«Модельно-ориентированная системная и программная инженерия (MBSE)» как базовый курс подготовки ИТ-специалистов и презентация учебника по данному курсу

28.01.2025

В.А. Сухомлин

Проф. МГУ им. Ломоносова sukhomlin@mail.ru

О системной инженерии (SE)

- Системы естественной и искусственной природы служат строительными блоками современного мира и бытия человека
- Поэтому изучение научных и прикладных аспектов, связанных с системами, а также процессами, методами и средствами, необходимыми для их создания и использования, имеют решающее значение для функционирования современного мира
- Значительную часть систем составляют системы, частично или полностью созданные человеком. Такие системы называются **инженерными**
- Научно-прикладная дисциплина, предназначенная для изучения инженерных систем, а также процессов, методов, средств их создания и использования называется **системной инженерией** (**system engineering – SE**)
- Область SE является многодисциплинарной и характеризуется быстрыми темпами развития, интеграцией с другими технологиями, в частности, такими, как: онтологический инжиниринг и искусственный интеллект
- Международный совет по системной инженерии (**INCOSE**) определяет **системную инженерию** (SE) как междисциплинарный подход и средства, позволяющие создавать успешные системы

О системной инженерии (SE)

Системную инженерию, которая сложилась до конца XX столетия, часто называют традиционной или **системной инженерией, основанной на документах (Document-Based Systems Engineering, или DBSE)**, ввиду того, что основными артефактами, сопровождающими систему на протяжении ее жизни, являлись документы на бумажных и/или машинных носителях.

С конца 1990-х специалисты по системному проектированию стали широко применять методы, основанные на моделях, используемых для облегчения коммуникации, управления сложностью дизайна, улучшения качества продукта, улучшения сбора и повторного использования знаний. Такой подход получил название **модельно-ориентированной системной инженерии (Model-Based Systems Engineering, или MBSE)**.

MBSE определяется как формализованное приложение графического моделирования с точными семантическими определениями для анализа операций, определения требований, разработки системного дизайна, верификации и испытаний, начиная с концептуальной фазы и продолжая на более поздних стадиях/этапах жизненного цикла [3].

Книга содержит 13 глав и 5 приложений

В.А. СУХОМЛИН, В.Ю. РОМАНОВ, Д.А. ГАПАНОВИЧ «ВВЕДЕНИЕ В МОДЕЛЬНО-ОРИЕНТИРОВАННУЮ СИСТЕМНУЮ И ПРОГРАММНУЮ ИНЖЕНЕРИЮ (MBSE)»

Содержание глав по темам следующее:

- **Глава 1. Введение в SE. Основные понятия**
- **Глава 2. Модели жизненного цикла систем**
- **Глава 3. Модели итеративного процесса разработки программно-го обеспечения**
- **Глава 4. Характерные черты и возможности языка UML**
- **Глава 5. Язык моделирования SysML и его применение для разработки моделей систем**
- **Глава 6. Инженерия требований**
- **Глава 7. MBSE — системная инженерия на основе моделей**
- **Глава 8. Описание архитектуры системы**
- **Глава 9. Цифровые двойники**
- **Глава 10. Система стандартов SE. Процессные стандарты**
- **Глава 11. Эталонная модель модельно-ориентированной системной и программной инженерии (MBSSE) и ее связь с процессными стандартами системной инженерии**
- **Глава 12. Интеграция системы и программного обеспечения. Методические аспекты**
- **Глава 13. Математические основы системной инженерии**

Глава 1. Введение в SE. Основные понятия

1.1. История и настоящее системной инженерии

1.2. Определение и область применения

1.3. Пользователи системной инженерии, область деятельности системных инженеров

1.4. Методологические основы системной инженерии. Системный подход и системное мышление. Принципы SE

1.5. Определение и классификация систем: концептуальная модель понятия «система»

1.6. Жизненный цикл систем и процессы жизненного цикла

1.7. Итеративность стадий и процессов жизненного цикла

1.1. История и настоящее системной инженерии

Системный подход (**А.А. Богданов** «Тектология. Всеобщая организационная наука» [1], первый том которой вышел в 1912 г., общая теория систем **Л. фон Берталанфи**, кибернетика и У. Эшби, ...

Одним из пионеров системной инженерии считается американский инженер **Джордж Бокс**, который в 1950-х годах предложил концепцию «системного подхода» к решению проблем, связанных со сложными инженерными системами

В 1960-х годах в рамках программы космических исследований были сформулированы основные принципы и методы системной инженерии, такие как анализ требований, анализ функциональности систем, архитектурное проектирование, верификация, испытания и т. д. Также были разработаны первые математические модели и инструменты для анализа и оптимизации систем

Большую известность приобрели работы по кибернетике Н. Винера (Weiner, 1948) [2], системной динамике (Forrester, 1961) [3], математическим основам системной инженерии Wymore, A. 1967 [4], общей теории систем (Bertalanffy, 1968) [5], математической теории системной инженерии (W.A. Wymore, 1977) [6] и др.

Именно с именем **Уэйна Уаймора (W.A. Wymore)**, связывают одну из первых попыток формального определения модели системы с использованием математического аппарата, изложенной в [6]

1.2. Определение и область применения SE

Как видно из рисунка, применение методов SE охватывает различные этапы жизненного цикла систем, включая:

- изучение операционных возможностей
- инженерия требований
- определение архитектуры
- синтез
- тестирование
- испытания
- развитие решений при рассмотрении проблемы в целом от исследования концепции системы до ее утилизации

Рисунок 1. Взаимосвязь трех дисциплин - **Системной инженерии**, **реализации систем** (system implementation), и **Управления проектами/системами** (project/systems management): https://sebokwiki.org/d/images/sebokwiki-farm!d/1/1a/Scope_BoundariesSE_PM_SM.png .

1.2. Определение и область применения SE

- SE взаимосвязана с такими дисциплинами как:

- **System Implementation -** внедрение/реализация систем и
- **Project/systems Management -** управление проектами/системами

  эта связь иллюстрируется с помощью диаграммы Венна (Рис. 1)

- аспекты производства программного обеспечения системы не входят в SE, а включены в область реализации системы (system implementation). Таким образом, разработка программного обеспечения не считается подмножеством системной инженерии
- С конца 1990-х специалисты по системному проектированию стали широко применять методы, основанные на моделях, используемых для облегчения коммуникации, управления сложностью дизайна, улучшения качества продукта, улучшения сбора и повторного использования знаний. Такой подход получил название **модельно-ориентированной системной инженерии** (Model-Based Systems Engineering, или **MBSE**)
- В последние годы наметилась тенденция более тесной интеграции системной инженерии с программной инженерией, что отражено в международном стандарте **ISO/IEC/IEEE 24641:2023-2023 — «Системная и программная инженерия. Методы и инструменты для модельно-ориентированной системной и программной инженерии»**, в котором представлены фундаментальные методологические решения в этой области

О системной инженерии (SE)

- Суть MBSE заключается в создании машиночитаемых моделей, представляющих все аспекты системы и поддерживающих все действия по проектированию, разработке, производству и эксплуатации системы на протяжении всего ее жизненного цикла
- Эти цифровые модели образуют **цифровой двойник** (Digital Twin — DT) целевой системы, основанный на общих схемах данных, формирующих **цифровой поток**, который объединял бы все заинтересованные стороны, участвующие в создании или приобретении новых систем
- MBSE предоставляет возможность **консолидировать информацию в доступном централизованном источнике**, обеспечивая частичную или полную автоматизацию многих процессов системного проектирования и облегчая интерактивное представление компонентов и поведения систем
- В SE основное внимание уделяется: целостному и одновременному пониманию потребностей заинтересованных сторон (в частности, заказчиков), изучению возможностей систем и требований к системам, их синтезу, проверке, испытаниям и разработке решений при рассмотрении проблемы в целом, от исследования концепции систем до их утилизации

1.3. Пользователи системной инженерии, область деятельности системных инженеров

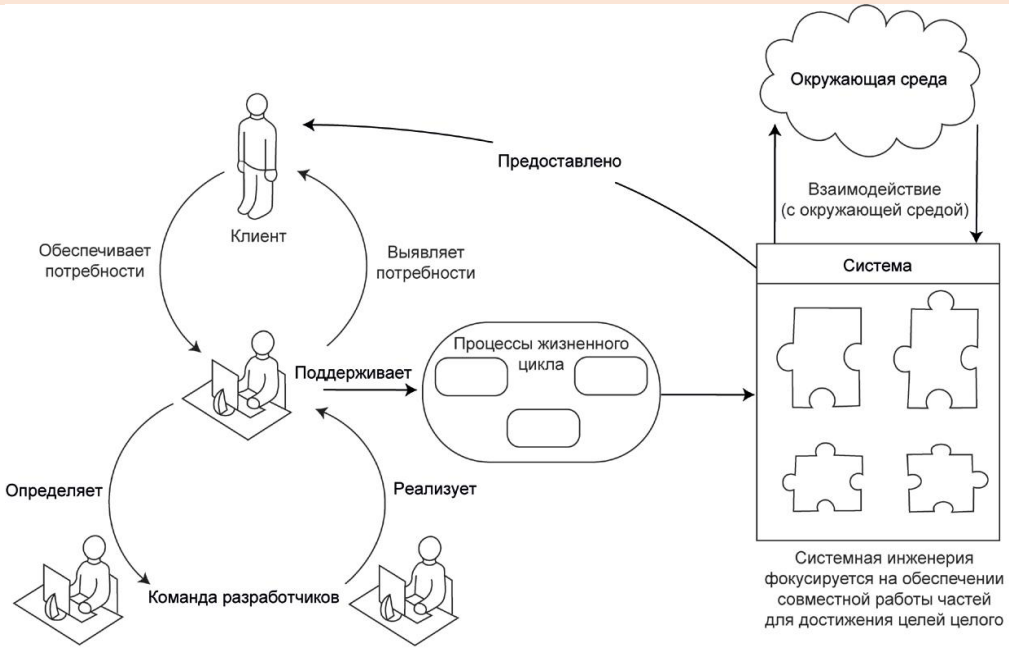


Рисунок 2. **Роль системного инженера** [https://sebokwiki.org/d/images/sebokwiki-farm!d/b/bd/SE_Key_Concepts.jpeg]

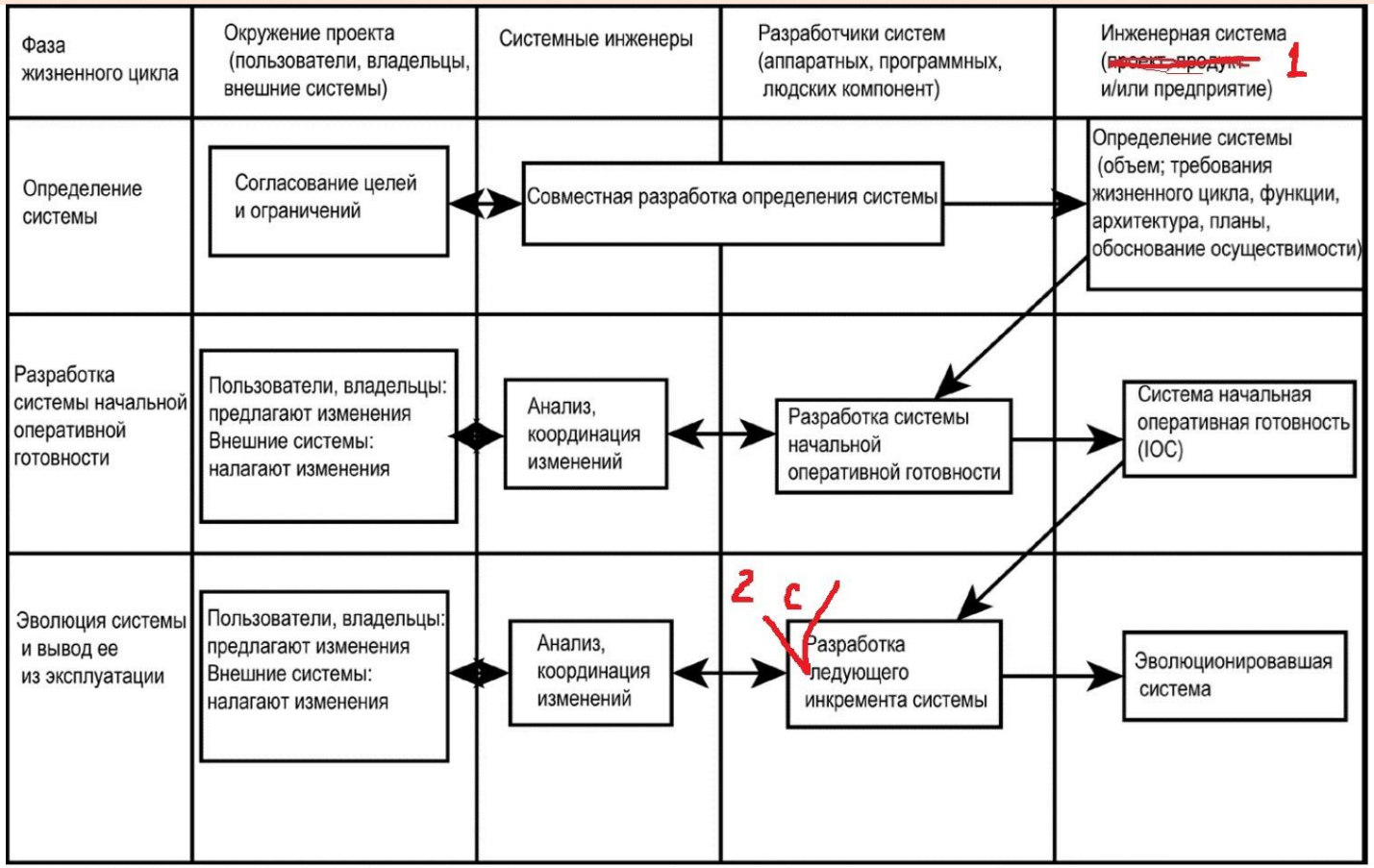


Рисунок 1.3. Контекст жизненного цикла проекта SE и инженерной системы и связанные с ними агенты, действия и артефакты. (SEBoK Original) [1]

1.4. Методологические основы SE. Системный подход и системное мышление. Принципы SE

- **системный подход и главные принципами системного подхода: А.А. Богданов, Л. фон Берталанфи,** (предложенные Берталанфи основные понятия и принципы, такие как целостность, централизация, дифференциация, ведущая часть системы, закрытая и открытая системы, финальность, эквифинальность, рост во времени, относительный рост, конкуренция идр., легли в основу системного подхода**...)**

- **системное мышление и принципы системного мышления -** мыслительная деятельность, построенная на использовании принципов системного подхода (Абстракция, Граница, Изменения, Дуализм, Инкапсуляция, Эквифинальность, Холизм или целостность, Взаимодействие, Иерархия слоев, Модульность, Сетевая организация, Синтез, Представление и т.д.

- **принципы системной инженерии (**SEPAT (The INCOSE Systems Engineering Principles Action Team)) в виде рекомендаций или руководящих указаний по реализации различных видов деятельности и процессов SE

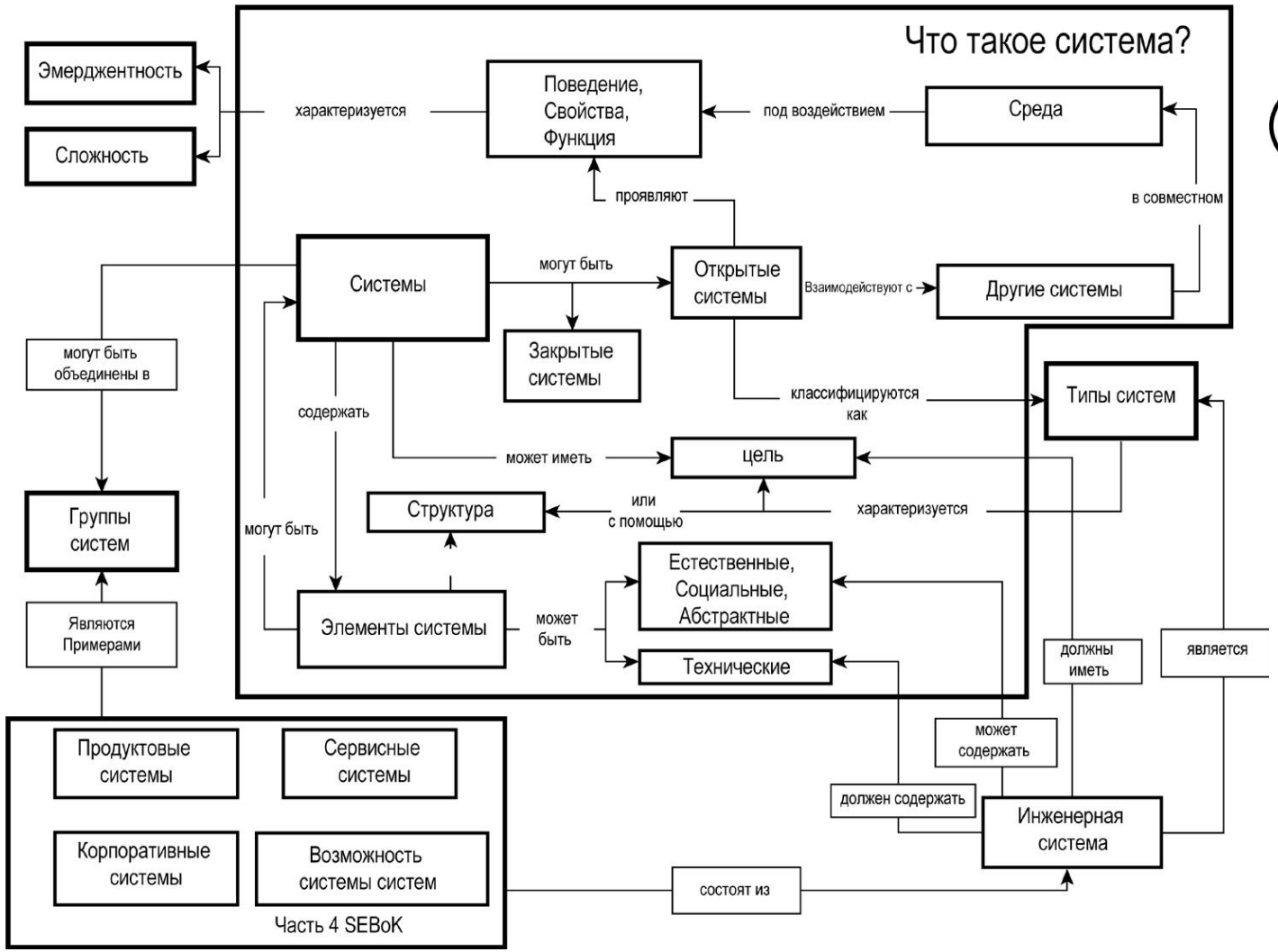
1. SE при использовании определяется потребностями заинтересованных сторон, пространством решений, результирующими системными решениями и контекстом на протяжении всего жизненного цикла системы.
2. SE имеет целостное системное представление, которое включает системные элементы и взаимодействия между ними, обеспечивающие системы и системную среду.
3. SE влияет и находится под влиянием внутренних и внешних ресурсов, а также политических, экономических, социальных, технологических, экологических и правовых факторов.
4. И политика, и закон должны быть правильно поняты, чтобы не ограничивать чрезмерно или недостаточно ограничивать внедрение системы.
5. Реальная система — это совершенное представление системы (модели являются только приближенными представлениями реальных систем)..

В центре внимания SE находится все более глубокое понимание взаимодействий, чувствительности, поведения системы и ее операционной среды, потребностей заинтересованных сторон.

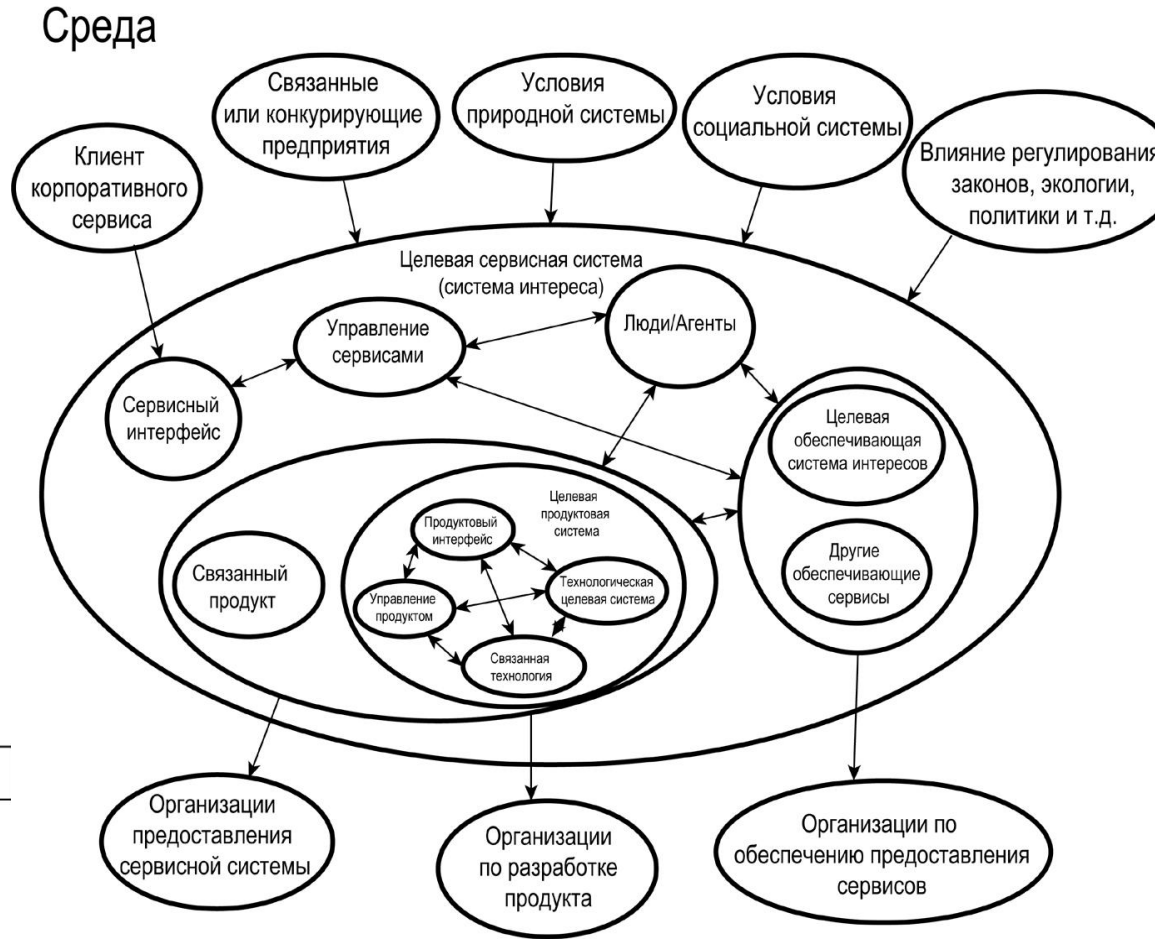
7. SE удовлетворяет изменяющиеся потребности заинтересованных сторон в течение жизненного цикла системы.
8. SE удовлетворяет потребности заинтересованных сторон, принимая во внимание бюджет, график и технические потребности, а также другие ожидания и ограничения.
9. Решения SE принимаются в условиях неопределенности с учетом риска.
10. Качество решения зависит от знаний о системе, обеспечиваю-щих системах и взаимодействующих системах, присутствующих в процессе принятия решения.

1.5. Определение и классификация систем: концептуальная модель понятия «система»

Концептуальная модель. Определения и классификация систем



Определения системного контекста

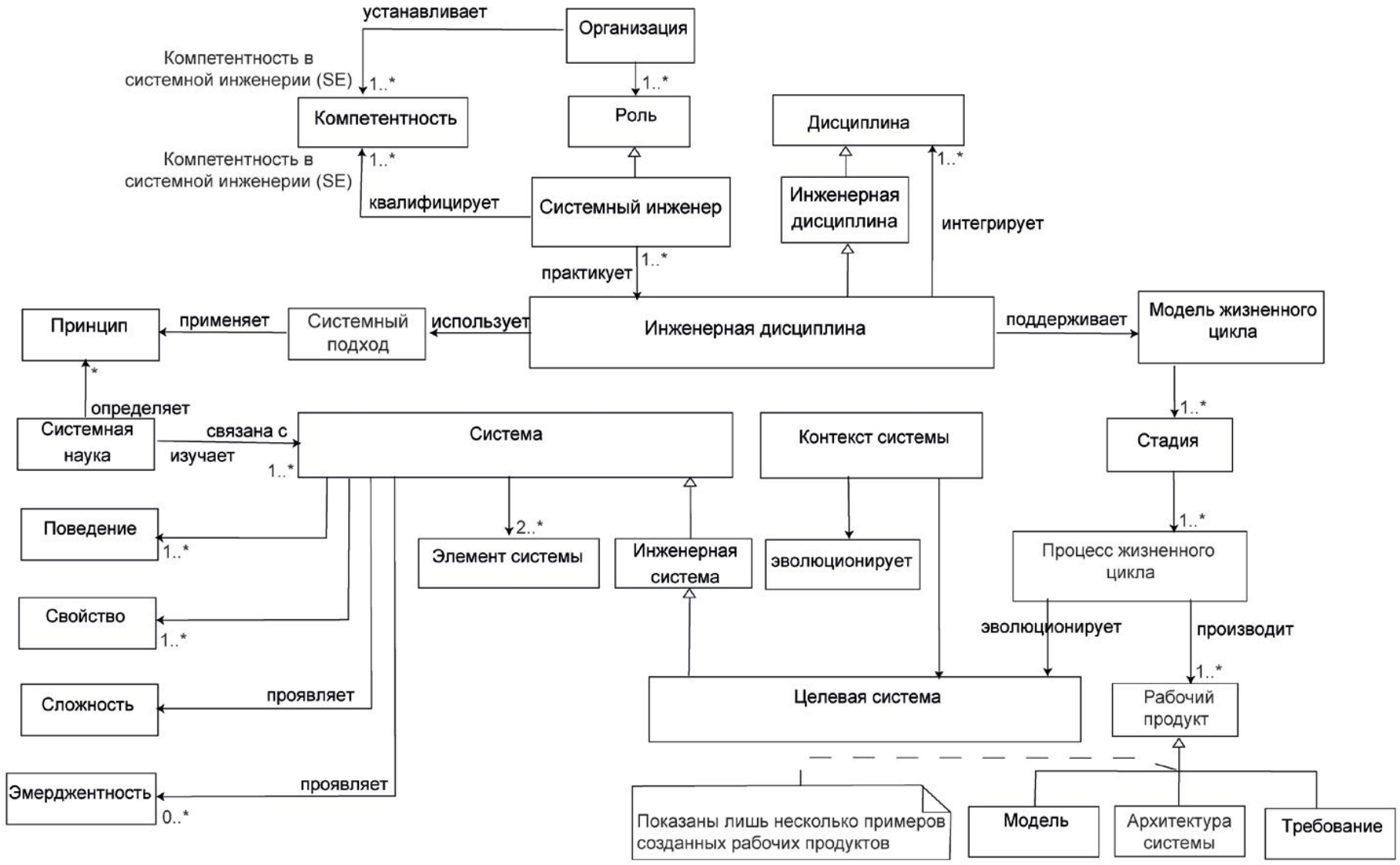


Классификация SOI

Рассмотрен общий случая целевой системы (SoI), которая на протяжении жизненного цикла взаимодействует с системами следующих видов:

- **Технологически ориентированная продуктовая система** SoI, встроенная в один или несколько интегрированных продуктов
- **Интегрированная мультитехнологическая продуктовая система** SoI, используемая непосредственно для предоставления услуги
- **Обеспечивающая сервисная система** SoI, поддерживающая несколько систем обслуживания
- **Сервисная система SoI**, созданная и поддерживаемая для непосредственного предоставления возможностей
- **Система предприятия** (Enterprise Systems)
- **Система систем (System of systems - SOS)** - термин SoS является дополнением к общей концепции системной иерархии, применимой ко всем системам. Поскольку технология, позволяющая интегрировать независимые системы, концепция SOS становится общим аспектом многих жизненных циклов SE

1.6. Жизненный цикл систем и процессы жизненного цикла



Концептуальная модель. Определения жизненного цикла систем

Понятия концепции жизненного цикла

Системный инженер — это роль в Организации, которая практикует инженерную дисциплину системной инженерии (SE) и имеет квалификацию набора компетенций SE

Системная инженерия объединяет другие дисциплины для поддержки модели жизненного цикла

Модель жизненного цикла состоит из **этапов жизненного цикла**, которые обычно включают стадии концепции, реализации, производства, поддержки, использования и вывода из эксплуатации (не показаны).

Каждая стадия жизненного цикла относится к **процессам жизненного цикла**, которые производят различные виды **рабочих продуктов (Модели, Описание системной архитектуры, Требования**

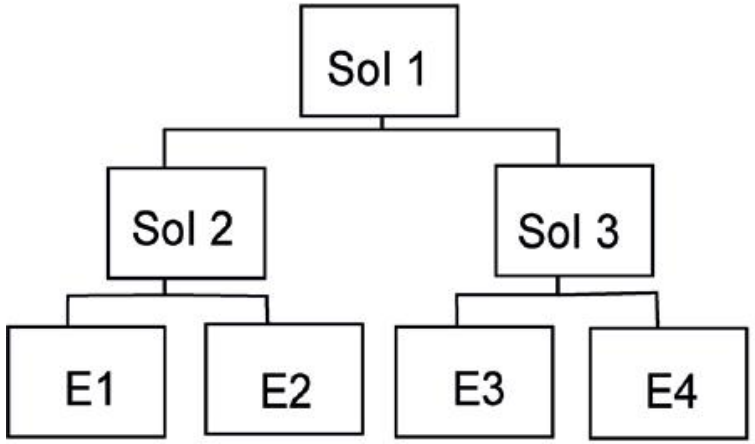
Процессы жизненного цикла создают **систему интереса** (SoI)

Существует множество **видов систем**, включая **природные системы**, **социальные системы** и **технологические системы** (не показаны)

Системы, созданные людьми и для людей, называются **инженерными системами**

Инженерная система, чей жизненный цикл находится в стадии рассмотрения, называется **целевой системой** или **система интереса** (SoI)

1.6. Жизненный цикл систем и процессы жизненного цикла. Общая парадигма SE



SOI - Целевая система
E - Элемент системы

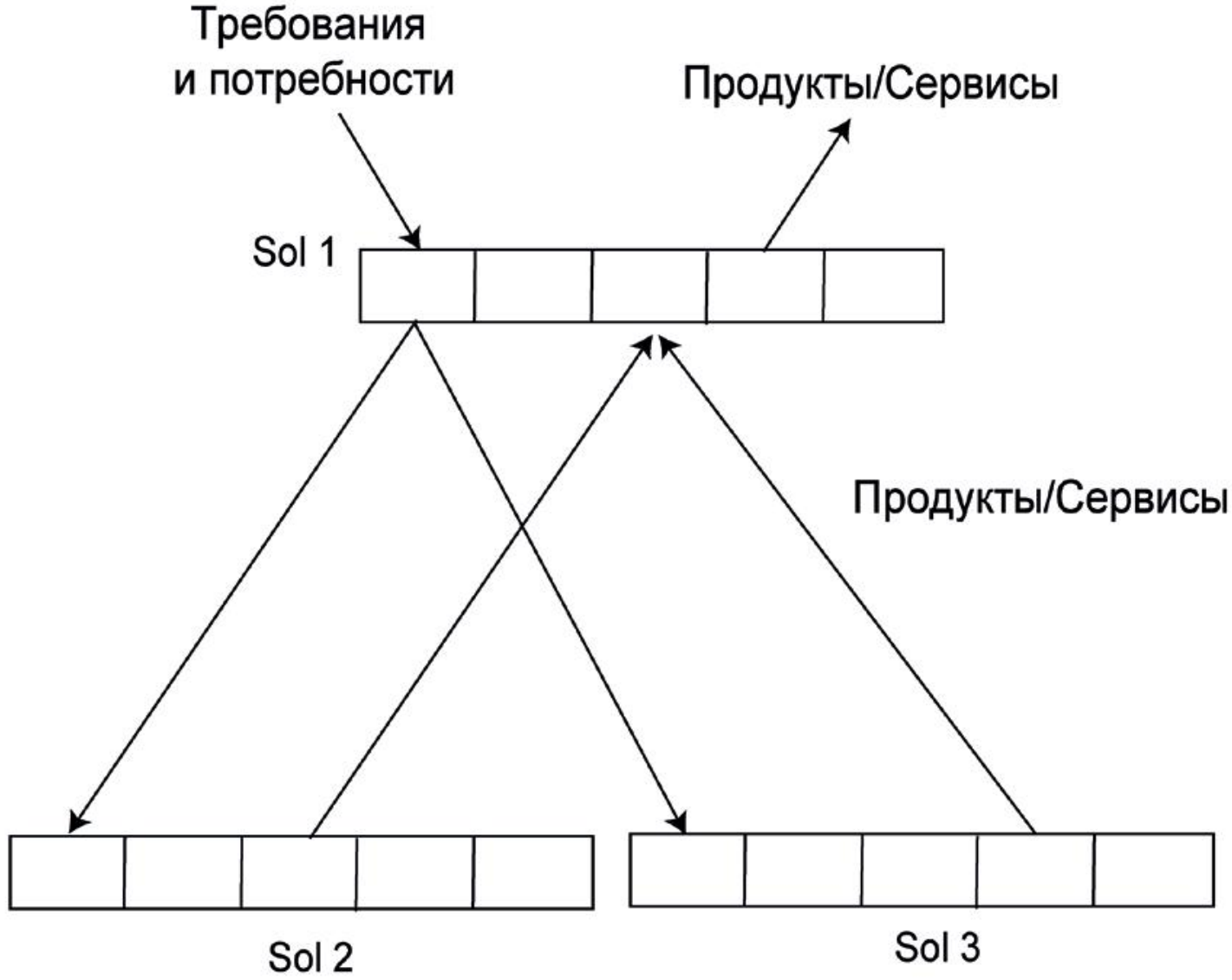


Рисунок 9. Общая парадигма SE - https://sebokwiki.org/d/images/sebokwiki-farm!d/6/6d/062211_BL_Paradigm.png

Общая модель жизненного цикла системы

Рисунок 10. **Обобщенная модель жизненного цикла SoI. -** https://sebokwiki.org/d/index.php?title=File%3AFig_1_A_generic_life_cycle_KF.png [SEBoK Original]

Стадия определения концепции начинается с решения вопроса об инвестировании создания некоторой системы (SoI). В частности, исследуется: целесообразность, возможность разработки, преимущество создания SoI, стоимость затрат на жизненный цикл SoI

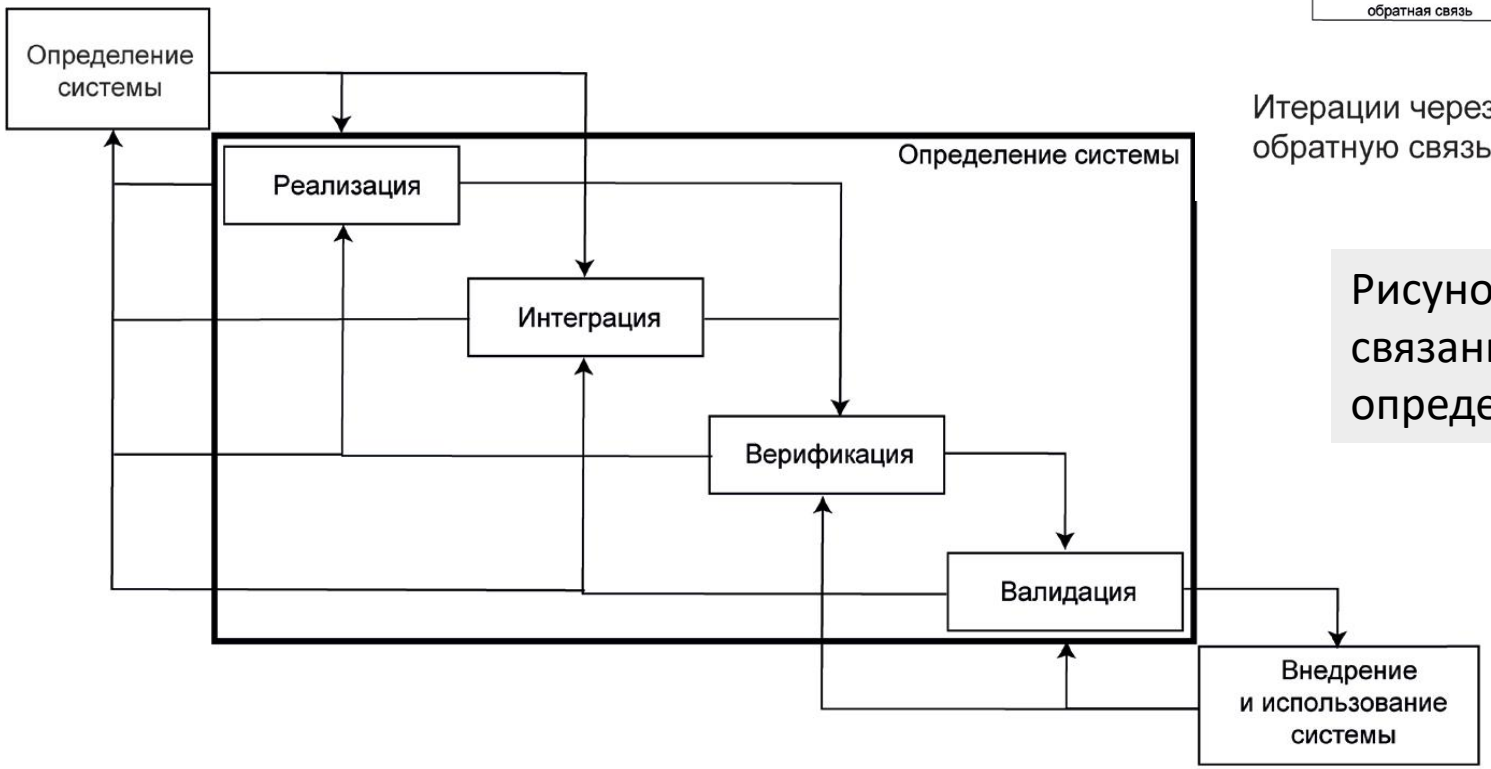
Стадия определения системы включает разработку системных архитектур; определение и согласование уровней системных требований; разработка планов жизненного цикла на системном уровне и выполнение системного анализа, чтобы проиллюстрировать совместимость и осуществимость итогового определения системы

Стадия реализации системы начинается есть уверенность, что создание SoI оправдает выделение ресурсов, необходимых для разработки и поддержания начальной операционной способности (**IOC**) или разработки полной эксплуатационной способности (**FOC**). Активности этапа включают: создание элементов системы, проверку и испытание элементов, их интеграцию и подготовку к производству

Последующими этапами жизненного цикла являются **стадии: Производства системы** (**Production**), **Поддержки** (**Support**), **Использования** (**Utilization**) с последующим **выводом из эксплуатации** (**Retirement**)

1.7. Итеративность стадий и процессов жизненного цикла

Рисунок 1.13. Пример итерации между другими процессами жизненного цикла [1]



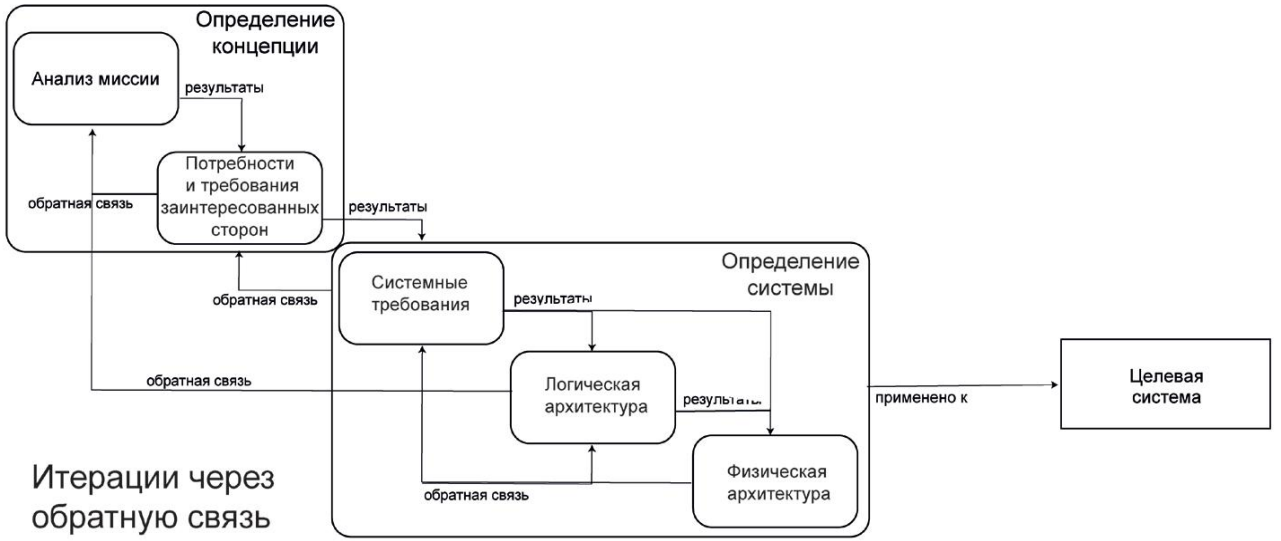


Рисунок 1.12. Пример итерации процессов жизненного ци связанных со стадиями определение концепции и определение системы [1]

Традиционная модель “V” системной инженерии [5]

Традиционная модель V системной инженерии [4]

Свойства модели SE «V»

- **Левая нисходящая ветвь** модели Vee включает стадии: Исследование осуществимости/Исследование концепции (Feasibility Study/Concept Exploration), Концепция эксплуатации (Concept of Operations), Системные требования (System Requirements), Высокоуровневое проектирование (High Level Design), Детальное проектирование (High Level Design), Сборка и интеграция конечного продукта системного уровня, проверка и валидация (Assemble & Integrate System-Level End Product & Verify & Validate), Разработка программного обеспечения/оборудования/Установка на месте (Software/Hardware Development/ Field Installation)
- Особенностью левой ветви является ее соответствие **фазе анализа системы**, начиная с анализа пользовательского представления системы в виде потребностей/требований, вплоть до детализации на уровне компонент низкого конструктивного уровня, например, модулей программной реализации программного обеспечения системы
- **Правая восходящая ветвь** модели Vee включает стадии: Тестирование модулей/устройств (Unit/Device Testing), Проверка подсистем (Subsystem Verification), Проверка и развертывание системы (System Verification & Deployment), Испытание системы (System Validation), Эксплуатация и обслуживание (Operations and Maintenance), Изменения и обновления (Changes and Upgrades), Вывод из эксплуатации/Замена (Retirement/ Replacement). Особенностью правой ветви соответствие **фазе синтеза**

Свойства модели SE «V»

Связи артефактов левой и правой ветвей реализуются в виде следующих планов:

- Плана тестирования модулей/устройств (Unit/Device Testing Plan)
- Плана проверки подсистем (Subsystem Verification Plan)
- Плана проверки системы (System Verification Plan)
- Плана испытания системы (System Validation Plan)

Модель SE «V» обладает рядом недостатков, в частности [5]:

- Компоновка «V» предполагает только последовательный процесс разработки
- Происхождение SE «V» относится к эпохе «**документоцентризма**», когда информация передавалась между отдельными людьми, группами и фазами жизненного цикла
- SE «V» также не отражает интегрированный и итеративный характер разработки продукта
- Попытки обновить символ «V» только усложнили его и сделали его еще более трудным для понимания
- Последовательность V-процесса затрудняет обмен данными и информацией в реальном времени между сложными взаимодействиями экосистемы MBE

Литература к главе 1

Литература

[1] Guide to the Systems Engineering Body of Knowledge (SEBoK), version 2.7. 2022.

[2] Wiener, N. 1948. Cybernetics or Control and Communication in the Animal and the Machine / N. Wiener. — New York, NY, USA: John Wiley & Sons Inc].

[3] Forrester, J. 1961. Industrial Dynamics / J. Forrester. — Winnipeg, Manitoba, Canada: Pegasus Communications].

[4] Wymore, A. 1967. A Mathematical Theory of Systems Engineering: The Elements / A. Wymore. — New York, NY, USA: John Wiley.

[5] Bertalanffy, L. von. 1968. General System Theory: Foundations, Development, Applications / L. Bertalanffy. New York, NY, USA: George Braziller].

[6] Wymore, A. W. 1977. A Mathematical Theory of Systems Engineering: The Elements / A. W. Wymore. Huntington, NY, USA: Robert E. Krieger].

[7] Wymore, A.W. 1993. Model-Based Systems Engineering / A.W Wymore. — Boca Raton, FL, USA: CRC Press, Inc].

[8] Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, Fourth Edition, INCOSE, 2015.

[9] {INCOSE. 2012. INCOSE Systems Engineering Handbook, version 3.2.2. San Diego, CA, USA: International Council on Systems Engineering (INCOSE), INCOSE-TP-2003-002-03.2.2}.

[10] Блауберг, И.В. Становление и сущность системного подхода / И.В. Блауберг, Э.Г. Юдин. — М.: Изд-во «Наука», 1973. — 272 с.

[11] Локтионов, М.В. А.А. Богданов как основоположник общей теории систем. Философия науки и техники 2016 / М.В. Локтионов. — Т. 21. — № 2. С. 80–96 УДК 141.2? Philosophy of Science and Technology 2016, — Vol. 21, no 2, pp. 80–96 DOI: 10.21146/2413-9084-2016-21-2-80-96.

[12] Берталанфи, Л. Общая теория систем — критический обзор / Л. Берталанфи // Исследования по общей теории систем: Сб. пер. / общ. ред. и вст. ст. В.Н. Садовского и Э.Г. Юдина; пер. с англ. Н.С. Юлиной. — М.: Прогресс, 1969. — С. 23–82.

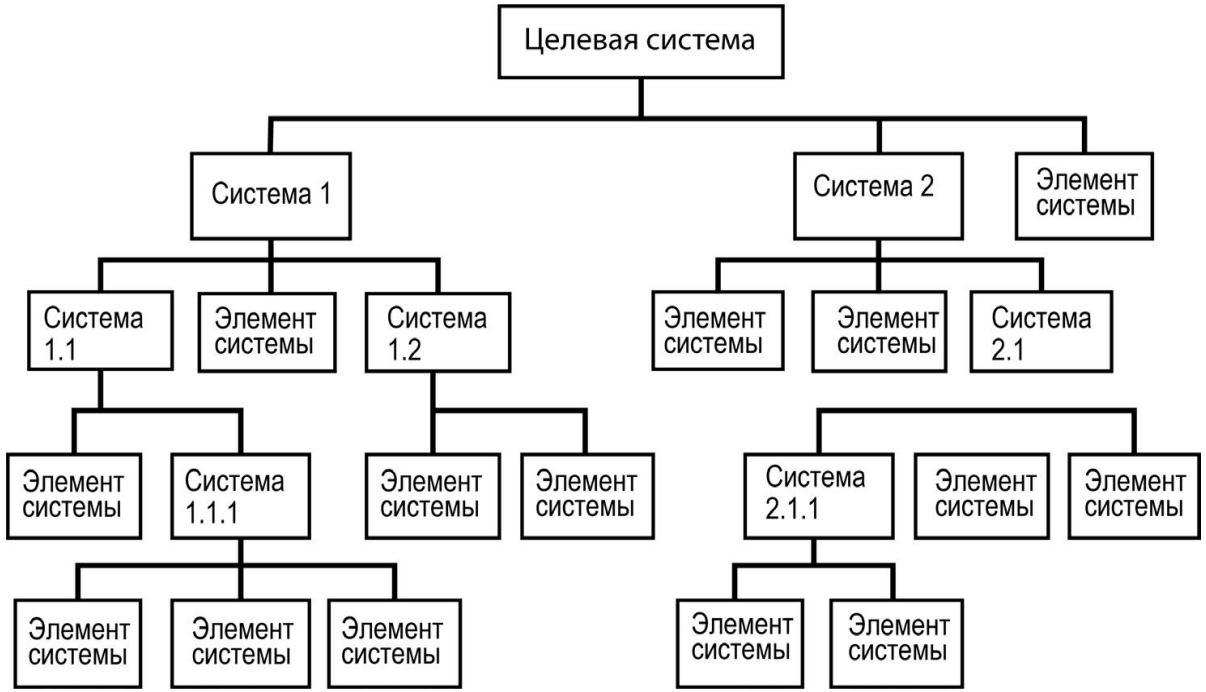
[13] Watson, M.D. 2018a. Engineering elegant systems: Postulates, principles, and hypotheses of systems engineering / M.D. Watson. — AIAA Complex Aerospace Systems Exchange (CASE) 2018, Future of Systems Engineering Panel, Orlando, FL, September 2018.

[14] Hitchins, D. 2009. What are the general principles applicable to systems? / D. Hitchins. — INCOSE Insight, — Vol. 12, no. 4. — P. 59–63.

Глава 2. Модели жизненного цикла систем

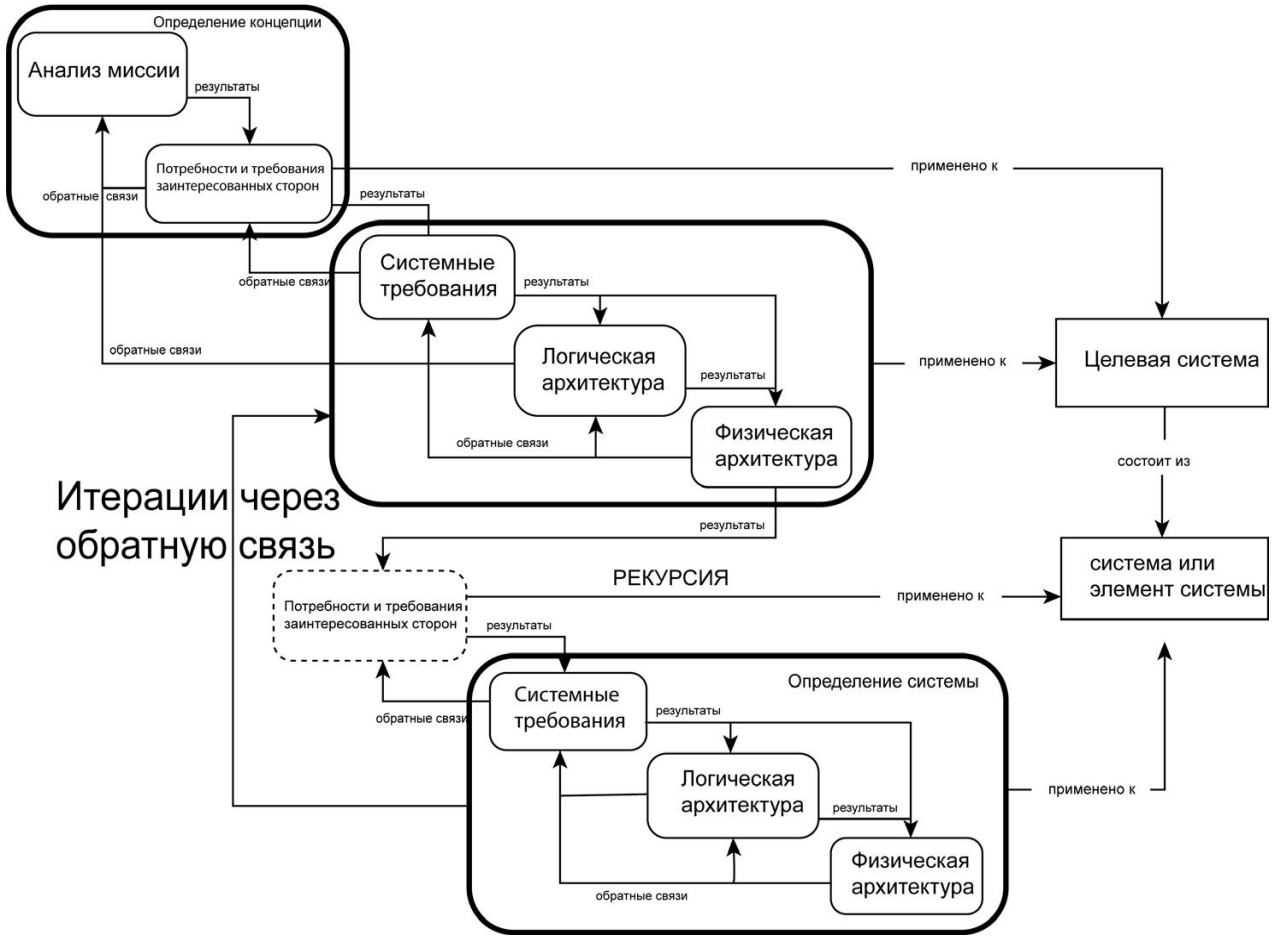
- 2.1. ЖИЗНЕННЫЕ ЦИКЛЫ СИСТЕМ И ИХ РЕКУРСИВНЫЙ ХАРАКТЕР
- 2.2. МОДЕЛИ ЖИЗНЕННОГО ЦИКЛА СИСТЕМ И ИХ КЛАССИФИКАЦИЯ
- 2.3. ФАЗЫ И СТАДИИ ЖИЗНЕННОГО ЦИКЛА СИСТЕМ
- 2.4. ПРЕДВАРИТЕЛЬНО ЗАДАННАЯ ОДНОШАГОВАЯ МОДЕЛЬ ЖИЗНЕННОГО ЦИКЛА SOI, МОДЕЛИ ТИПА VEE
- 2.5. ИНКРЕМЕНТАЛЬНЫЕ МОДЕЛИ ПРОЦЕССОВ ЖИЗНЕННОГО ЦИКЛА СИСТЕМ
- 2.6. ЭВОЛЮЦИОННЫЕ МОДЕЛИ ПРОЦЕССОВ ЖИЗНЕННОГО ЦИКЛА СИСТЕМ
- 2.7. ГИБКИЕ МОДЕЛИ ПРОЦЕССОВ ЖИЗНЕННОГО ЦИКЛА СИСТЕМ (AGILE SYSTEMS ENGINEERING)
- 2.8. ВОПРОСЫ ЗАЩИТЫ ИНФОРМАЦИИ В ПРОЦЕССЕ УПРАВЛЕНИЯ МОДЕЛЬЮ ЖИЗНЕННОГО ЦИКЛА СИСТЕМЫ

2.1. ЖИЗНЕННЫЕ ЦИКЛЫ СИСТЕМ И ИХ РЕКУРСИВНЫЙ ХАРАКТЕР



Иерархическая декомпозиция SoI

Рекурсивность процессов жизненного цикла в соответствии с уровнями конструктивной иерархии SoI



2.2. МОДЕЛИ ЖИЗНЕННОГО ЦИКЛА СИСТЕМ И ИХ КЛАССИФИКАЦИЯ

Модели делятся на следующие основные категории:

1) предварительно заданные одноэтапные (pre-specified, single-ste and sequential processes (e.g. the single-step waterfall model)), также известные как традиционные или последовательные процессы

2) предварительно заданные многоэтапные (pre-specified, multi-step processes (e.g. the multi-step waterfall model))

3) Инкрементальные или пошаговые модели жизненного цикла систем

4) эволюционно-последовательные

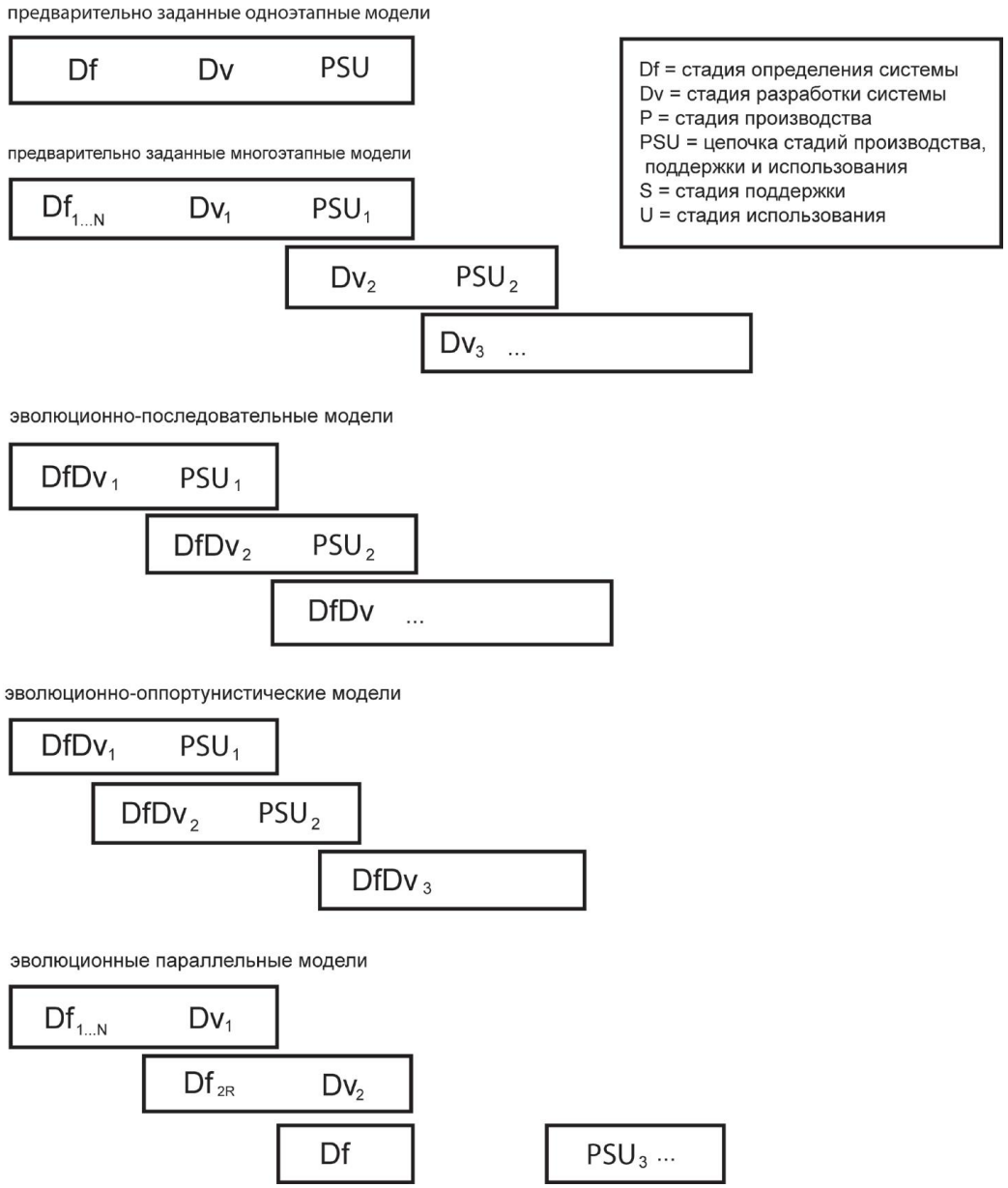
5) эволюционно-оппортунистические

6) эволюционные параллельные (или инкрементно-гибкие — incremental agile)

Процессы жизненного цикла системы

1. Предварительно заданная одношаговая модель (Pre-specified Single-step), примером которой является традиционная каскадная модель или последовательная модель Vee, применяется, если требования к SoI предварительно определены и имеют низкую вероятность значительных изменений, а также если нет возможности или смысла предоставлять частичные возможности продукта.
2. Предварительно заданная многоэтапная модель (Pre-specified multi-step models) разделяет разработку на части, чтобы задействовать ранние начальные эксплуатационные возможности (pre-planned product improvements — P3I) на протяжении жизненного цикла. Для данного подхода желательно, чтобы все возможности продукта были бы указаны заранее и вероятность значительных изменений мала. Он применяется, когда ожидание разработки полной системы влечет за собой потерю важных и полезных дополнительных возможностей конечной цели создания системы.
3. Эволюционно-последовательная модель (Evolutionary Sequential model) развивает первоначальные операционные возможности (IOC) и совершенствует их на основе опыта эксплуатации. Чистая гибкая (Agile) разработка программного обеспечения соответствует этой модели (если обнаружены дефекты в ПО и их нужно изменить, то исправления появятся через 30 дней в следующем релизе. Быстрое развертывание также подходит для этой модели для больших или программно-аппаратных систем. Его главное достоинство заключается в том, чтобы обеспечить возможности быстрого реагирования в процессе использования.
4. Эволюционно-оппортунистическая модель (Evolutionary Opportunistic model) может быть принята в случаях, когда следующее приращение откладывается до тех пор, пока: появляется достаточно привлекательная возможность, желаемая новая технология или до тех пор, пока не станут доступными другие факторы, такие как дефицитные компоненты или ключевой персонал. Такой подход также подходит для синхронизации обновлений нескольких готовых коммерческих продуктов (multiple commercial-off-the-shelf — COTS). Этот подход лучше всего использовать, когда для инкремента системы не требуется ждать оперативной обратной связи о ее использовании, а может потребоваться ожидание таких факторов для реализации инкремента, как технологическая зрелость, возможности внешней системы, необходимые ресурсы или новые возможности добавления ценности.
5. Эволюционно-параллельная модель (Evolutionary Concurrent model) реализуется на основе спиральной модели возрастающих обязательств [1]. Для нее характерным является работа постоянной команды системных инженеров, обрабатывающих трафик изменений планы и спецификации для следующего шага проекта, а также стабильной команды разработчиков для своевременной и надежной реализации текущего шага и использование одновременной проверки и валидации (V&V) для обеспечения более высокого уровня надежности

2.2. МОДЕЛИ ЖИЗНЕННОГО ЦИКЛА СИСТЕМ И ИХ КЛАССИФИКАЦИЯ

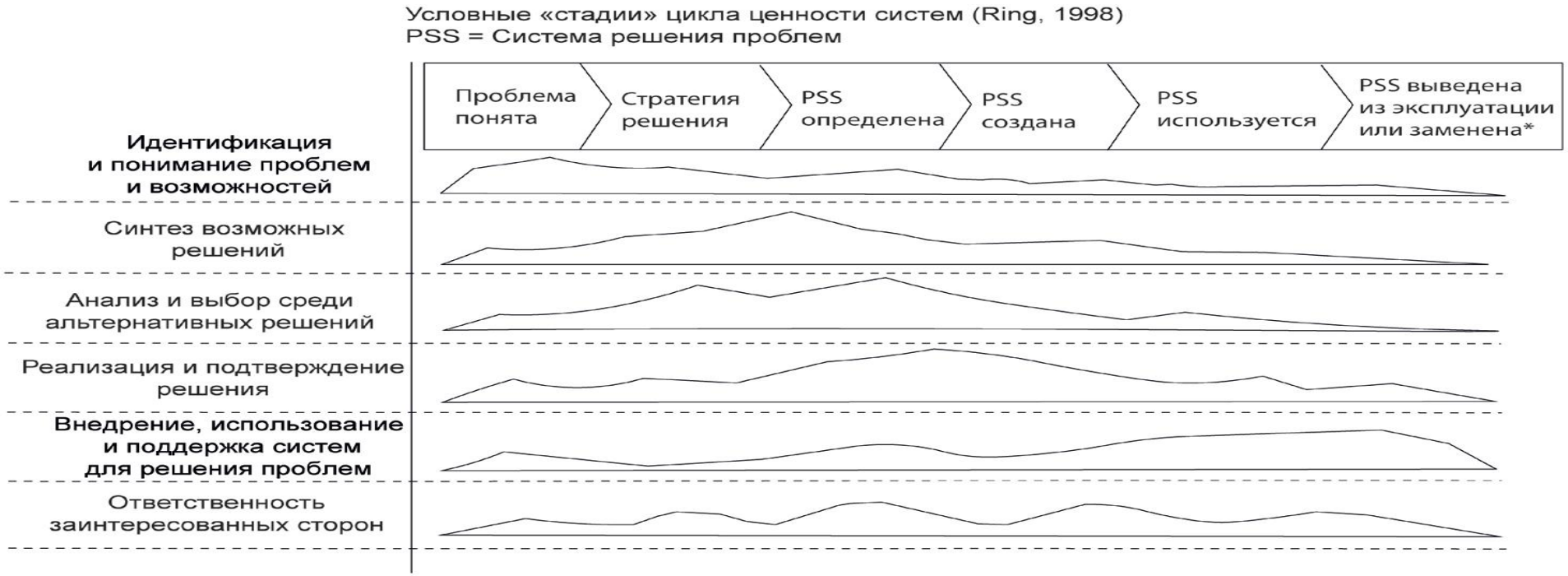


Pre-specified, Single-Step (PS) — **предварительно заданные одноэтапные модели**
Pre-specified, Multi-Step (PM) — **предварительно заданные многоэтапные модели**
Evolutionary Sequential (ES) — **эволюционно-последовательные модели**
Evolutionary Opportunistic — **эволюционно-оппортунистические модели**
Evolutionary Concurrent (EC) — **эволюционные параллельные модели**

Процессы жизненного цикла системы

Процесс – это последовательность действий или шагов, предпринятых для достижения определенной цели, преобразующие входные данные в выходные

Процессы жизненного цикла системной инженерии определяют технические и управленческие действия, выполняемые на одном или нескольких этапах для предоставления информации, необходимой для принятия решений в течение жизненного цикла; и обеспечить реализацию, использование и поддержку интересующей системы (SoI)



Показывает одновременную деятельность в рамках «одного цикла» системного подхода * PSS выведена из эксплуатации или заменена может означать переоценку проблемной ситуации и формулировку нового понимания проблемы.

Предварительно заданная одношаговая модель жизненного цикла SoI, модели Vee

Примеры стадий, их процессов и решателей

Типичный вариант традиционного многоэтапного предварительно заданного жизненного цикла модели SoI [2)]

Данная модель включает следующие стадии (описанные в стандарте ISO ISO/IEC 15288):

- **Исследовательская (Exploratory)**
- **Концепция (Concept)**
- **Разработка (Development)**
- **Производство (Production)**
- **Использование (Utilization)**
- **Поддержка (Suppot)**
- **Вывод из эксплуатации (Retirement)**

После завершения каждой стадии выполняется процедура **Решатель (Decision Gate**), которая принимает одно из следующих альтернативных решений:

- **Прейти к следующей стадии**
- **Прейти к следующей стадии, но открыть активности по элементам, которые должны быть перепроектированы**
- **Нет готовности: повторить предыдущую стадию Завершить проект**

Предварительно заданная одношаговая модель жизненного цикла SoI, модель Vee

Важным свойством модели Vee являются горизонтальные соответствия между стадиями одного уровня, по существу показывающие связь между процессами проверки и валидации с соответствующими им спецификациям, по которым эти процессы будут тестировать соответствующие компоненты систем.

Рекурсивная и параллельная Vee модели жизненного цикла систем

В том случае, когда целевая система представляет собой композицию, состоящую из многих элементов, которые сами могут рассматриваться как SoI, применяется **рекурсивная декомпозиция исходной системы**, в которой реализация каждого системного элемента осуществляется повторным использованием модели жизненного цикла системы, но на следующем более низком уровне конструктивной иерархии, рассматривая системный элемент как самостоятельный SoI

Общая структура Т-стадий жизненного цикла систем

Инкрементальные модели процессов жизненного цикла системы

- Инкрементальные или пошаговые модели жизненного цикла систем широко используются тогда, когда с помощью рассмотренного выше традиционного или водопадного (traditional or waterfall) подхода создана SoI с первоначальными операционными возможностями (**initial operational capability - IOP**), а затем планируется провести улучшение созданной базовой версии SoI
- Методы инкрементной разработки позволяют быстро создать действующую системы с минимальными возможностями, а затем с помощью инкрементального проекта поэтапно (пошагово) развивать базовую версию SoI до конечной SoI с заданными требованиями.
- Инкрементальный подход используется, когда:

- желательно осуществить быстрое исследование и внедрение части SoI

- требования изначально неясны

- финансирование ограничено

- заказчик желает оставить SoI открытой для возможности внедрения новой технологии в более позднее время и/или - требуется экспериментирование для разработки последовательных версий прототипа

- В этой модели применяются три **точки обзора проекта** (решатели):
- предварительного анализа проекта (**preliminary design review** - **PDR**) для проверки и утверждения набора системных требований, артефактов проекта и элементов обоснования в конце очередного цикла проектирования (также известного как решатель или ворота «проектирование-до»);
- критического обзора проекта (**critical design review - CDR**) для проверки и испытаний набора системных требований, артефактов проекта и элементов обоснования в конце цикла проектирования;
- точки обзора типа TRR (**test readiness review**) - многопрофильной проверки, предназначенной для обеспечения того, чтобы проверяемая подсистема или система была готова перейти к формальному тестированию, т.е. испытаниям -

Инкрементальные модели процессов жизненного цикла системы

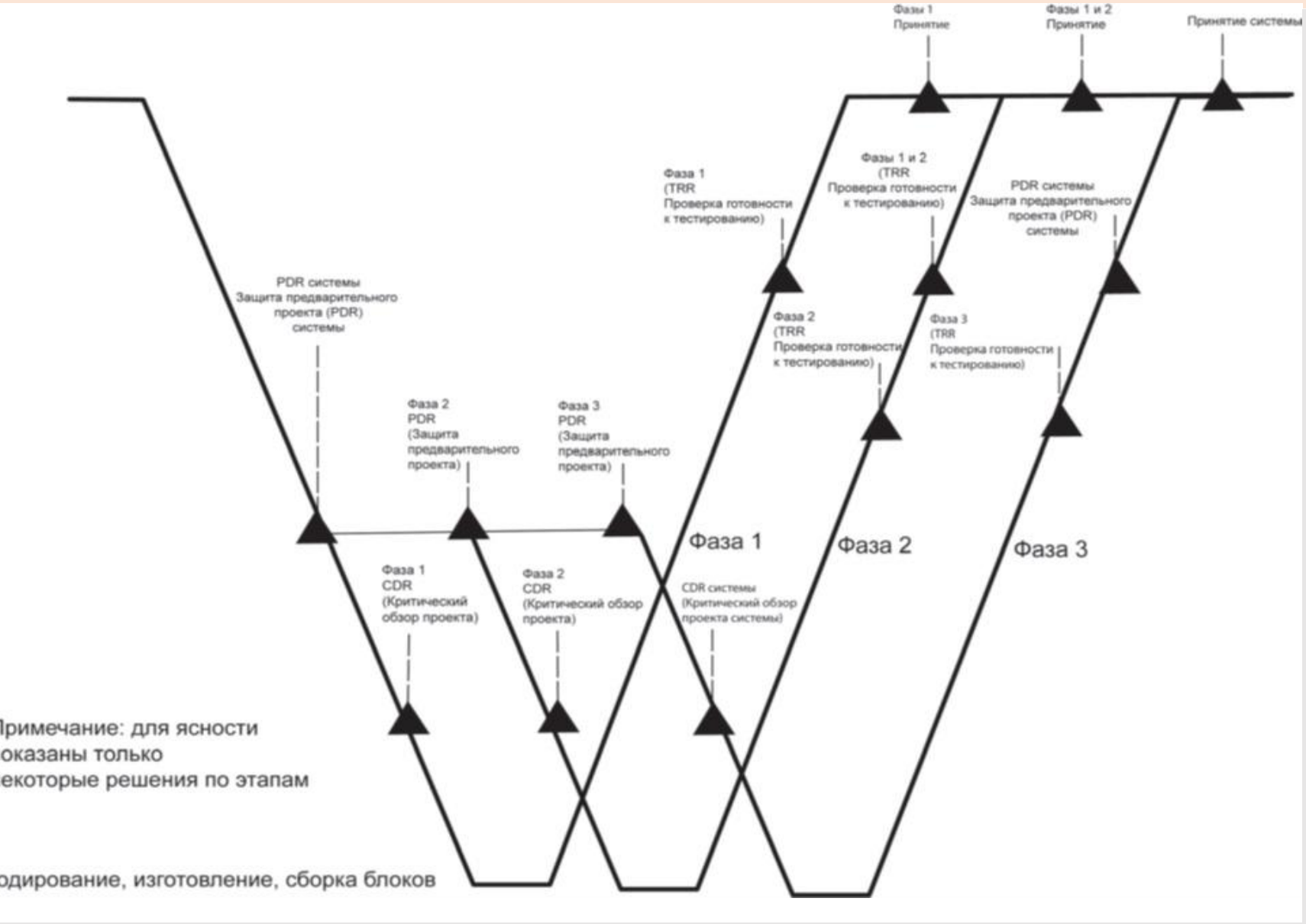
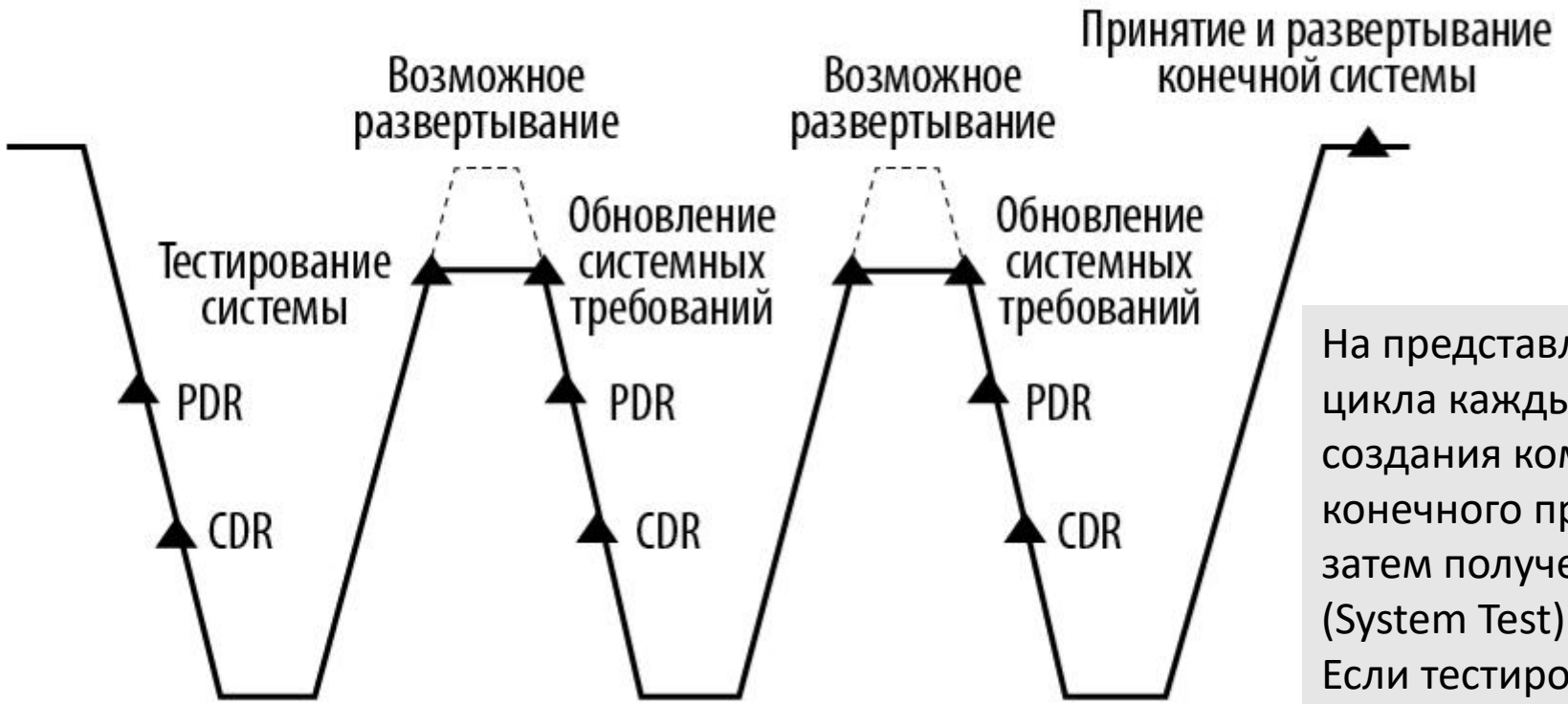


Рисунок 9. Инкрементальный подход с множеством шагов/фаз развития базовой версии системы и поставками (deliveries) продукта конечному пользователю [1]

В этой модели применяются три **точки обзора проекта** (решатели): предварительного анализа проекта (**preliminary design review** - **PDR**) для проверки и утверждения набора системных требований, артефактов проекта и элементов обоснования в конце очередного цикла проектирования критического обзора проекта (**critical design review - CDR**) для проверки и испытаний набора системных требований, артефактов проекта и элементов обоснования в конце цикла проектирования точки обзора типа TRR (**test readiness review**) - многопрофильной проверки, предназначенной для обеспечения того, чтобы проверяемая подсистема или система была готова перейти к формальному тестированию, т.е. испытаниям -

Эволюционные модели процессов жизненного цикла системы



На представленной на Рис. 10 модели жизненного цикла каждый проектный шаг начинается с создания компонент и сборки очередной версии конечного продукта (Code, Fab, -, Assemble Units), затем полученная версия проходит тестирование (System Test).
Если тестирование прошло успешно, то возможно развертывание системы, далее выполняется обновление системных требований (**Update System Requirements**), после чего начинается нисходящая ветвь разработки новой версии системы с двумя точками обзора (решателями) проекта: предварительного анализа проекта PDR и критического обзора проекта CDR

Рис. 10. Эволюционная общая модель - https://sebokwiki.org/d/images/sebokwiki-farm!d/b/b2/Evolutionary_Generic_Model.PNG [1].

Гибкие модели процессов жизненного цикла системы: Agile Systems Engineering

- В современном мире сложные и взаимосвязанные инженерные системы сталкиваются с быстрым устареванием под давлением технологических изменений, изменений окружающей среды и быстро меняющихся потребностей. Чтобы эти системы оставались устойчивыми к изменениям, они должны быть спроектированы гибкими, адаптируемыми, способными к расширению своей функциональности
- Разработка новых методологических решений сформировало новое направление гибких технологий проектирования инженерных систем, получившее название процессов **Agile SE**, которое базируется на особой культуре работы системных инженеров
- Такая культура в свою очередь основывается на так называемом **Agile-мышлении** или **Agile Mindset** - наборе убеждений и действий, основанных на **agile-ценностях**, которые делают акцентированное внимание на максимальном использовании возможностей исполнителей проектов, доверие работникам умственного труда в поиске наилучшего решения, улучшение продукта и процесса посредством регулярных демонстраций и ретроспектив, частое планирование для реализации извлеченных уроков.
- В процессе **Agile SE** системный инженер работает итеративно, поэтапно, постоянно моделируя, анализируя, разрабатывая и торгуя вариантами, чтобы сосредоточить внимание на определении системных решений, руководствуясь **Agile-принципами**, основные из которых являются:

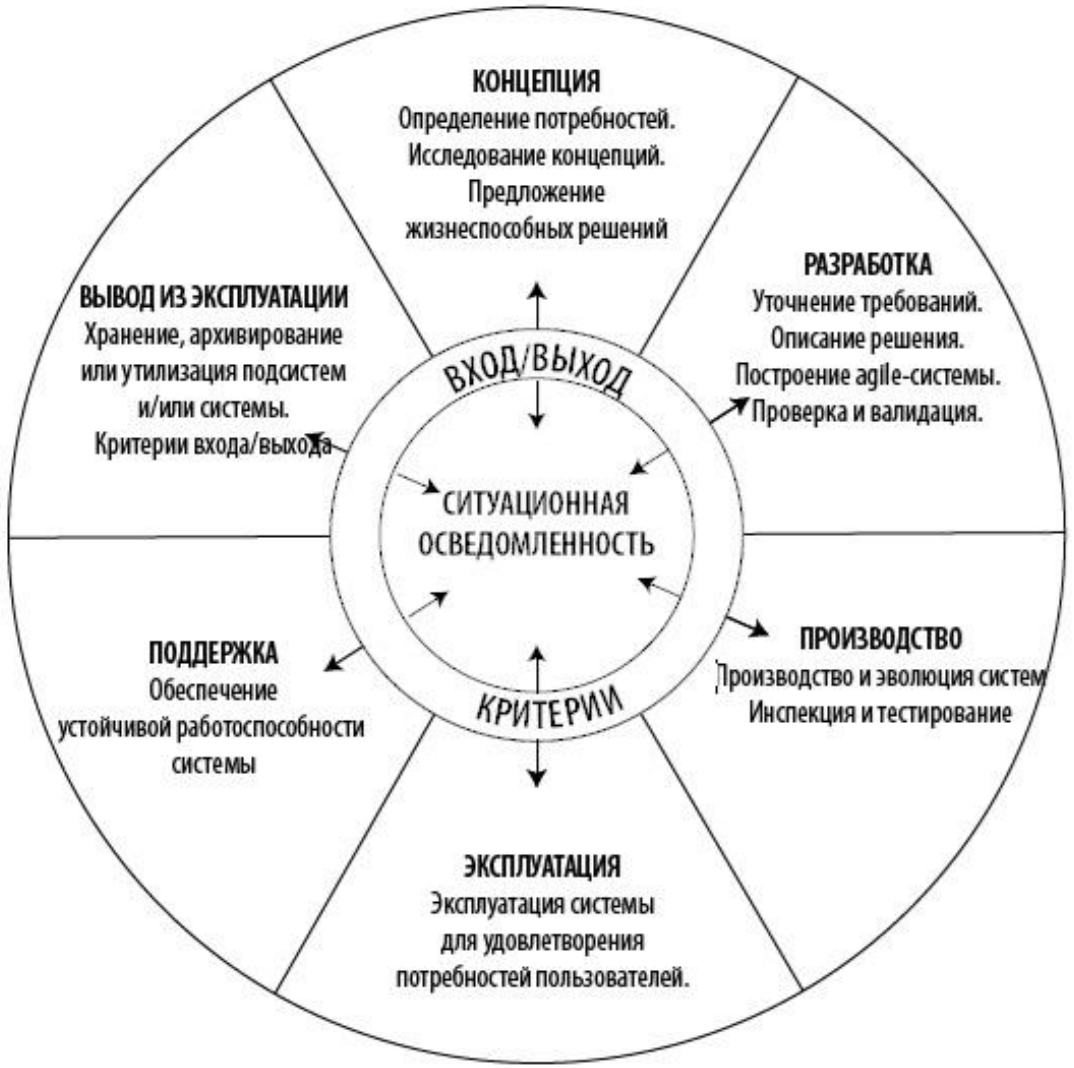
Принципы гибкой разработки. Agile Systems Engineering

1. Удовлетворить клиента за счет раннего и непрерывного предоставления ценных возможностей
2. Планируйте меняющиеся требования и сохраняйте столько гибкости, сколько необходимо на протяжении всей разработки; гибкие процессы используют изменения для клиента, особенно когда изменения приводят к конкурентному преимуществу
3. Предоставляйте рабочие возможности часто, от пары недель до пары месяцев, отдавая предпочтение более коротким временным рамкам
4. Бизнес-персонал, клиенты или их сторонники и исполнители должны ежедневно работать вместе на протяжении всего проекта
5. Стройте проекты вокруг мотивированных людей. Обеспечьте им необходимые условия и поддержку и доверьте им выполнение работы
6. Самый оперативный и действенный способ передачи информации — личный разговор
7. Рабочие способности являются основным мерилом прогресса
8. Гибкие процессы способствуют устойчивому развитию. Спонсоры, разработчики и пользователи должны иметь возможность поддерживать постоянный темп на неопределенный срок
9. Постоянное внимание к техническому совершенству и хорошему дизайну повышает гибкость
10. Простота «искусство максимизировать объем невыполненной работы» имеет важное значение, особенно в группе внедрения. По-настоящему гибкий проект разработки не навязывает искусственную отчетность и технологические требования группе внедрения
11. Лучшие архитектуры, требования и проекты создаются самоорганизующимися командами, основанными на минимальном наборе руководящих принципов
12. Через регулярные промежутки времени команда размышляет о том, как стать более эффективной, затем соответствующим образом настраивает и корректирует свое поведение

Принципы гибкой разработки

- Согласно стандарта ISO/IES/IEEE 24748-1 «Системная и программная инженерия. Управление жизненным циклом. Часть 1. Руководство по управлению жизненным циклом» существует шесть асинхронных и параллельных стадий жизненного цикла системы: **Concept**, **Development**, **Production**, **Utilization**, **Support**, **Retierement**.
- В представленной на Рис. 11 модели добавлена седьмая стадия жизненного цикла - **ситуационная осведомленность** (***Situational Awareness***) [Dove 2019].
- Используя эту модель жизненного цикла разработки гибких систем (ASELCM), проект может начинаться на стадии «Концепция», переходить в фазу ситуационной осведомленности, затем переходить в фазу разработки, обратно в фазу ситуационной осведомленности и так далее по кругу.
- Эта фаза ситуационной осведомленности позволяет акцентировать внимание на проактивном осознании ситуационных возможностей и рисков как для процессов, так и для продуктов на протяжении всего процесса системной инженерии и жизненного цикла целевой системы.
- Эта упреждающая осведомленность и последующая деятельность по смягчению последствий вдыхают жизнь в процесс разработки гибких систем, выводя его за рамки повторяющегося выполнения традиционных приращений разработки, выполняющих невыполненные запланированные функции.
- Структура гибкой модели жизненного цикла допускает постоянную эволюцию после первоначальной доставки и требует, чтобы на стадии разработки создавалась гибкая целевая система для поддержки эволюции. Стадия ситуационной осведомленности является отправной точкой для всей остальной деятельности стадии и включает в себя исходное осознание того, что необходима новая система.

Гибкие модели процессов жизненного цикла системы: Agile Systems Engineering



ис. 11. Цели для каждой стадии модели жизненного цикла, даптированные из (ISO/IEC/IEEE 2018, стр. 17), с добавлением стадии ситуационной осведомленности» - [7]

В стандарте ISO/IES/IEEE 24748-1 определе-
ны шесть асинхронных и параллельных стадий жизненного цикла системы:
Concept (Концепция), **Development** (Разработка), **Production** (Производство), **Utilization** (Использование), **Support** (Поддержка), **Retierement** (Вывод из эксплуатации).
В представленной на рис. модели добавлена седьмая стадия жизненного цикла — ситуационная осведомленность (**Situational Awareness**) [7].
Используя эту модель жизненного цикла разработки гибких систем (ASELCM), проект может начинаться на стадии «Концепция», переходить в стадию ситуационной осведомленности, затем переходить в стадию разработки, обратно в стадию ситуационной осведомленности и так далее по кругу.

Глава 3. Модели итеративного процесса разработки программного обеспечения

3.1. КЛАССИФИКАЦИЯ СИСТЕМ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ И МОДЕЛЕЙ ИХ ЖИЗНЕННОГО ЦИКЛА

3.2. ИНКРЕМЕНТАЛЬНАЯ СБОРКА (INCREMENTAL-BUILD)

3.3. ПРОТОТИПИРОВАНИЕ (PROTOTYPING) В РАЗРАБОТКЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

3.4. СПИРАЛЬНЫЕ МОДЕЛИ ЖИЗНЕННОГО ЦИКЛА

3.5. ГИБКАЯ МОДЕЛЬ РАЗРАБОТКИ ПО (AGILE) — ИТЕРАТИВНАЯ ЭВОЛЮЦИЯ ТРЕБОВАНИЙ И КОДА

- **3.6.** ЭВОЛЮЦИОННЫЙ ПОСЛЕДОВАТЕЛЬНЫЙ ПРОЦЕСС СКРАМ (SCRUM)
- **3.7.** ПРИНЦИПЫ БЕРЕЖЛИВОГО ПРОИЗВОДСТВА И РАЗРАБОТКИ ПО
- **3.8.** РУКОВОДСТВО ПО СВОДУ ЗНАНИЙ В ОБЛАСТИ ПРОГРАММНОЙ ИНЖЕНЕРИИ SWEBOK

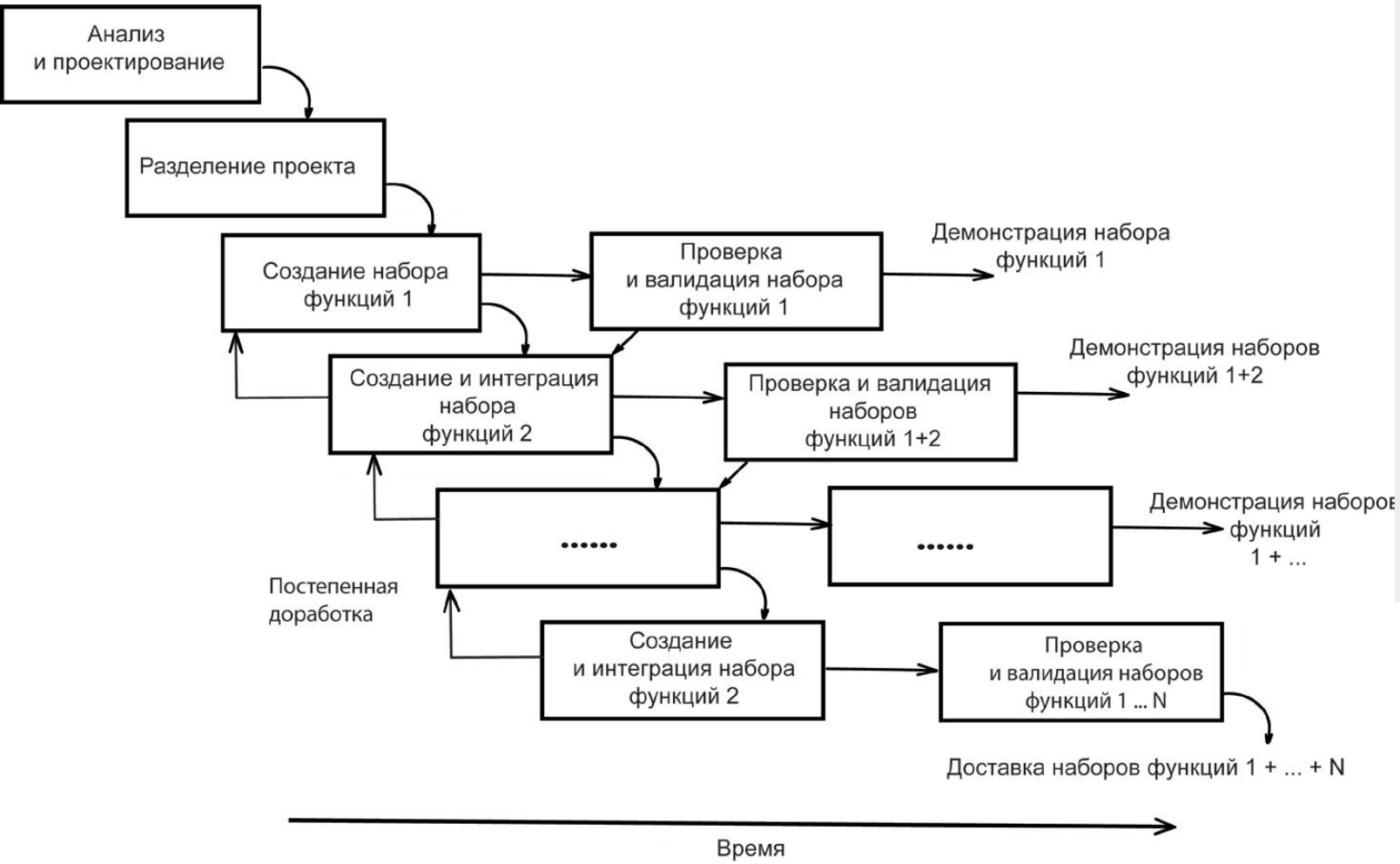
Модели итеративного процесса разработки программного обеспечения

- Итеративный подход подразумевает, что реализация проекта организована в виде последовательности итераций
- Каждая итерация модели жизненного цикла разработки программной системы добавляет код к растущей базе ПО, завершаясь получением согласованных результатов выполнения итерации (артефактов системы), включающих работающий код программного релиза системы, соответствующий данной итерации, модели проекта и документацию
- Расширенная кодовая база тестируется, при необходимости перерабатывается и демонстрируется, что она удовлетворяет пользовательским требованиям
- Каждый инкремент добавляет функциональность к основному коду создаваемой программной системы, однако при этом, возможно, не привнося качественных изменений свойств системы. В этом случае подход можно назвать чисто итеративным
- В случае, когда итерация приводит к качественным изменениям свойств проекта итерационный подход становится итерационным и эволюционным, так как под эволюцией понимается качественное изменение свойств проекта.

Рассматриваются следующие модели процессов для разработки ПО:

- **Инкрементальная сборка (Incremental-build) - итеративные циклы внедрения-проверки-валидации-демонстрации;**
- **Прототипирование** (**Prototyping**) – первоначальная разработка упрощенного макета целевого продукта;
- **Спиральный** (**Spiral**) - итеративный риск-ориентированный анализ альтернативных подходов и оценка результатов;
- **Гибкая модель** (**Agile**) - итеративная эволюция требований и кода

Инкрементальная сборка (Incremental-build)

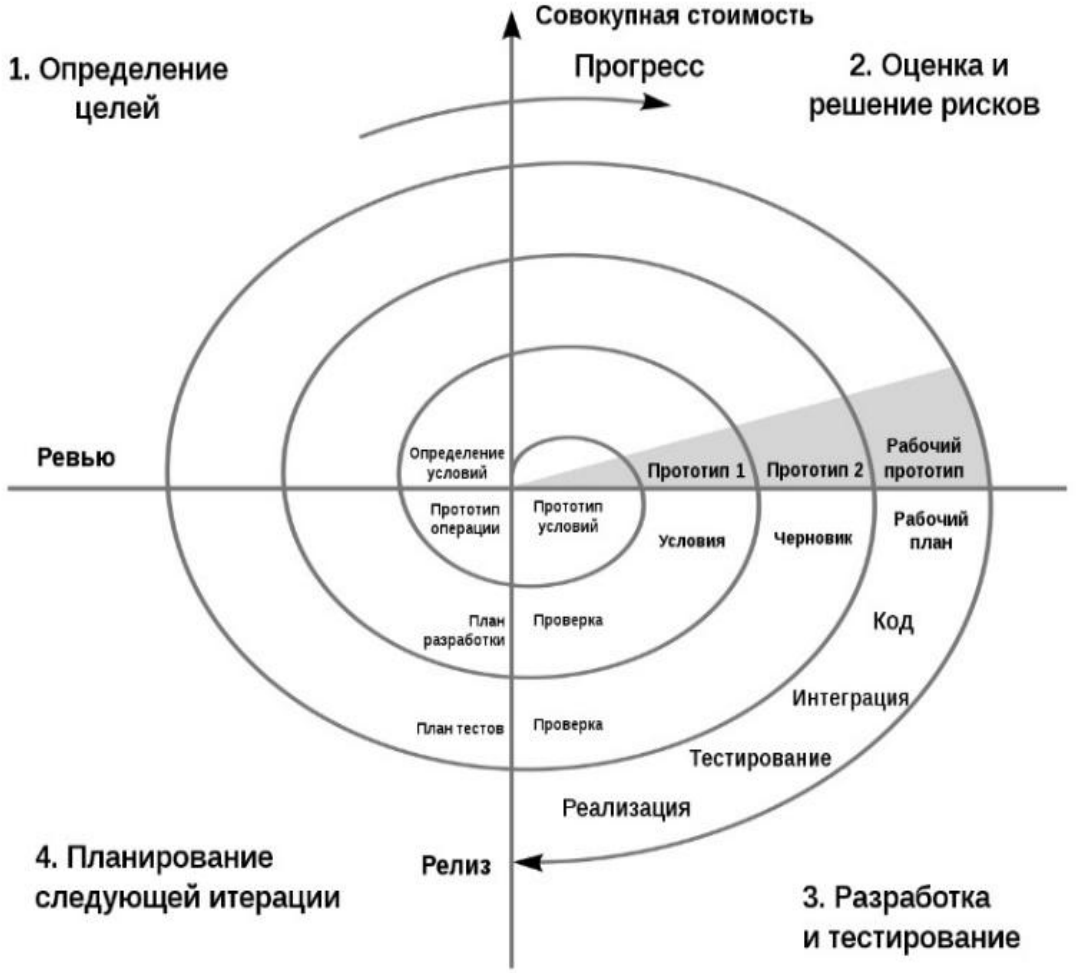


Процесс инкрементной сборки обычно хорошо работает с небольшими командами, но его можно масштабировать для более крупных проектов.
Достоинствами данного подхода являются:
- непрерывная интеграция, проверка и проверка развивающегося продукта;
- частые демонстрации прогресса;
раннее выявление дефектов;
- раннее предупреждение о технологических проблемах; и
- систематическая отработка переделок, вызванных устранением обнаруженных ошибок

Прототипирование (prototyping) в разработке программного обеспечения

- В программной инженерии (Software Engineering - SwE) прототип — это макет желаемой функциональности некоторой части системы, что отличается от физических систем, где прототип обычно представляет собой первую полнофункциональную версию системы, которую передают на серийное производство.
- Одной из основных задач прототипирования и является вовлечение заказчика в отработку требований к создаваемой системе. Это достигается посредством техники быстрых релизов упрощенной экспериментальной или прототипной системы, которая легко модифицируется и позволяет итеративно и быстро отрабатывать требования к проектируемому продукту и с помощью оперативной обратной связи с заказчиком, демонстрировать ему полученные результаты, благодаря чему заказчик становится активным участников процесса создания системы. В результате такого подхода заказчики (конечные пользователи) системы принимают активное участие в проектировании системы, анализируя результаты работы быстрого прототипа они выдают свои замечания до тех пор, пока прототип не будет удовлетворять всем их требованиям.
- Такой подход часто называют структурной эволюционной технологией быстрого прототипирования. При создании прототипа ПО, полученные в ходе его разработки программные решения, как правило, не используется в готовой версии программного продукта. Во многих случаях эффективнее создавать рабочий код конечного продукта с нуля, используя знания и опыт, полученные при прототипировании, чем перепроектировать существующий код прототипа.

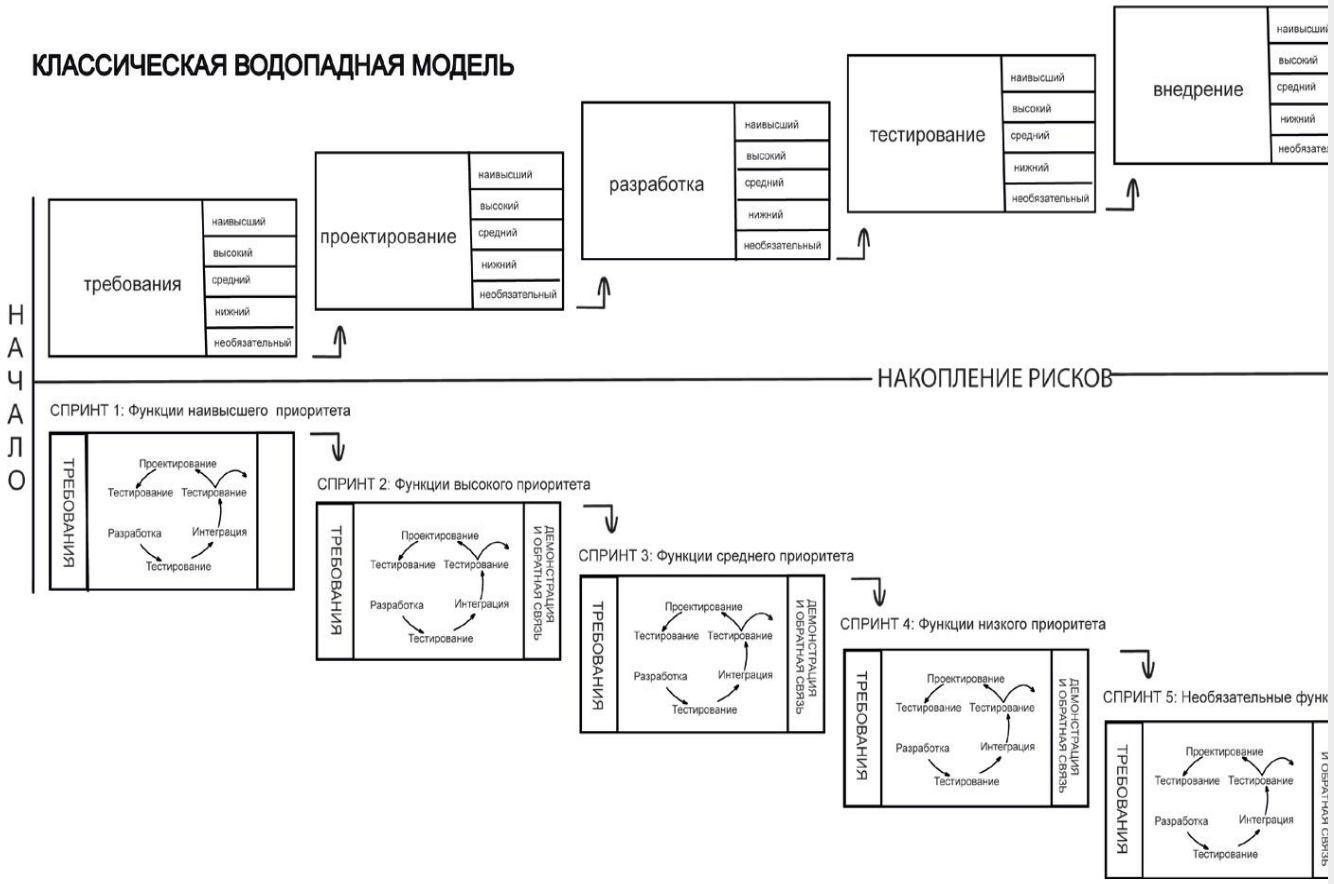
Спиральные модели жизненного цикла



Спиральная модель процесса разработки ПО предложена Барри Боэмом в 1986 году [5].
Она представляет собой модель процесса разработки ПО, сочетающий в себе как инкрементальность (пошаговость), так и итеративность.
Особое внимание в этой модели уделено рискам, влияющим на результат реализации жизненного цикла ПО.
Боэм формулирует десять наиболее распространённых (по приоритетам) рисков:
- Дефицит специалистов
- Нереалистичные сроки и бюджет
- Реализация несоответствующей функциональности
- Разработка неправильного пользовательского интерфейса «Золотая сервировка», перфекционизм, ненужная оптимизация и оттачивание деталей
- Непрекращающийся поток изменений
- Нехватка информации о внешних компонентах, определяющих окружение системы или вовлечённых в интеграцию.

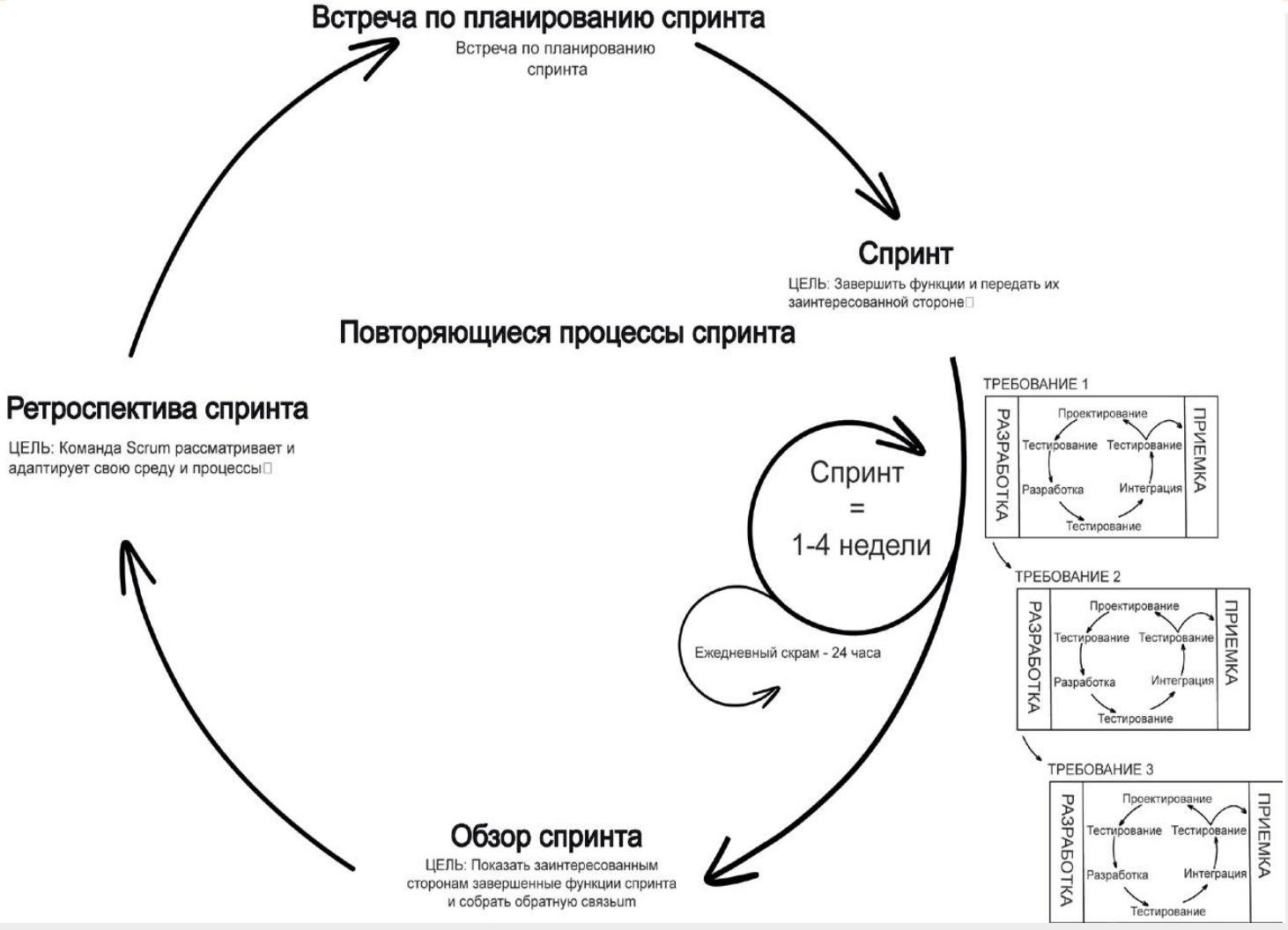
Рисунок 10. Спиральная модель Барри Боэма - https://commons.wikimedia.org/wiki/File:%D0%A1%D0%BF%D0%B8%D1%80%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_%D0%91%D0%B0%D1%80%D1%80%D0%B8_%D0%91%D0%BE%D1%8D%D0%BC%D0%B0.svg [6].

Гибкая модель разработки ПО (Agile) - итеративная эволюция требований и кода



Agile - гибкий неформализованный подход в управлении проектами [6]. В отличие от классических моделей жизненного цикла данный подход не требует подробного задания на разработку точных спецификаций требований, также не применяется поэтапное планирование проекта и управление жизненным циклом. Реализация гибкого процесса строится на выполнении работ по проекту на коротких временных отрезках, так называемых **спринтах**, длиной в одну–две недели, по окончании которых происходят совещания членов команды, на которых делается оценка выполнения задач, поставленных на предыдущей встрече, и ставятся задачи для следующего спринта. Различие между классической водопадной моделью жизненного цикла ПО и моделью гибкого подхода иллюстрируется на Рис. 12.

Гибкая модель разработки ПО (Agile) - итеративная эволюция требований и кода



Работа спринта состоит в следующем:
- Во время спринта проводятся постоянные проверки, чтобы оценить прогресс в достижении цели спринта и, следовательно, в достижении цели релиза.
- Проводятся ежедневные собрания по схватке (scrum) для организации работы на день, анализа того, что команда сделала вчера, и согласования того, над чем она будет работать сегодня.
- По сути, скрам-команда проверяет свой прогресс в достижении цели спринта.
- В конце спринта проводится ретроспективное совещание по спринту, чтобы оценить производительность и спланировать необходимые адаптации.

Рисунок 3.4. Работа спринта в Agile-методологии [5]. [5] Mark, C. Layton and Steven J Ostermiller / C. Mark. Agile Project Management For Dummies®, 2nd Edition Published by: John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030-5774, www.wiley.com. Copyright © 2017 by John Wiley & Sons, Inc., Hoboken, New Jersey.

Эволюционный последовательный процесс Скрам (Scrum)

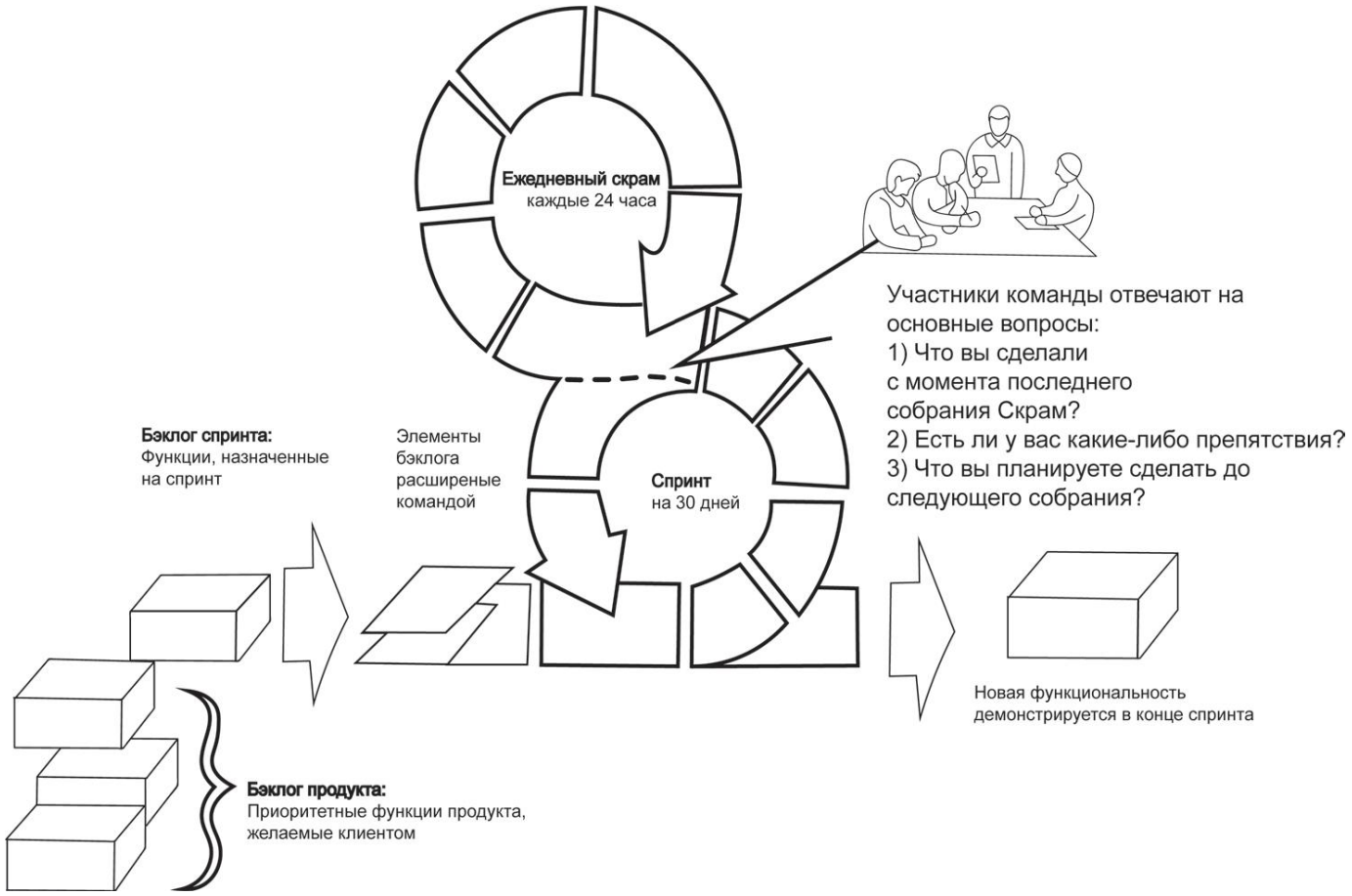


Рисунок 16. Пример Agile-процесса: Scrum (Бем и Тернер, 2004 г.) - https://sebokwiki.org/d/images/sebokwiki-farm!d/0/0c/Tale_of_Two_Implementations_Schwaber.jpg [1].

Рисунок 3.5. Состав Agile-команды [6]

ГЛАВА 4. ХАРАКТЕРНЫЕ ЧЕРТЫ И ВОЗМОЖНОСТИ ЯЗЫКА UML

4.1. ОСНОВНЫЕ ПРИНЦИПЫ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПОДХОДА (АБСТРАКЦИЯ, ИНКАПСУЛЯЦИЯ, МОДУЛЬНОСТЬ, ОБОБЩЕНИЕ И НАСЛЕДОВАНИЕ, АГРЕГАЦИЯ И КОМПОЗИЦИЯ, ИНТЕРФЕЙСЫ, ПОЛИМОРФИЗМ)

4.2. ИСТОРИЯ СОЗДАНИЯ И НАЗНАЧЕНИЕ ЯЗЫКА МОДЕЛИРОВАНИЯ UML

4.3. СРЕДСТВА ЯЗЫКА UML ДЛЯ МОДЕЛИРОВАНИЯ СТРУКТУРЫ СИСТЕМЫ (ОБЪЕКТЫ, КЛАССЫ, СТРУКТУРИРОВАННЫЙ КЛАСС, ПАКЕТЫ, АКТЕРЫ)

4.4. СРЕДСТВА ЯЗЫКА UML ДЛЯ МОДЕЛИРОВАНИЯ ПОВЕДЕНИЯ СИСТЕМЫ (СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ, ДЕЯТЕЛЬНОСТИ, ДИАГРАММЫ ПОСЛЕДОВАТЕЛЬНОСТИ, КОМБИНИРОВАННЫЙ ФРАГМЕНТ, КОНЕЧНЫЕ АВТОМАТЫ, ВРЕМЕННЫЕ ДИАГРАММЫ)

4.5. РЕАЛИЗАЦИЯ МОДЕЛИРОВАНИЯ (ДИАГРАММЫ КОМПОНЕНТОВ, ДИАГРАММЫ РАЗВЕРТЫВАНИЯ)

ГЛАВА 5. ЯЗЫК МОДЕЛИРОВАНИЯ SYSML И ЕГО ПРИМЕНЕНИЕ ДЛЯ РАЗРАБОТКИ МОДЕЛЕЙ СИСТЕМ

5.1. ИСТОРИЯ СОЗДАНИЯ И НАЗНАЧЕНИЕ ЯЗЫКА МОДЕЛИРОВАНИЯ SYSML

5.2. СРЕДСТВА ЯЗЫКА SYSML ДЛЯ МОДЕЛИРОВАНИЯ ТРЕБОВАНИЙ К СИСТЕМЕ

5.3. СРЕДСТВА ЯЗЫКА SYSML ДЛЯ МОДЕЛИРОВАНИЯ СТРУКТУРЫ СИСТЕМЫ

5.4. СРЕДСТВА ЯЗЫКА SYSML ДЛЯ МОДЕЛИРОВАНИЯ ПОВЕДЕНИЯ СИСТЕМЫ

5.5. СРЕДСТВА ЯЗЫКА SYSML ДЛЯ ПОСТРОЕНИЯ МОДЕЛИ

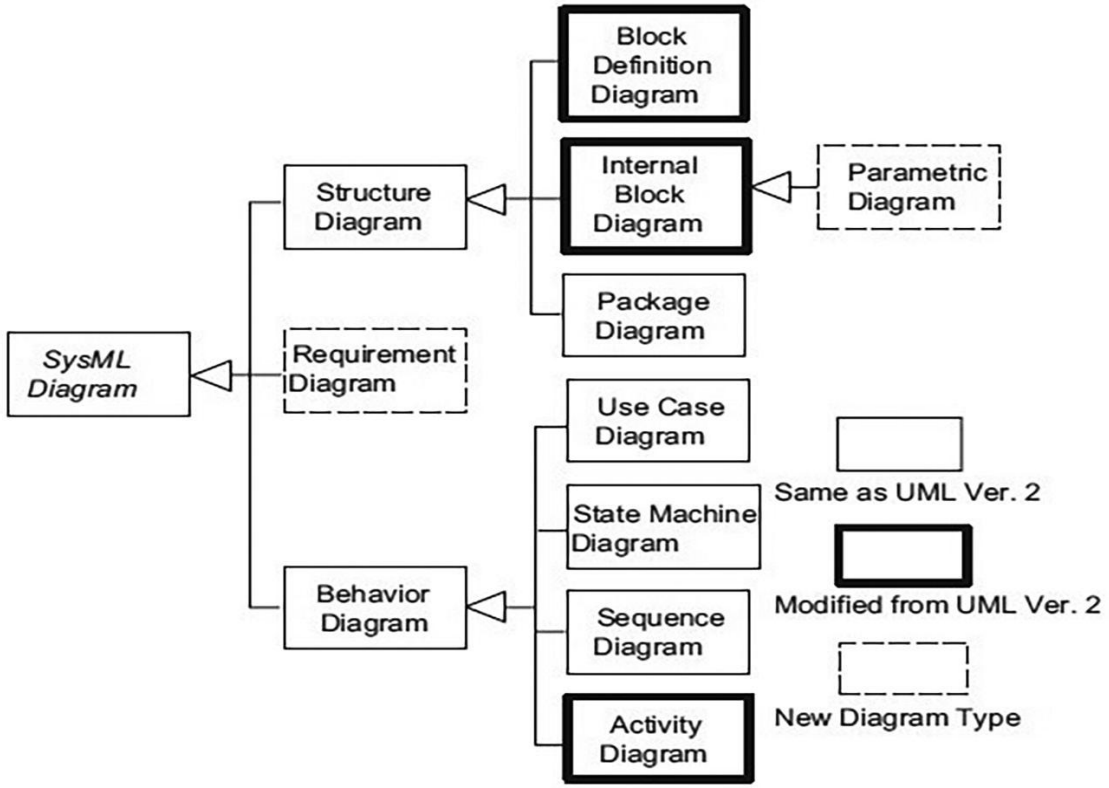


Fig. B.2 SysML Diagrams

ГЛАВА 6. ИНЖЕНЕРИЯ ТРЕБОВАНИЙ

6.1. ИНЖИНИРИНГ ТРЕБОВАНИЙ. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

6.2. ТРЕБОВАНИЯ В ЖИЗНЕННОМ ЦИКЛЕ СИСТЕМ

6.3. ПРЕОБРАЗОВАНИЕ ПОТРЕБНОСТЕЙ ПРЕДПРИЯТИЯ В ТРЕБОВАНИЯ

6.4. КОНСТРУКЦИЯ ТРЕБОВАНИЙ

6.5. ПРОЦЕССЫ И ДЕЯТЕЛЬНОСТИ ИНЖЕНЕРИИ ТРЕБОВАНИЙ В ЖИЗНЕННОМ ЦИКЛЕ СИСТЕМЫ

6.6. ПОТРЕБНОСТИ И ТРЕБОВАНИЯ БЕЗОПАСНОСТИ

6.7. ПРИМЕНЕНИЕ ЯЗЫКА SYSML ДЛЯ МОДЕЛИРОВАНИЯ ТРЕБОВАНИЙ

6.1. ИНЖИНИРИНГ ТРЕБОВАНИЙ. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

Инжиниринг требований (requirements engineering)один из важнейших разделов SE, от качественного исполнения которого целиком и полностью зависит успешность любого проекта по созданию и использованию инженерных систем

Под инжинирингом требований понимается ряд последовательных действий по разработке, декомпозиции, использованию спецификаций пользовательских потребностей и требований, включая требований заинтересованных сторон и системных требований, с переходом к требованиям проектирования низкого уровня

Требование — утверждение, которое идентифицирует эксплуатационные, функциональные параметры, характеристики или ограничения проектирования продукта или процесса, которое однозначно, проверяемо и измеримо. Требования необходимы для приемки продукта или процесса (потребителем или внутренним руководящим принципом обеспечения качества) [1 ISO 29148 - https://analytics.infozone.pro/requirements-in-iso-iec-29148].

Требование – это условие или возможность, которая должна выполняться системой или системным элементом, чтобы соответствовать контракту, стандарту, спецификации или другому официально установленному документу [2 IEEE 610.12].

Требование - требуемая (ожидаемая) количественная или качественная характеристика или свойство объекта, а также связанные ограничения и условия [3 ГОСТР 59194— 2020].

В жизненном цикле разработки и поддержки системы обычно выделяют несколько классов требований, каждый из которых имеет своё назначение. Ниже перечислены основные классы требований и их назначение [1]:

Классы требований

Требование заинтересованных сторон (Stakeholder Requirement) или **требования пользователя** (User requirements - **StRS**) включают: требуемые **входные данные/выборы/информационные наблюдения**, которые пользователи/операторы/специалисты по обслуживанию должны выполнять посредством использования системы; **выходные данные**, которые им требуются от системы для выполнения этих задач; любые применимые условия или ограничения, регулирующие их взаимодействие с системой ...

Бизнес-требования (Business Requirement) определяют включают: **цели и задачи бизнеса, которые должна решать система, функциональность системы**

Системные требования (System Requirement), включают следующие классы требований проекта:

- **Функциональные требования** (Functional requirements)
- **Требования к удобству использования** (Usability requirements).
- **Требования к производительности** (Performance requirements).
- **Системные интерфейсы** (System interfaces)
- **Требования к интеграции человека и системы** (Human system integration requirements)
- **Требования к ремонтопригодности** (Maintainability)
- **Требования к надежности** (Reliability)
- **Физические требования** (Physical requirements)
- **Требования к адаптивности** (Adaptability requirements)
- **Требования к безопасности системы** (System security) — требования к безопасности
- **Нефункциональные требования** (Non-functional requirements)
- **Требования к данных (**Data requirements**)**
- **Требования к процессам (**Process requirements) …

6.2. ТРЕБОВАНИЯ В ЖИЗНЕННОМ ЦИКЛЕ СИСТЕМ

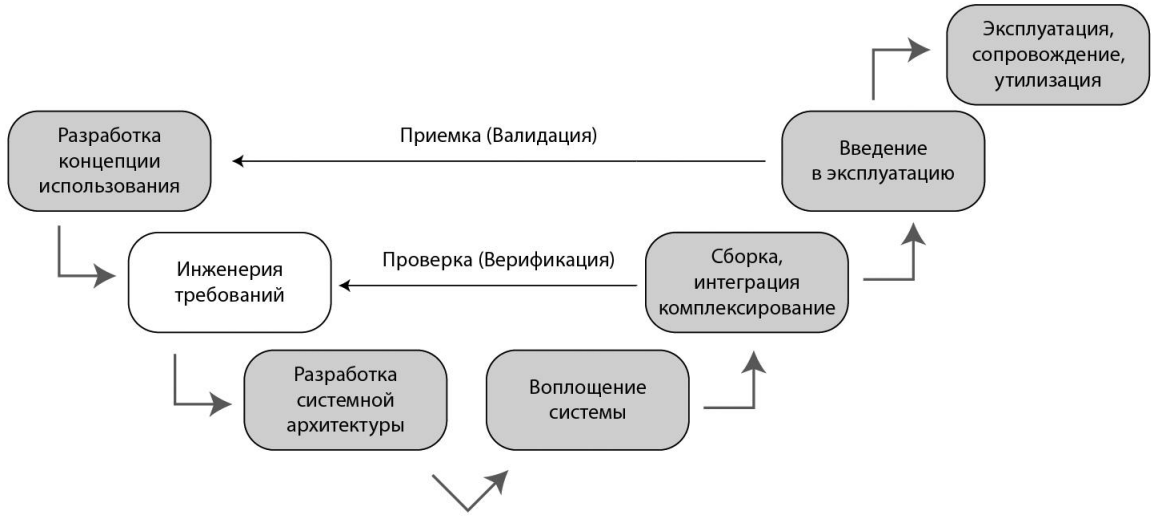
Жизненный цикл системы начинается со стадии «**определения концепции**». На первых шагах этой стадии заказчик (или предприятие) идентифицирует новые желательные возможности системы в виде **потребностей (needs) заинтересованных сторон**. Они недостаточно проработаны

Затем проводится **бизнес-анализ** или **анализ миссии**, результатом которого являются высокоуровневый набор стратегий и потребностей, выраженные в виде **бизнес-требований**, отражающих миссию проекта

Далее проводится анализ возможных решений, направленных на удовлетворение потребности заинтересованных сторон, с целью уточнений потребностей и их трансформации в **требования заинтересованных сторон** (представление системы с точки зрения пользователя)

Следующей стадией жизненного цикла систем является стадия **«определения системы»**. На этой стадии требования заинтересованных сторон преобразуются в **системные требования**

Описанные выше процессы и составляют основу инженерии требований, место которой в жизненном цикле систем иллюстрируется на слайде [4]



Поскольку определение системы формируется рекурсивно, то далее определяются элементы системы посредством декомпозиции системных требований с использованием деталей более низкого уровня решения в требования более низкого уровня абстракции

На самом высоком уровне идеальное требование не зависит от реализации и, следовательно, не относится к конкретному решению, допуская ряд возможных решений

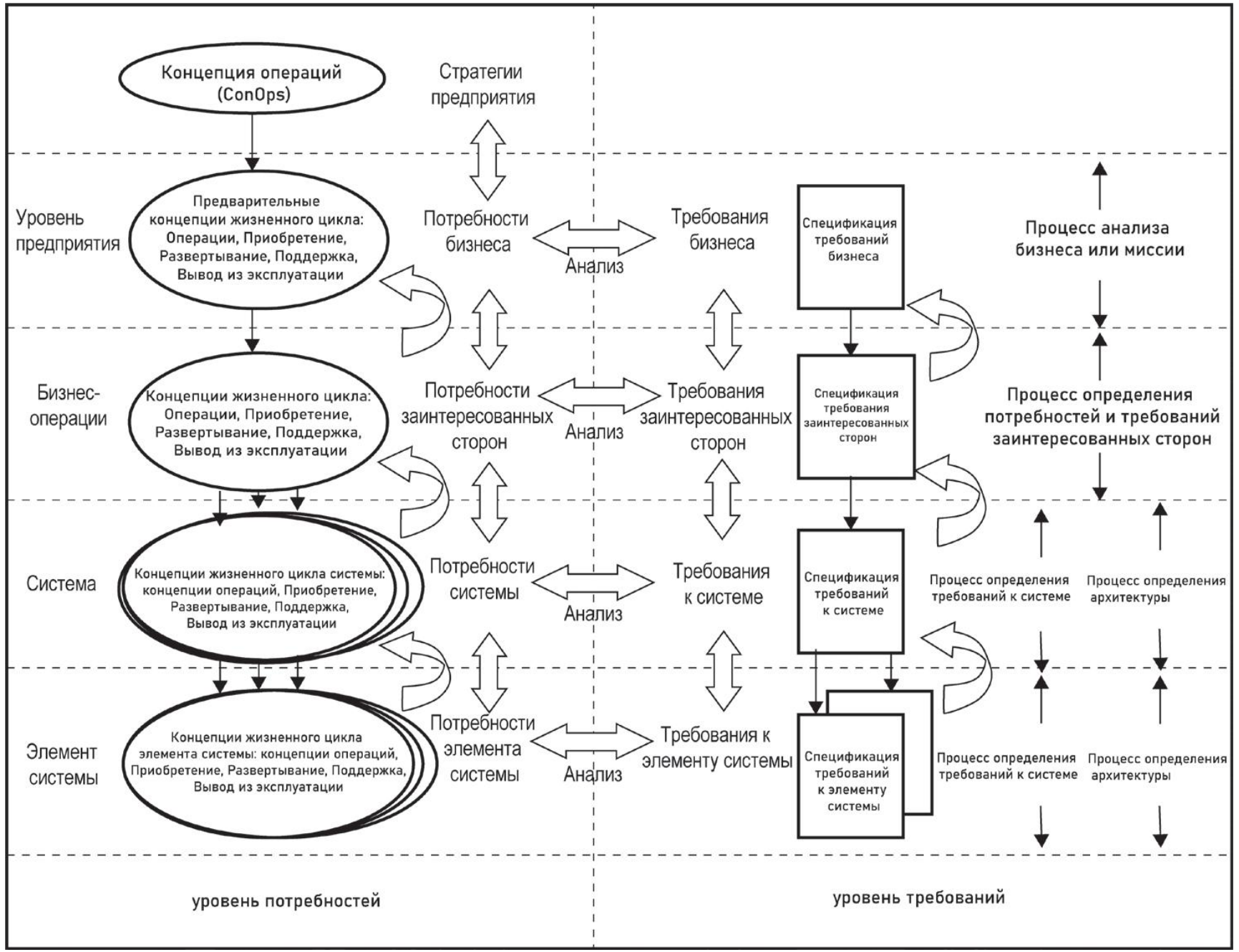
На самом низком уровне формулировки требований могут стать более конкретными для выбранного решения

Рекурсивный и итеративный характер процессов разработки требований иллюстрируется на следующем слайде [ISO/IEC ISO/IEC 29148:2018]

Рекурсивное применение описанных выше процессов позволяет создать требования к системным элементам более низкого уровня

Рекурсивный и итеративный характер процессов разработки требований

6.3. ПРЕОБРАЗОВАНИЕ ПОТРЕБНОСТЕЙ ПРЕДПРИЯТИЯ В ТРЕБОВАНИЯ



Классы спецификаций требований системы

Stakeholder Requirements Specification (StRS) — спецификации требований заинтересованных сторон
System Requirements Specification (SyRS) — системные спецификации требований
Software Requirements Specification (SRS) — спецификации требований к программному обеспечению

6.4. КОНСТРУКЦИЯ ТРЕБОВАНИЙ

- Требования заинтересованных сторон, требования к системе и требования к элементам системы должны быть четко сформулированными [1]
- Правильно сформулированное требование — это утверждение, которое позволяет проверить, удовлетворяет ли система потребностям конечного пользователя (заказчика или заинтересованного лица)
- Оно может быть написано на формальном или естественном языках
- Все термины, используемые для разработки требований, должны быть формально определены
- В требовании должен быть указан **субъект (subject)** требования (например, система, программное обеспечение и т. д.) и то, **что должно быть сделано** (shall be done) (например, выполнять действия в соответствии с установленным планом)
- В случае выражения в форме естественного языка, требование должно строиться по следующим синтаксическим правилам:

[Условие] [Субъект] [Действие] [Объект] [Ограничение]

ПРИМЕР: Когда на вход системы приходит сигнал «цель обнаружена» [Условие], система [Субъект] должна создать и инициировать [Действие] процесс «сопровождение цели» [Объект] в течение 100 млсек [Ограничение]

Или

[Условие] [Действие или ограничение] [Значение]

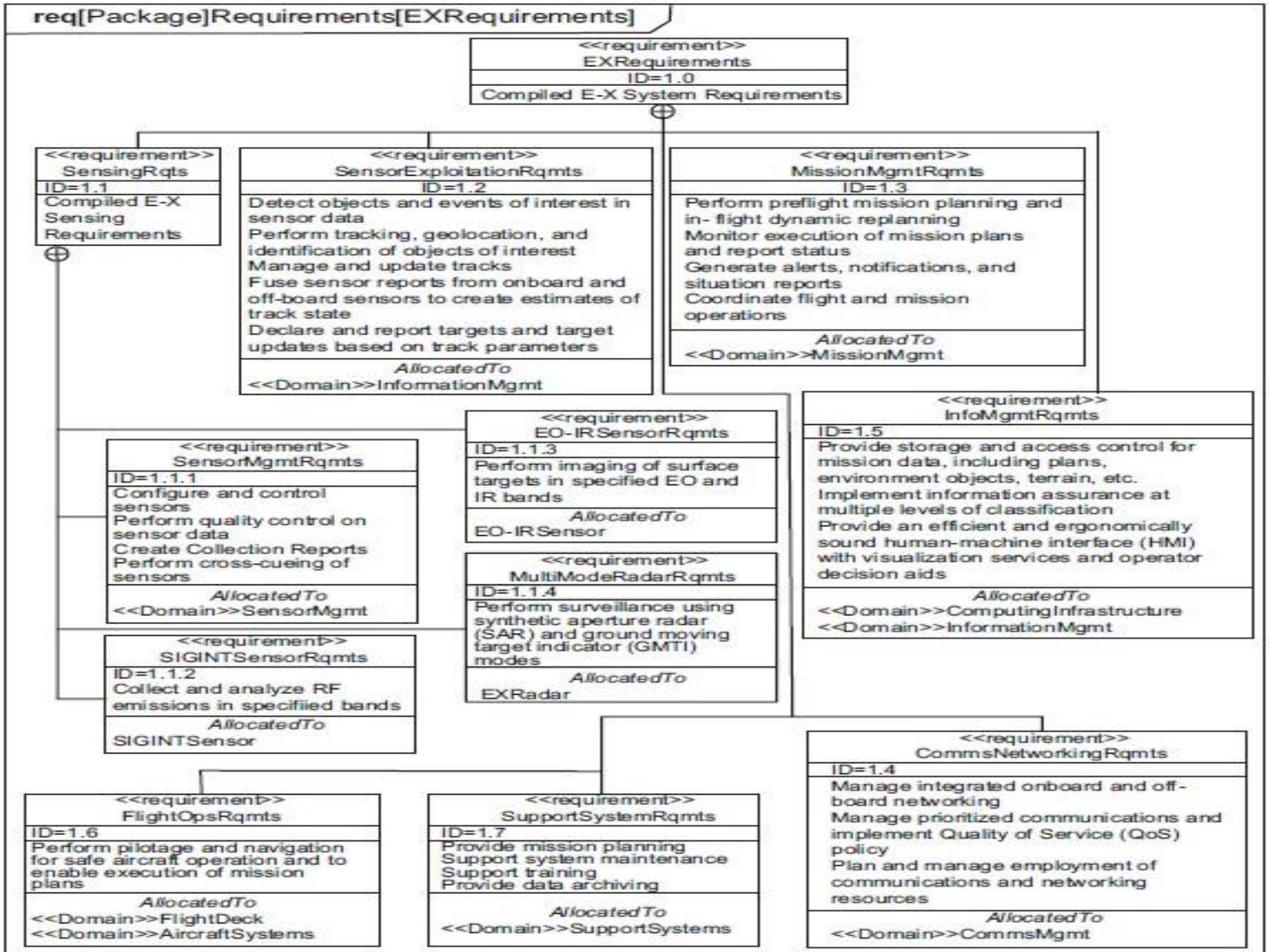
ПРИМЕР: При вхождении цели в зону обнаружения [Условие] радиолокационная система должна обнаруживать цель на расстоянии до [Действие или ограничение] 100 км [Значение]

Или

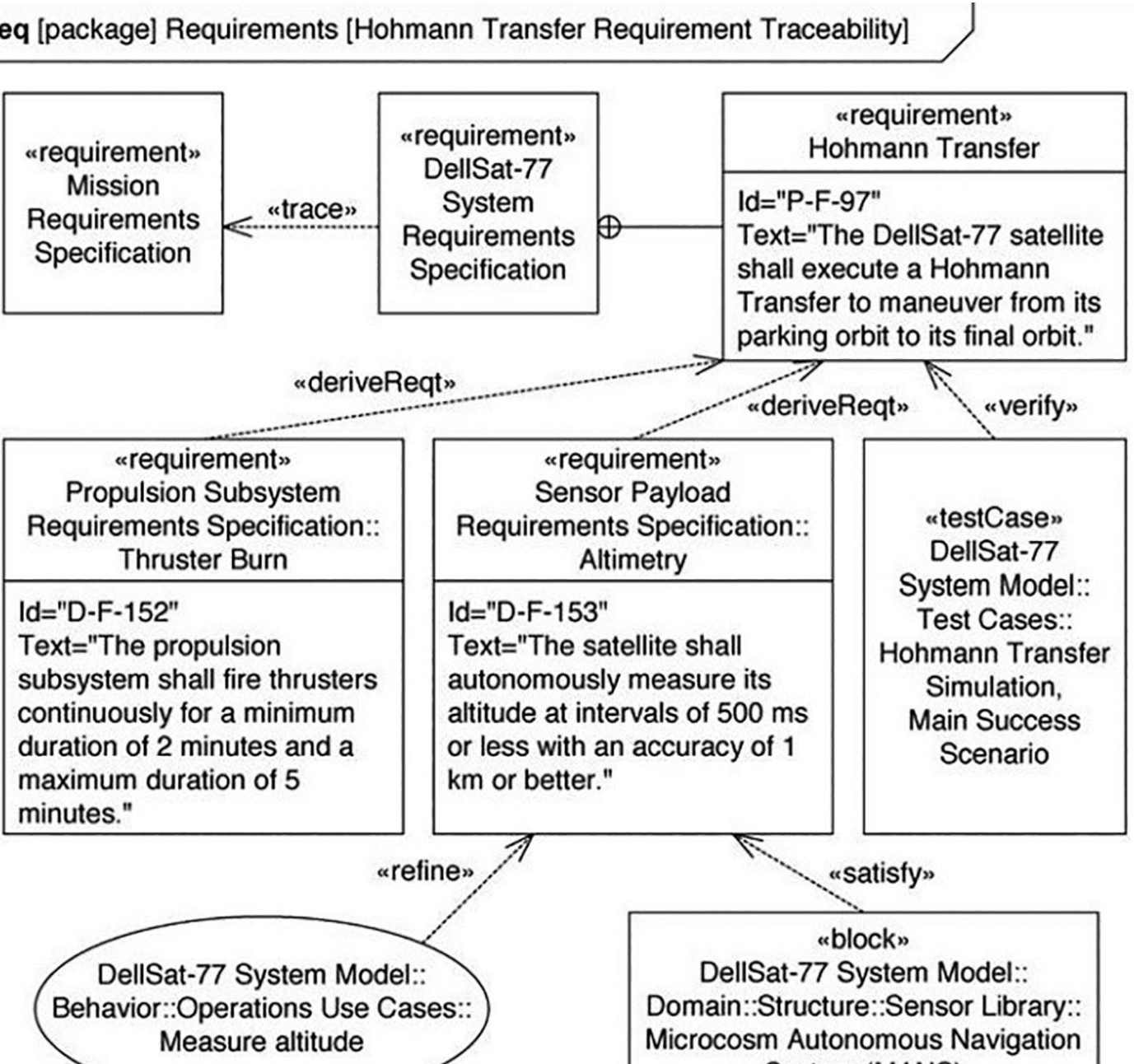
[Субъект] [Действие] [Значение]

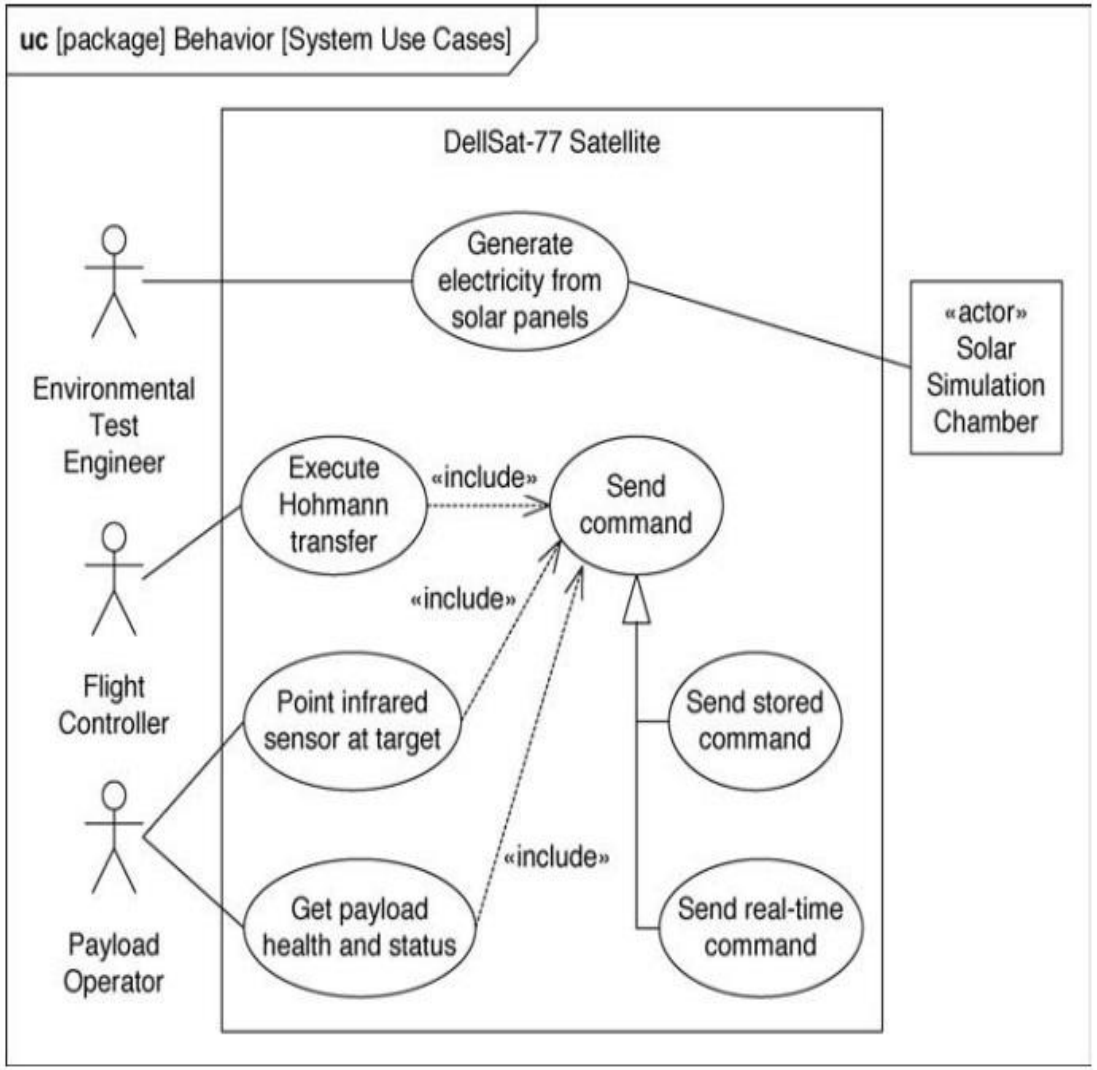
ПРИМЕР: Система обслуживания счетов [Субъект] должна отображать ожидающие счета клиентов [Действие] в порядке возрастания [Значение], в котором счета должны быть оплачены

Пример диаграммы требований на языке SysML



Пример диаграммы требований на языке SysML

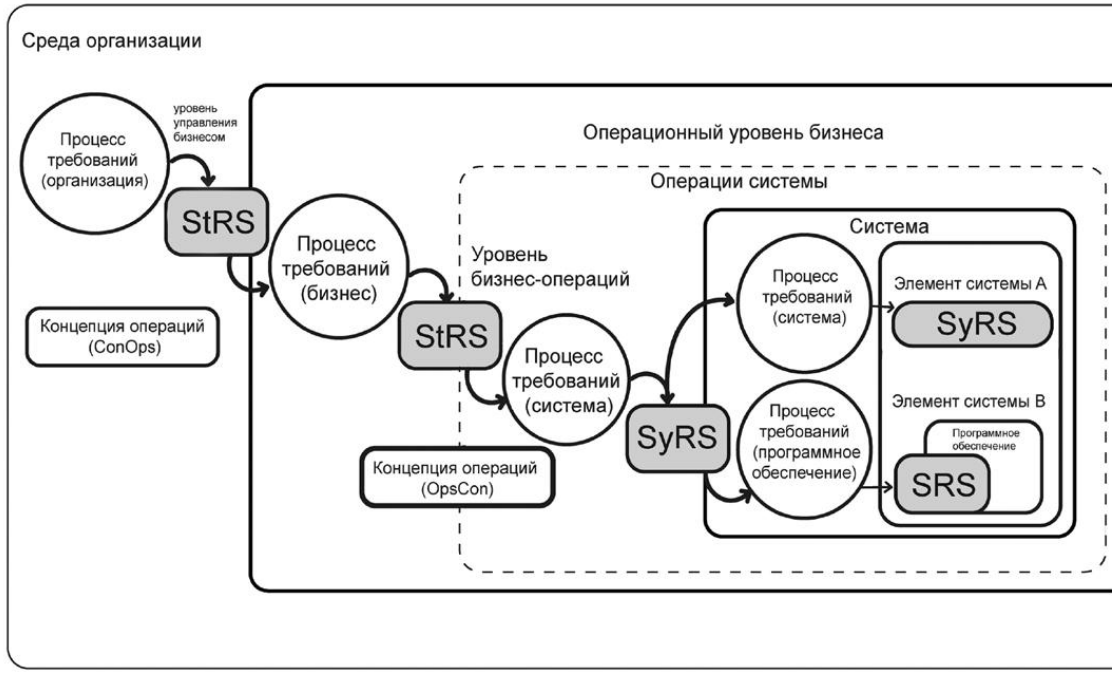




6.5. ПРОЦЕССЫ И ДЕЯТЕЛЬНОСТИ ИНЖЕНЕРИИ ТРЕБОВАНИЙ В ЖИЗНЕННОМ ЦИКЛЕ СИСТЕМЫ

В жизненном цикле создаваемой инженерной системы должны реализовываться следующие процессы требований, определенные в ISO/IEC 15288:2008 (IEEE Std 15288-2008) «Процессы жизненного цикла системы» и ISO/IEC 12207:2008 (IEEE Std 12207-2008) «Процессы жизненного цикла программного обеспечения», в зависимости от соблюдения одного или обоих стандартов ISO/IEC 15288 и ISO/IEC 12207:

- **Процесс определения требований заинтересованн сторон** (ISO/IEC 15288:2008 (IEEE Std 15288-2008), подраздел 6.4.1 или ISO/IEC 12207:2008 (IEEE Std 12 2008), подраздел 6.4.1).
- **Процесс анализа требований** (ISO/IEC 15288:2008 (Std 15288-2008), подраздел 6.4.2, или ISO/IEC 12207:2008 (IEEE Std 12207-2008), подраздел 6.4.2).
- **Процесс анализа требований к программному обеспечению** (ISO/IEC 12207:2008 (IEEE Std 12207-2008), подраздел 7.1.2) для приобретения или поставки программных продуктов.



6.6. ПОТРЕБНОСТИ И ТРЕБОВАНИЯ БЕЗОПАСНОСТИ (PROTECTION NEEDS AND REQUIREMENTS)

Понимание природы и объема потребностей в защите является фундаментальной обязанностью инженеров по обеспечению безопасности систем. Потребности в защите определяются на основе анализа потери активов (**assets loss**) и связанных с этим последствий.

Анализ проводится с трех взаимосвязанных точек зрения, включая:

- **Перспективу заинтересованных сторон**: потребности миссии/бизнеса; цели и показатели операционной деятельности; концепции жизненного цикла; законы; нормативные, законодательные и сертификационные критерии; регулирующая политика; устойчивость к потерям и риску; аккредитация, и пр.
- **Системная перспектива**: возможность самозащиты системы; системная архитектура, системный дизайн и решения по внедрению системы; стандарты разработки, изготовления, производства; безопасное управление системой и пр.
- **Перспектива профессий бизнеса**: требования бизнеса; инжиниринг бизнеса, бизнес обработки рисков и др.

Литература

[1] ISO 29148 – 2018. Systems and software engineering — Life cycle processes — Requirements engineering. -- https://analytics.infozone.pro/requirements-in-iso-iec-29148

[2] **IEEE 610.12** Institute of Electrical and Electronics Engineers (IEEE) Std. 610.12-1990, *IEEE Standard Glossary of Software Engineering Terminology*, December 1990.

[3] ГОСТР 59194— 2020 УПРАВЛЕНИЕ ТРЕБОВАНИЯМИ Основные положения/

[4] Бухарин М.А. Инженерия требований 19.11.2020 - https://edunano.ru/upload/iblock/43d/Buharin_Requirements_Engineering_19112020.pdf

[5] Guide to the Systems Engineering Body of Knowledge (SEBoK), version 2.7. 2022.

[6] John M. Borky • Thomas H. Bradley. Effective Model-Based Systems Engineering, ISBN 978-3-319-95668-8 ISBN 978-3-319-95669-5 (eBook) https://doi.org/10.1007/978-3-319-95669-5 Library of Congress Control Number: 2018950394/ © Springer International Publishing AG, part of Springer Nature 2019. P. 788.

ГЛАВА 7. MBSE — СИСТЕМНАЯ ИНЖЕНЕРИЯ НА ОСНОВЕ МОДЕЛЕЙ

7.1. ВВЕДЕНИЕ В MBSE. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

7.2. МОДЕЛИ ЖИЗНЕННОГО ЦИКЛА SE И MBSE. V-МОДЕЛЬ В СРЕДЕ, ОСНОВАННОЙ НА МОДЕЛЯХ, РОМБ SE И ЦИФРОВОЙ ДВОЙНИК

7.3. СТРУКТУРА ДОМЕНОВ ДЛЯ МОДЕЛИ АРХИТЕКТУРЫ СИСТЕМ

7.4. АРХИТЕКТУРНО-АКЦЕНТИРОВАННЫЙ ПРОЦЕСС MBSE (MBSAP THE MODEL-BASED SYSTEM ARCHITECTURE PROCESS — MBSAP) РАЗРАБОТКИ МОДЕЛЕЙ СИСТЕМ

7.1. ВВЕДЕНИЕ В MBSE. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

- **Системная инженерия на основе моделей [MBSE]** — это парадигма, которая использует формализованные представления систем, известные как модели, для поддержки и облегчения выполнения задач системной инженерии (SE) на протяжении всего жизненного цикла системы [1]
- Международный совет по системной инженерии (INCOSE) определяет **MBSE** как реализация подхода SE, на основе принципов цифрового моделирования на уровне определения системы и имитации физического и рабочего поведения на протяжении всего жизненного цикла системы [2]
- Модели SE обычно выражаются на стандартизированном языке моделирования таком, как, например, язык моделирования систем **SysML** [3], они выражают ключевую информацию о системе в кратком, последовательном, правильном и последовательном формате
- При правильном применении модели MBSE обеспечивают стандартизированную консолидацию и интеграцию системных знаний между инженерными дисциплинами и подсистемами и оптимизируют ключевые задачи системного проектирования, а также минимизируют риски разработки
- Определение модели ...
- Свойства модели ...
- Характеристики системной модели ...
- Критерии эффективных моделей MBSE ...
- Системные модели ...

7.2. МОДЕЛИ ЖИЗНЕННОГО ЦИКЛА SE И MBSE. V-МОДЕЛЬ В СРЕДЕ, ОСНОВАННОЙ НА МОДЕЛЯХ, РОМБ SE И ЦИФРОВОЙ ДВОЙНИК

Модель SE «V» обладает рядом недостатков, в частности [5]:

- Компоновка «V» предполагает только последовательный процесс разработки
- Происхождение SE «V» относится к эпохе «**документоцентризма**», когда информация буквально передавалась между отдельными людьми, группами и фазами жизненного цикла SE «V» также не отражает интегрированный и итеративный характер разработки продукта Попытки обновить символ «V» только усложнили его и сделали его еще более трудным для понимания
- Последовательность V-процесса затрудняет обмен данными и информацией в реальном времени между сложными взаимодействиями экосистемы MBE

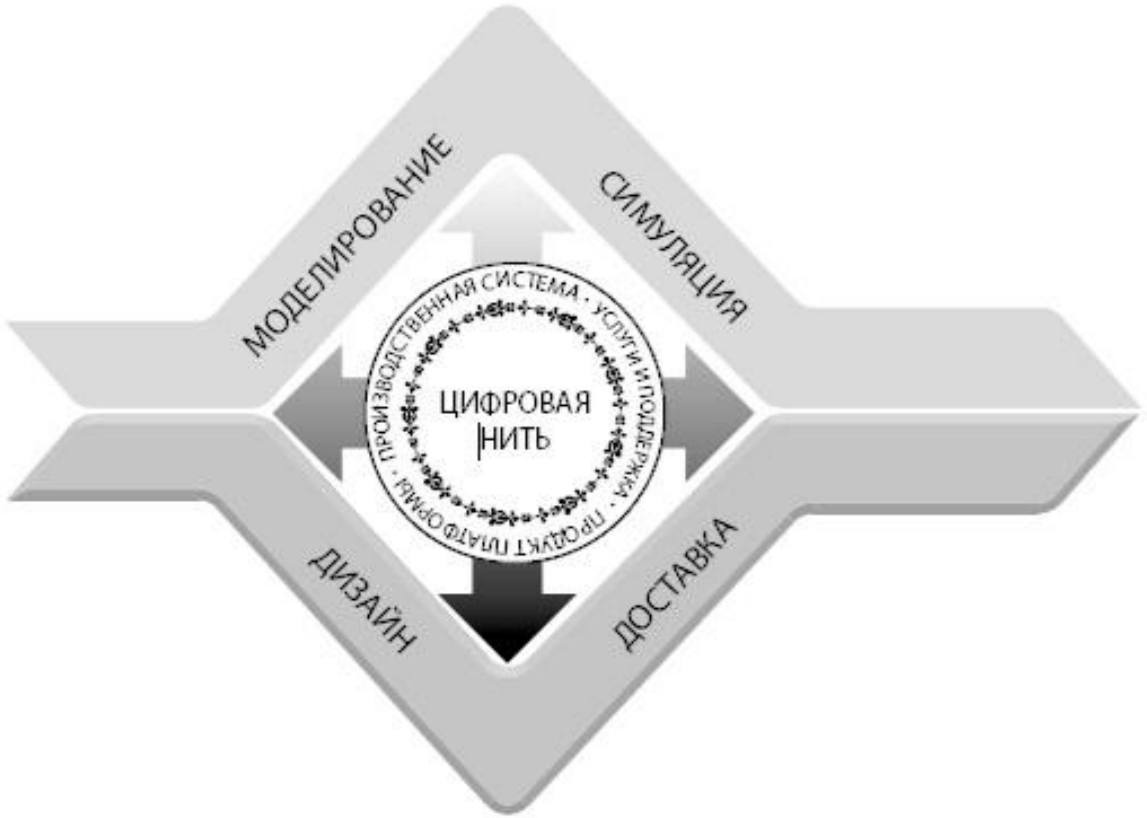
Цифровая нить (цифровой поток)

MBSE предполагает переход от мышления, ориентированного на документы, к мышлению цифрового проектирования, которое использует **цифровую нить или цифровой поток (Digital thread)** на протяжении всего жизненного цикла

Центральной задачей становится разработка, как физического, так и виртуального представления целевой системы

Это само по себе приводит к **циклу разработки виртуального продукта**, который можно рассматривать как «**зеркальное отражение**» цикла разработки физического продукта

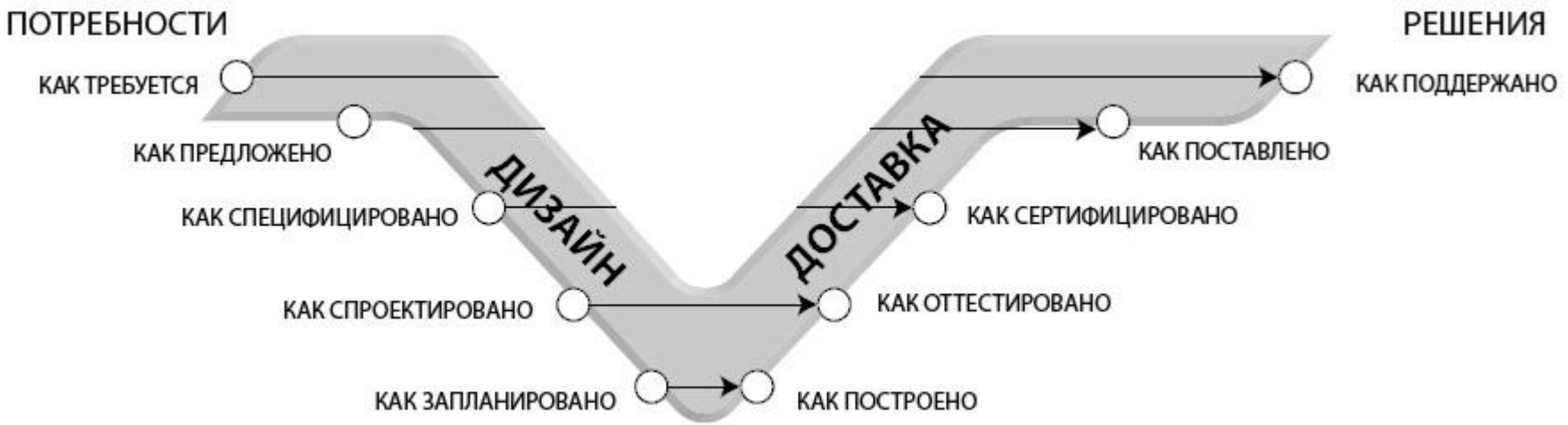
Компания **Boeing** разработала подход (**SE Diamond**), демонстрирующий принципы использования конфигураций цифровых двойников как самостоятельных продуктов, которые зеркально отражают артефакты проекты физического продукта: **«как спроектировано»**, **«как построено»**, **«как поставлено» и т.п**.



MBE - многомерный итеративный процесс создания как физических, так и виртуальных реализаций

- Именно разработка виртуальных систем или цифровых двойников, которые влияют на развитие физических систем - одно из основных изменений при переходе от SE к MBE
- Ромб SE обновляет традиционную букву «V», добавляя виртуальное представление продукта, производственной системы, а также услуг и систем поддержки
- **Ромб SE разделяет физический и виртуальный пути на две V, которые обращены друг к другу (сверху и снизу)**
- Нижняя половина MBE Diamond, совпадает с традиционной SE «V» и ориентирована на создание экземпляров физических систем на протяжении всего жизненного цикла (с сохранением традиционного потока SE «V»)
- «V» начинается с потребностей клиентов и заканчивается готовыми решениями для удовлетворения этих потребностей
- И для нижней буквы «V», и для верхней буквы «V»существует симметрия между артефактами слева и справа. Модели, разработанные на левой стороне Ромба SE, используются для все более точного моделирования на правой стороне

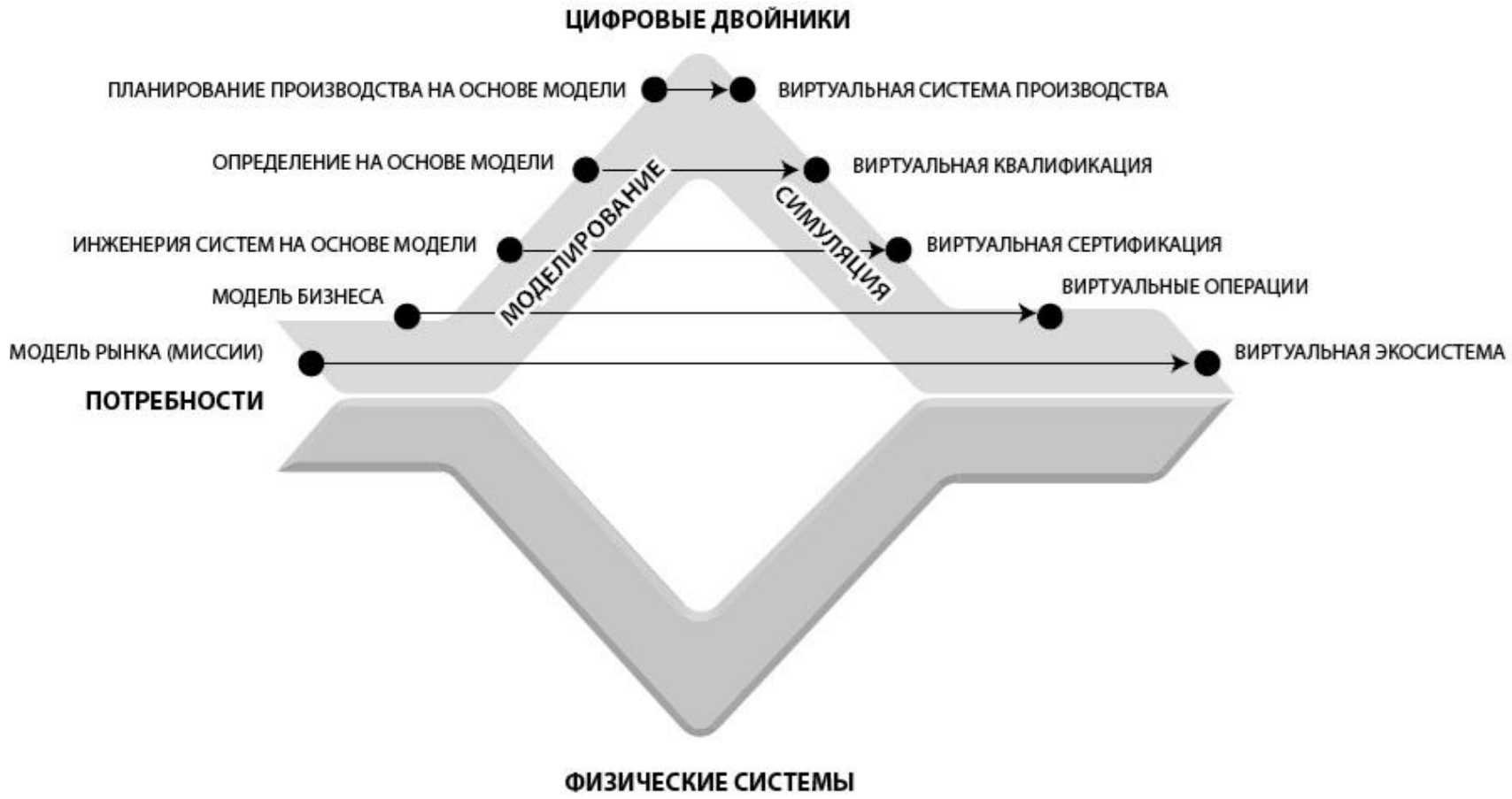
Нижняя часть Ромба SE [4]



Нижняя часть Ромба SE [4]

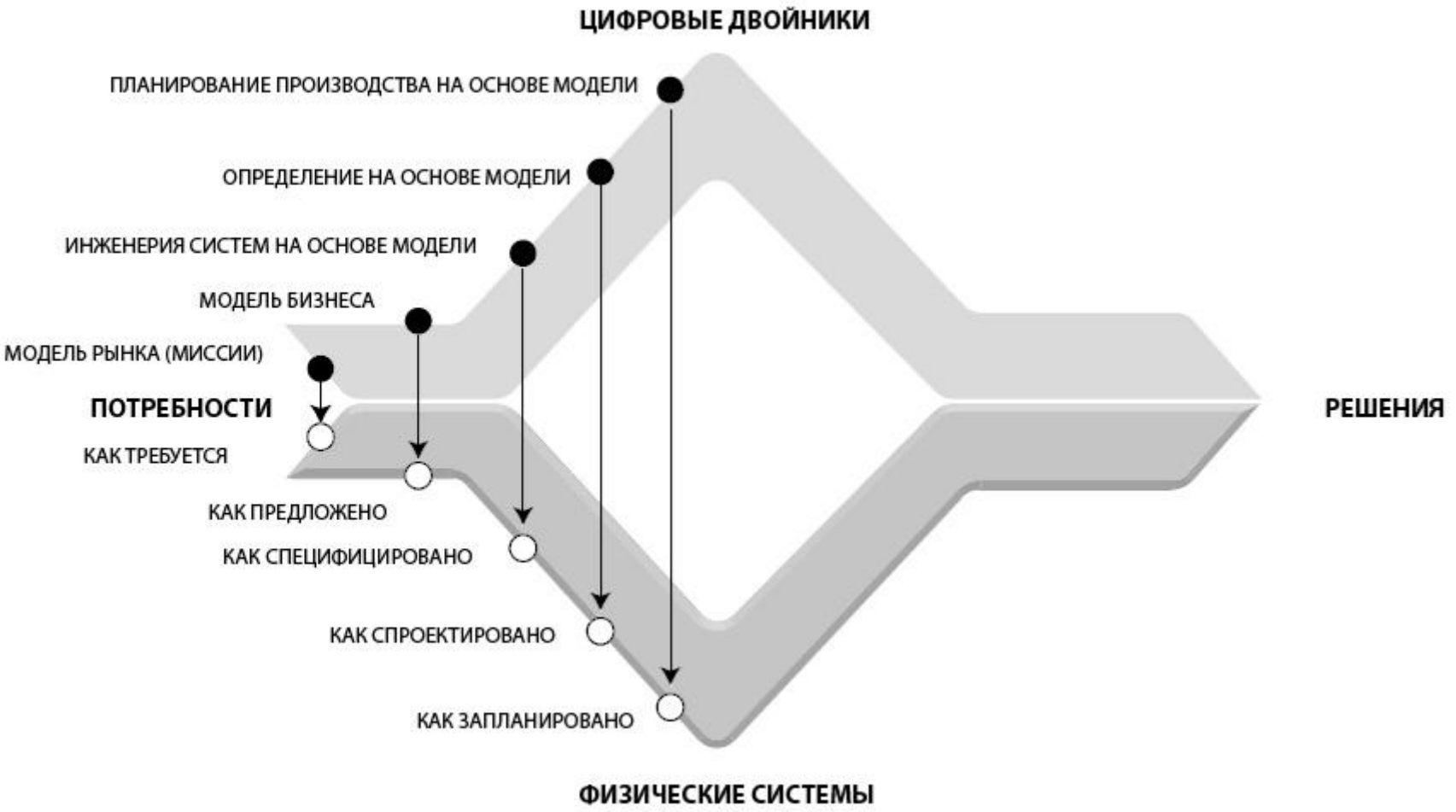
Нижняя половина MBE Diamond совпадает с традиционной SE V и ориентирована на создание экземпляров физических систем на протяжении всего жизненного цикла (с сохранением традиционного потока SE V). V начинается с потребностей клиентов и заканчивается готовыми решениями для удовлетворения этих потребностей. Она включает следующие стадии: КАК ТРЕБУЕТСЯ (AS NEEDED), КАК ПРЕДЛОЖЕНО (AS OFFERED), КАК СПЕЦИФИЦИРОВАНО (AS SPECIFIED), КАК СПРОЕКТИРОВАНО (AS DESIGNED), КАК ЗАПЛАНИРОВАНО (AS PLANNED), КАК ПОСТРОЕНО (AS BUILT), КАК ОТТЕСТИРОВАНО (AS TESTED), КАК СЕРТИФИЦИРОВАНО (AS CERTIFIED), КАК ПОСТАВЛЕНО (AS DELIVERED), КАК ПОДДЕРЖАНО (AS SUPPORTED).

Ромб SE (SE diamond)



Верхняя половина Ромба SE [4]

Ромб SE (SE diamond)



Цифровые двойники и физические системы (слева) [4]

Ромб SE (SE diamond)

Симметрия между артефактами для виртуальных и физических систем. Физические системы могут использоваться для информирования виртуальных систем; виртуальные системы могут использоваться для информирования физических систем. Например, аналитика производственных данных из цеха может использоваться для обновления имитационных моделей производства для проведения анализа «что, если» в режиме реального времени, связанного с изменениями производительности. Также данные высокоточного виртуального моделирования системы, могут быть использованы для уменьшения потребности в физическом тестировании.

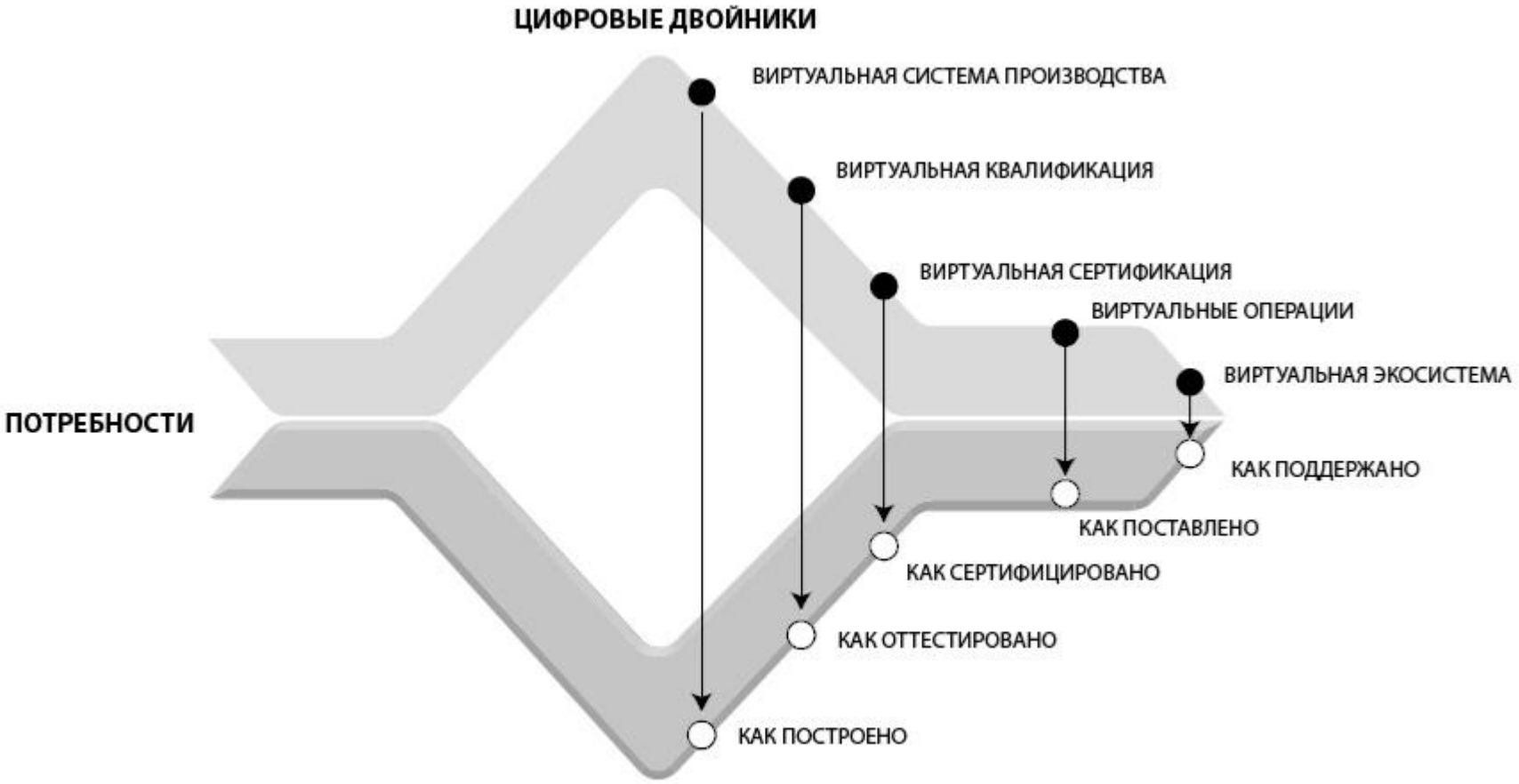


Рисунок 7.7. Цифровые двойники и физические системы (справа) [4]

MBE - интегрированный характер разработки как физического, так и виртуального продукта

MBE Diamond рассматривает разработку цифрового двойника и физического двойника как параллельные пути, которые информируют друг друга на протяжении жизненного цикла продукта

Цифровая нить (Digital thread), представленная внутри ромба, связывает модели/моделирование (цифровые близнецы) с физическим дизайном систем (продукт-платформа, производственная система, услуги и поддержка)

Процесс интегрирован через цифровую нить, включая: продукт платформы, производственную систему, а также услуги и систему поддержки

7.3. СТРУКТУРА ДОМЕНОВ ДЛЯ МОДЕЛИ АРХИТЕКТУРЫ СИСТЕМ

Рисунок 7.9. Структура основных доменов системной инженерии и соответствующих артефактов, определяющих структуру архитектурной и функциональной модели системы

7.4. АРХИТЕКТУРНО-АКЦЕНТИРОВАННЫЙ ПРОЦЕСС MBSE (MBSAP THE MODEL-BASED SYSTEM ARCHITECTURE PROCESS) РАЗРАБОТКИ МОДЕЛЕЙ СИСТЕМ

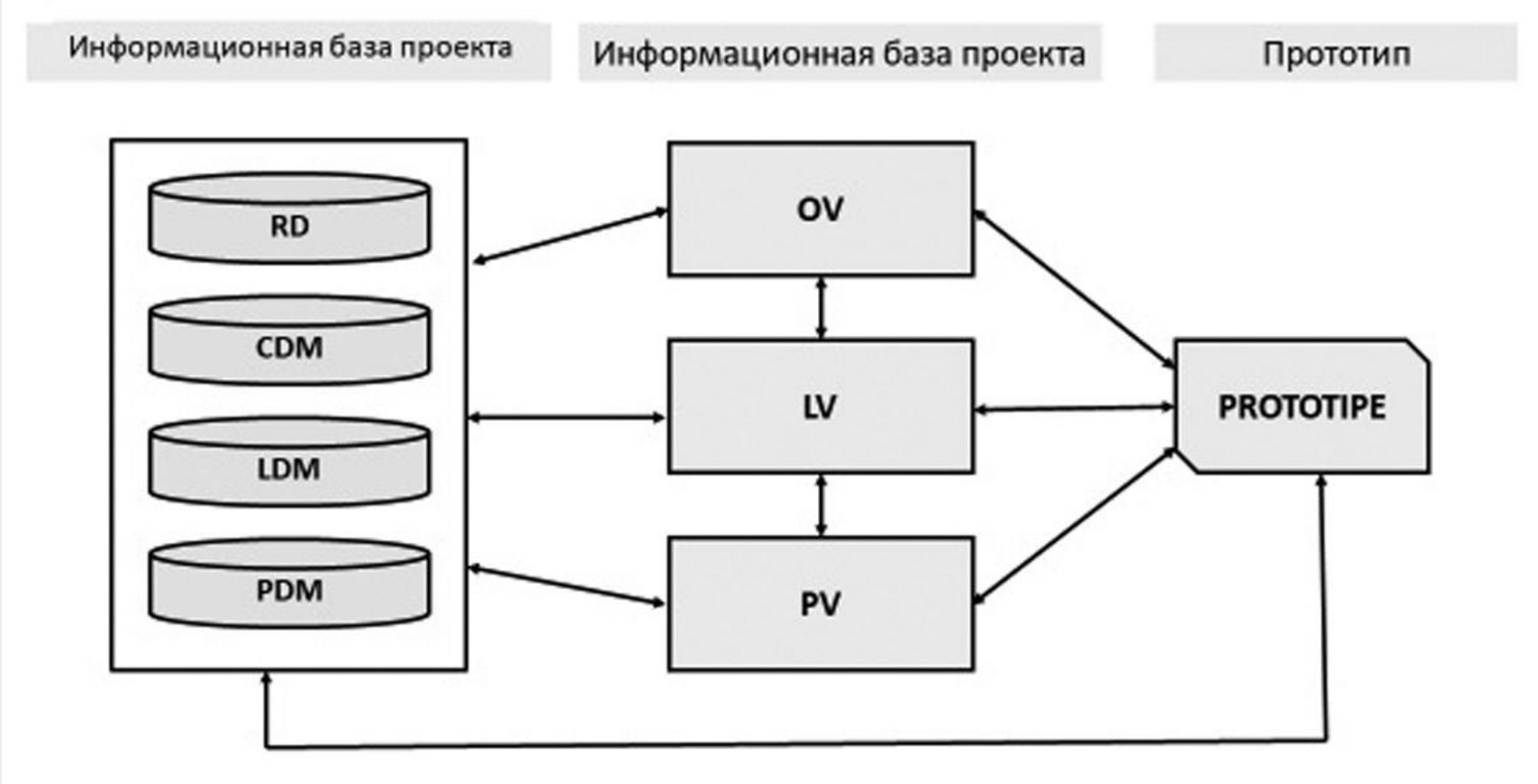


Рисунок 7.10. Общая модель подхода MBSAP, где RD – база требований (возможностей), CDM – концептуальная модель данных, LDM – логическая модель данных, PDM – физическая модель данных, стадии: OV - операционной точки зрения, LV - логической/функциональной точки зрения, PV – физической точки зрения

Литература к главе 7

[1] Guide to the Systems Engineering Body of Knowledge (SEBoK), version 2.7. 2022.

[2] Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, Fourth Edition, INCOSE, 2015.

[3] OMG Systems Modeling Language (OMG SysML™). Version 1.4. http://www.omg.org/spec/SysML/1.4/

[4] Model-Based Systems Engineering and Model-Based Safety Analysis: Final Report. January 2021. Final report. U.S. Department of Transportation. Federal Aviation Administration. DOT/FAA/TC-20/42. Federal Aviation Administration. William J. Hughes Technical Center Aviation Research Division. Atlantic City International Airport. New Jersey 08405/

[5] Jeff, A. Estefan. INCOSE MBSE Initiative. Survey of Candidate Model-Based Engineering (MBSE) Methodologies Page 1 of 70 Rev / A. Jeff. — B May 23, 2008. INCOSE MBSE Initiative. Survey of Model-Based Systems Engineering (MBSE) Methodologies. Jet Propulsion Laboratory. California Institute of Technology. Pasadena, California, U.S.A.

[6] John M. Borky , Thomas H. Bradley Effective Model-Based Systems Engineering © Springer International Publishing AG, part of Springer Nature 2019, 788 p., ISBN 978-3-319-95668-8 ISBN 978-3-319-95669-5 (eBook)/https://doi.org/10.1007/978-3-319-95669-5

[7]. The Open Group (2017) TOGAF Version 9 Enterprise Edition. http://www.opengroup.org/togaf/. Accessed 18 May 2017

[8]. Zachman JA (2008) A concise definition of the Zachman framework. https://www.zachman.com/about-the-zachman-framework. Accessed 12 May 2017

ГЛАВА 8. ОПИСАНИЕ АРХИТЕКТУРЫ СИСТЕМЫ

8.1. ОПИСАНИЕ АРХИТЕКТУРЫ (ГОСТ Р 57100-2016/ISO/IEC/IEEE 42010)

8.1.1. ОПРЕДЕЛЕНИЯ ПРОЦЕССА АРХИТЕКТУРИЗАЦИИ

8.1.3. АРХИТЕКТУРА И ОПИСАНИЕ АРХИТЕКТУРЫ

8.1.4. ТРЕБОВАНИЯ К ОПИСАНИЮ АРХИТЕКТУРЫ

8.2. МОДЕЛЬ ЗАХМАНА ОПИСАНИЯ АРХИТЕКТУРЫ ПРЕДПРИЯТИЯ

8.3. ПРИМЕР ОПИСАНИЯ АРХИТЕКТУРЫ ДЛЯ ПРИЛОЖЕНИЙ ИНТЕРНЕТА ВЕЩЕЙ (IoT)

8.1. ОПИСАНИЕ АРХИТЕКТУРЫ (ГОСТ Р 57100-2016/ISO/IEC/IEEE 42010)

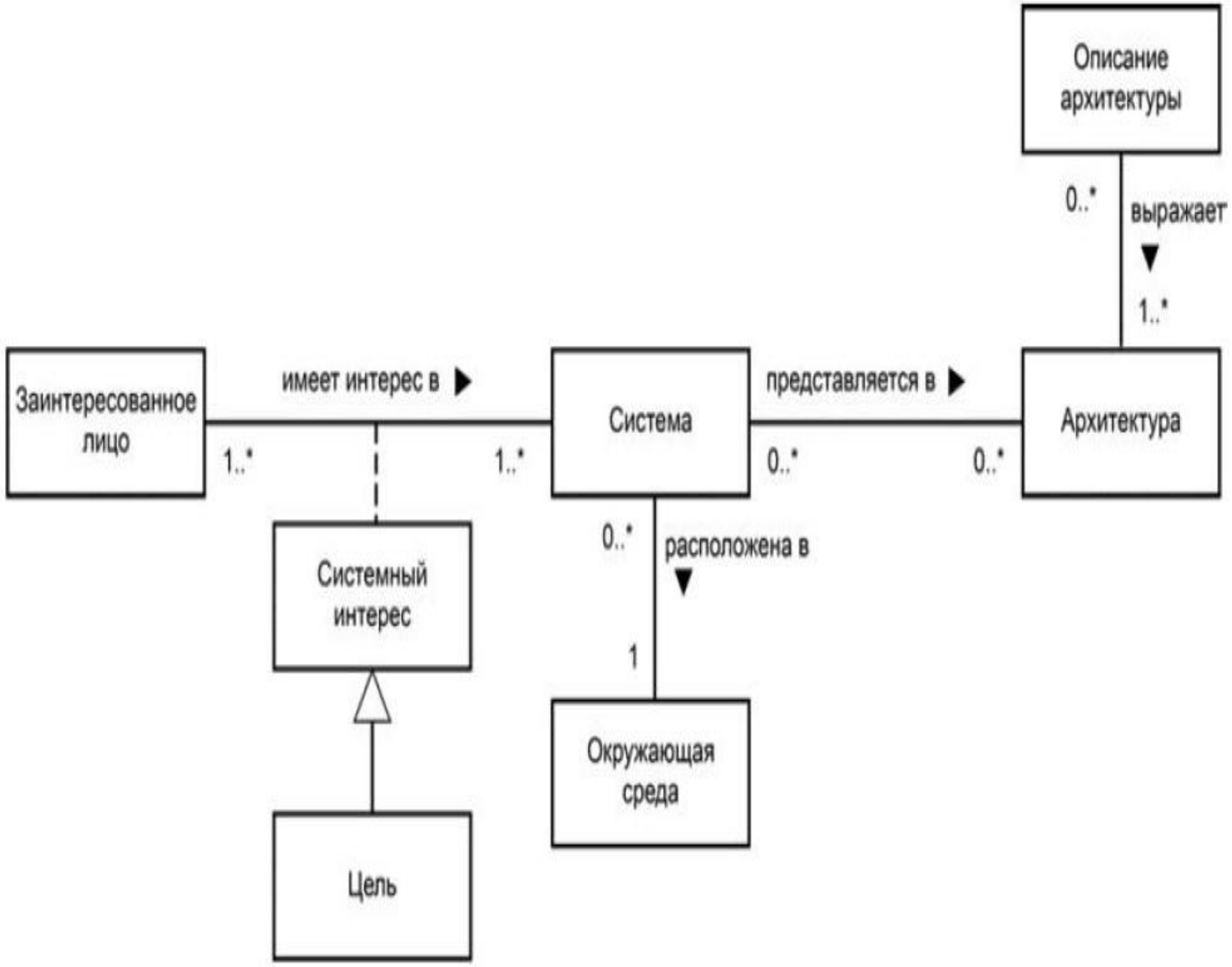
•Сложность искусственных систем достигла предельного уровня, что приводит к необходимости представления таких систем через призму абстракции, выявляющую основные свойства и конструктивные особенности таких систем, т. е. через архитектурные представления систем.

•**Концептуализация системной архитектуры** выражается в описании архитектуры и помогает понять сущность системы и ключевые свойства, определяющие ее поведение, состав и эволюцию, в свою очередь, влияющие на такие факторы, как целесообразность, полезность и управляемость системой

•Для создания описания архитектуры системы применяются понятия, принципы и процедуры **процесса архитектуризации**. Роль описания архитектуры системы чрезвычайно высока, так как способствует пониманию системной сути и основных свойств, имеющих отношение к ее поведению, составу и развитию.

•Концепции процесса архитектуризации, принципам создания и свойствам описания архитектуры системы посвящен стандарт **ГОСТ Р 57100-2016/ISO/IEC/IEEE 42010:2011** [1], который является одним из основных стандартов системной инженерии и рассматривается в первой части главы

8.1.2. КОНЦЕПТУАЛЬНАЯ МОДЕЛЬ ОПИСАНИЯ АРХИТЕКТУРЫ



Модели в системной архитектуре могут быть представлены диаграммами, схемами, финансовыми отчетами и другими условностями.

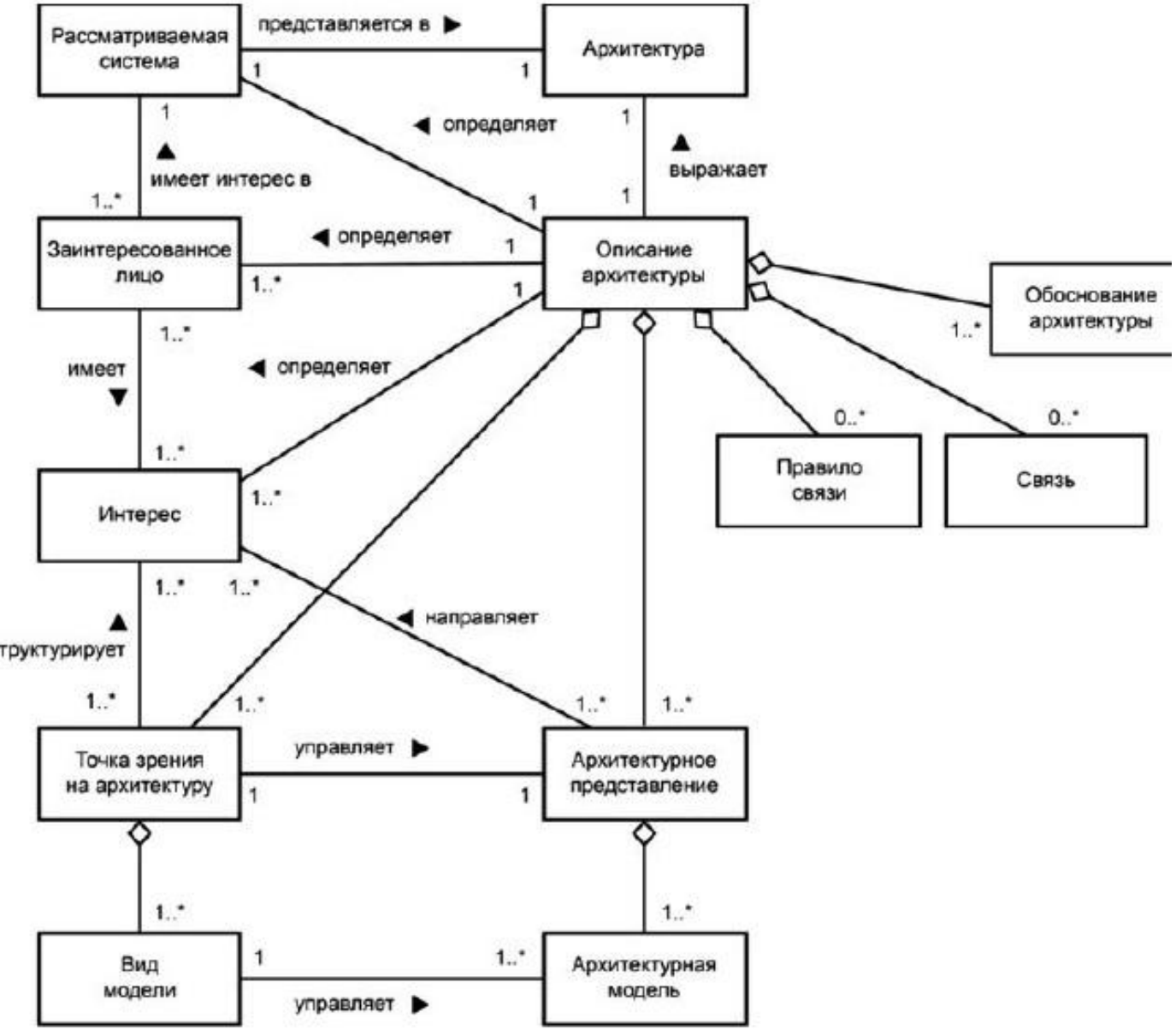
Заинтересованные стороны — это индивиды или организации, имеющие интерес в системе.

Интересы системы — это цели, выгоды или проблемы, которые важны для заинтересованных сторон.

Архитектура системы — это основные концепции, элементы и отношения внутри системы, которые воплощены в её проекте и развитии.

Описание архитектуры фиксирует структуру системы и помогает заинтересованным сторонам понять, как она функционирует.

8.1.3. АРХИТЕКТУРА И ОПИСАНИЕ АРХИТЕКТУРЫ
ГОСТ Р 57100-2016 и ISO/IEC 15288: Основы системной архитектуры



- Согласно стандартам ГОСТ Р 57100-2016 и ISO/IEC 15288, системы состоят из следующих компонентов: аппаратные и программные средства, данные, люди, процессы, процедуры, оборудование, материалы.
- Различие между архитектурой системы и описанием архитектуры:
  - **Архитектура системы** — это абстракция, отображающая ключевые свойства и принципы проектирования системы.
  - **Описание архитектуры** — это рабочий продукт, документирующий архитектуру.
- Описание архитектуры системы включает две основные группы понятий:
  - Понятия, описывающие контекст системы (её среду и границы).
  - Понятия, описывающие модели системы (структуру и взаимодействие её компонентов).

8.1.5. РАЗВИТИЕ АРХИТЕКТУРНЫХ СТАНДАРТОВ

ISO/IEC/IEEE 42030 определяет средства для организации и регистрации оценок архитектуры для предприятий, систем и областей применения программного обеспечения. Целью этого документа является обеспечение возможности проведения оценок архитектуры, которые используются для:

a) подтверждения того, что архитектура отвечает интересам заинтересованных сторон;

b) оценки качества архитектур в соответствии с их целевым назначением;

c) оценки ценности архитектур для заинтересованных сторон;

d) определения, соответствуют ли объекты архитектуры своему прямому назначению;

e) предоставления знаний и информации об объектах архитектуры;

f) оценки прогресса в достижении целей архитектуры;

g) уточнения понимания проблемного пространства, а также потребностей и ожиданий заинтересованных сторон;

Рисунок 8.3. Семейство стандартов 420x0

ISO/IEC/IEEE 42020 определяет **процессы архитектуры**, такие как:

• Управление архитектурой (Architecture Governance) — создание и поддержание соответствия архитектур в коллекции архитектур целям, политике и стратегиям предприятия, а также связанным с ними архитектурам;

• Менеджмент архитектурой (Architecture Management) — внедрение директив по управлению архитектурой для своевременного, эффективного и действенного достижения целей по сбору архитектурных данных;

• Концептуализация архитектуры (Architecture Conceptualization).Определение проблемного пространства и определение подходящих решений, которые решают проблемы заинтересованных сторон, достигают целей архитектуры и отвечают соответствующим требованиям;

• Оценка архитектуры (Architecture Evaluation) — определение, в какой степени одна или несколько архитектур соответствуют своим целям, решают проблемы заинтересованных сторон и соответствуют соответствующим требованиям;

• Разработка архитектуры (Architecture Elaboration) — описание и документирование достаточно полным и правильным образом для предполагаемого использования архитектуры;

• Поддержка архитектуры (Architecture Enablement) — разработка, поддержание и улучшение возможностей, услуг и ресурсов, необходимых для выполнения других архитектурных процессов

8.2. МОДЕЛЬ ЗАХМАНА ОПИСАНИЯ АРХИТЕКТУРЫ ПРЕДПРИЯТИЯ

	Данные ЧТО	Функции КАК	Дислокация ГДЕ	Люди КТО	Время КОГДА	Мотивация ПОЧЕМУ	
Бизнес-руководители	Список важных понятий и объектов	Список основных бизнес-процессов	Территориальное расположение	Ключевые организации	Важнейшие события	Бизнес-цели и стратегии	Сфера действия (контекст)
	Концептуальная модель данных	Модель бизнес-процессов	Схема логистики	Модель потока работ (workflow)	Мастер-план реализации	Бизнес-план	Модель предприятия
ИТ-менеджеры и разработчики	Логические модели данных	Архитектура приложений	Модель распределенной архитектуры	Архитектура интерфейса пользователя	Структура процессов	Роли и модели бизнес-правил	Модель системы
	Физическая модель данных	Системный проект	Технологич. архитектура	Архитектура презентации	Структуры управления	Описания бизнес-правил	Технологическая (физическая) модель
	Описание структуры данных	Программный код	Сетевая архитектура	Архитектура безопасности	Определение временных привязок	Реализация бизнес-логики	Детали реализации
	Данные	Работающие программы	Сеть	Реальные люди, организации	Бизнес-события	Работающие бизнес-стратегии	Работающее предприятие

Рисунок 8.5. Модель Захмана [https://bstudy.net/1003370/tehnika/model_zahmana]

8.3. ПРИМЕР ОПИСАНИЯ АРХИТЕКТУРЫ ДЛЯ ПРИЛОЖЕНИЙ ИНТЕРНЕТА ВЕЩЕЙ (IoT)

Литература к Главе 8

[1] ГОСТ Р 57100-2016/ISO/IEC/IEEE 42010:2011. Системная и программная инженерия. Описание архитектуры. Электронный ресурс] URL: https://files.stroyinf.ru/Data/634/63426.pdf

[2] ISO/IEC/IEEE FDIS 15288:2023. Systems and software engineering — System life cycle processes.

[3] ISO/IEC/IEEE 12207. Systems and software engineering. Software life cycle processes.

[4] ISO/IEC/IEEE 42020. Software, systems and enterprise — Architecture processes.

[5] ISO/IEC/IEEE 42030. Software, systems and enterprise — Software, systems and enterprise — Architecture processes.

[6] JOHN ZACHMAN'S CONCISE DEFINITION OF THE ZACHMAN FRAMEWORK™ BY JOHN A. ZACHMAN © 2008 John A. Zachman, Zachman International, Inc. Электронный ресурс] URL: https://www.zachman.com/about-the-zachman-framework, Электронный ресурс] URL: https://bstudy.net/1003370/tehnika/model_zahmana.

[7] The Open Group Architecture Framework. The Open Group Architecture Framework (TOGAF). Электронный ресурс] URL: — https://sparxsystems.com/enterprise_architect_user_guide/15.2/model_domains/togaf.html

[8] International Journal of Open Information Technologies ISSN: 2307-8162 — Vol. 6, no.7, 2018. О сложностях киберзащиты информационных систем. М.А. Шнепс-Шнеппе, В.А. Сухомлин, Д.Е. Намиот.

[9] Зиндер, Е.З. Методы архитектурного подхода для обеспечения результативности и эффективности электронного правительства: Учебное пособие / Е.З. Зиндер. — СПб.: НИУ ИТМО,

ГЛАВА 9. ЦИФРОВЫЕ ДВОЙНИКИ

9.1. КРАТКАЯ ИСТОРИЯ И ОПРЕДЕЛЕНИЯ ЦИФРОВЫХ ДВОЙНИКОВ

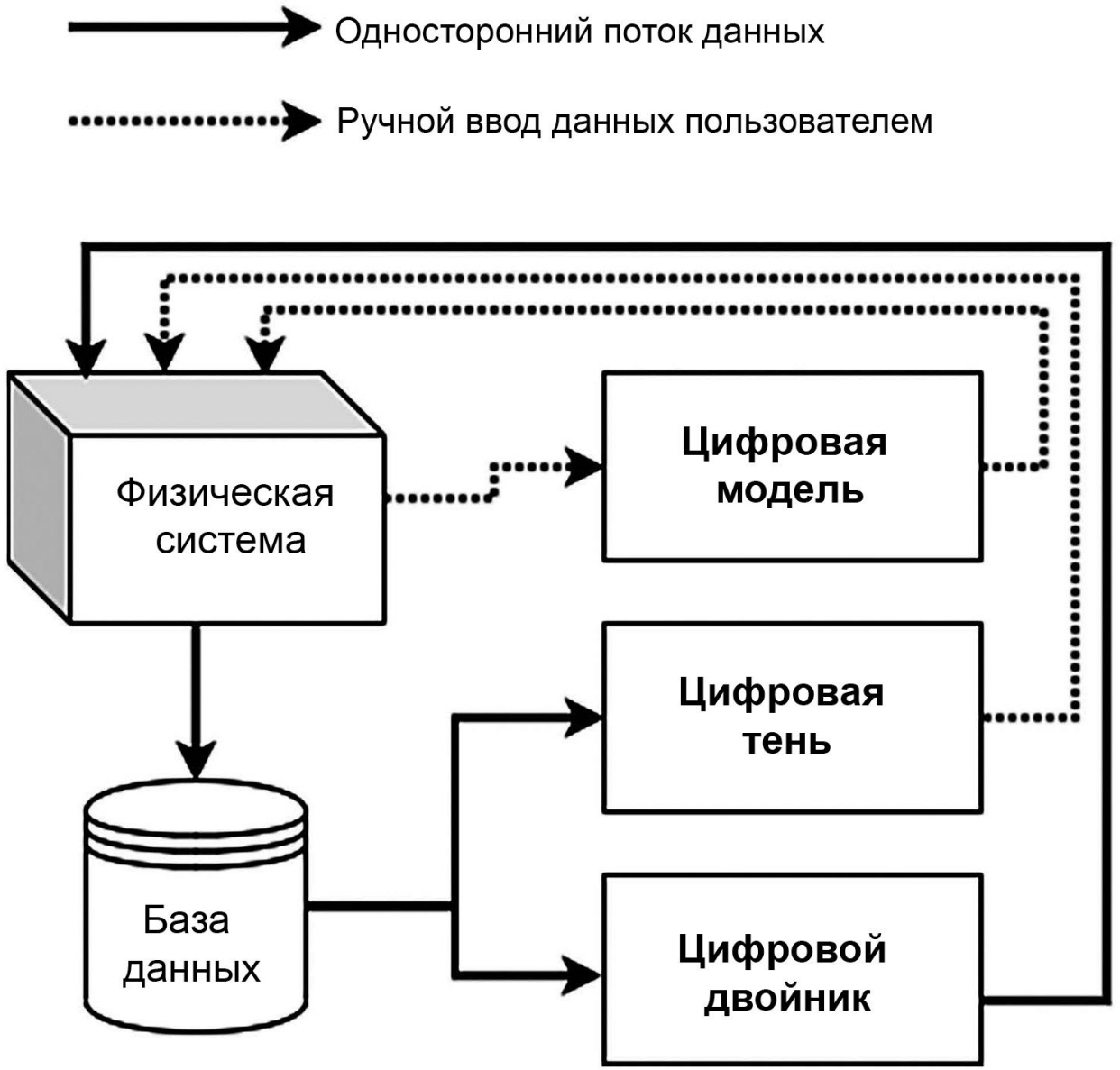


Рисунок 9.1. Поток данных от физической системы к виртуальной модели для цифровых двойников разных типов [34]

9.2. АНАЛИЗ ПРОЦЕССА СТАНДАРТИЗАЦИИ В ОБЛАСТИ ЦИФРОВЫХ ДВОЙНИКОВ

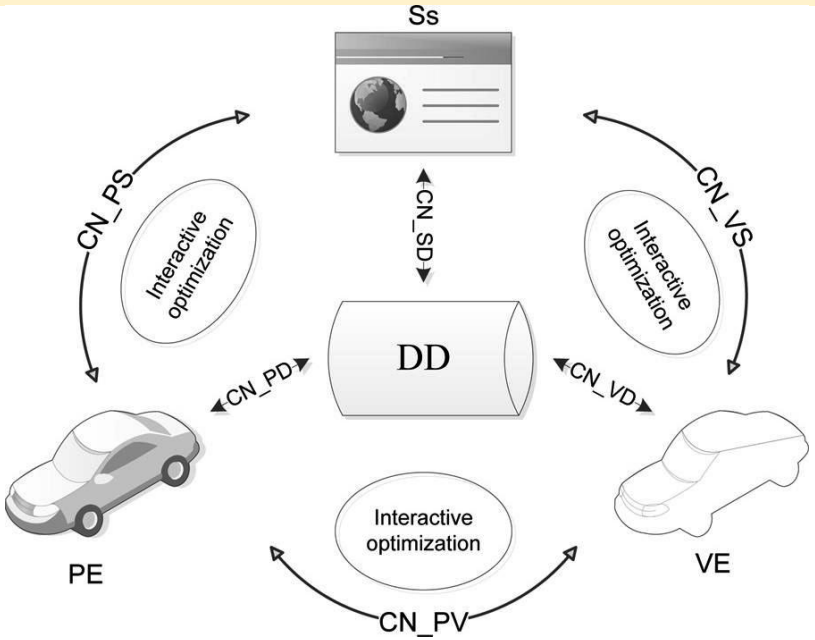


Рисунок 9.2. Структура стандартов DT [35]

9.2.2. ПРОМЫШЛЕННЫЕ СТАНДАРТЫ ЦИФРОВЫХ ДВОЙНИКОВ

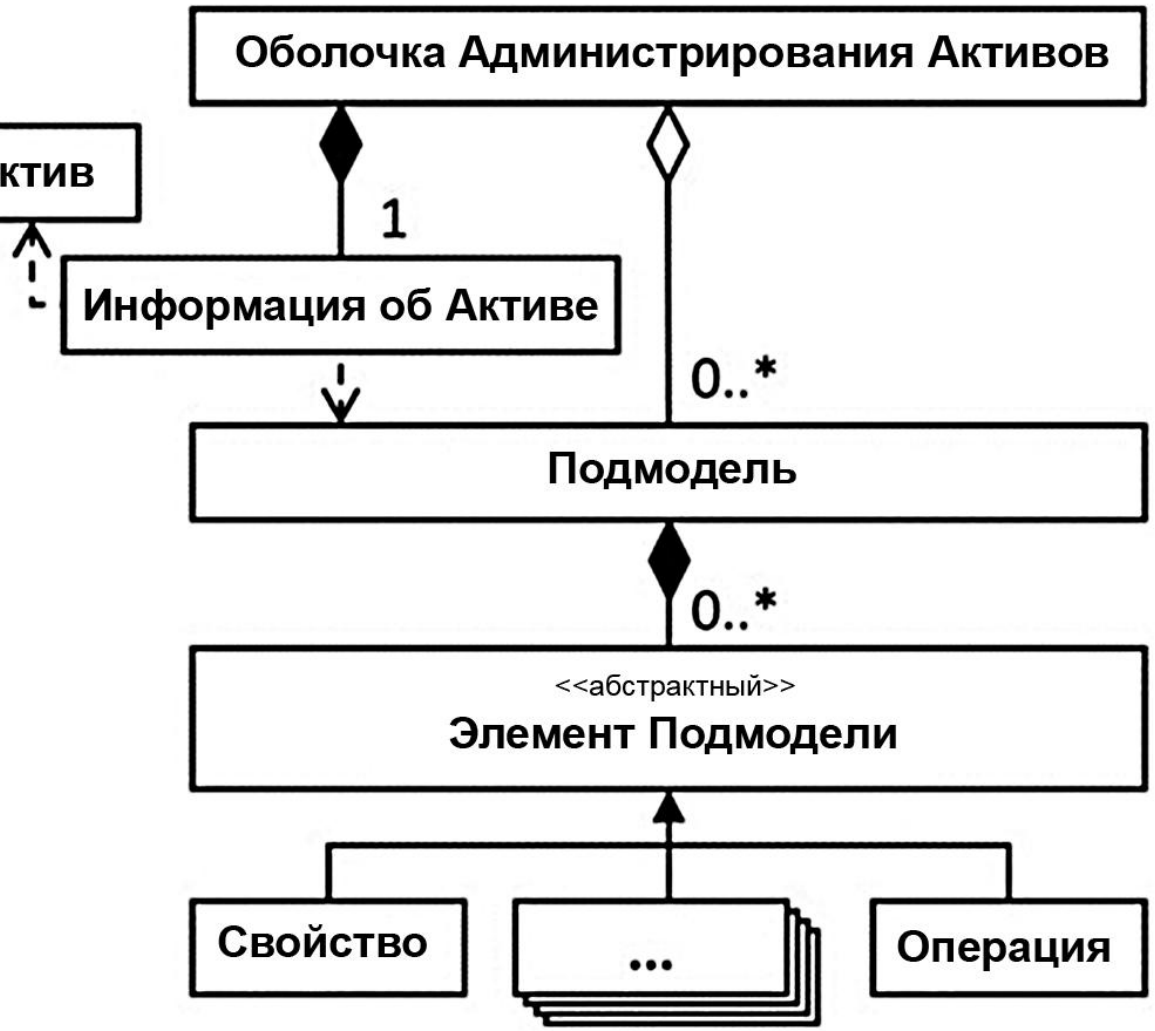


Рисунок 9.4. Модель AAS с его подмоделями [45]

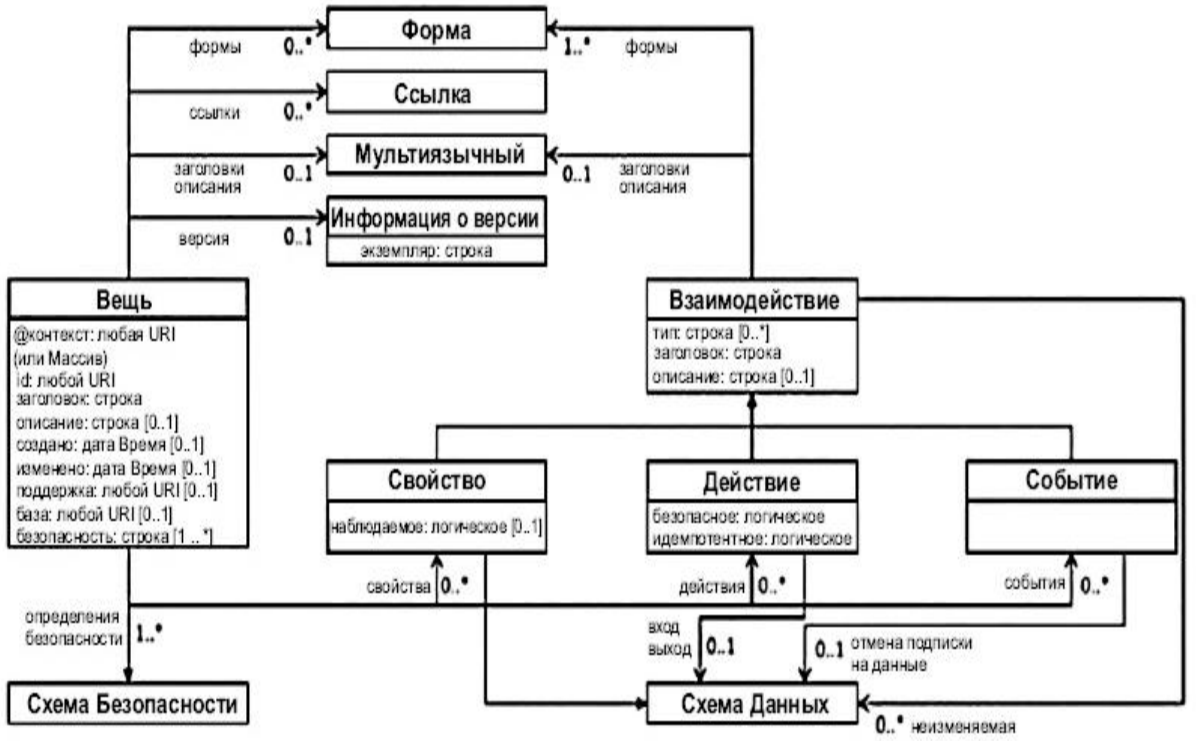


Рисунок 9.6. Диаграмма классов, изображающая метамодель Web of Things Thing Description (на основе рисунка 1 из [48])

9.2.3. НАЦИОНАЛЬНЫЕ СТАНДАРТЫ ЦИФРОВЫХ ДВОЙНИКОВ

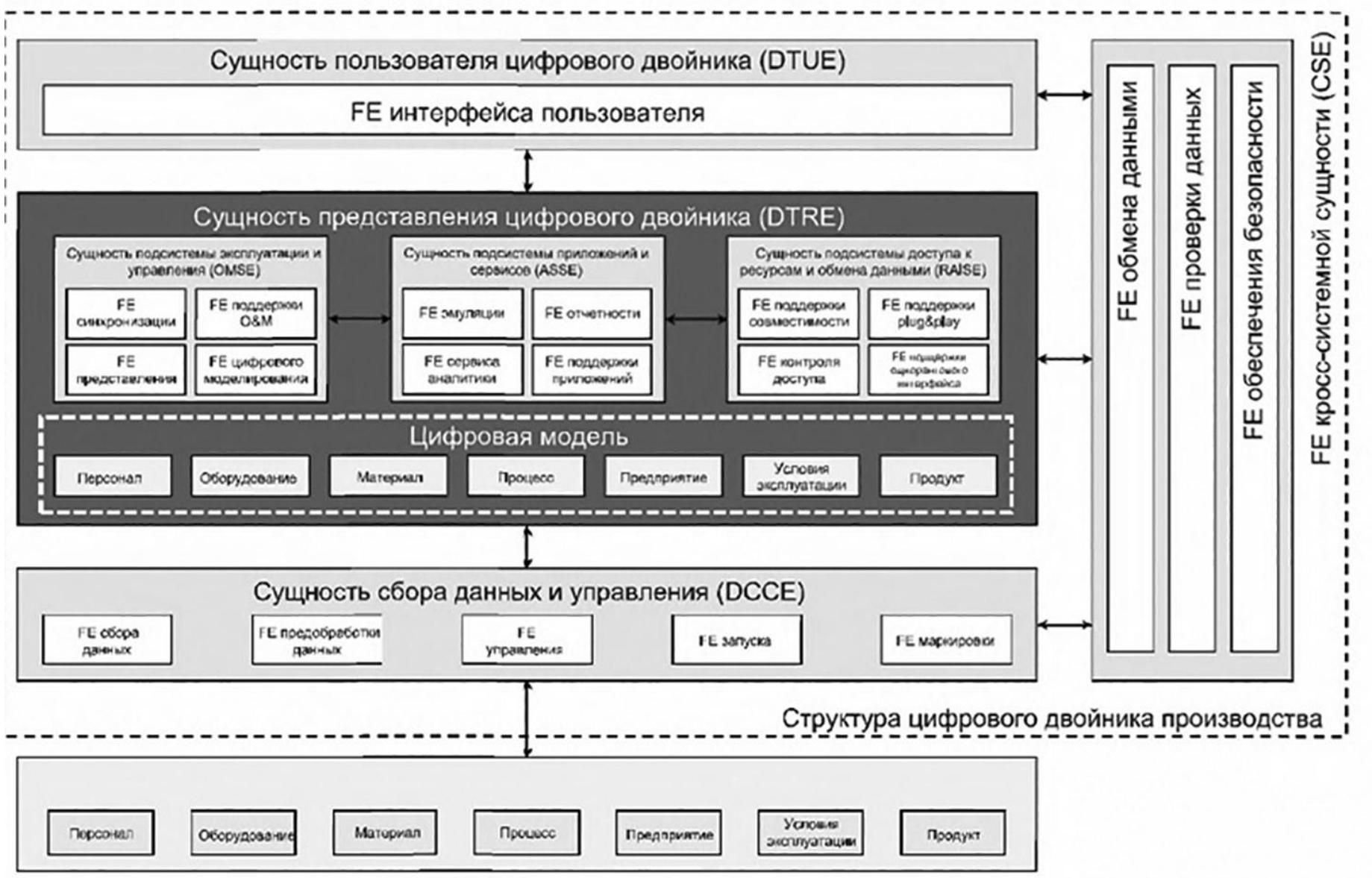


Рисунок 9.10. Производственные элементы в типовой архитектуре цифрового двойника производства [51] ПНСТ 431-2020

9.3. АНАЛИЗ АРХИТЕКТУРНЫХ РЕШЕНИЙ ДЛЯ ЦИФРОВЫХ ДВОЙНИКОВ

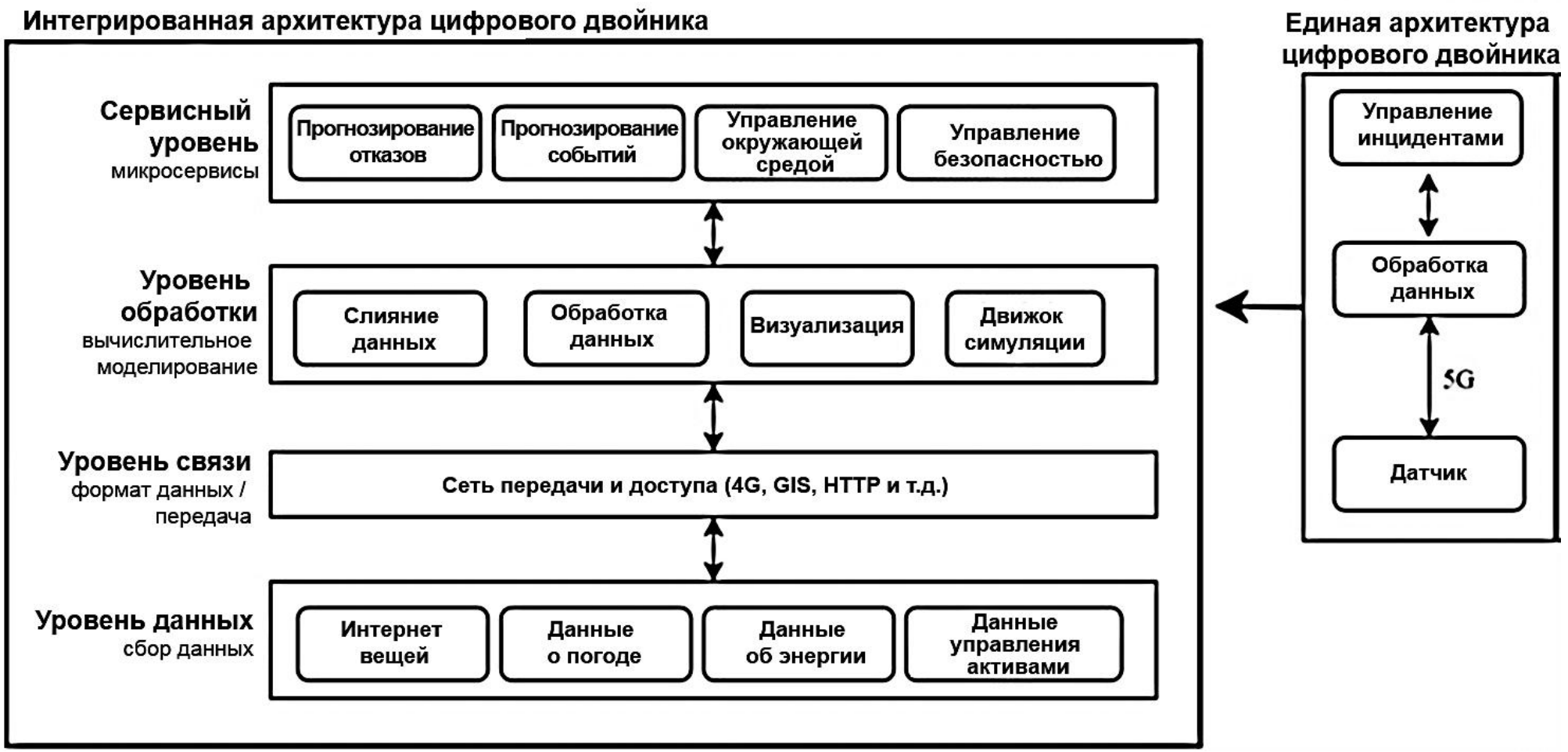


Рисунок 9.14. Интегрированная (полная) архитектура цифрового двойника [59]

9.3. АНАЛИЗ АРХИТЕКТУРНЫХ РЕШЕНИЙ ДЛЯ ЦИФРОВЫХ ДВОЙНИКОВ

Рисунок 9.17. Шестимерный цифровой двойник цеха [65]

9.3. АНАЛИЗ АРХИТЕКТУРНЫХ РЕШЕНИЙ ДЛЯ ЦИФРОВЫХ ДВОЙНИКОВ

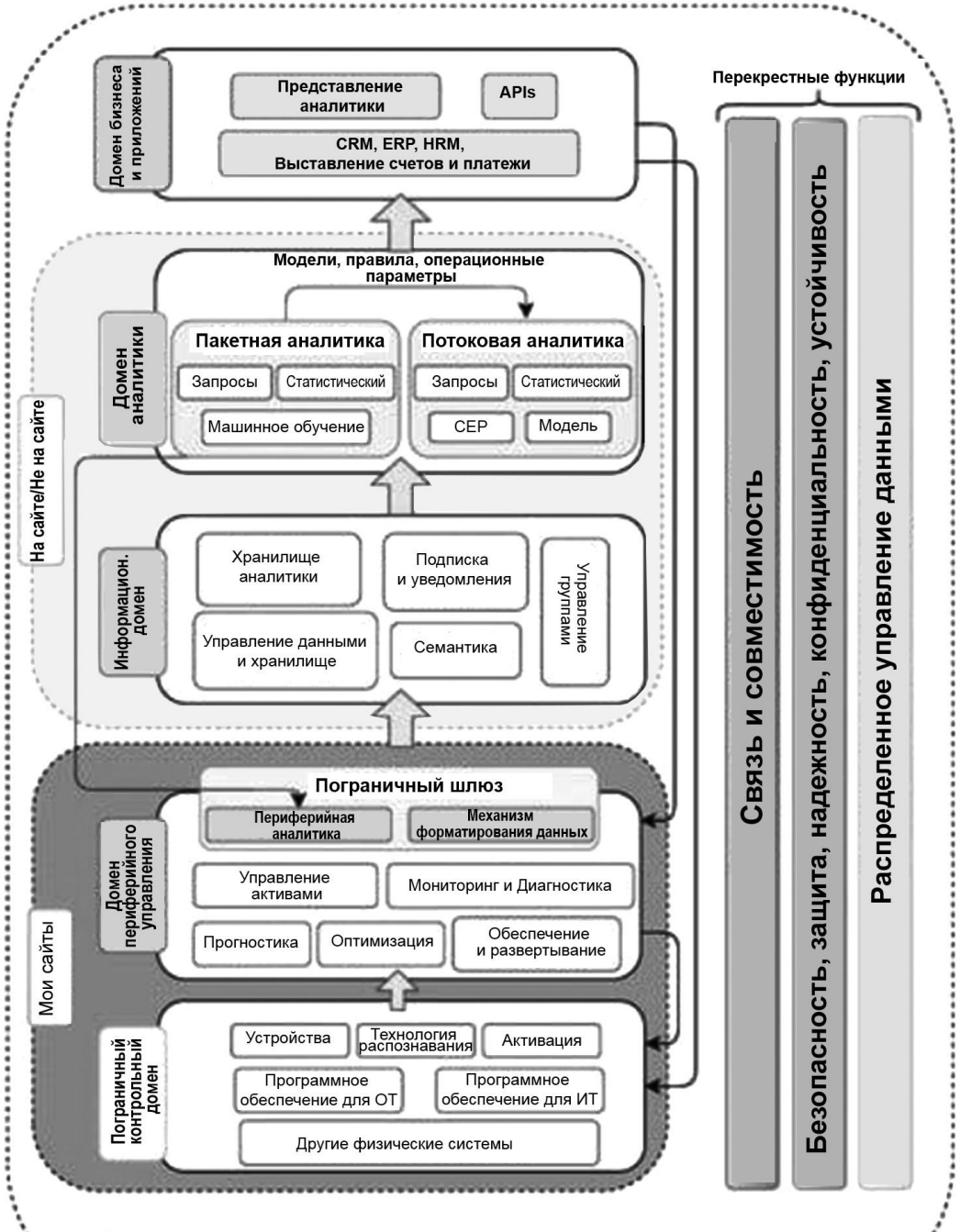
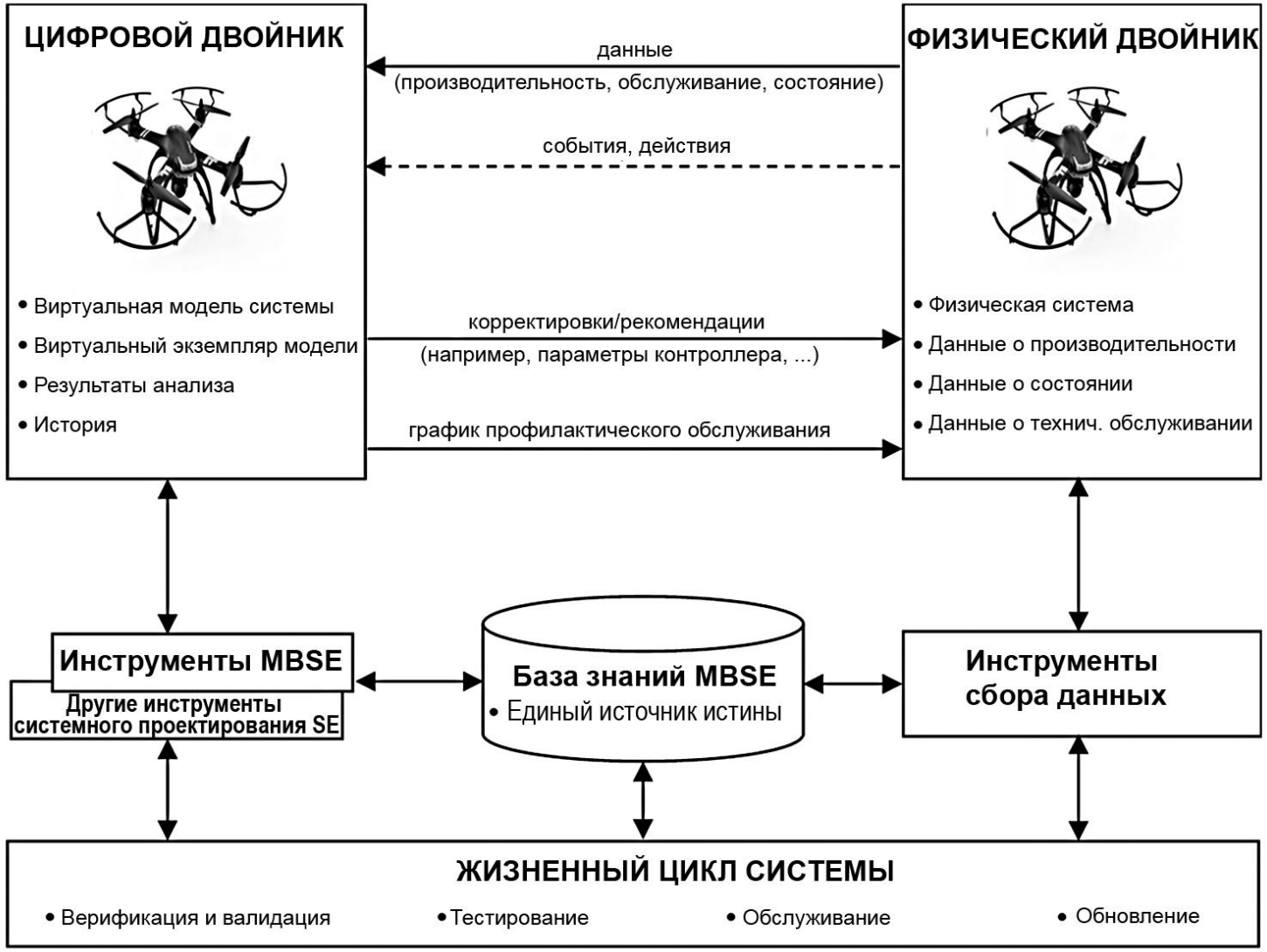
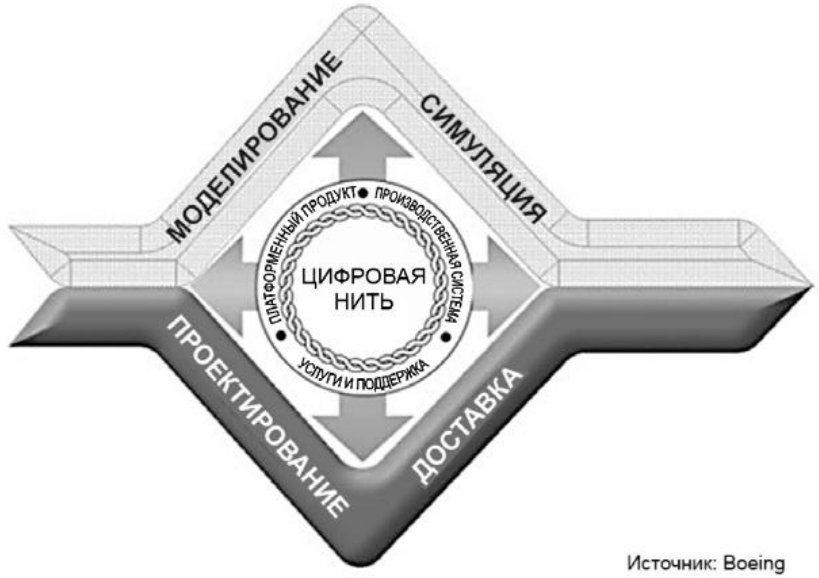


Рисунок 9.18. Синтезированная высокоуровневая архитектура IIoT дл
горнодобывающей промышленности [66]

9.4. ТЕХНОЛОГИЯ ЦИФРОВЫХ ДВОЙНИКОВ И MBSE
9.4.1. ЦИФРОВЫЕ ДВОЙНИКИ КАК ОСНОВА ПОДХОДА MBSE





Источник: Boeing

С помощью создаваемого таким образом цифрового двойника создается цифровая нить информационного потока, пронизывающая все стадии жизненного цикла продукта

Рисунок 9.19. Концепция цифрового двойника в рамках MBSE (системное проектирование на основе моделей) [70]

9.4. ТЕХНОЛОГИЯ ЦИФРОВЫХ ДВОЙНИКОВ И MBSE
9.4.1. ЦИФРОВЫЕ ДВОЙНИКИ КАК ОСНОВА ПОДХОДА MBSE

Рисунок 9.23. Платформа Digital Twin и Digital Thread для эффективного управления данными о продуктах [70]

Интеллектуальный цифровой двойник (Intelligent Digital Twin) — Он обладает всеми возможностями адаптивного цифрового двойника (включая контролируемое машинное обучение). Кроме того, он обладает возможностью самостоятельного машинного обучения. Эта тема глубоко исследуется в работе [79].

Двухнитиевая структура включает создание централизованной системы управления данными о продукте (product data management — **PDM**). Система PDM обеспечивает взаимодействие сервисов и платформ, участвующих в проекте, и помогает стандартизировать форматы файлов, принять общие подходы к хранению и представлению данных, а также вести контроль версий файлов данных на разных платформах. Кроме этого, система PDM позволяет разработчикам постоянно общаться и сотрудничать с другими заинтересованными сторонами посредством единой и последовательной структуры представления данных с целью доставки соответствующих данных нужному человеку в нужное время и в режиме реального времени

9.4.3 ИНТЕЛЛЕКТУАЛЬНЫЕ ЦИФРОВЫЕ ДВОЙНИКИ

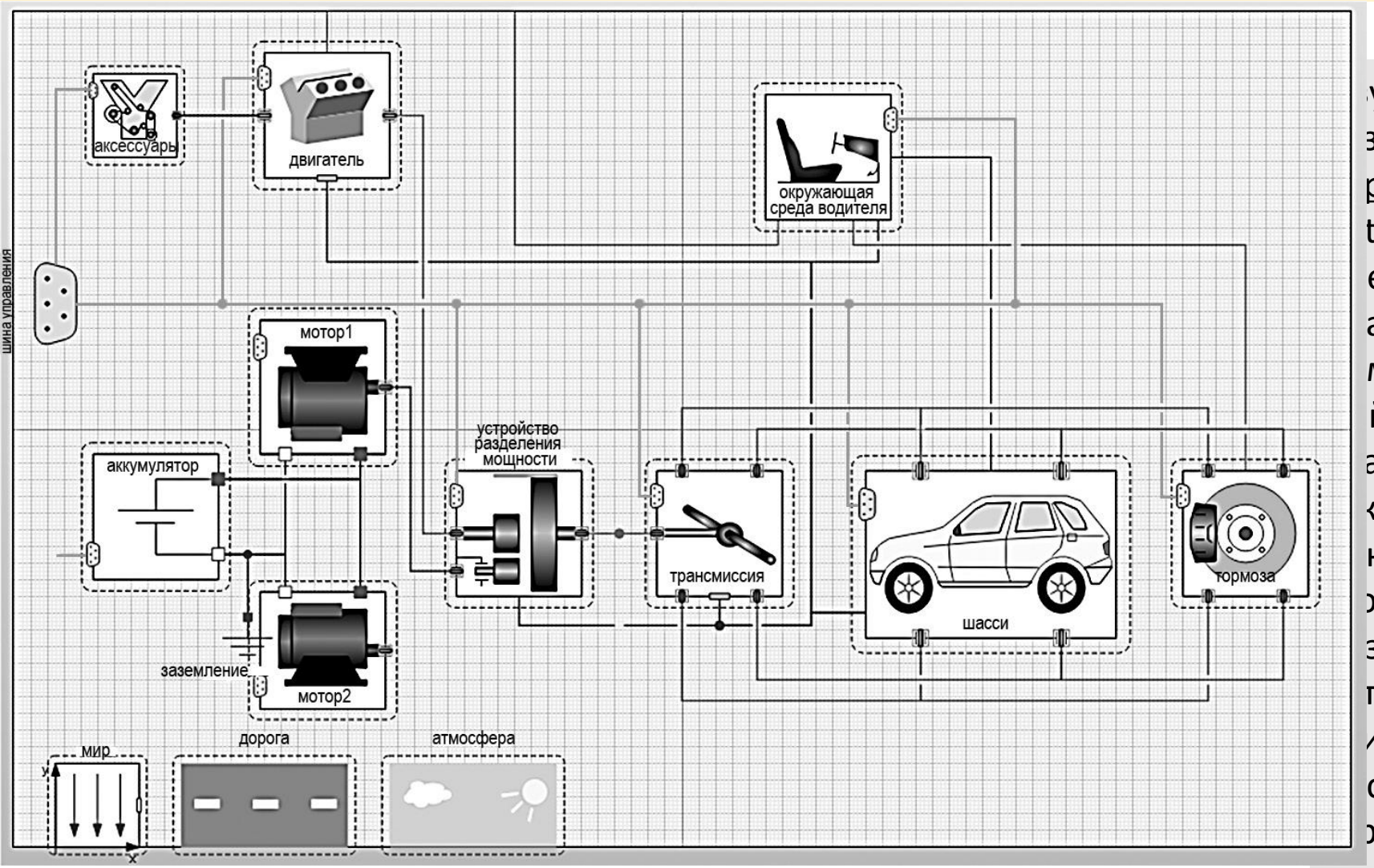


Рисунок 9.24. Модель цифрового двойника, созданная с использованием библиотеки с открытым исходным кодом автомобильных интерфейсов [70]

3. Адаптивный цифровой двойник (Adaptive Digital

ухнитиевая структура включает здание централизованной системы равления данными о продукте (product ta management — **PDM**). Система PDM еспечивает взаимодействие сервисов и атформ, участвующих в проекте, и могает стандартизировать форматы йлов, принять общие подходы к анению и представлению данных, а же вести контроль версий файлов нных на разных платформах. Кроме ого, система PDM позволяет зработчикам постоянно общаться и трудничать с другими интересованными сторонами средством единой и последовательной руктуры представления данных с целью доставки соответствующих данных нужному человеку в нужное время и в режиме реального времени

9.5. ЦИФРОВЫЕ ДВОЙНИКИ ДЛЯ СОЗДАНИЯ ПРОМЫШЛЕННОЙ МЕТАВСЕЛЕННОЙ

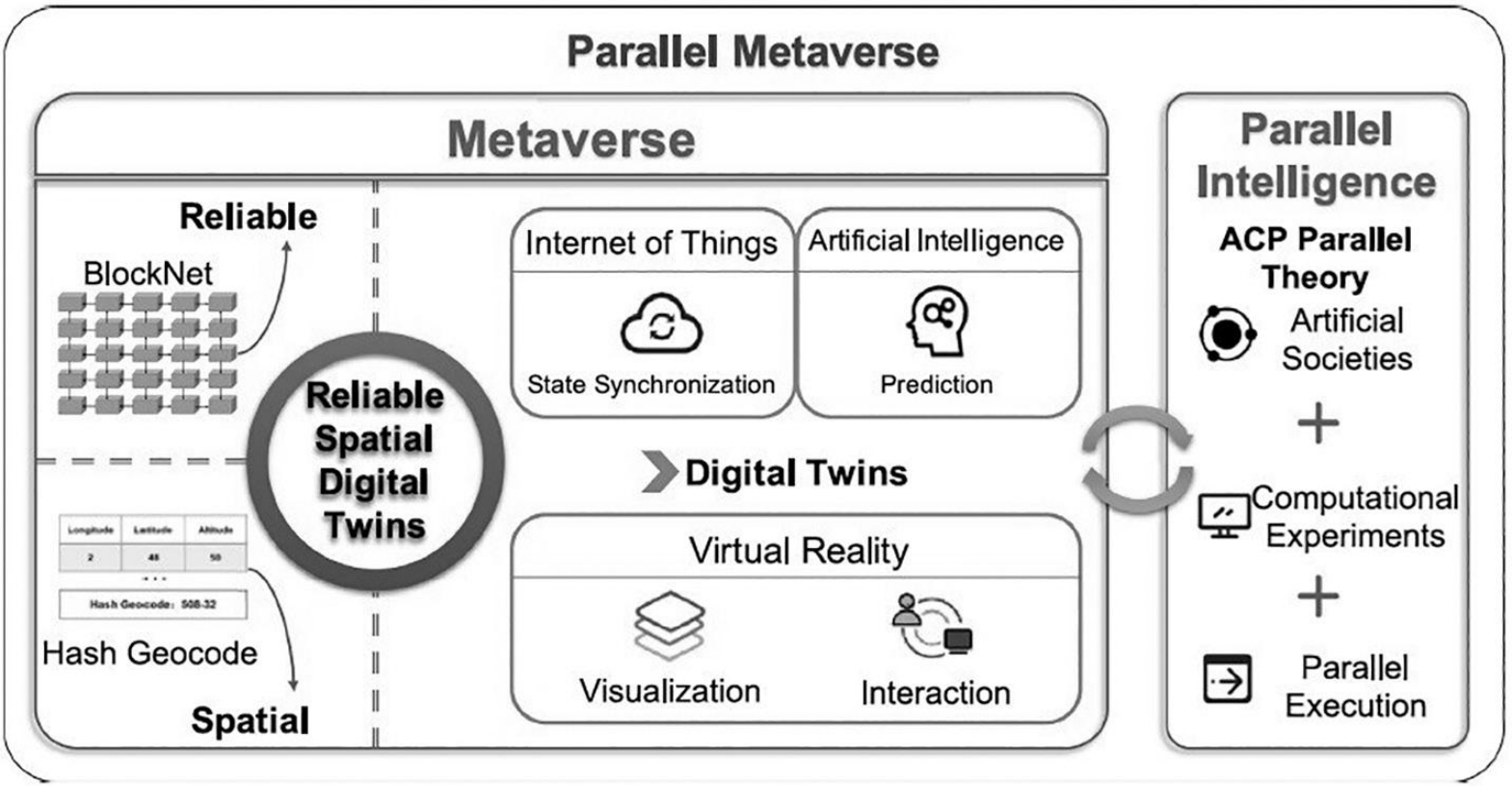


Рисунок 9.26. От надежных пространственных цифровых двойников к параллельной метавселенной [88]

каждая метавселенная строится из надежных пространственных цифровых двойников (spatial digital twin), использующих критически важные технологии блокчейна (BlockNet) для безопасного хранения многомерных данных и безопасности процесса цифрового картографирования метавселенной, а также других базовых технологий (цифровые двойники, движки виртуальной реальности, технологии ИИ, Интернета вещей, интеллектуального интерфейса, блокчейна).

Таким образом, цифровые двойники — это основа промышленной метавселенной и фундаментальный инструмент для построения виртуальных миров

Литература к Главе 9

Литература

[1] Прохоров, А. Цифровой двойник. Анализ, тренды, мировой опыт. Издание первое, исправленное и дополненное / А. Прохоров, М. Лысачев; научный редактор профессор А. Боровков. — М.: ООО «АльянсПринт», 2020. — 401 с., ил.

[2] ПНСТ 429-2020. Умное производство. ДВОЙНИКИ ЦИФРОВЫЕ ПРОИЗВОДСТВА. Часть 1. Общие положения. Издание официальное. — М.: Стандартинформ, 2020.

[3] Цифровые двойники в высокотехнологичной промышленности экспертно-аналитический доклад. Инфраструктурный центр «Технет» СПбПУ. 2019. https://technet-nti.ru/article/ekspertno-analiticheskij-dokladcifrovye-dvojniki-v-vysokotehnologichnoj-promyshlennosti.

[4] Madni, A.M. Transdisciplinary Systems Engineering: Exploiting Convergence in a Hyper-Connected World; Foreword by Norm Augustine / A.M. Madni; Springer: Berlin, Germany, 2017.

[5] West, T.D.; Pyster, A. Untangling the Digital Thread: The Challenge and Promise of Model-Based Engineering in Defense Acquisition. Insight 2015, 18, 45—55.

[6] Bachelor, G. Model-based design of complex aeronautical systems through digital twin and thread concepts / G. Bachelor, B. Eugenio, F. Davide, M. Andreas. — IEEE Systems Journal 14, no. 2 (2019): 1568-1579.

[7] Laukotka, F. Digital twins of product families in aviation based on an MBSE-assisted approach / F. Laukotka, H. Michael, K. Dieter. Procedia CIRP 100 (2021). — P. 684–689.

[8] Ghaboura, S. Digital Twin for Railway: A Comprehensive Survey / S. Ghaboura, R. Ferdousi, F. Laamarti, C. Yang and A.E. Saddik // IEEE Access. — Vol. 11. — P. 120237–120257, 2023, doi: 10.1109/ACCESS.2023.3327042.

[9] Doubell, G.C. et al. A Digital Twin System for Railway Infrastructure / G.C. Doubell et al // R&D Journal. — 2023. — Т. 39. — С. 23–34 // https://www.scielo.org.za/scielo.php?pid=S2309-89882023000100003&script=sci_arttext

[10] Eneyew, D. D. Toward Smart-Building Digital Twins: BIM and IoT Data Integration / D.D. Eneyew, M.A. M. Capretz, G.T. Bitsuamlak // IEEE Access. — 2022. — Vol. 10. — P. 130487–130506 // doi: 10.1109/ACCESS.2022.3229370. https://ieeexplore.ieee.org/document/9987476

[11] Marinelli, M. From Industry 4.0 to Construction 5.0: Exploring the Path towards Human—Robot Collaboration in Construction / M. Marinelli. — Systems 2023, 11, 152. https://doi.org/10.3390/systems11030152

[12] Neugebauer, J., Heilig, L. & Voß, S. Digital Twins in the Context of Seaports and Terminal Facilities. Flex Serv Manuf J. — 2024. https://doi.org/10.1007/s10696- 023-09515-9

[13] Mahmoud, E.G. (2022). The Future of Digital Twins for Autonomous Systems: Analysis and Opportunities. In: Hassanien, A.E., Darwish, A., Snasel, V. (eds) Digital Twins for Digital Transformation: Innovation in Industry / E.G. Mahmoud, A. Darwish, A.E. Hassanien. — Studies in Systems, Decision and Control, vol 423. Springer, Cham. https://doi.org/10.1007/978-3-030-96802-1_10

[14] Stary, C. Digital Process Twins as Intelligent Design Technology for Engineering Metaverse / C. Stary // XR Applications. Sustainability. — 2023. — 15. — 16062 // https://doi.org/10.3390/su152216062

ГЛАВА 10. СИСТЕМА СТАНДАРТОВ СИСТЕМНОЙ ИНЖЕНЕРИИ. ПРОЦЕССНЫЕ СТАНДАРТЫ

10.1. ТАКСОНОМИЯ БАЗОВЫХ СТАНДАРТОВ СИСТЕМНОЙ ИНЖЕНЕРИИ

10.2. ПРОЦЕССЫ ЖИЗНЕННОГО ЦИКЛА СИСТЕМ — ISO/IEC/IEEE 15288

10.3. ПРОЦЕССЫ ЖИЗНЕННОГО ЦИКЛА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ (ISO/IEC 12207)

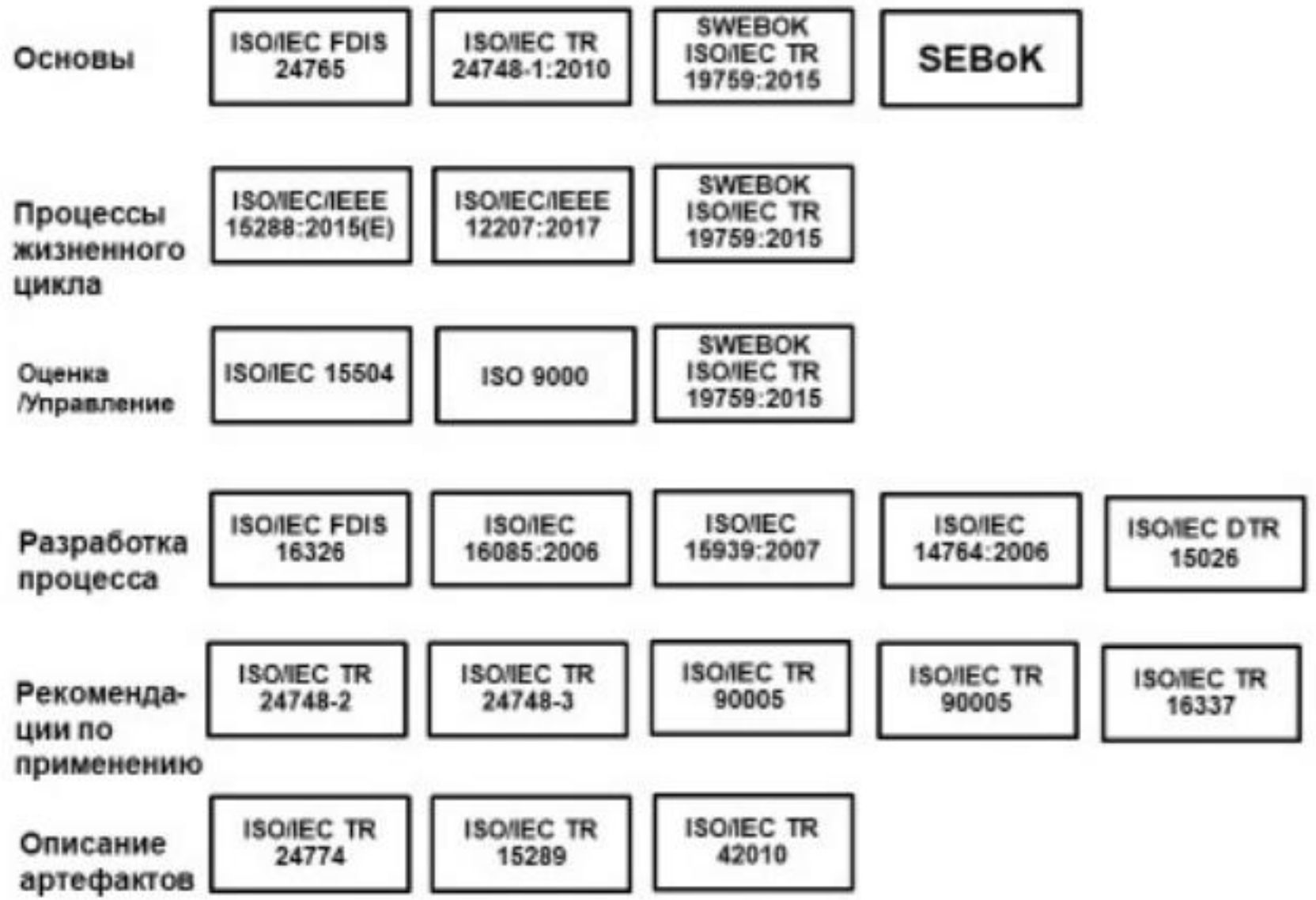
10.4. КАТЕГОРИЯ ПРОЦЕССОВ В КОНТЕКСТЕ СИСТЕМЫ

10.5. КАТЕГОРИЯ СПЕЦИАЛЬНЫХ ПРОЦЕССОВ ПРОГРАММНЫХ СРЕДСТВ)

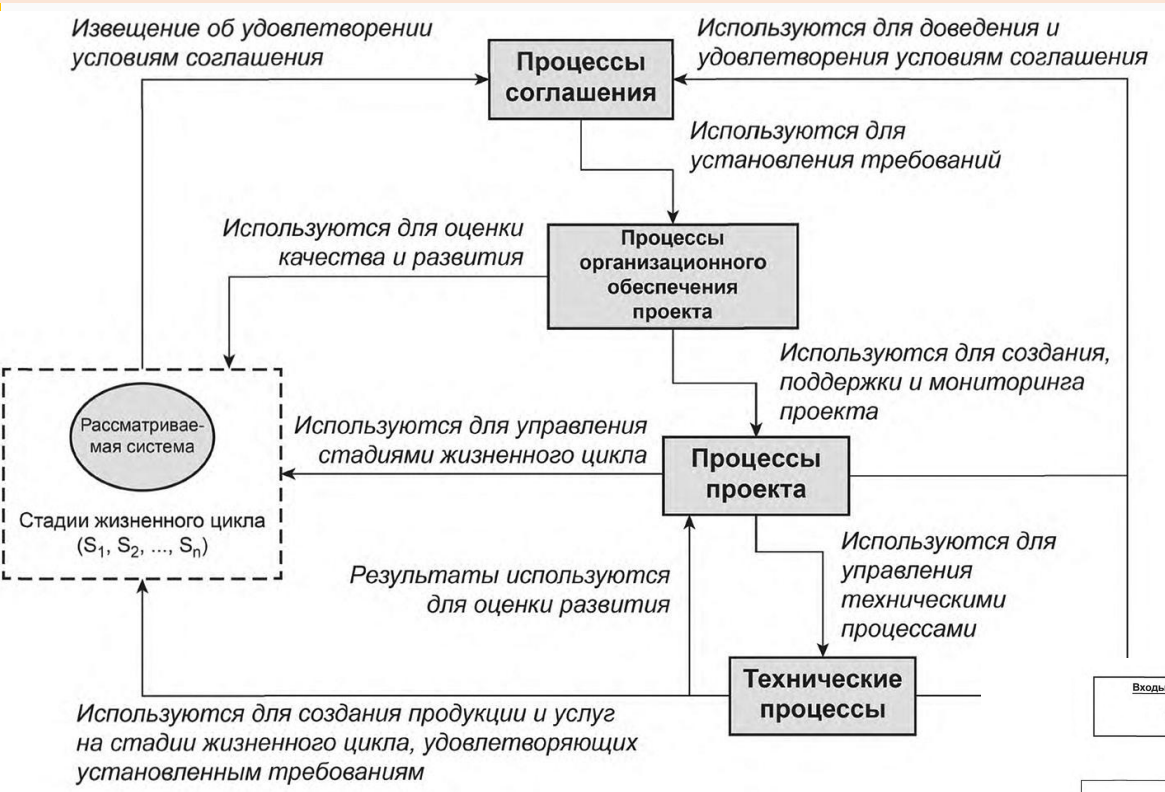
10.6. СРАВНЕНИЕ ISO/IEC/IEEE 15288:2023 С ISO/IEC/IEEE 15288:2015

10.7. МОДЕЛЬНО-ОРИЕНТИРОВАННАЯ СИСТЕМНАЯ И ПРОГРАММНАЯ ИНЖЕНЕРИЯ (MODEL-BASED SYSTEMS AND SOFTWARE ENGINEERING (MBSSE)»

10.1. ТАКСОНОМИЯ БАЗОВЫХ СТАНДАРТОВ СИСТЕМНОЙ ИНЖЕНЕРИИ



10.2. ПРОЦЕССЫ ЖИЗНЕННОГО ЦИКЛА СИСТЕМ — ISO/IEC/IEEE 15288 [4]



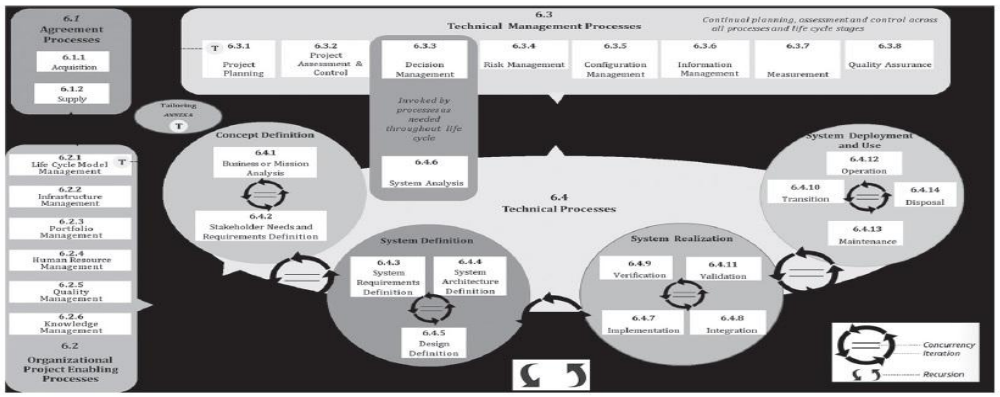
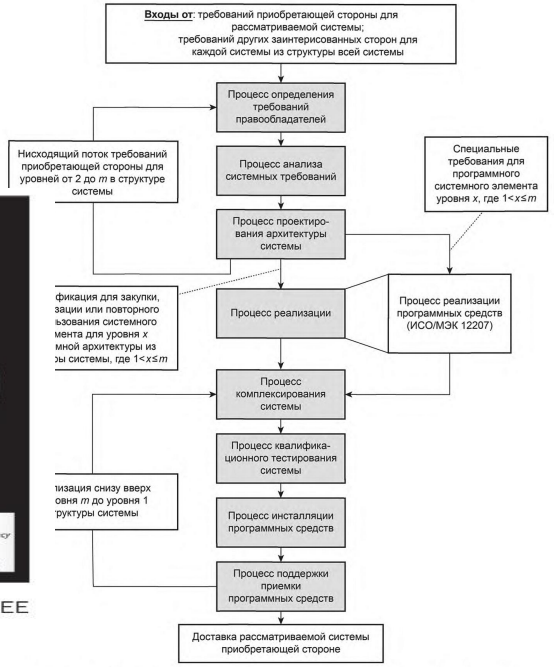
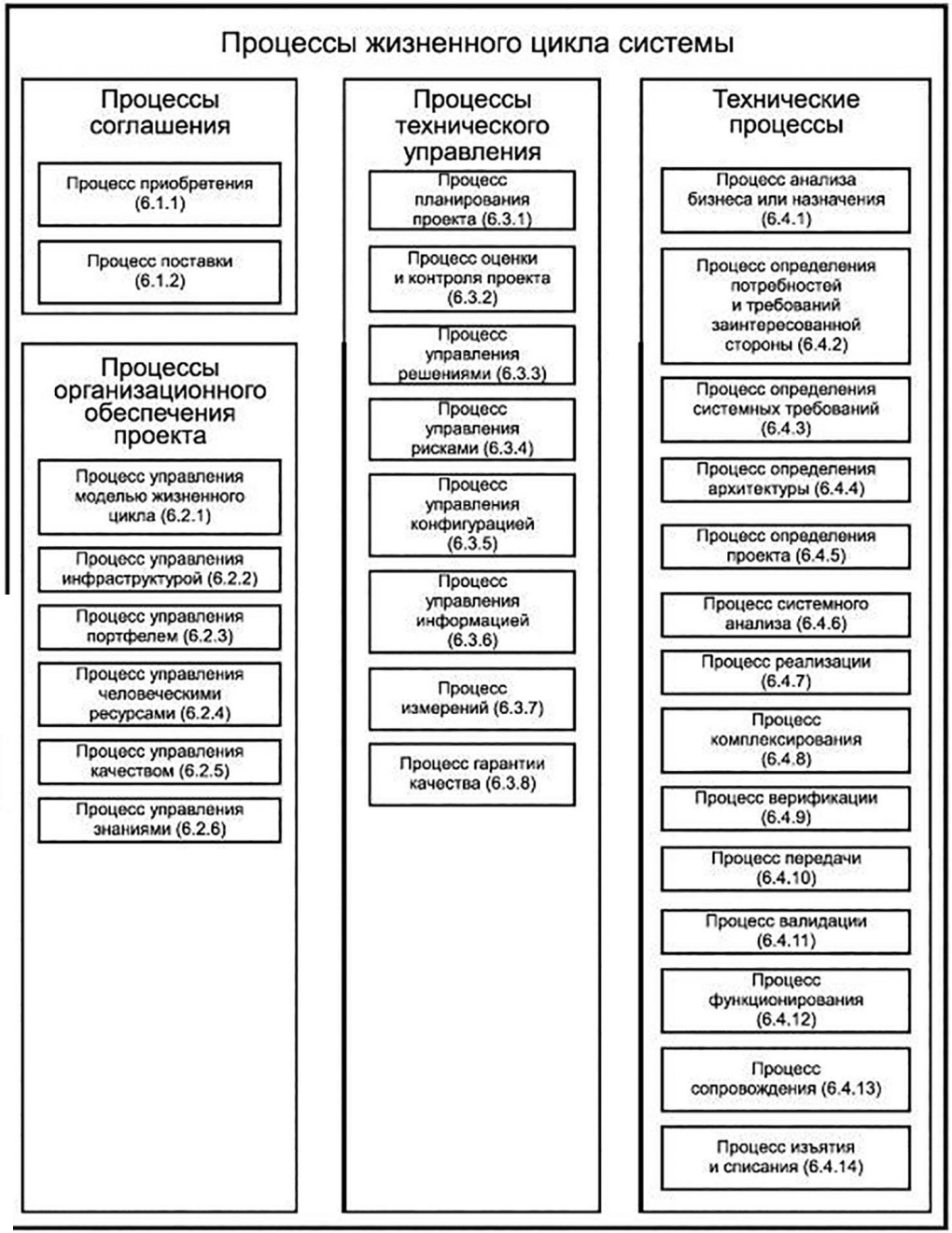
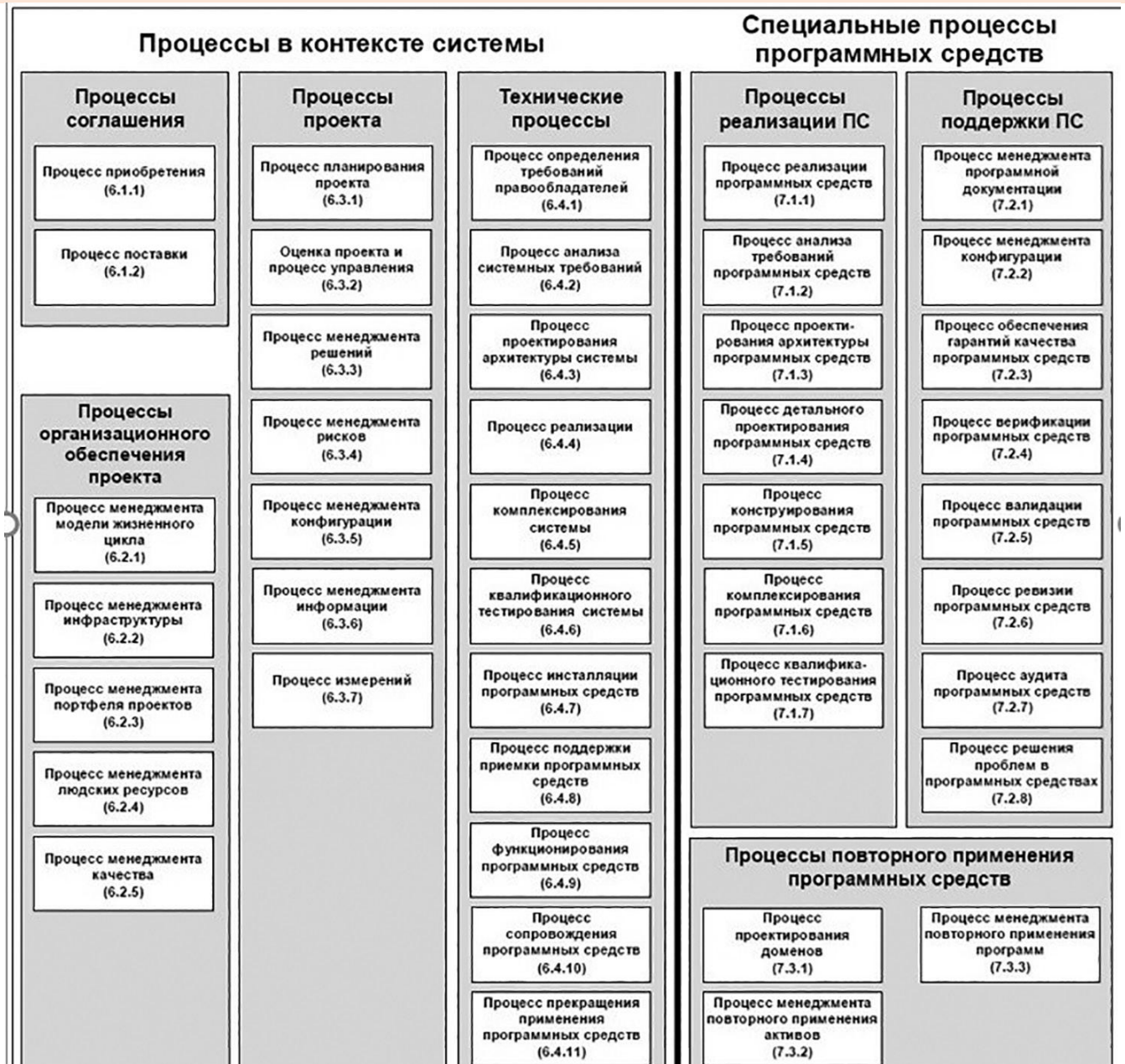


Рисунок 10.9. Модель взаимодействия жизненного цикла систем [ISO/IEC/IEEE 15288:2023]





10.3. ПРОЦЕССЫ ЖИЗНЕННОГО ЦИКЛА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ (ISO/IEC 12207)



10.4. СРАВНЕНИЕ ISO/IEC/IEEE 15288:2023 С ISO/IEC/IEEE 15288:2015

10.5. МОДЕЛЬНО-ОРИЕНТИРОВАННАЯ СИСТЕМНАЯ И ПРОГРАММНАЯ ИНЖЕНЕРИЯ (MODEL-BASED SYSTEMS AND SOFTWARE ENGINEERING (MBSSE))

Литноптура к Главе 10

[1] Guide to the Systems Engineering Body of Knowledge (SEBoK), version 2.7. 2022.
[2] SWEBOK (Software Engineering Body of Knowledge) —ISO/IEC TR 19759:2015 и SEBoK (Guide to the Systems Engineering Body of Knowledge).
[3] SEBoK (Guide to the Systems Engineering Body of Knowledge).
[4] ISO/IEC/IEEE FDIS 15288:2023. Systems and software engineering — System life cycle processes (ГОСТ Р ИСО/МЭК 15288-2016 Информационная технология (ИТ). Системная инженерия. Процессы жизненного цикла систем).
[5] ГОСТ Р 57102 — 2016/ ISO/IEC TR 24748-2:2011 Информационные технологии. Системная и программная инженерия. УПРАВЛЕНИЕ ЖИЗНЕННЫМ ЦИКЛОМ. Часть 2. Руководство по применению ИСО/МЭК 15288 (ISO/IEC TR 24748-2:2011).
[6] ISO/IEC/IEEE 12207:2017 «Systems and software engineering — System life cycle software» (ГОСТ Р ИСО/МЭК 12207-2010 Информационная технология (ИТ). Системная и программная инженерия. Процессы жизненного цикла программных средств).
[7] ISO/IEC/IEEE FDIS 24748-1:2023 Systems and software engineering — Life cycle management — Part 1: Guidelines for life cycle management.
[8] ISO/IEC/IEEE FDIS 24748-2:2023 Systems and software engineering — Life cycle management — Part 2: Guidelines for the application of ISO/IEC/IEEE 15288 (system life cycle processes).

Глава 11. Эталонная модель модельно-ориентированной систем-ной и программной инженерии (MBSSE) и ее связь с процессными стандартами системной инженерии

11.1. МЕТОДЫ И ИНСТРУМЕНТЫ ДЛЯ МОДЕЛЬНО-ОРИЕНТИРОВАННОЙ СИСТЕМНОЙ И ПРОГРАММНОЙ ИНЖЕНЕРИИ (ISO/IEC/IEEE 24641:2023)

11.2. ОСНОВНЫЕ ПОНЯТИЯ СТАНДАРТА ISO/IEC/IEEE 24641:2023

11.3. ЭТАЛОННАЯ МОДЕЛЬ MBSSE

11.4. ОПИСАНИЕ ЦЕЛЕЙ ПРОЦЕССОВ ЭТАЛОННОЙ МОДЕЛИ

11.5. ВЗАИМОСВЯЗЬ ISO/IEC/IEEE 24641 С ГЛАВНЫМИ МЕЖДУНАРОДНЫМИ СТАНДАРТАМИ СИСТЕМНОЙ ИНЖЕНЕРИИ

11.6. ТРЕХМЕРНАЯ ЭТАЛОННАЯ СТРУКТУРА (ФРЕЙМВОРК) MBSSE

11.7. MBSSE-РАЗМЕРНОСТИ СИСТЕМНОЙ МОДЕЛИ

11.8. ПОТЕНЦИАЛЬНО ВОЗМОЖНЫЕ MBSSE-РОЛИ

–

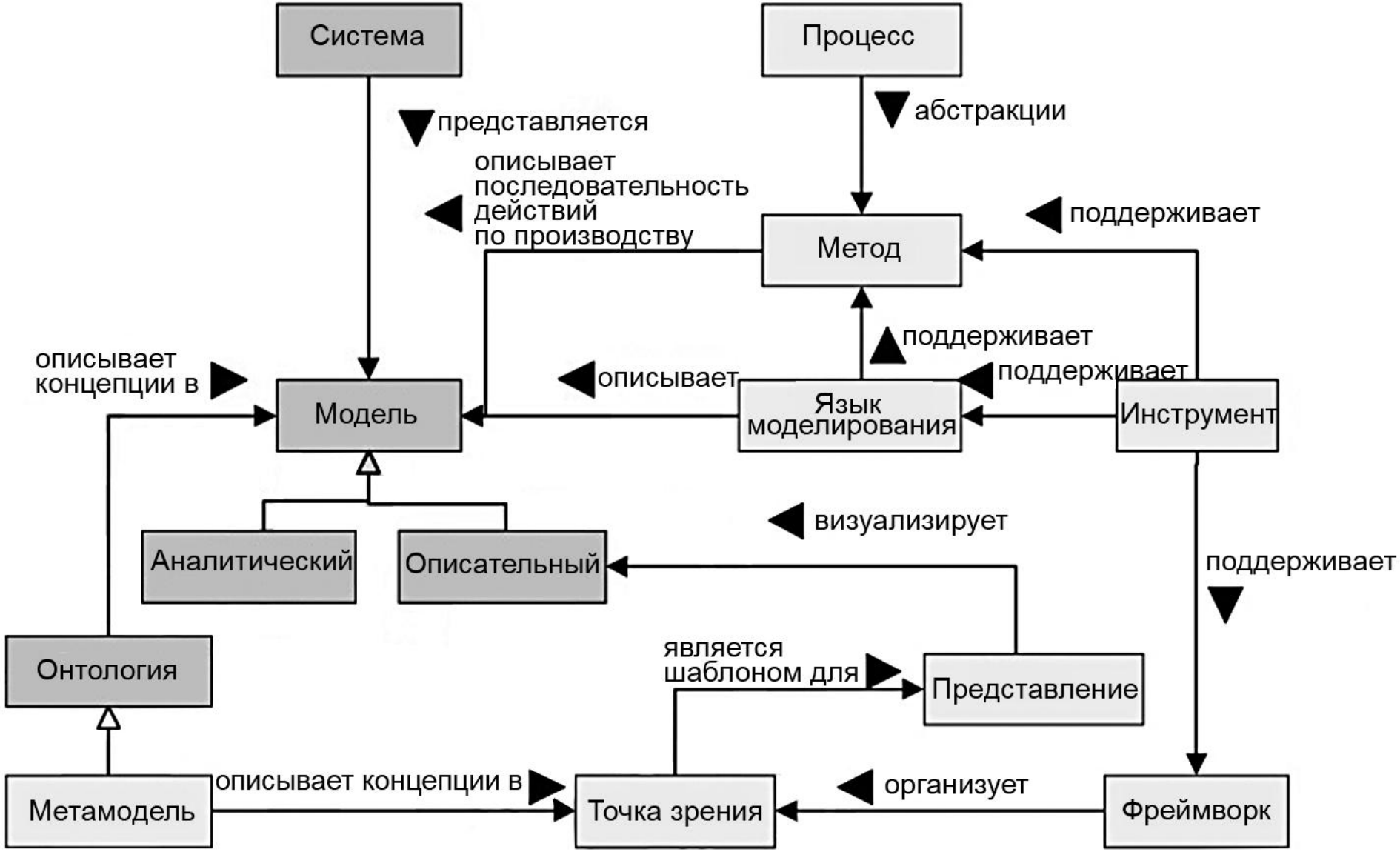
11.3. ЭТАЛОННАЯ МОДЕЛЬ MBSSE

•**Стандарт ISO/IEC/IEEE 24641:2023 — Системная и программная инженерия. Методы и инструменты для модельно-ориентированной системной и программной инженерии**, посвящен возможностям и методам разработки систем и программного обеспечения на основе моделей (MBSSE) — это один из первых стандартов, который можно отнести к фундаментальным методологическим решениям в области MBSE, отражающим тенденцию на интеграцию системной инженерии с программной инженерией

•Данный стандарт определяет методологические основы подхода MBSSE, в частности, в нем определяются:

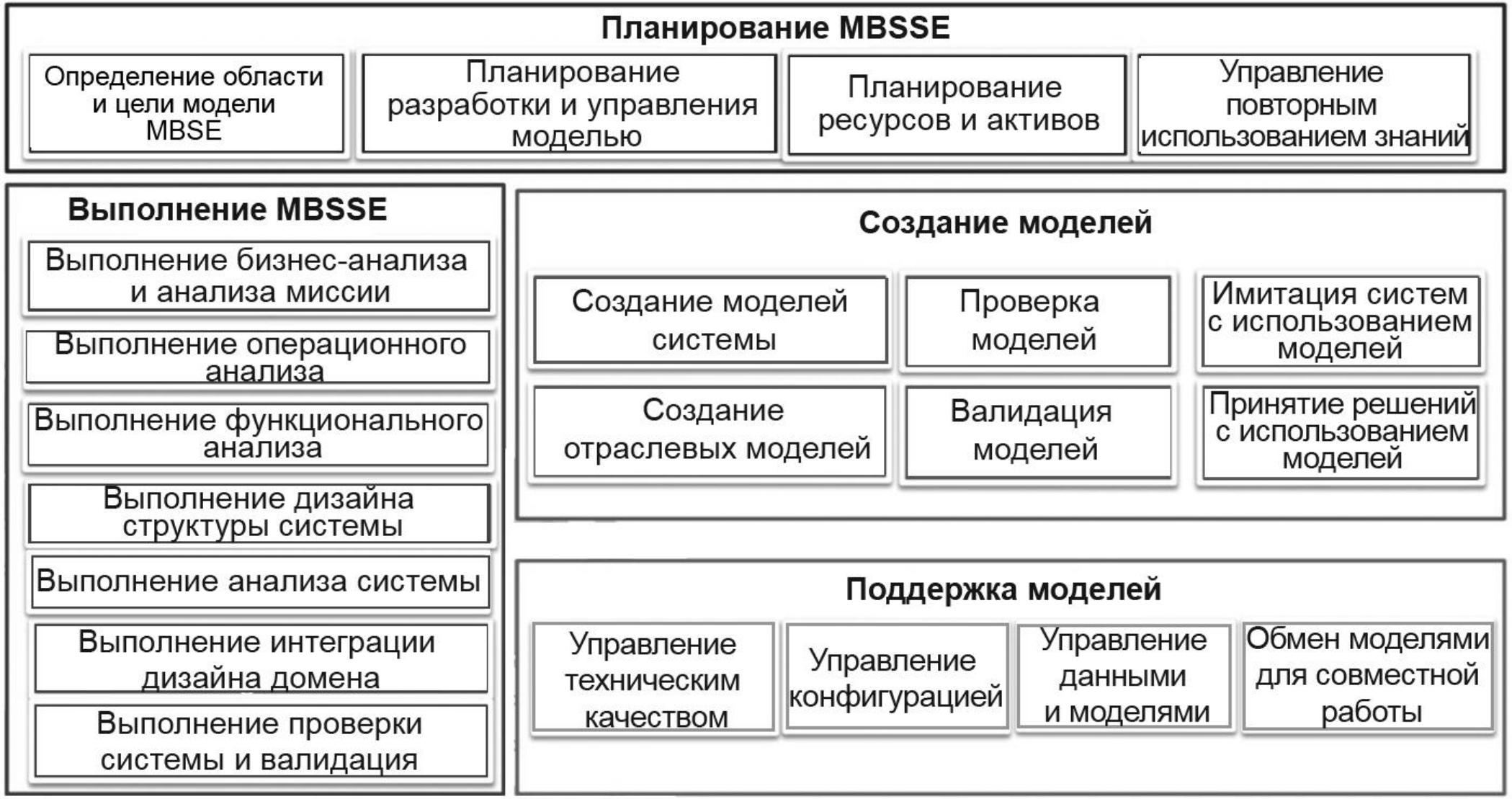
- эталонная модель для общей структуры и процессов, специфичных для MBSSE
- взаимосвязь между компонентами эталонной модели
- специфические для MBSSE процессы, которые описаны с точки зрения цели, входных данных, результатов и задач
- методы для поддержки определенных задач каждого процесса, специфичного для MBSSE
- возможности инструмента для автоматизации задач или методов

ОСНОВНЫЕ ПОНЯТИЯ СТАНДАРТА ISO/IEC/IEEE 24641:2023



MBSSE ISO/IEC/IEEE 24641

Эталонная модель ДЛЯ общего фреймворка и процессов MBSSE

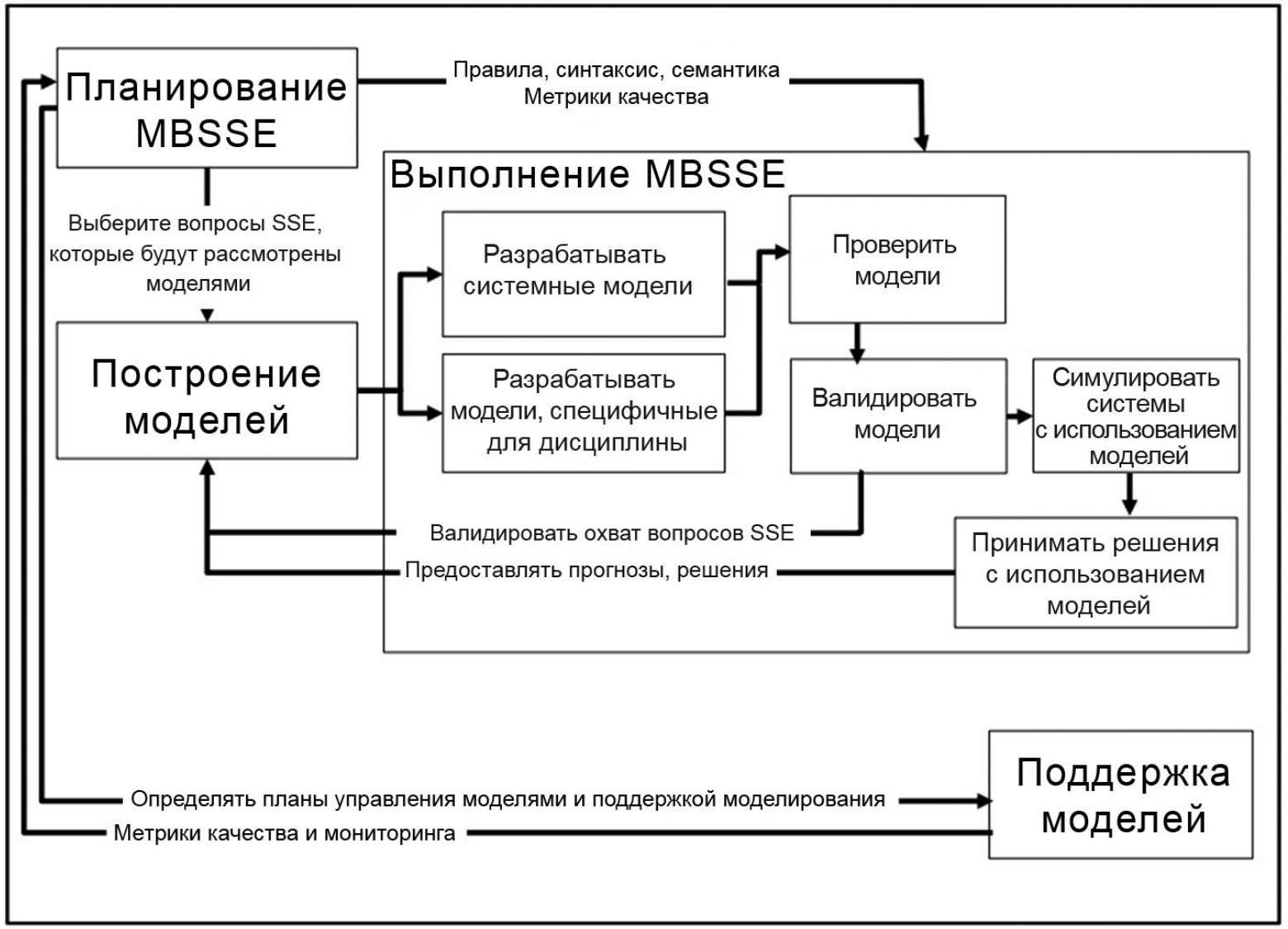


Эталонная модель MBSSE ISO/IEC/IEEE 24641

Эталонная модель для общего фреймворка и процессов MBSSE

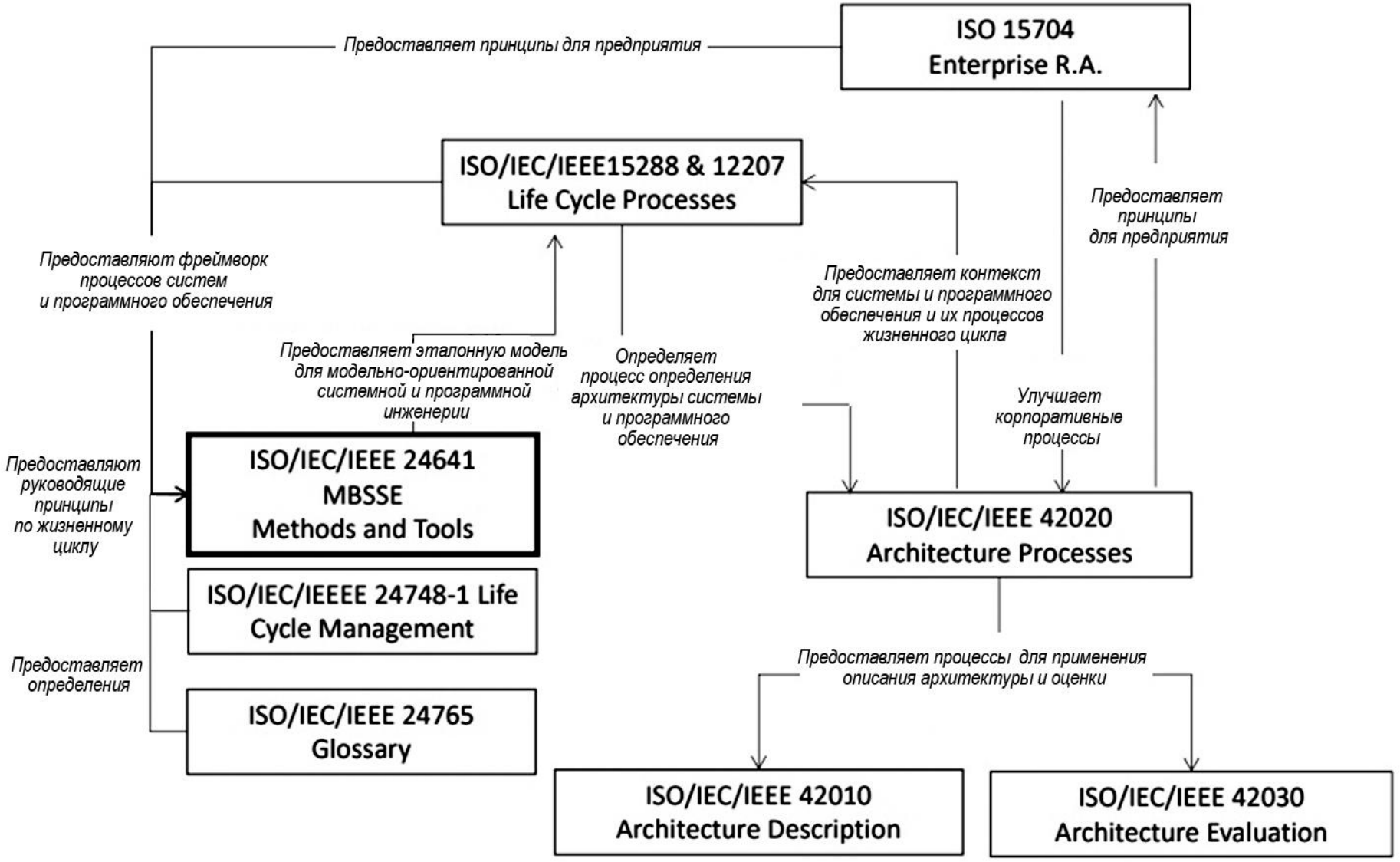
- **Группа процессов «План MBSSE» (Plan MBSSE)** определяет процессы, необходимые в качестве предварительных условий для моделирования
- Процессы группы «План MBSSE» выполняются в начале жизненного цикла моделирования систем и программного обеспечения, в частности, с помощью их определяются объем моделирования и цели для задачи разработки систем, определяются процессы, предназначенные для организации разработки модели, которые могут считаться необходимыми для каждого конкретного проекта моделирования (например, планирование разработки модели и управления ею или планирование ресурсов и активов)
- Любое развертывание MBSSE должно пройти через эти процессы
- **Процессы группы «Построение моделей» (Build Models)** представляют собой базовые процессы моделирования, т. е. создания полезных моделей, и обеспечивают согласованность в репозитории моделей
- **Процессы группы «Выполнение MBSSE» («Perform MBSSE»)**, включенные в план управления MBSSE, должны определять задачи системного проектирования (SE), выявленные в ходе процессов «Планирования MBSSE». Эти процессы определяют, какие модели должны быть созданы, они используют процессы, описанные в трех основных группах процессов
- **Процессы группы «Поддержка моделей» («Support Models»)** касаются технических аспектов и выполняются в течение всего жизненного цикла

Эталонная модель для общего фреймворка и процессов MBSSE



Отношения между группами процессов MBSSE и процессами группы построения моделей [MBSSE ISO/IEC/IEEE 24641]

Эталонная модель для общего фреймворка и процессов MBSSE



Взаимосвязь ISO/IEC/IEEE 24641 с главными международными стандартами системной инженерии [13]

Эталонная модель для общего фреймворка и процессов MBSSE

Согласно определения в ISO/IEC/IEEE 15288:2023 и ISO/IEC/IEEE 24748-1:2018, модель жизненного цикла созданных человеком систем и объектов имеет две размерности:

(1) стадию эволюции (или системная)

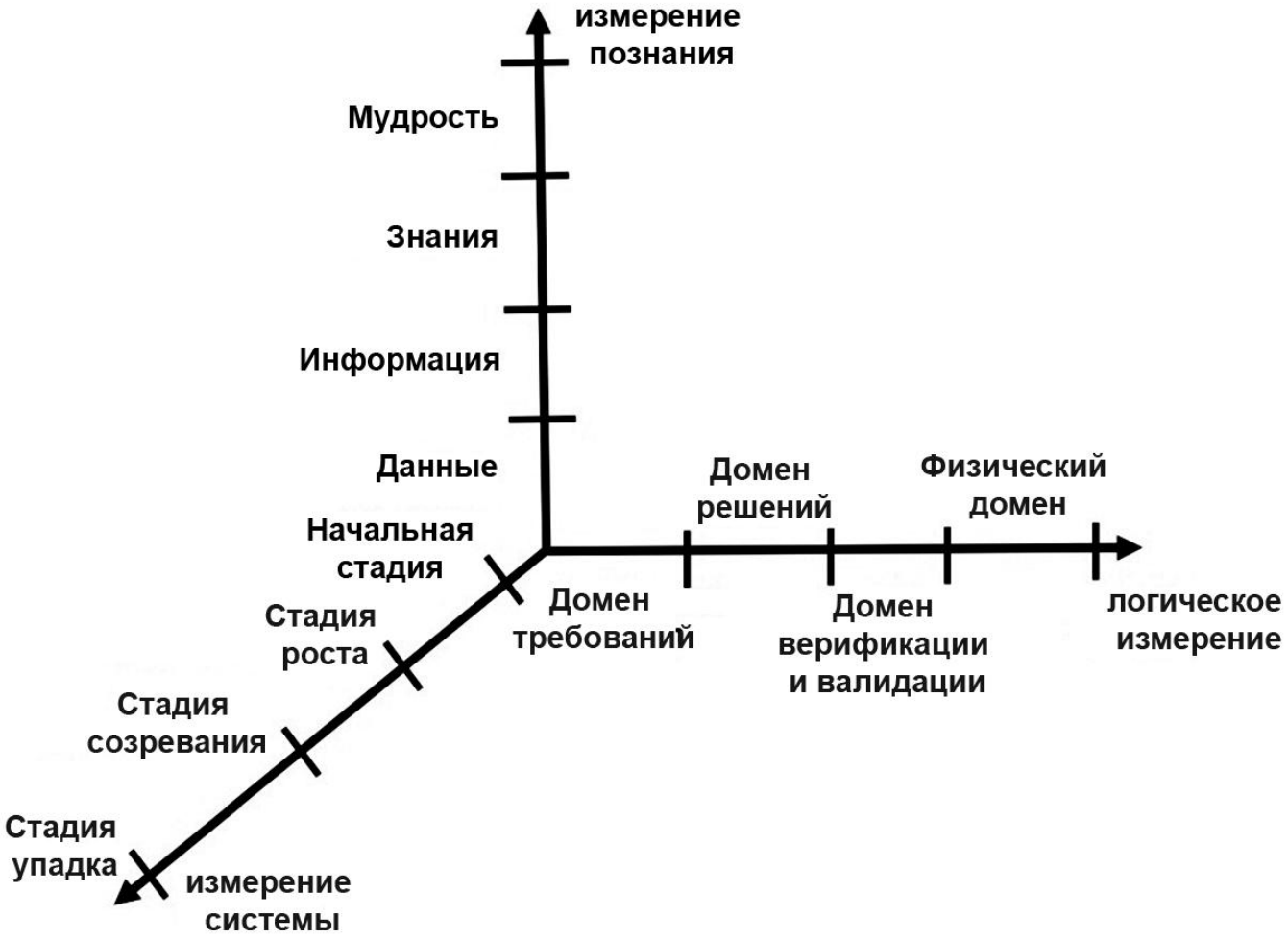
(2) структуру процессов и деятельностей (логическая).

Третья размерность – это когнитивная, или познавательная (cognition dimension), размерность фиксирует процессы и результаты понимания и преобразования объективного мира. Это измерение личности и организации

Взаимосвязь между процессами построения моделей и пирамидой «Данные — Информация — Знания — Мудрость»

Эталонная модель для общего фреймворка и процессов MBSSE

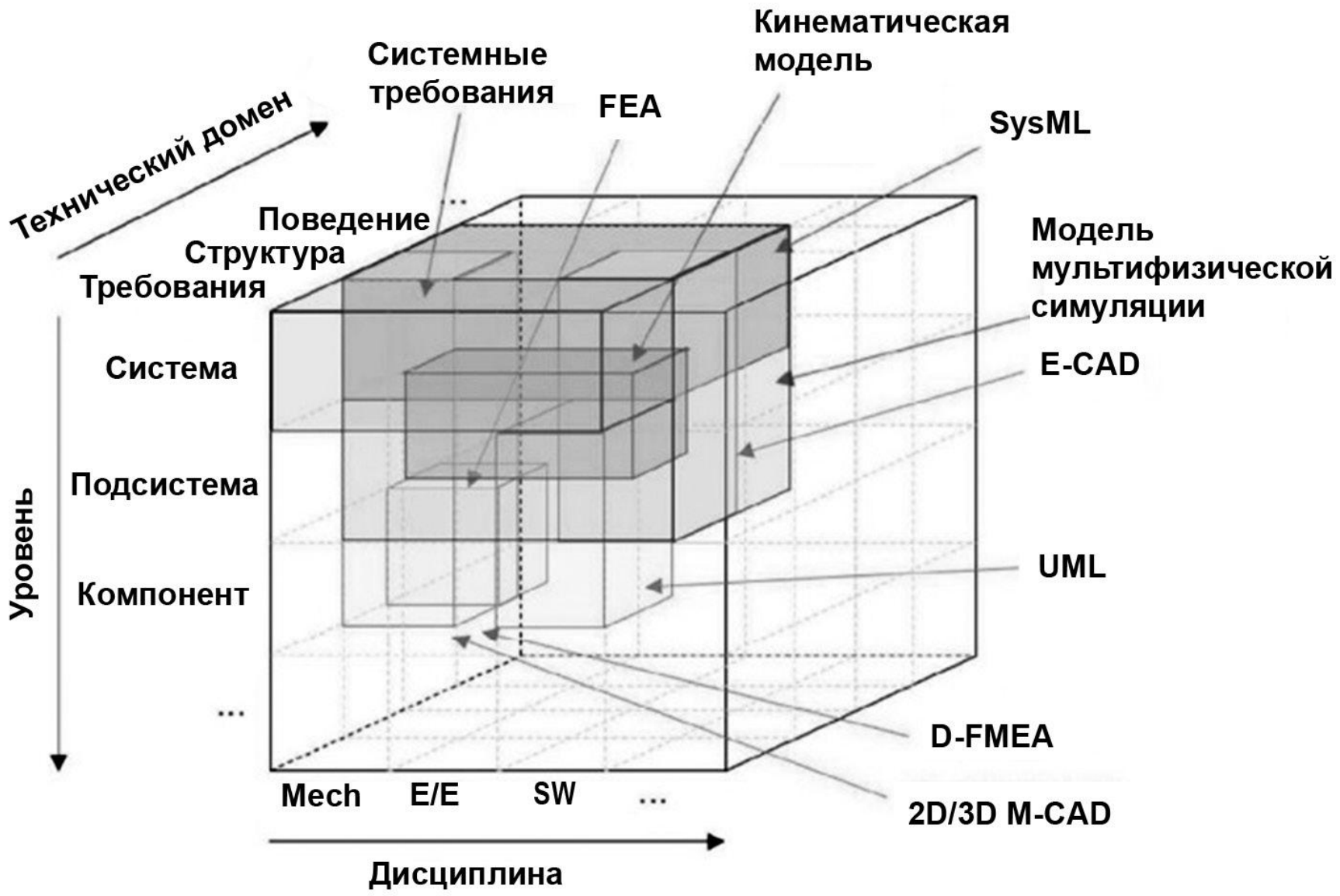
- В рассматриваемом стандарте предлагается трехмерная модель фреймворка MBSSE RF (MBSSE reference framework), ориентированная на поддержку подхода MBSSE для организации и формализации выполнения процессов жизненного цикла, описанных в процессных стандартах
- Третья размерность – это когнитивная, или познавательная (cognition dimension), размерность фиксирует процессы и результаты понимания и преобразования объективного мира. Это измерение личности и организации



Эталонная модель для общего фреймворка и процессов MBSSE

Общий пример экземпляра MBSSE RF [MBSSE ISO/IEC/IEEE 24641]

Эталонная модель для общего фреймворка и процессов MBSSE



Общий пример экземпляра MBSSE RF [MBSSE ISO/IEC/IEEE 24641]

11.8. ПОТЕНЦИАЛЬНО ВОЗМОЖНЫЕ MBSSE-РОЛИ

В приложении D стандарта ISO/IEC/IEEE 24641 описан набор возможных MBSSE-ролей для проектов, реализуемых на основе подхода MBSE. Роли описаны с помощью трех (некоторые двух) атрибутов: имя роли, ответственность и KSA (Knowledge, Skills and Abilities — Знания, Навыки и Способности)

1. Роль: **тренер MBSSE по методам и моделированию или тренер по развертыванию** (MBSSE Method and modelling coach or Deployment coach)
2. Роль: **разработчик моделей MBSSE** (обычно системный инженер или архитектор) — MBSSE modeller (typically a system engineer or architect)
3. Роль: **рецензент модели MBSSE** (MBSSE model reviewer)
4. Роль: **читатель моделей MBSSE** (MBSSE model reader). Обязанности: использовать модели в соответствии с планом управ
ления MBSSE
5. Роль: **администратор инструментов и сред MBSSE** (MBSSE tools and environments administrator).
6. Роль: **управление базой данных администратора модели** (Model Administrator Database management)
7. Роль: **менеджер по управлению данными** (Data governance manager).

Литература к Главе 11

[1] Systems engineering vision 2020, Proc. Int. Council Syst. Eng. (INCOSE), Sep. 2007.
[2] Guide to the Systems Engineering Body of Knowledge (SEBoK), version 2.7. 2022.
[3] Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, Fourth Edition, INCOSE, 2015.
[4] Wymore, A.W. Model-Based Systems Engineering, Boca Raton / A.W. Wymore. — FL, USA. — CRC Press. — 1993.
[5] Ahmad, E. Model-Based System Engineering of the Internet of Things: A Bibliometric Literature Analysis / E. Ahmad. in IEEE Access. — Vol. 11. P.50642–50658, 2023, doi: 10.1109/ACCESS.2023.3277429. https://ieeexplore.ieee.org/abstract/document/10128114
[6] Beery, P. Application of model-based systems engineering concepts to support mission engineering / P. Beery, E. Paulo, Systems, — Vol. 7. — № 3. — P.44. — Sep. 2019 [online] Available: https://www.mdpi.com/2079-8954/7/3/44
[7] Neureiter, C. A domain-specific model based systems engineering approach for cyber-physical systems / C. Neureiter, C. Binder. Systems. — Vol. 10. — №2. — P. 42. — Mar. 2022, [online] Available: https://www.mdpi.com/2079-8954/10/2/42
[8] Ahmad, E. Hybrid annex: An AADL extension for continuous behavior and cyber-physical interaction modeling / E. Ahmad, B. R. Larson, S. C. Barrett, N.Zhan, Y. Dong. — Proc. ACM SIGAda Annu. Conf. High Integrity Lang. Technol. — P. 29–38. — Oct. 2014.
[9] Acheson, P. Model based systems engineering for system of systems using agent-based modeling / P. Acheson, C. Dagli, N. Kilicay-Ergin, Proc. Comput. Sci. — 2013. — Vol. 16. — P. 11–19. — [online] Available: https://www.sciencedirect.com/science/article/pii/S1877050913000033.
[10] Madni, A. Leveraging digital twin technology in model-based systems engineering / A. Madni, C. Madni, S. Lucero, Systems, — Vol. 7. — № 1. — 7, Jan.2019, [online] Available: https://www.mdpi.com/2079-8954/7/1/7
[11] ISO/IEC/IEEE FDIS 15288:2023. Systems and software engineering — System life cycle processes
[12] ISO/IEC/IEEE 12207. Systems and software engineering. Software life cycle processes
[13] ISO/IEC/IEEE FDIS 24641:2022(E). Systems and software engineering — Methods and tools for model-based systems and software engineering.

ГЛАВА 12. ИНТЕГРАЦИЯ СИСТЕМЫ И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ. МЕТОДИЧЕСКИЕ АСПЕКТЫ

12.1. НАЗНАЧЕНИЕ СТАНДАРТА ISO/IEC/IEEE FDIS 24748-6 ИНТЕГРАЦИЯ СИСТЕМ И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ (SYSTEM AND SOFTWARE INTEGRATION)
12.2. КОНЦЕПЦИЯ СИСТЕМНОЙ ИНТЕГРАЦИИ И ЕЕ ОСНОВНЫЕ ПОНЯТИЯ
12.3. ПЛАНИРОВАНИЕ И ЦЕЛИ ПРИМЕНЕНИЯ ПРОЦЕССА ИНТЕГРАЦИИ
12.4. ДРУГИЕ ПРОЦЕССЫ, СВЯЗАННЫЕ С ПРОЦЕССОМ ИНТЕГРАЦИИ
12.5. ТРЕБОВАНИЯ К ИНФОРМАЦИОННЫМ ЭЛЕМЕНТАМ
12.6. МАТРИЦЫ СВЯЗИ (COUPLING MATRICES)

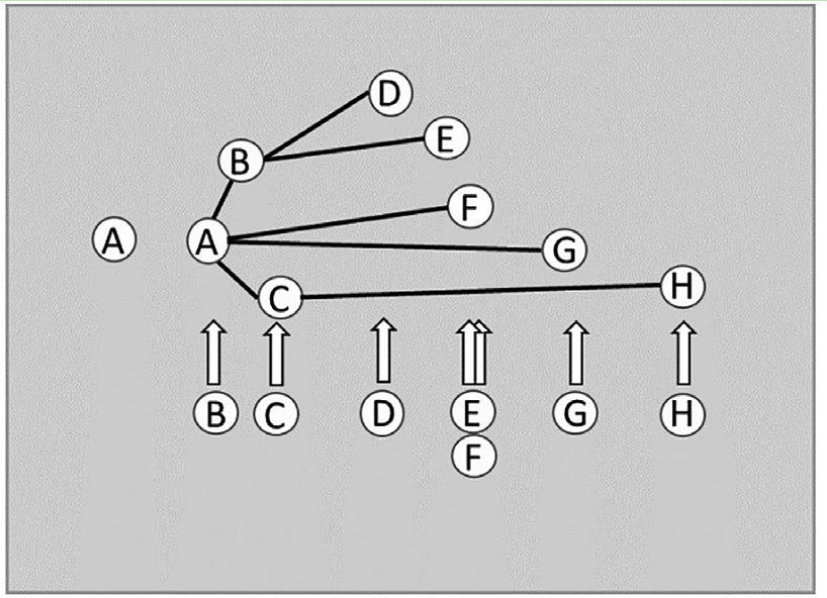
–

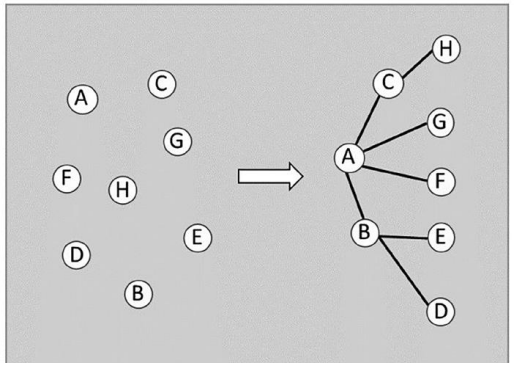
ГЛАВА 12. ИНТЕГРАЦИЯ СИСТЕМЫ И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ. МЕТОДИЧЕСКИЕ АСПЕКТЫ

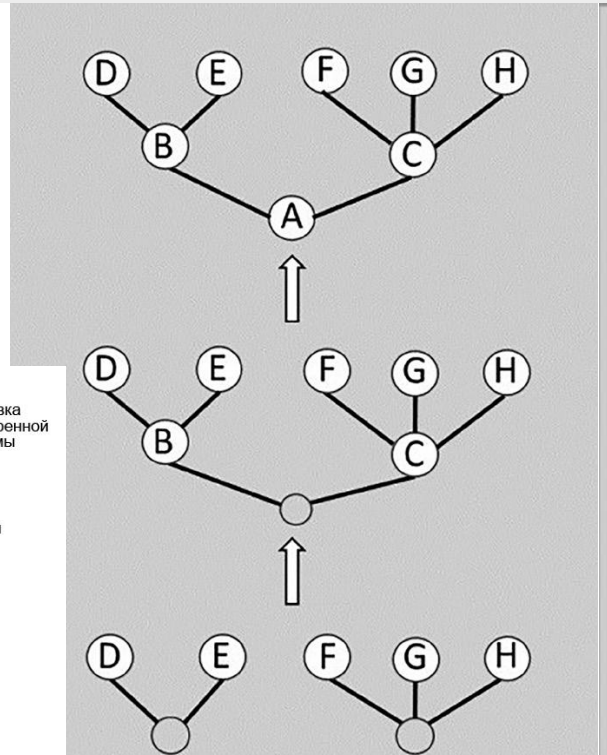
12.3.3. СЦЕНАРИИ ИНТЕГРАЦИИ 12.3.4. МЕТОДЫ ИНТЕГРАЦИИ

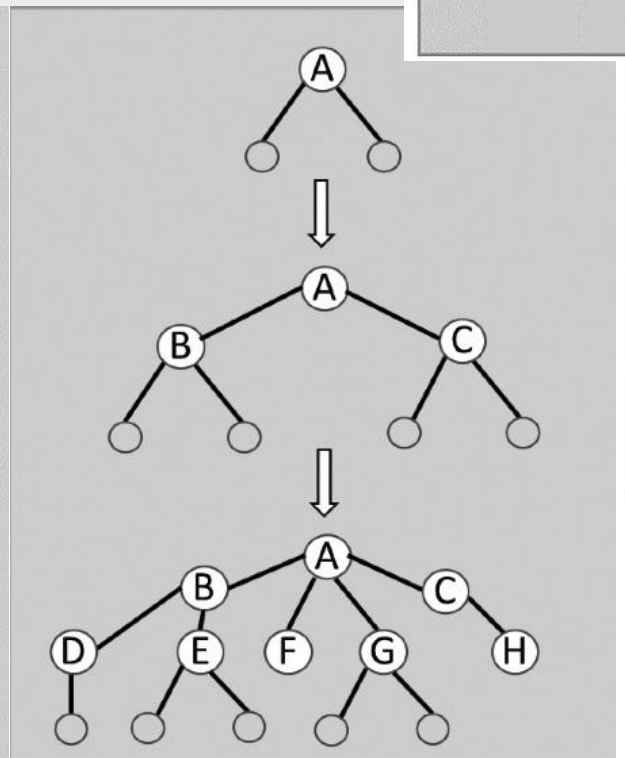
На практике используются такие методы интеграции, как:

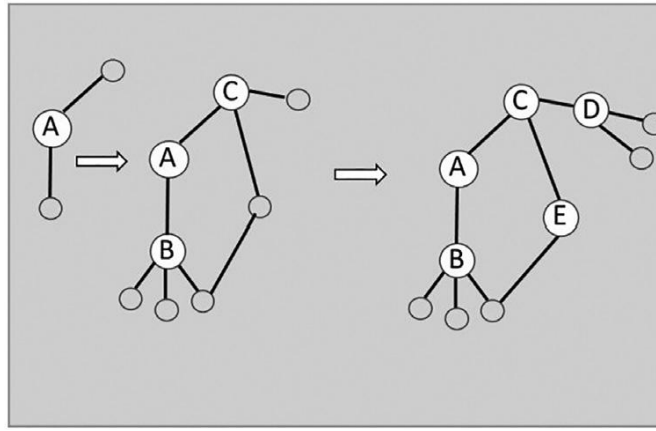
- Глобальная интеграция (Global integration);
- Интеграция снизу вверх (Bottom-up integration);
- Интеграция сверху вниз (Top-down integration);
- Интеграция «с потоком» (Integration «with the stream»);
- Инкрементальная интеграция (Incremental integration);
- Интеграция подмножества за подмножеством (Subset-by-subset integration);
- Интеграция на основе критериев (Criterion-driven integration).

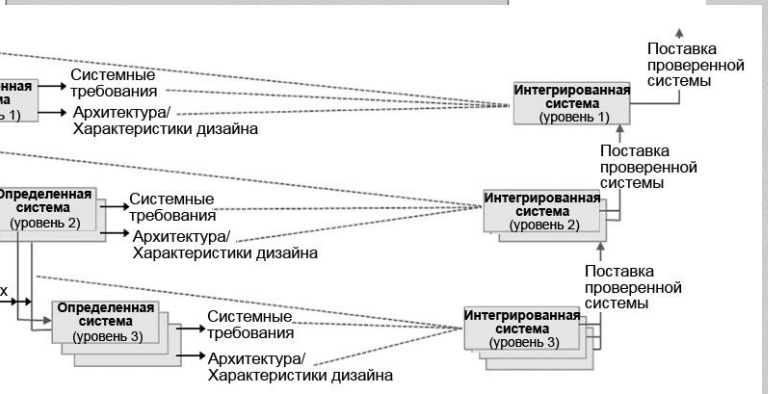


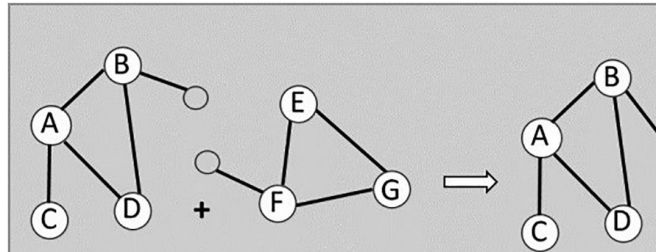








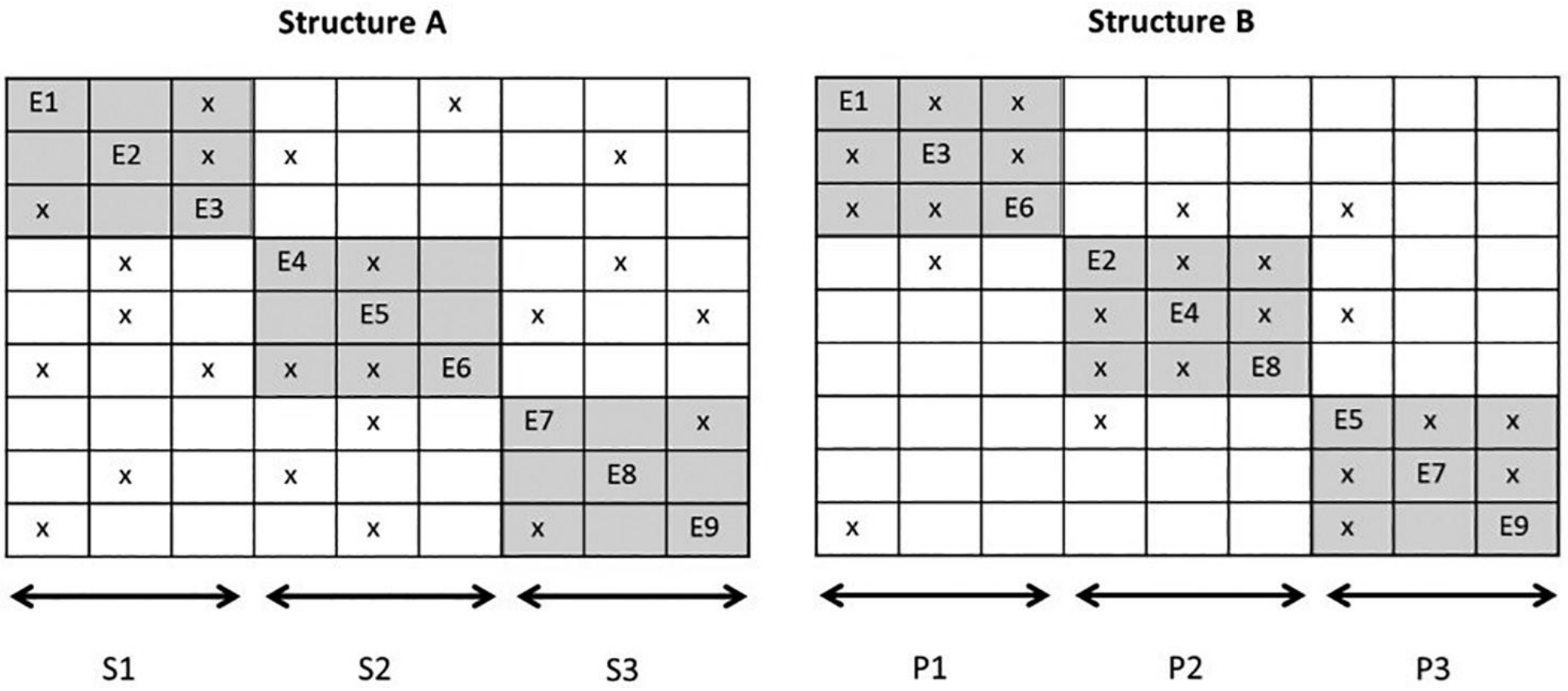




12.3.2. РОЛИ И КОМПЕТЕНЦИИ КОМАНДЫ ИНТЕГРАЦИИ

• **инженеры по требованиям,** которые обеспечивают связь с концепциями, потребностями, ожидаемыми операциями, анализом проблем и разрешением изменений требований с учетом многочисленных воздействий на систему
• **архитекторы**, обладающие знаниями в определении и описании архитектуры системы, особенно в отношении интерфейсов
• **проектировщики (дизайнеры)**, обладающие знаниями в области проектирования системных элементов, тестирования системных элементов и технологий, из которых состоят системные элементы
• **интеграторы**, которые определяют стратегию интеграции, процедуры или операции интеграции, принимают поставку системных элементов, инструментов и ресурсов, выполняют интеграцию системных элементов и обеспечивающих систем, проверяют интерфейсы, подтверждают, что полученная совокупность элементов готова к проверке, и пишут отчеты об интеграции и отчеты о проблемах
• **технологи по верификации**, которые проводят тесты для определения соответствия интерфейсов системным требованиям;
• **технологи по валидации**, которые применяют тесты для определения соответствия функциональности проверенного интерфейса требованиям заинтересованных сторон
• **технологи обеспечения качества**, подтверждающие, что результаты интеграции на каждом этапе дают необходимые результаты

12.6. МАТРИЦЫ СВЯЗИ (COUPLING MATRICES)



Литература к Главе 12

Литература

[1] ISO/IEC/IEEE FDIS 24748-6 System and software integration

[2] INCOSE MBSE Initiative Survey of Candidate Model-Based Engineering (MBSE) Methodologies Page 1 of 70 Rev. B May 23, 2008 INCOSE MBSE Initiative Survey of Model-Based Systems Engineering (MBSE) Methodologies Jeff A. Estefan Jet Propulsion Laboratory California Institute of Technology Pasadena, California, U.S.A. https://www.omgsysml.org/MBSE_Methodology_Survey_RevB.pdf

В. А. Сухомлин, В. Ю. Романов, Д. А. Гапанович

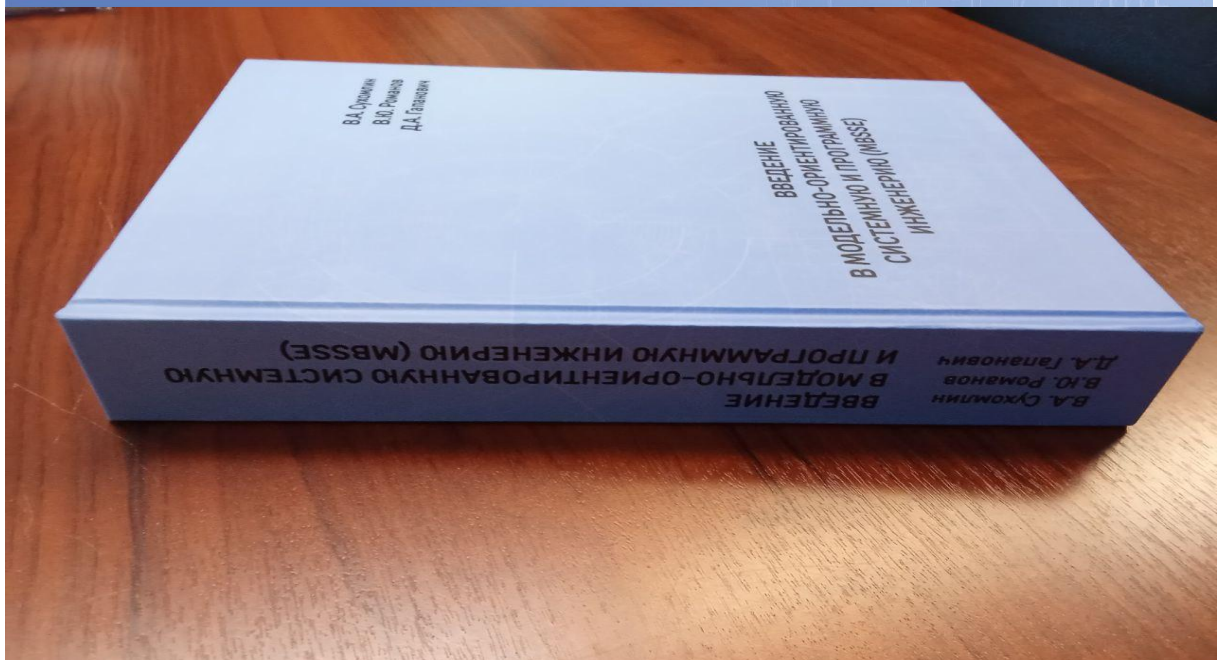
Введение в модельно-ориентированную системную и программную инженерию (MBSSE)

Учебник для вузов

Рекомендовано ФУМО по укрупненной группе специальностей и направлений подготовки 02.00.00 Компьютерные и информационные науки в качестве учебного пособия для студентов высших учебных заведений, обучающихся по направлениям подготовки: «Фундаментальная информатика и информационные технологии», «Математическое обеспечение и администрирование информационных систем», «Математика и компьютерные науки»

МАКС ПРЕСС

МОСКВА – 2024



В.А. СУХОМЛИН, В.Ю. РОМАНОВ, Д.А. ГАПАНОВИЧ «ВВЕДЕНИЕ В МОДЕЛЬНО-ОРИЕНТИРОВАННУЮ СИСТЕМНУЮ И ПРОГРАММНУЮ ИНЖЕНЕРИЮ (MBSE)»

УДК 004.413(075)
ББК 32.973.2-018-02я7
С91

https://elibrary.ru/wdwbit

Рецензенты:
М.А. Посыпкин – член-корр. РАН, д.ф.-м.н., профессор;
Д.Е. Намиот – д.т.н., ведущий н.с. МГУ имени М.В. Ломоносова

Сухомлин В. А., Романов В. Ю., Гапанович Д. А.

С 91 **Введение в модельно-ориентированную системную и программную инженерию (MBSSE)** : учебник / В.А. Сухомлин, В.Ю. Романов, Д.А. Гапанович. – Москва : Фонд «Лига интернет-медиа»; МАКС Пресс, 2024. – 672 с.
ISBN 978-5-317-07289-6
https://doi.org/10.29003/m4300.978-5-317-07289-6

Системная инженерия является трансдисциплинарным научно-прикладным направлением, объединяющим принципы системного подхода, методы, инструменты и стандарты, предоставляющие интегрированную методологию и соответствующие технологии для проектирования и разработки сложных систем, их всестороннего анализа и проверки, эффективного и безопасного использования. Современный подход к системной инженерии характеризуется акцентированным моделированием систем на протяжении их жизненного цикла с конечной целью создания их цифровых двойников. Он получил название модельно-ориентированной системной инженерии (MBSE, Model-Based Systems Engineering). Последовательная интеграция MBSE с программной инженерией ведет к формированию направления MBSSE (Model-Based Systems and Software Engineering). Именно этому направлению и посвящен данный учебник, главной задачей которого является изучение концептуальных и научно-методических основ MBSE и MBSSE, моделей и методов управления жизненным циклом систем и программного обеспечения, базовых стандартов системной инженерии, изучение и овладение языками моделирования систем UML и SysML и соответствующими инструментами моделирования, изучение методов инженерии требования и разработки описания архитектуры системы, ознакомление с концепцией цифровых двойников и их ролью в MBSE, изучение эталонной модели MBSSE и ее связи с процессными стандартами системной инженерии, изучение методических аспектов процесса интеграция системы и программного обеспечения. Заключительная глава учебника посвящена изучению математических основ системной инженерии. Рассматриваются как основные аспекты математической теории систем А. Уэйна Уаймора, ставшей важным стимулом развития модельно-ориентированного подхода в системной инженерии, так и современные исследования, включая аппарат конечных автоматов Дэвида Харела, формализм моделирования дискретных систем DEVS, исследования теории категорий как формальной математической основы для системного проектирования на основе моделей. Авторы рекомендуют данный курс в качестве базового при подготовке ИТ-профессионалов.

Ключевые слова: системная инженерия, SE, модельно-ориентированная системная инженерия,

Спасибо за внимание!