования программных средств

Процесс проектирования доменов

1. Выбираются формы представления модели и архитектуры домена
2. Определяются границы домена и его взаимосвязи с другими доменами
3. Разрабатывается модель домена, которая объединяет в себе существенные общие и различные свойства, возможности, концепции и функции в этом домене
4. Разрабатывается архитектура домена, описывающая семейство систем в пределах домена, включая их общность и изменчивость
5. Специфицируются активы, относящиеся к домену
6. Соответствующие активы приобретаются или разрабатываются и поддерживаются в течение всего жизненного цикла
7. Модели и архитектуры домена поддерживаются в течении всего их жизненного цикла

Стандарт ЖЦ согласно ГОСТ 34

Разработка автоматизированной системы управления (АСУ)

1. Формирование требований
2. Разработка концепции
3. Техническое задание
4. Эскизный проект
5. Технический проект
6. Рабочая документация
7. Ввод в действие
8. Сопровождение

3
Архитектурное представление

Основной целью архитектуры является определение требований, которые влияют на структуру приложения. Хорошо продуманная архитектура снижает бизнес-риски, связанные с созданием технического решения, и устанавливает мост между бизнес- и техническими требованиями.

Архитектурные решения

1. Выбор структурных элементов и их интерфейсов, из которых состоит система
2. Поведение, определенное взаимодействием между этими элементами
3. Объединение этих структурных и поведенческих элементов в большую подсистему
4. Архитектурные решения согласуются с бизнес-целями
5. Архитектурные стили направляют организацию
6. Проектирование программного обеспечения

Механизмы для описания архитектуры

ТОЧКИ ЗРЕНИЯ НА АРХИТЕКТУРУ

ЯЗЫКИ ОПИСАНИЯ АРХИТЕКТУРЫ

АРХИТЕКТУРНЫЕ ФРЕЙМВОРКИ

Примеры точек зрения

1. Функциональная точка зрения
2. Логическая точка зрения
3. Точка зрения информации/данных
4. Точка зрения модуля
5. Точка зрения компонентов и соединителей
6. Точка зрения требований
7. Точка зрения разработчика/внедрения
8. Точка зрения Concurrency/process/runtime/thread/execution
9. Точка зрения производительности
10. Точка зрения безопасности
11. Физическая точка зрения/ точка зрения развертывания/установки
12. Точка зрения действия пользователя/обратной связи

Языки описания архитектуры

1. **ADL** - Architecture description language, язык описания архитектуры

2. **UML** – Unified Modeling Language — унифицированный язык моделирования

3. **ArchiMate** - язык архитектурного описания корпоративных и инженерных систем (моделирования архитектуры предприятия)

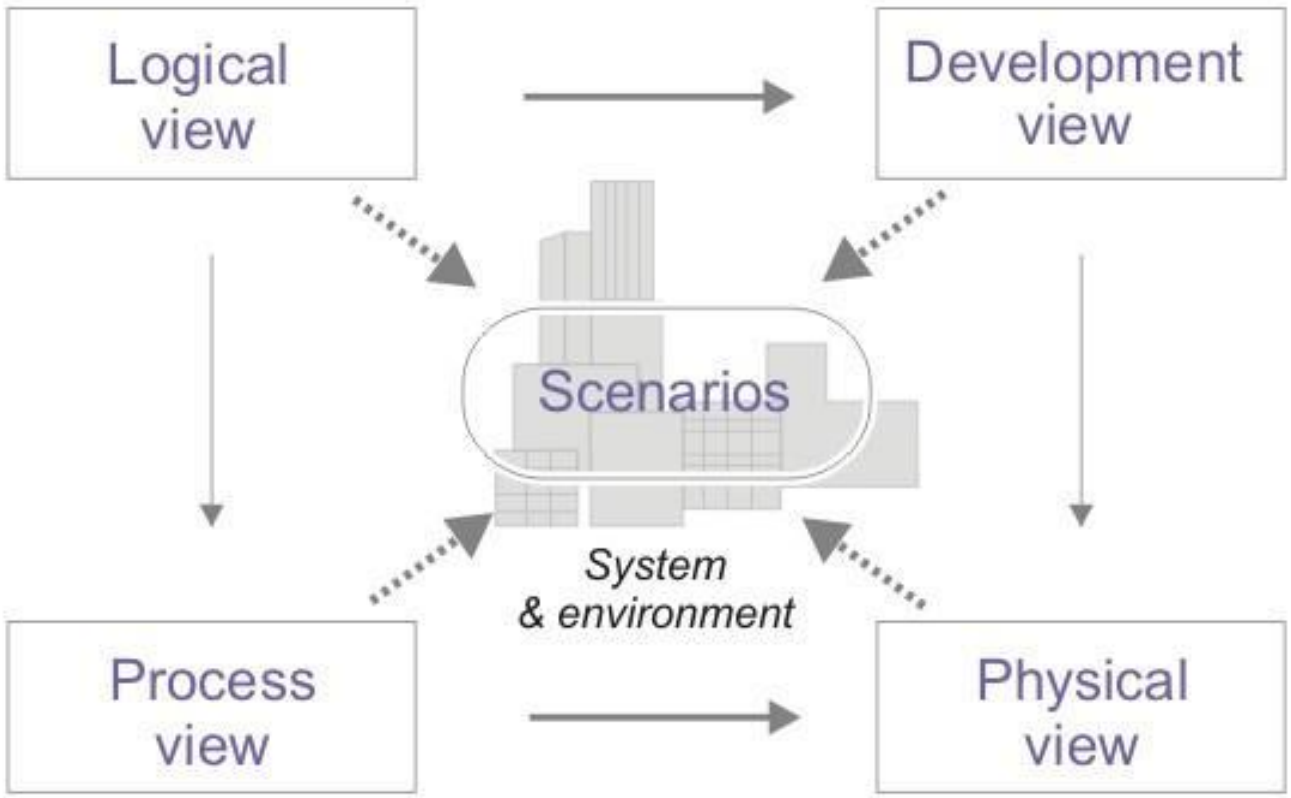
Архитектурные каркасы - фреймворки

- 4+1
- Views and Beyond
- TOGAF – ArchiMate
- ARIS

1. Архитектурная модель 4+1 в нотации UML

Логическое представление

Представление разработчика



Процессное представление

Физическое представление

Варианты использования (сценарии)

Logical view

Логическое представление сфокусировано на функциональности, предоставляемой системой для конечных пользователей. В этом представлении используются UML-диаграммы классов, связей и последовательностей

Development view

Представление разработки показывает систему с точки зрения разработчика и касается управления программой. Это представление также известно как представление реализации. Здесь используются UML-диаграммы компонентов и пакетов для описания компонентов системы и их объединения в логические пакеты (например, слои).

Process view

Представление процесса отражает динамические аспекты системы, показывает происходящие в системе процессы и связи между ними. Представление процесса фокусируется на поведении системы во время выполнения программы. Представление процесса отражает параллелизм выполнения, распределение, интеграцию, производительность, масштабирование и т.п. Представление процесса использует UML-диаграммы активности

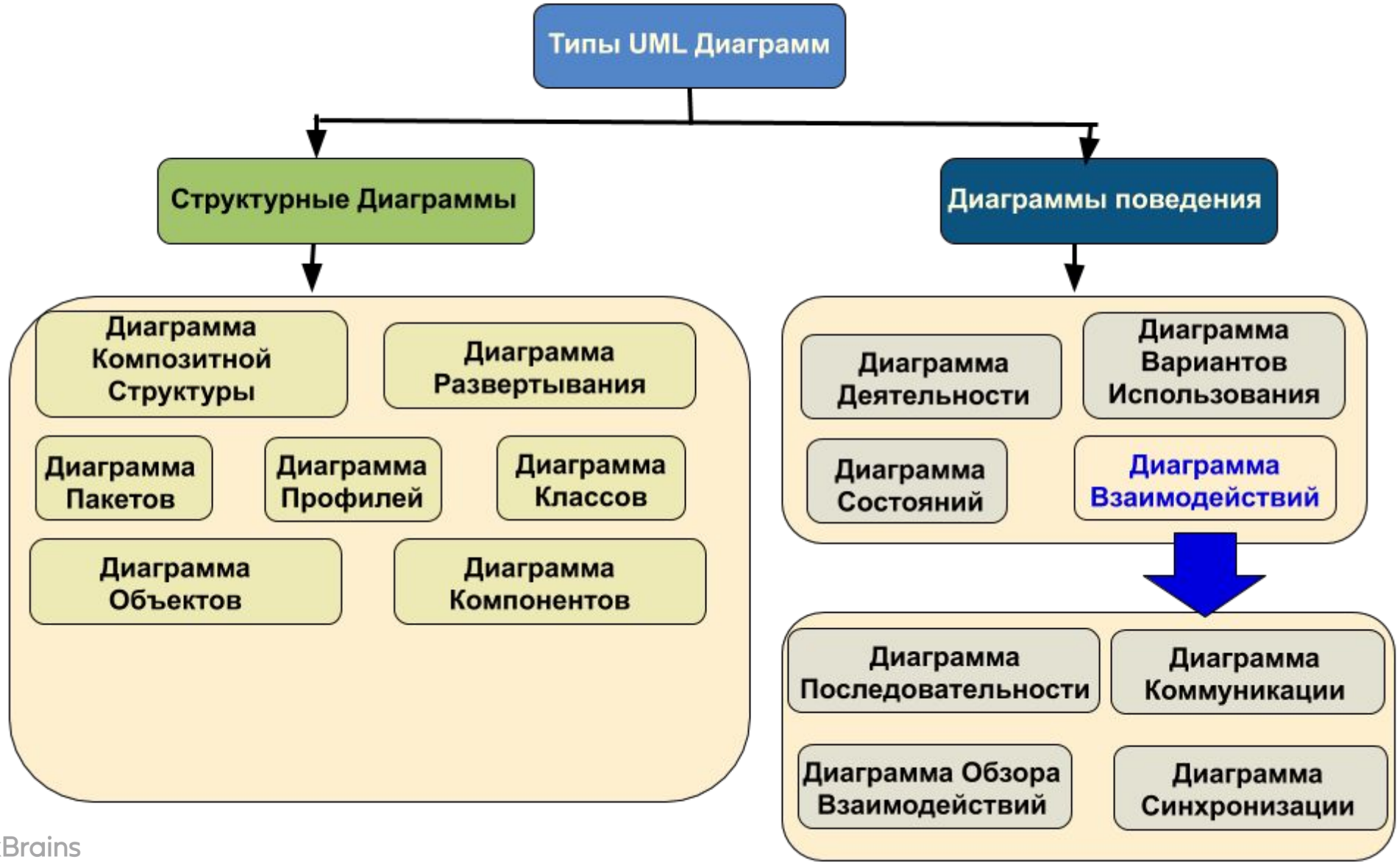
Physical view

Представление физической структуры показывает систему с точки зрения системного инженера. Она показывает распределение программных компонентов по физическим уровням и физические каналы связи между уровнями. Это представление известно также как представление развёртывания системы. Представление физической структуры системы использует UML-диаграмму развёртывания

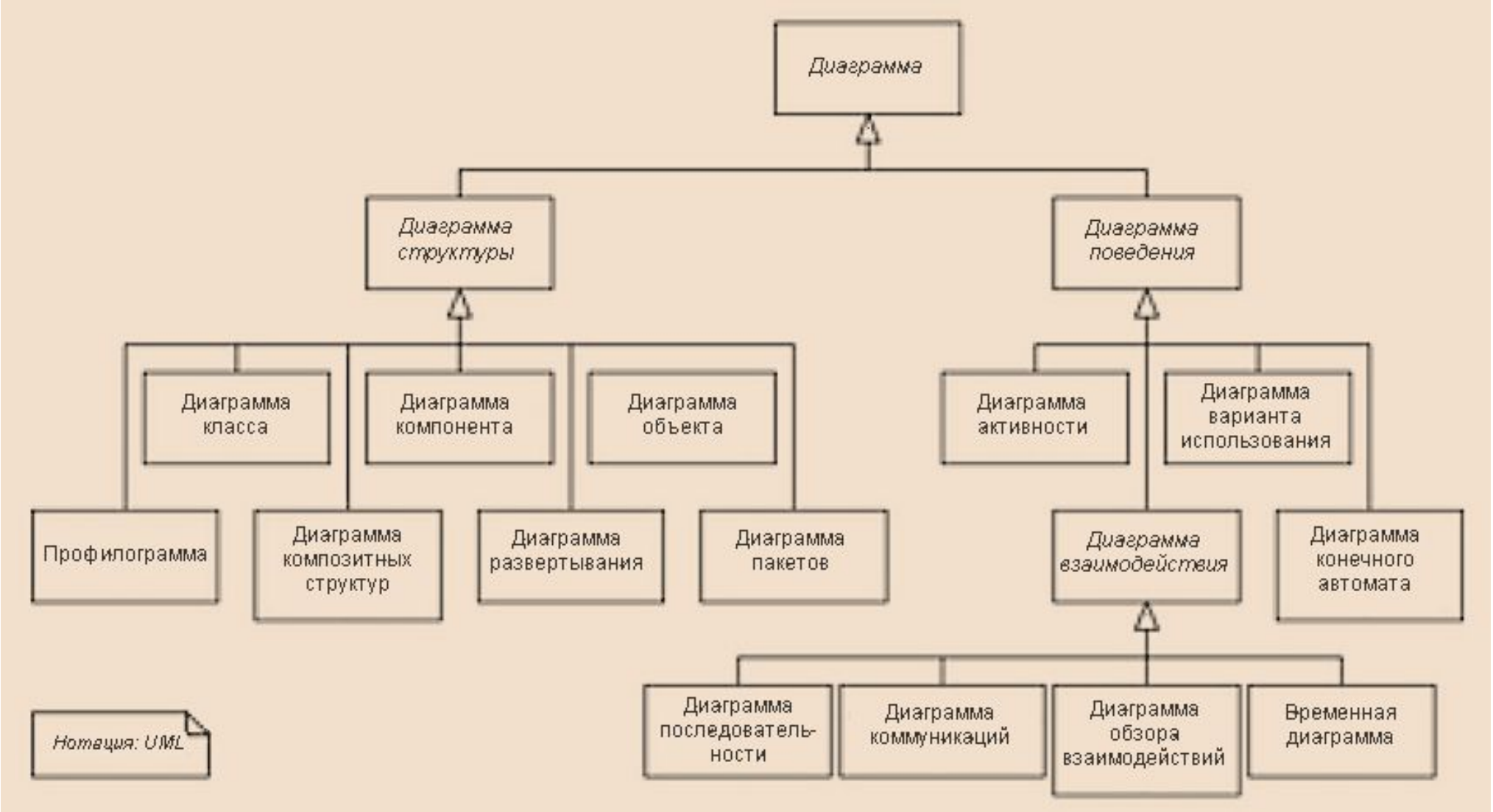
Scenarios

Описание архитектуры системы должно быть проиллюстрировано небольшим набором вариантов использования, или сценариев. Этот набор сценариев составляет пятое представление модели Крачтена. Сценарии должны описывать взаимодействие между объектами и между процессами. Сценарии используются для идентификации архитектурных элементов и для иллюстрирования и проверки качества архитектуры системы. Они также служат отправной точкой для тестирования архитектурного прототипа системы. Сценарии описываются UML-диаграммами вариантов использования.

Декомпозиция UML диаграмм



Декомпозиция UML диаграмм (вариант)



2. Подход Views and Beyond

(любые доступные нотации)

Подход Views and Beyond не только дает рекомендации по созданию документации, но и описывает распространенные архитектурные шаблоны.

В подходе Views and Beyond для архитектурного представления предлагается следующая структура:

1. Основная презентация (main presentation)
2. Каталог элементов и отношений между ними
3. Контекстные диаграммы

В этом архитектурном представлении могут сочетаться несколько архитектурных стилей.

Например, в одном представлении модуля мы можем отразить зависимости использования между различными подмодулями (стиль Use case) вместе с функциональной декомпозицией системы на различные подмодули (стиль Decomposition UML).

Руководство по стилю V&B охватывает следующие стили архитектуры

1. Модульные представления (Module views) - описывает как программная система представлена в виде значимых частей программного кода.
2. Стили модульной архитектуры:
    - стиль декомпозиции
    - стиль использования
    - стиль обобщения
    - многослойный стиль
    - стиль аспектов

Руководство по стилю V&B охватывает следующие стили архитектуры

3. Стили компонентов и коннекторов:
    - стили событий: стиль публикации и подписки, другие
    - стиль потока данных
    - стиль репозитория
4. Стиль вызова и возврата:
    - стиль SOA, клиент-сервер, одноранговый стиль.
    - стили распределения

Руководство по стилю V&B охватывает следующие стили архитектуры

5. Стиль развертывания

6. Стиль распределения работы.

7. Компонентные представления (Component-and-connector views) - показывает программную систему с точки зрения взаимодействующих друг с другом элементов в процессе выполнения.

8. Представления развертывания и распределения (Allocation views) - показывает в программной системе используются аппаратные средства.

3. Представление TOGAF и ArchiMate

TOGAF и ArchiMate имеют свои спецификации, и они могут использоваться по отдельности, независимо друг от друга, или совместно с другими стандартами и фреймворками

TOGAF

обеспечивает метод разработки архитектуры, в состав которого входят разделенный на фазы процесс разработки, руководства и техники

ArchiMate

обеспечивает язык с необходимыми элементами, отношениями и графическими обозначениями для моделирования архитектур

ArchiMate - язык

позволяет создавать как отдельные модели, соответствующие представлениям TOGAF

так и модели, объединяющие различные домены архитектуры

ArchiMate можно позиционировать как средство интегрированного высокоуровневого моделирования и анализа различных доменов предприятия и зависимостей между доменами

Модели BSC
Модели 5 Forces
Модели Value Chain
SWOT-анализ

Модели ArchiMate

Модели BPMN
Модели UML
Модели ERD

Базовые понятия языка

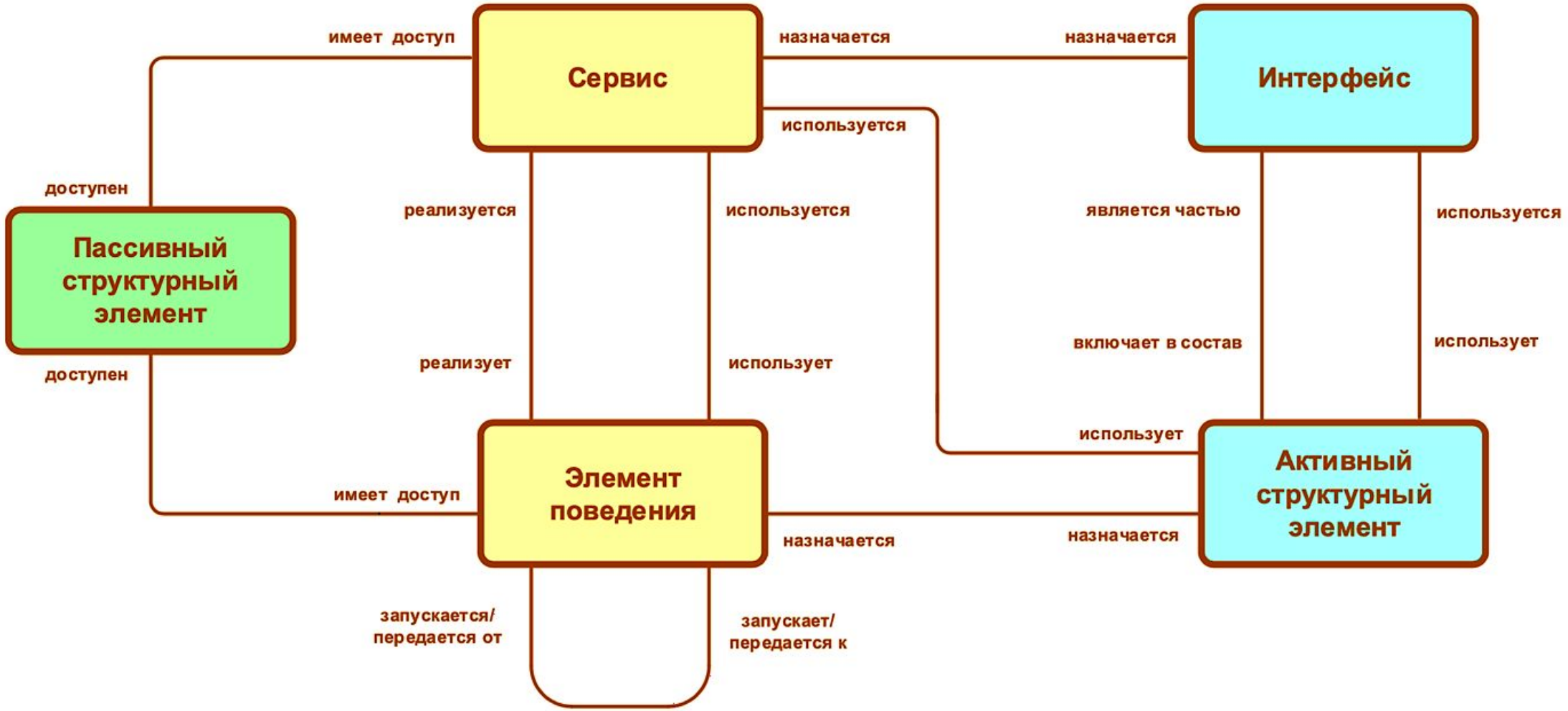
Элемент

всего базовых элементов – 32

Отношение

всего отношений – 12

Обобщенная метамодель - основные понятия языка



Структура языка ArchiMate и разделение элементов напоминают структуру естественных (человеческих) языков, в состав предложений которых входят существительное-подлежащее (субъект), глагол-сказуемое (действие) и существительное- дополнение (объект).

В языке ArchiMate:

- существительное-подлежащее – это активный структурный элемент, то есть субъект поведения;
- глагол-сказуемое – это элемент поведения или действие, то есть выполнение поведения;
- существительное-дополнение - пассивный структурный элемент, то есть объект, на котором или с которым выполняется поведение.

4. Представление архитектуры в методологии ARIS

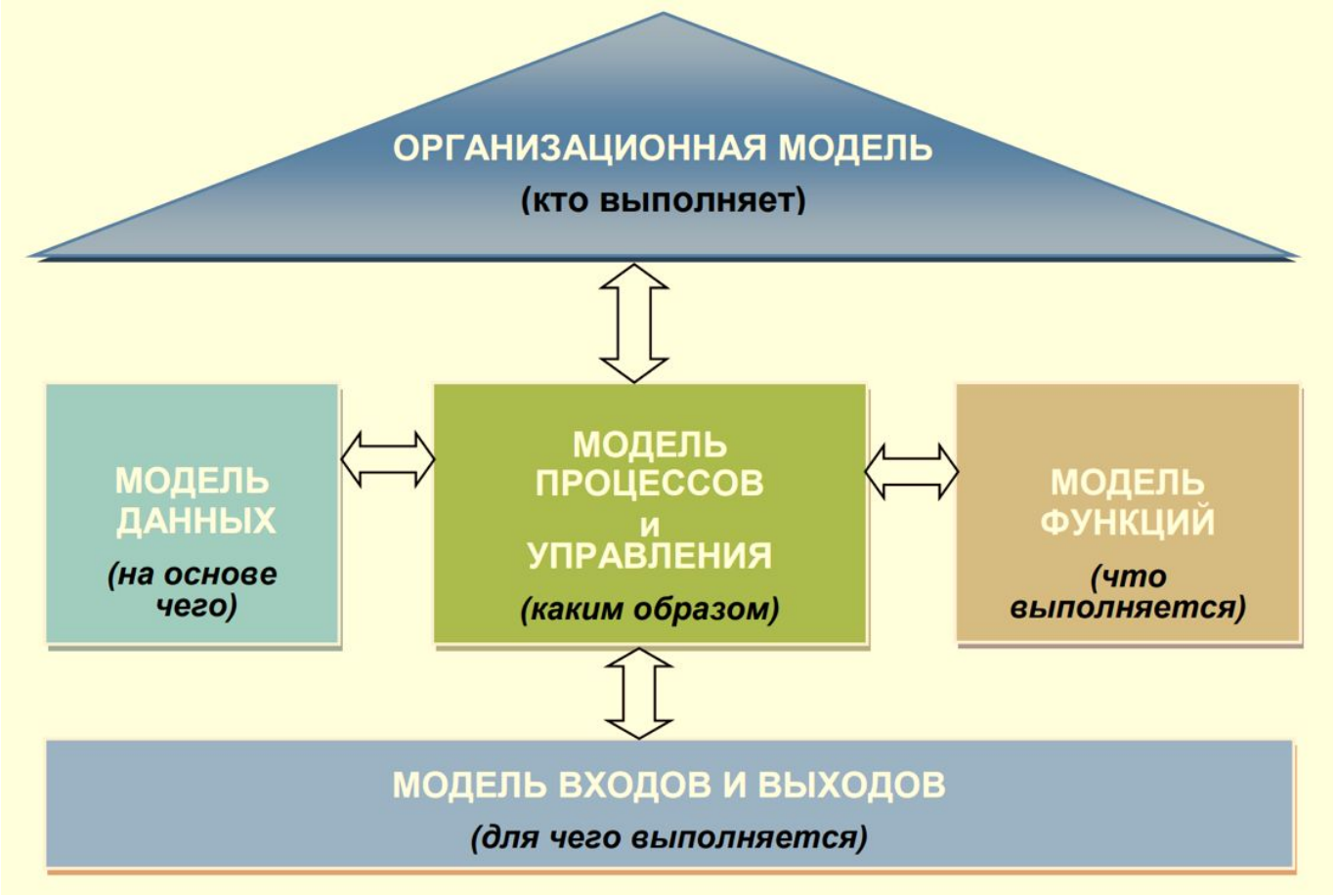
ARIS – это сокращенное английское выражение (Architecture of Integrated Information Systems), что означает: архитектура интегрированных информационных систем.

Под архитектурой подразумевается совокупность технологий, обеспечивающих проектирование, управление, применение и реализацию бизнеса в виде «деловых» процедур бизнес-процессов предприятий и организаций, а также проектирование и создание интегрированных информационных систем поддержки бизнес-процессов.

Основы методологии **ARIS** состоят в том, что любая организация рассматривается и визуально представляется во всех аспектах, т.е. как единая система, описание которой предусматривает четыре различных «взгляда»

1. Организационная структура
2. Данные (потоки и структура)
3. Функции («деревья» функций)
4. Контроль и управление (деловые процессы)

В методологии ARIS выделено пять типов представлений основных моделей, отражающих основные аспекты организации



1. **Организационные модели**, описывающие иерархическую структуру системы, т.е. иерархию организационных подразделений, должностей, полномочий конкретных лиц, многообразие связей между ними, а также территориальную привязку структурных подразделений

2. **Функциональные модели**, описывающие функции (процессы, операции), выполняемые в организации

3. **Информационные модели** (т.е. модели данных), отражающие структуру информации, необходимой для реализации всей совокупности функций системы

4. **Модели процессов или управления**, представляющие комплексный взгляд на реализацию деловых процессов в рамках системы и объединяющие вместе другие модели

5. **Модели входов и выходов**, описывающие потоки материальных и нематериальных входов и выходов, включая потоки денежных средств.

Для построения моделей и проведения структурного анализа в ARIS используют следующие методы и средства визуального описания:

- **DFD (Data Flow Diagrams)** – диаграммы потоков данных для анализа и функционального проектирования моделей систем. Описывают источники и адресаты данных, логические функции, потоки данных и хранилища данных к которым осуществляется доступ

- **STD (State Transition Diagrams)** – диаграммы перехода состояний для проектирования систем реального времени

- **ERD (Entity-Relationship Diagrams)** – диаграммы сущность-связь, описывающие объекты (сущности), свойства этих объектов (атрибуты) и их отношения объектов (связи);

- **SADT (Structured Analysis and Design Technique)** - технология структурного анализа, проектирования и моделирования иерархических многоуровневых модульных систем

- **IDEF0 (Integration Definitionfor Function Modeling)** – подмножество SADT – стандарт описания бизнес-процессов в виде иерархически взаимосвязанных функций

- **IDEF1** – стандарт описания движения информации; используется для определения структуры информационных потоков, правил движения, принципов управления информацией, связей потоков, выявления проблем некачественного информационного менеджмента

• **IDEF1X** – стандарт разработки логических схем баз данных, основанный на концепции сущность-связь

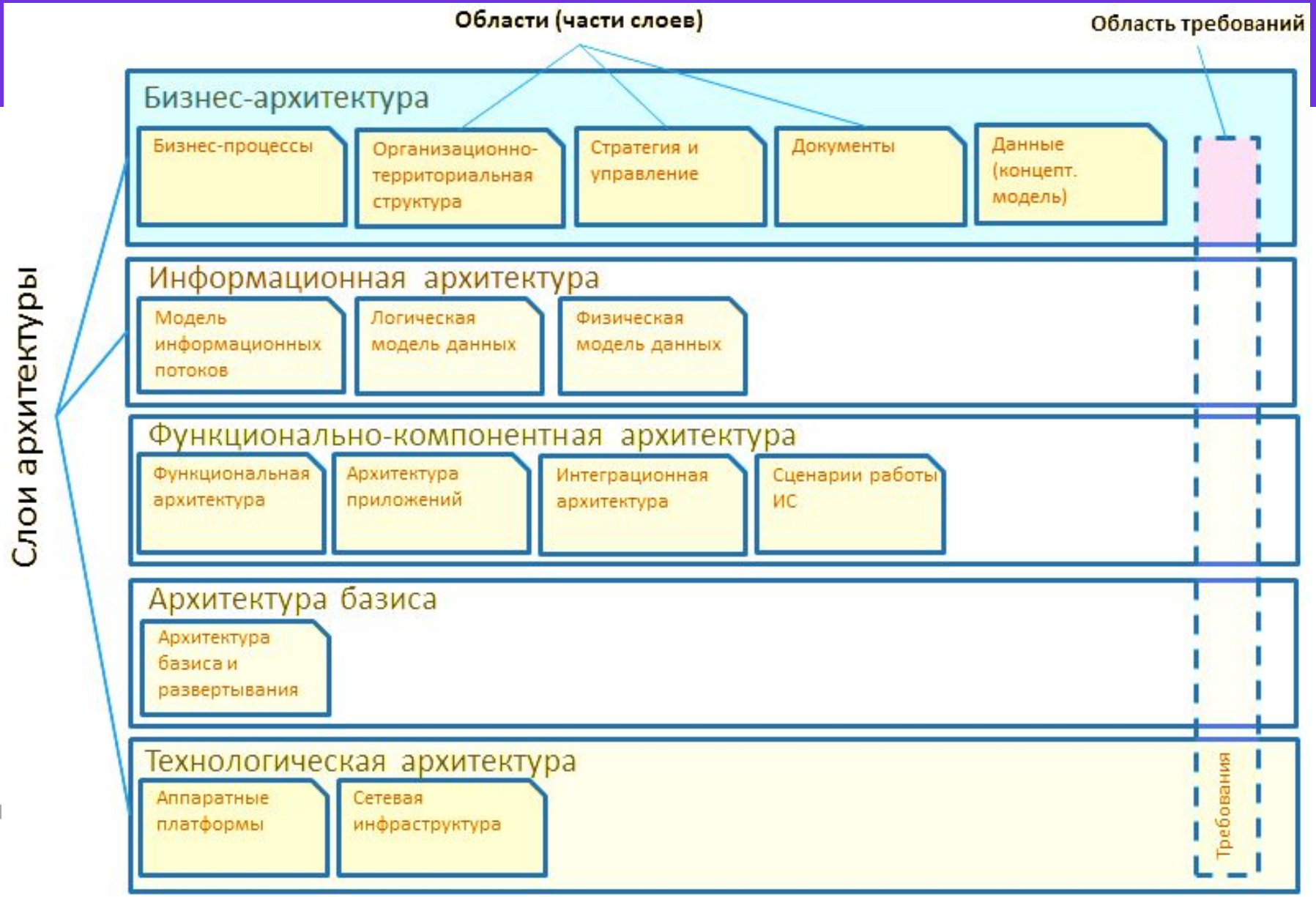
• **IDEF3** – стандарт описания процессов, основанная на сценариях. Сценарий есть описание последовательности изменения свойств объекта в рамках некоторого процесса. Стандарт позволяет описать последовательность этапов изменения свойств объекта (Process Flow Description Diagrams - PFDD) и состояния объекта на этапах (Object State Transition Network - OSTN). Стандарт позволяет решать задачи документирования и оптимизации процессов

• **IDEF4** – стандарт описания структуры объектов и заложенных принципов их взаимодействия; позволяет анализировать и оптимизировать сложные объектноориентированные системы

• **IDEF5** – стандарт, позволяющий описать совокупность терминов, правил комбинирования терминов в утверждения для описания свойств и связей объектов, построить модель на основе этих утверждений. Такие модели позволяют изучать онтологию объектов. Онтология – это знания о совокупности фундаментальных свойств некоторого объекта или области, определяющих их поведение и изменение, собранные для детальной формализации

- **UML (Unified Modeling Language)** – объектно-ориентированный унифицированный язык визуального моделирования. Позволяет описывать диаграммы действий, диаграммы взаимодействия, диаграммы состояний, диаграммы классов и компонент. Используется как для анализа, так и для проектирования моделей информационных систем.

Структура слоев и областей ИТ архитектуры

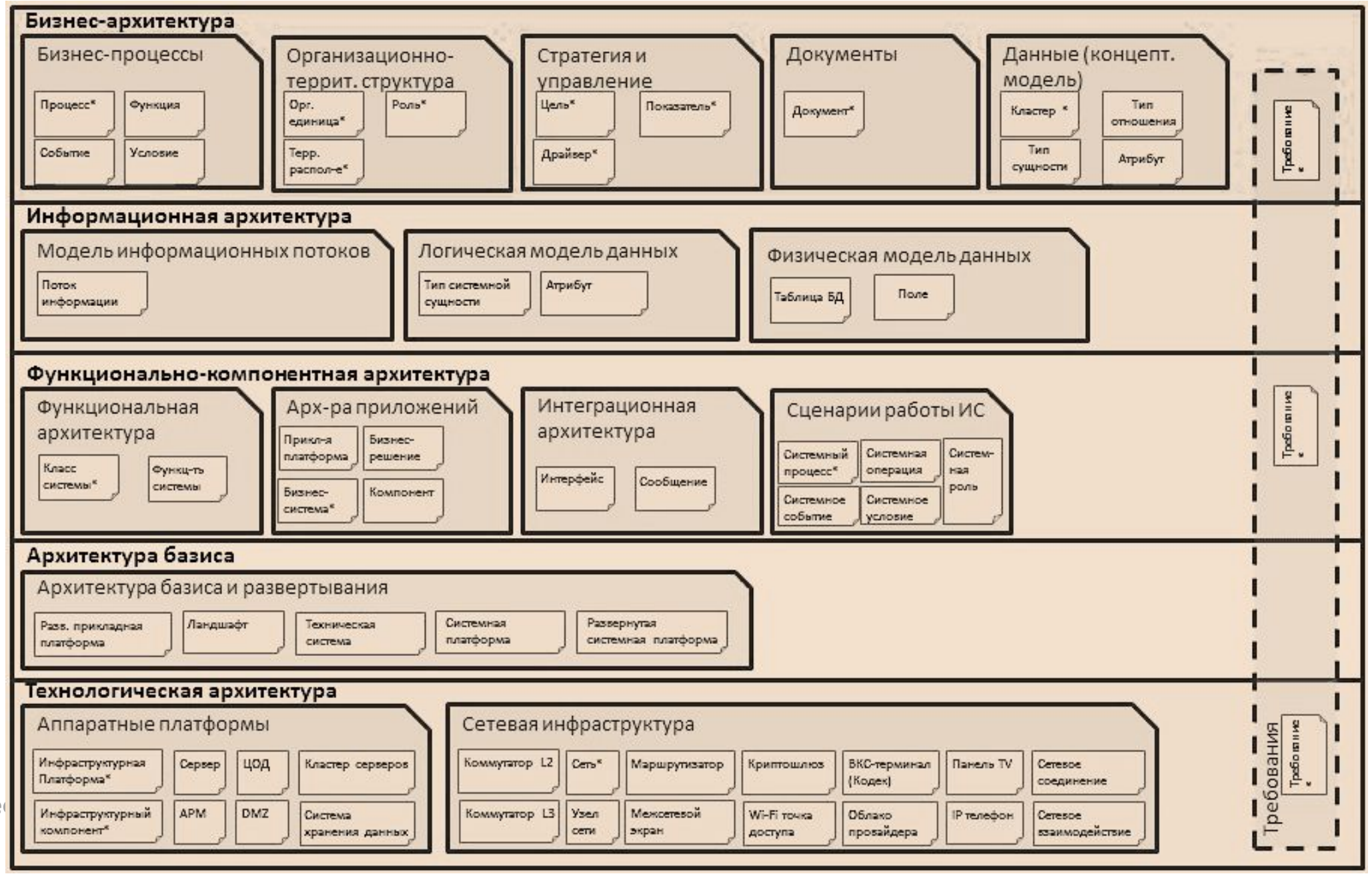


Описание архитектурных слоев

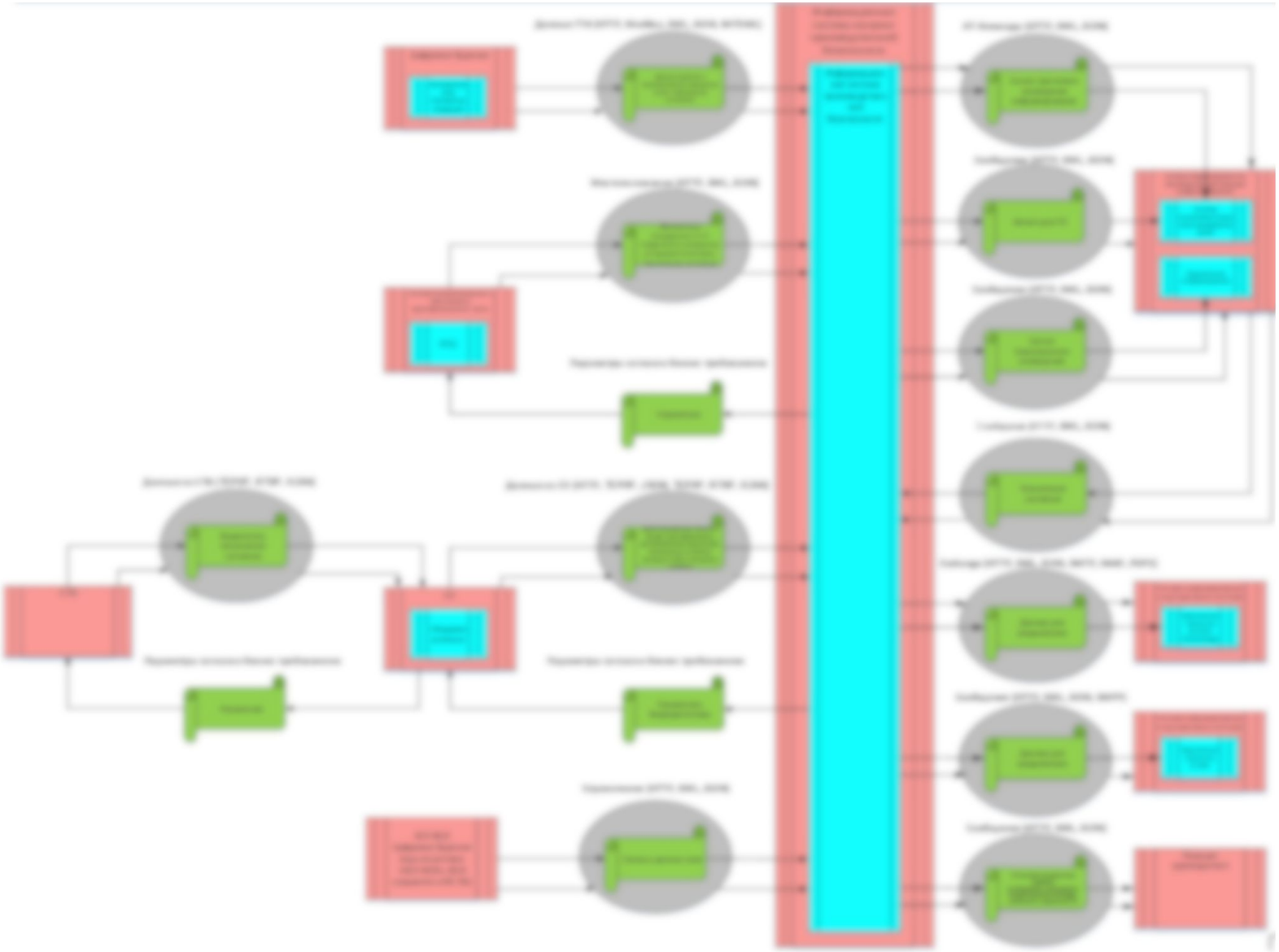
Слой	Описание
Бизнес-архитектура	Модели существенных аспектов организационной, управленческой и процессной деятельности Компании
Информационная архитектура	Модели структуры информации в Компании, информационного взаимодействия и носителей данных
Функционально-компонентная архитектура	Модели структурного и функционального состава информационных систем Компании, а также их интеграции
Архитектура базиса	Модели прикладных и системных платформ и развертывания прикладных компонентов информационных систем
Технологическая архитектура	Модели технического обеспечения информационных систем Компании
Требования	Модели, описывающие требования (бизнес-требования, функциональные и нефункциональные требования)

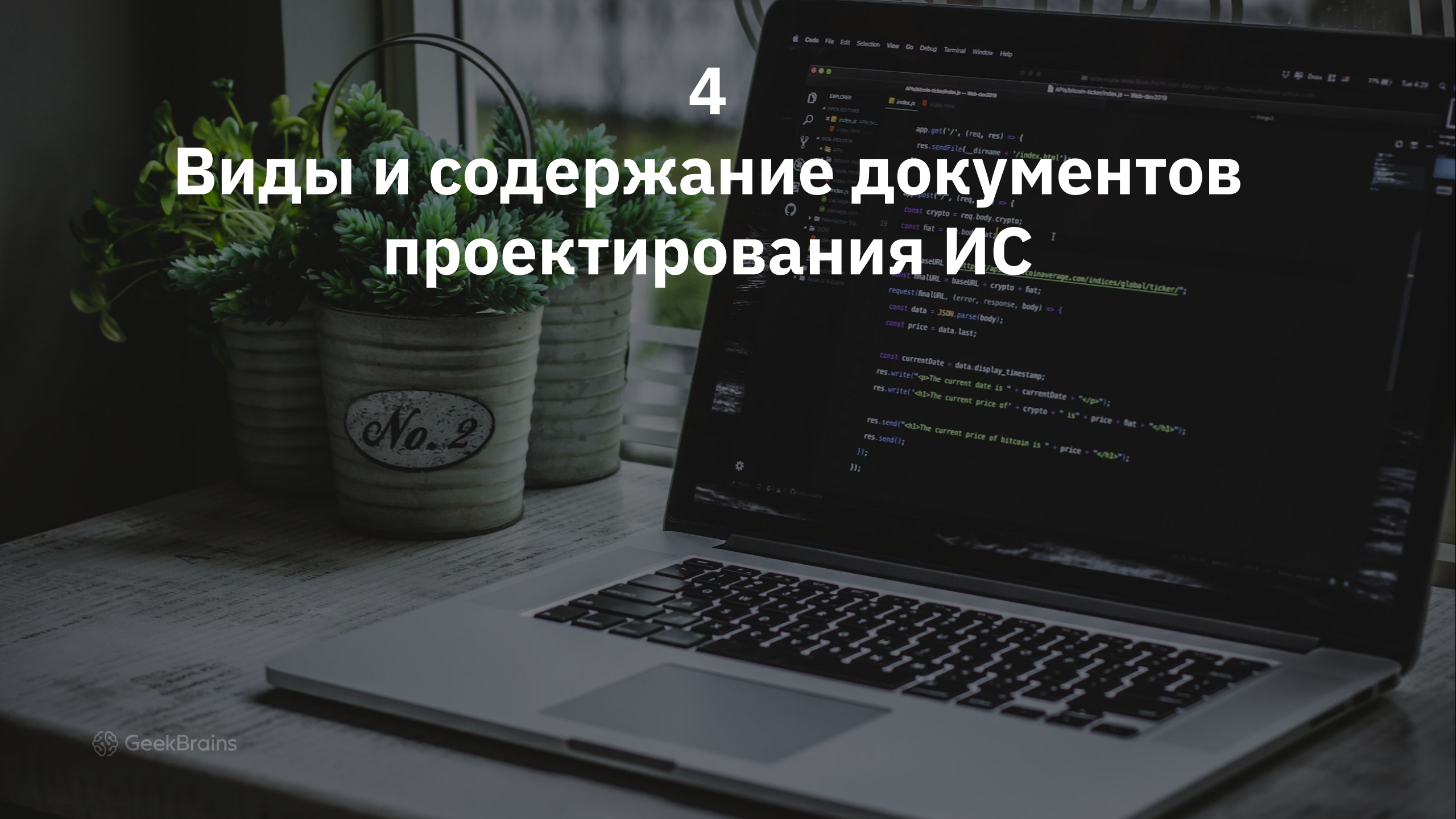
Метамодель (или «модель моделей») служит «словарем», определяющим разновидности объектов, составляющих архитектуру ИС и возможные взаимоотношения (связи) между такими объектами, используемые при моделировании

Пример метамодели архитектуры ИС



Пример документа в нотации ARIS




4
Виды и содержание документов проектирования ИС
GeekBrains

Для формирования необходимого и достаточного покрытия разработки ПО и ИС можно выделить следующий перечень документов:

1. SPMP (Software Project Management Plan)
2. SRS (Software Requirements Specification) – Требования к программному продукту, Техническое Задание, Функциональные-технические требования;
3. SDD (Software Design Document), AD (Architectural Design), SAD: Systematic Architecture Design – эскизный проект, описание программы, etc.
4. SVVP (Software Verification and Validation Plan)
5. SQAP (Software Quality Assurance Plan)
6. STD (Software Test Documentation) – Программа и методики тестирования
7. SCMP (Software Configuration Management Plan)
8. Source code

Software Requirement Specification — SRS

Требования к программному обеспечению могут быть задокументированы в техническом задании или документе Software Requirements Specification.

SRS — специальная документация для ПО которая содержит в себе информацию о том, как должна себя вести система, какие функции должна выполнять, какую нагрузку должна выдерживать и т.д.

Рекомендуемая стандартом IEEE 830 структура SRS:

Введение

- Цели
- Соглашения о терминах
- Предполагаемая аудитория и последовательность восприятия
- Масштаб проекта
- Ссылки на источники

Рекомендуемая стандартом IEEE 830 структура SRS:

Общее описание

- Видение продукта
- Функциональность продукта
- Классы и характеристики пользователей
- Среда функционирования продукта (операционная среда)
- Рамки, ограничения, правила и стандарты
- Документация для пользователей
- Допущения и зависимости

Рекомендуемая стандартом IEEE 830 структура SRS:

Функциональность системы

- Функциональный блок X (таких блоков может быть несколько)
- Описание и приоритет
- Причинно-следственные связи, алгоритмы (движение процессов, workflows)
- Функциональные требования

Рекомендуемая стандартом IEEE 830 структура SRS:

Требования к внешним интерфейсам

- Интерфейсы пользователя (UX)
- Программные интерфейсы
- Интерфейсы оборудования
- Интерфейсы связи и коммуникации

Рекомендуемая стандартом IEEE 830 структура SRS:

Нефункциональные требования

- Требования к производительности
- Требования к сохранности (данных)
- Критерии качества программного обеспечения
- Требования к безопасности системы

Рекомендуемая стандартом IEEE 830 структура SRS:

Прочие требования

- Приложение А: Глоссарий
- Приложение Б: Модели процессов и предметной области и другие диаграммы
- Приложение В: Список ключевых задач

Рекомендуемая стандартом IEEE 830 структура SRS:

Введение

- Цели
- Соглашения о терминах
- Предполагаемая аудитория и последовательность восприятия
- Масштаб проекта
- Ссылки на источники

Software design description – SDD

Описание разработки программного обеспечения (так называемый дизайн программного документа или SDD , просто дизайн документ , также дизайн Спецификация программного обеспечения) является представлением разработки программного обеспечения, которое будет использоваться для записи информации о конструкции, решении различных проблем проектирования, и передавать эту информацию в дизайнах заинтересованным сторонам.

SDD обычно сопровождает диаграмму архитектуры с указателями на подробные характеристики более мелких частей проекта.

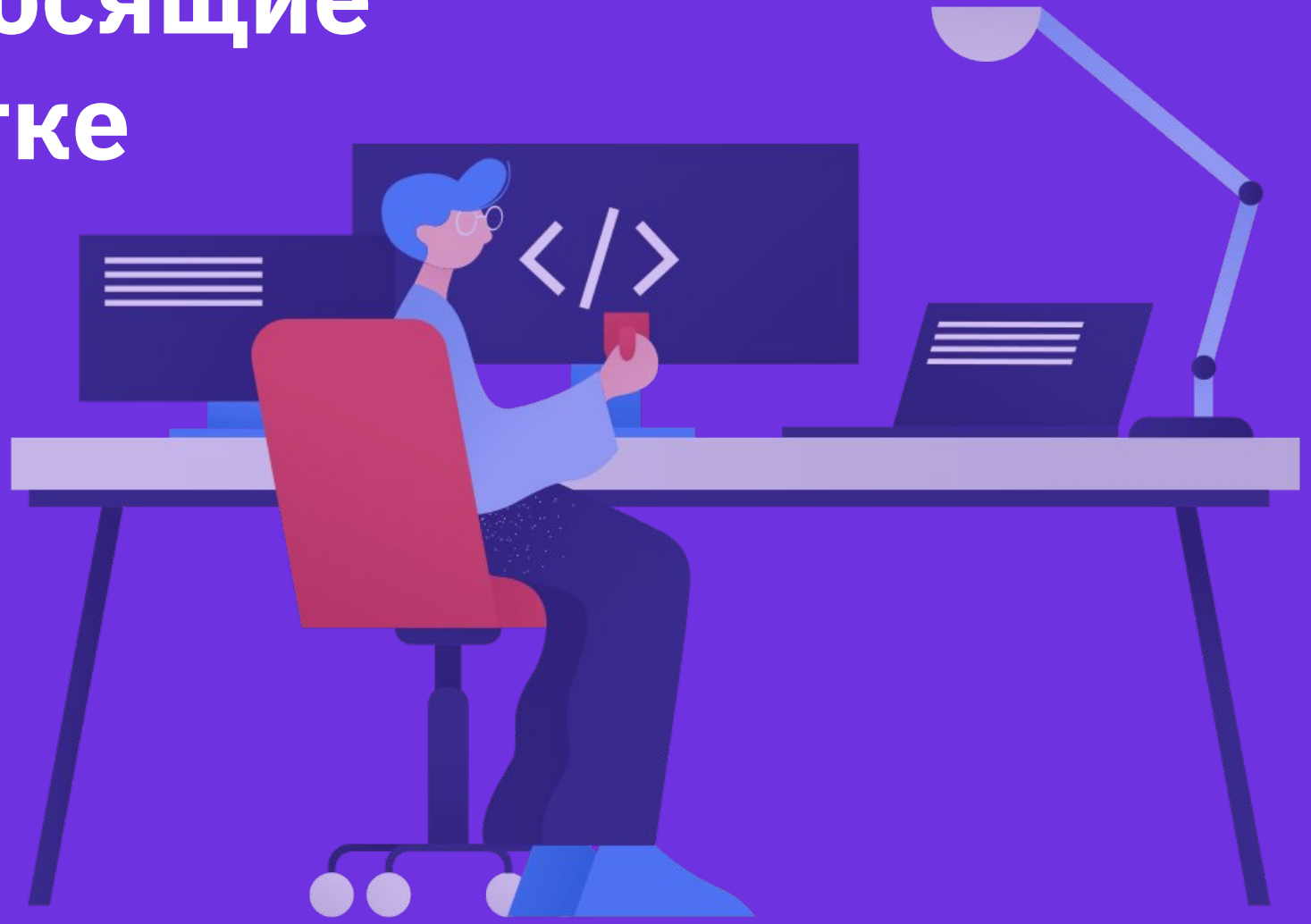
SDD обычно содержит следующую информацию:

1. Дизайн данных описывает структуры, которые находятся в программном обеспечении.
2. Атрибуты и отношения между объектами данных диктуют выбор структур данных . В дизайне архитектуры использует информацию , протекающие характеристики, и отображают их в структуру программы.
3. Метод преобразования преобразования применяется для демонстрации четких границ между входящими и исходящими данными.

SDD - это живой документ, и в любой момент времени он может не содержать всей перечисленной здесь информации:

1. Контекст
2. Функциональный вид
3. Представление процесса
4. Нефункциональный вид
5. Ограничения
6. Принципы
7. Логический взгляд
8. Интерфейсный взгляд
9. Вид дизайна
10. Вид инфраструктуры
11. Представление развертывания
12. Операционный вид
13. Представление безопасности
14. Представление данных
15. Выбор технологии
16. Обоснование архитектуры

5
Стандарты относящие к разработке ПО и ИС



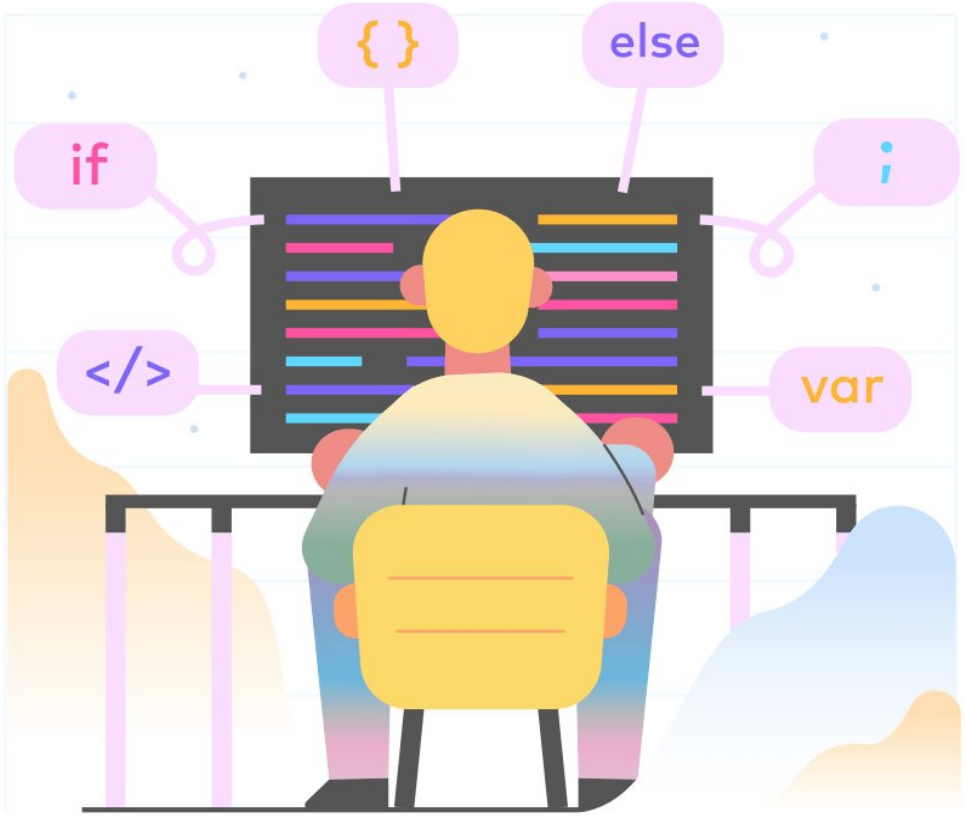
GeekBrains

1. Международные стандарты
2. Международные стандарты адаптированные под ГОСТ
3. ГОСТ Общего назначения
4. Стандарты АСУ ГОСТ 34
5. Стандарты ЕСПД ГОСТ 19
6. Международные стандарты для документации

Практическое задание

Собрать комплект архитектурной документациипроекта (рабочий, либо личный проект):

- Указать фазу жизненного цикла ИС или ПО
- Вид и назначение документа, нотация
- Указать кто потребитель данных документов



Спасибо!

Каждый день вы становитесь лучше :)