

CUSTIS®

Системное мышление и его место в работе аналитика

Максим Цепков

Главный архитектор решений CUSTIS
Навигатор в мире Agile, бирюзовых организаций
и спиральной динамики



**ANALYST
DAYS #18**

Analyst Days, Санкт-Петербург

24–25 мая 2024

Зачем этот доклад?

1

На прошлой Analyst Days я делал доклад [«Рациональное и системное мышление: практики и компетенции аналитика»](#), и в обсуждениях прозвучало: «Зачем так сложно, есть специализированные модели, почему их недостаточно?»

2

В этом докладе я покажу, где именно играет системное мышление при применении известных моделей, таких как BPMN, ArchiMate, C4 model, и какие ошибки совершают, когда их применяют формально

3

Доклад — это не обучение системному мышлению. Системно изложить его в кратком докладе невозможно, но я показываю место применения и хочу побудить к изучению



Лучшее, на мой взгляд, обучение методам рационального и системного мышления — в [школе системного менеджмента Анатолия Левенчука](#) (не реклама)

Что даёт рациональное и системное мышление

Рациональное мышление — представление мира в виде моделей

- Структурирование мира и текстов: выделение объектов и связей
- Умение обобщать и строить абстракции, построение моделей для описания мира
- Понимание чужих моделей и терминов, согласование различных картин мира
- Соотнесение моделей с реальным миром, работа с адекватностью представлений

Системное мышление: мир из систем — способ работы со сложными моделями

- Выделение систем, условность их границ
- Многоплановый взгляд: viewpoint, надсистемы и подсистемы, уровни и слои
- Система в развитии и её окружение: создающая и эксплуатирующая системы



Системное мышление учит эффективным способам смотреть на системы, выделять в них правильные объекты внимания, отличать важное от неважного

Весь мир состоит из систем

- Система — выделенный набор взаимодействующих объектов, для которых внутренние связи сильнее, чем внешние
- Система может образовывать целостность, **большую, чем сумма составляющих**: есть синергетический эффект, эмерджентность, и в этом случае описания частей недостаточно для описания целого
- Первоначально систему понимали как объективное устройство объекта с физическими границами, как вскрытие чёрного ящика
- Применение в биологии и социальных науках привело к понятию **soft system** с нечёткими границами: экосистема, система безопасности — от устройства мира перешли к способу мышления о мире
- Выделение системы зависит от цели: для одних целей система «человек» ограничена его телом, а для других включает личные вещи, обеспечивающие функционирование, например смартфон

Пример: Система доставки интернет-магазина

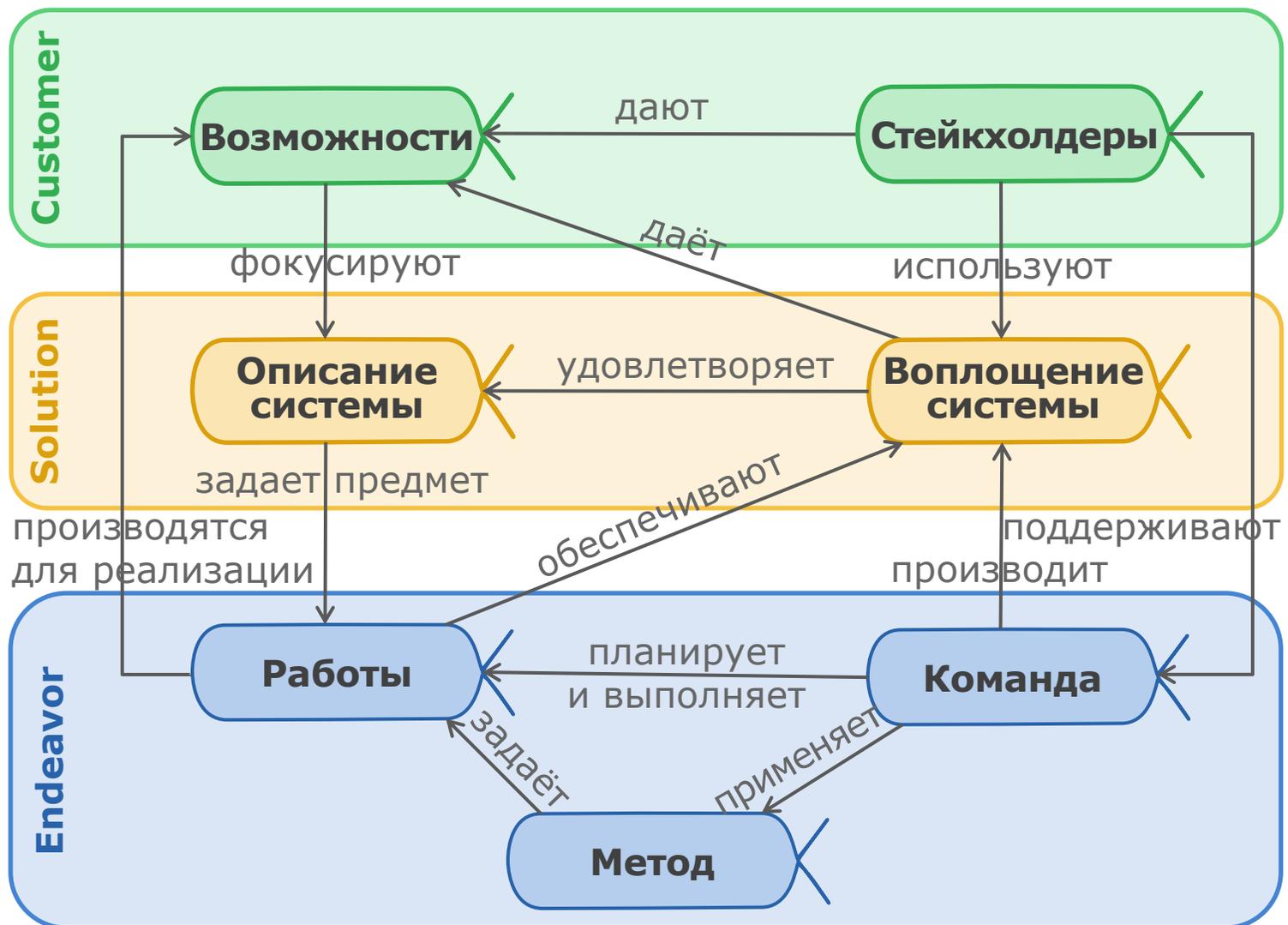
- Понятный процесс доставки: от склада до получателя
- Серая зона: формирование условий доставки при заказе клиента
- Мы используем внешние курьерские компании; это материал для элементов нашей системы доставки, SLA задаёт требуемую морфологию материала
- Если есть собственная служба доставки, то она подобна внешним или нет?
- Если интернет-магазин в составе большой розничной сети, то у него отдельная служба доставки (система), или удобнее рассматривать всю службу логистики как одну систему?
- Может ли автономная собственная служба начинать выполнять чужие заказы?
- Что изменится, если наш интернет-магазин начинает продавать чужие заказы?

Все эти вопросы не только про бизнес, но и про гибкость ИТ-решений

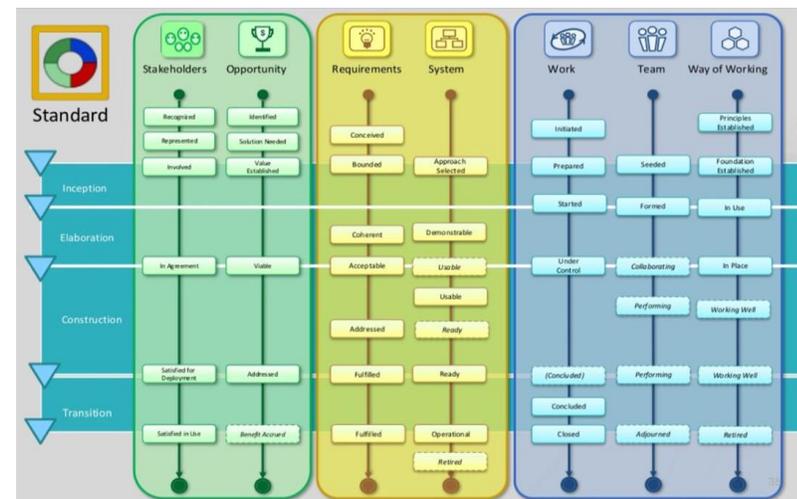
OMG Essence

Стандарт создан для ИТ и принят OMG,
Адаптирован **Анатолием Левенчуком** для
системной инженерии и общего менеджмента

CUSTIS



Концепция: управление деятельностью через состояние **объектов внимания**



Есть **описания всех методов** разработки и сопровождения в ИТ в этом формализме: [в стандарте](#) и [библиотеке Ивара Якобсона](#)

Система и её окружение

Целевая система — та, над которой мы работаем в настоящий момент:

- Различаем описание системы и её саму в реальном мире

Целевая система встроена в **надсистему**:

- Выполняет определённые функции в ней
- Отвечает на интересы агентов надсистемы

Эксплуатирующая (обеспечивающая) система обеспечивает работу целевой, обычно включает **команду** и другие части, живые и неживые.

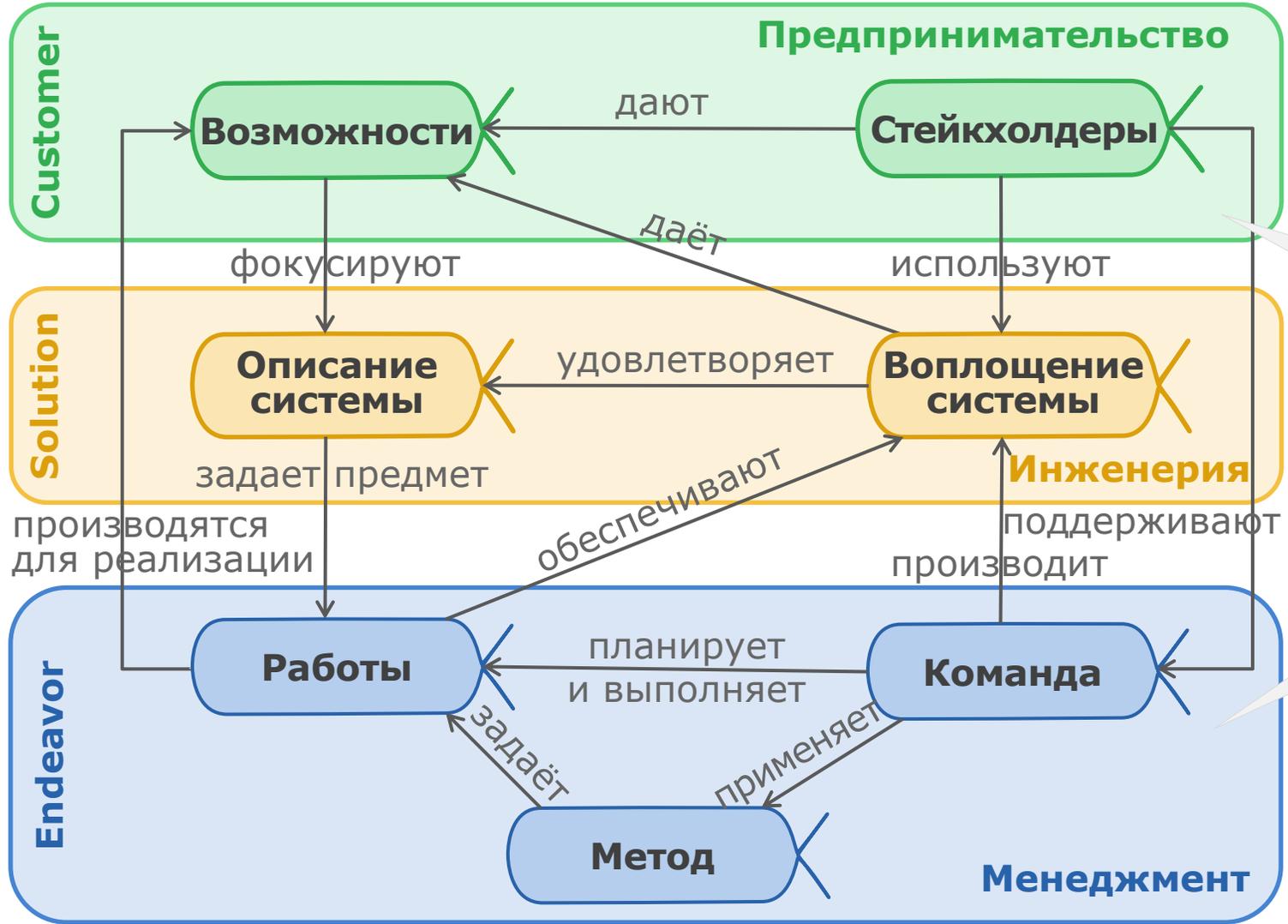
Создающая система — та, что создаёт и развивает целевую.

Окружение системы — смежные системы, с которыми целевая взаимодействует в процессе своей работы.



Нетривиальные связи — **закон Конвея**: отражение в структуре создаваемой ИТ-системы орг. структуры команд создающей системы

OMG Essence: сопряжение систем



Три системы: надсистема, целевая и создающая; для ИТ — бизнес, софт и команда

Сопряжение с надсистемой: ценность и люди

Создающая, эксплуатирующая система

Каждую из систем изучает своя дисциплина: предпринимательство, инженерия и менеджмент



Бизнес и софт



Это две разных системы или два уровня в одной системе?
Ответ зависит от встройки софта в бизнес и от целей рассмотрения

Классический подход

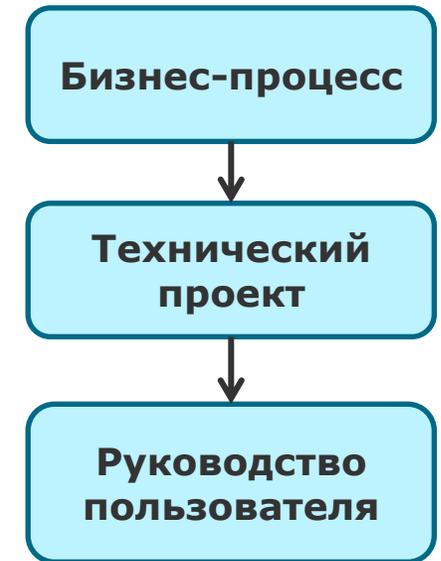
Разные системы, поэтому несколько отдельных документов:

- Описание бизнес-процессов, которые система поддерживает
- Руководство пользователя по работе с системой
- Технический проект системы: при проектировании именно показывал связь и объяснял, как система будет поддерживать бизнес-процессы

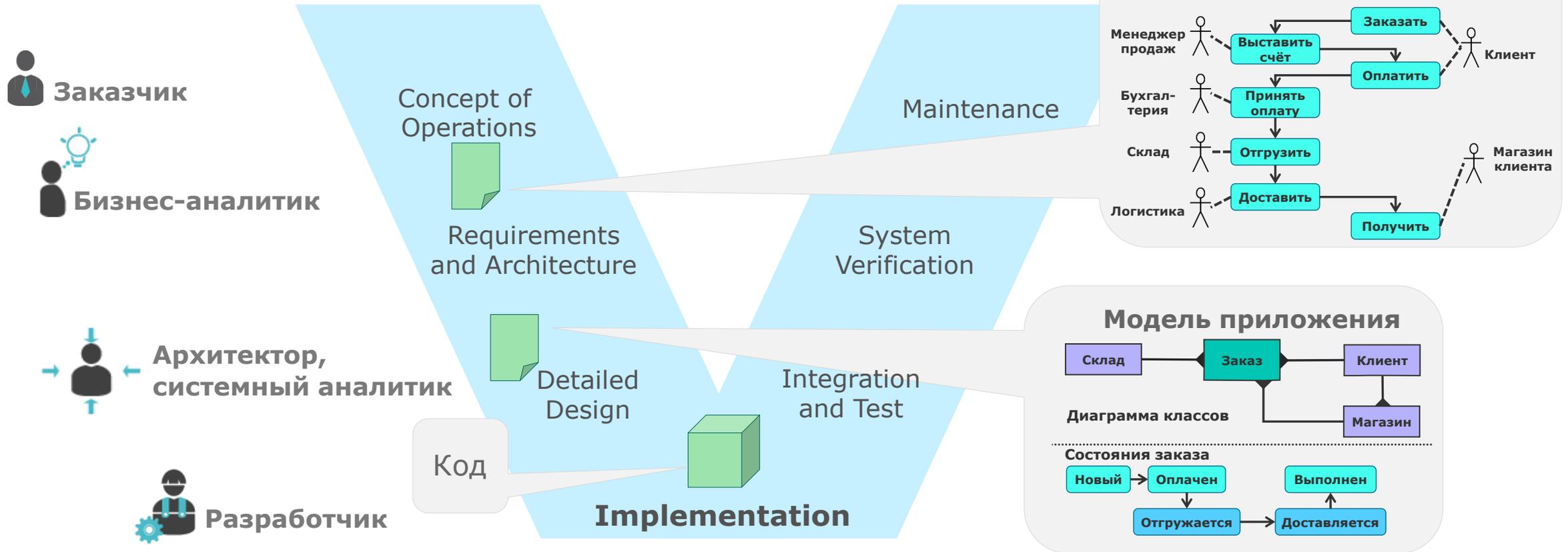
Проблема. Системы сильно связаны, а описания живут независимо, устаревают и перестают соответствовать друг другу:

- Руководство пользователя описывает экраны, связь с процессом неочевидна
- Описание бизнес-процессов обычно сильно отстаёт от фактической работы
- Технический проект оказывается ориентирован на админов

Способ борьбы — ограниченное описание бизнес-контекста в виде user story, или use case, или context diagram, включённые в описание ИТ-системы.



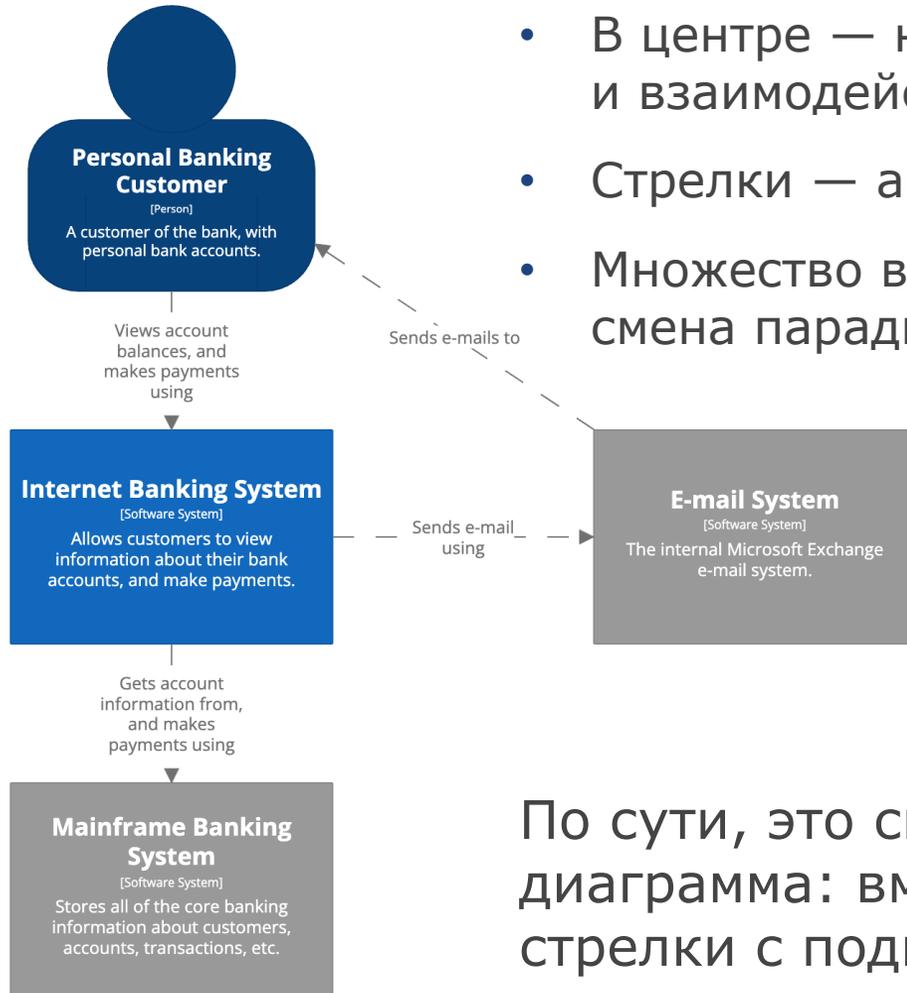
Разделение труда предполагало две системы



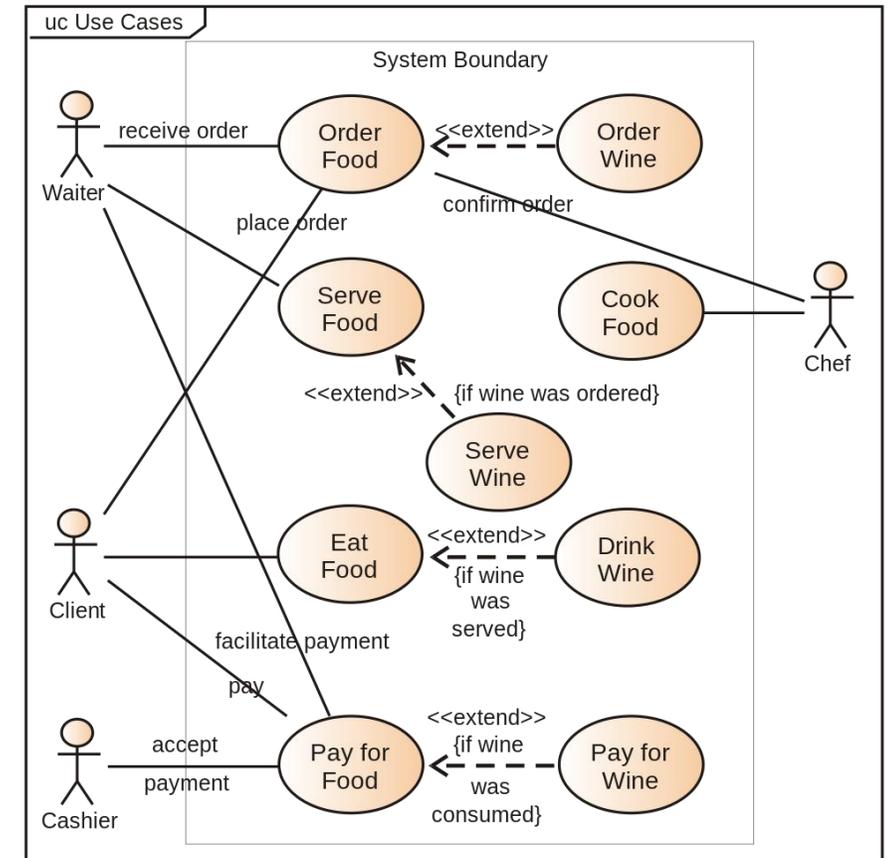
Опыт показал, что такое разделение приводит к несоответствию ИТ-системы ожиданиям бизнеса. Agile-методы это поменяли, но подход к описаниям остался

Контекстная диаграмма C4 model — система в окружении

- В центре — наша система, вокруг — пользователи и взаимодействующие системы
- Стрелки — акты взаимодействия
- Множество внешних систем — смена парадигмы на сервисы



По сути, это свёрнутая use case диаграмма: вместо овалов — стрелки с подписями



Зачем нужно знать пользователя?

Кейс: запрос на доработку. Сейчас при вводе оплаты система автоматически привязывает её к первому не закрытому договору клиента (так сделали MVP). Требуется уметь указывать договор, так как клиент может внести предоплату по новому договору, не закрыв старый, чтобы ему отгрузили новый товар.

Очевидное решение: добавляем поле договора на форму ввода платежа; дорабатываем сервис создания платежей из выписки, чтобы он не заполнял договор; подсвечиваем платежи без договора в списке для разбора.

Скрытая проблема: входящие платежи обрабатывает бухгалтер, и ему нужно сделать это быстро, а договор может определить только менеджер.

- Если при согласовании постановки явно видно, кто выполняет действие, то проблема будет выявлена до разработки, иначе — только на демо или даже в эксплуатации

При неверном решении бухгалтер просто ставит первый попавшийся договор.



Указание пользователя конкретизирует встройку в надсистему, указывая конкретный элемент, и мы проверяем работоспособность связи — компетенции пользователя, необходимые для её работы

Фокуса на одной системе мало

Если при проработке постановки указывать только взаимодействия, явно относящиеся к системе, то не все шаги бизнес-процесса могут быть поддержаны.

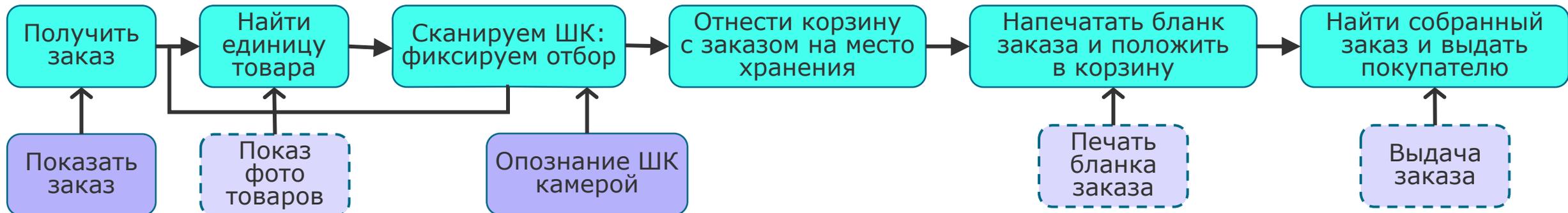
Кейс: запрос на доработку. Поддержать на мобильном рабочем месте отбор одежды по заказу в торговом зале магазина.

Очевидное решение — два шага: получить заказ и фиксировать товар поштучно.

Проблемы:

- Как продавцу быстро найти нужное? Модели и цвета похожи, посмотреть этикетки — долго
- После отбора заказ надо куда-то положить, чтобы потом выдать. Как?

Проблемы видны, только если представить процесс отбора по шагам, и для каждого шага указать функции поддержки, например диаграммой в ArchiMate.



Archimate — связь бизнеса и софта



Domain Driven Design

Пример: снабжение магазинов

Бизнес-процесс — Activity Diagram

Объекты — Class Diagram

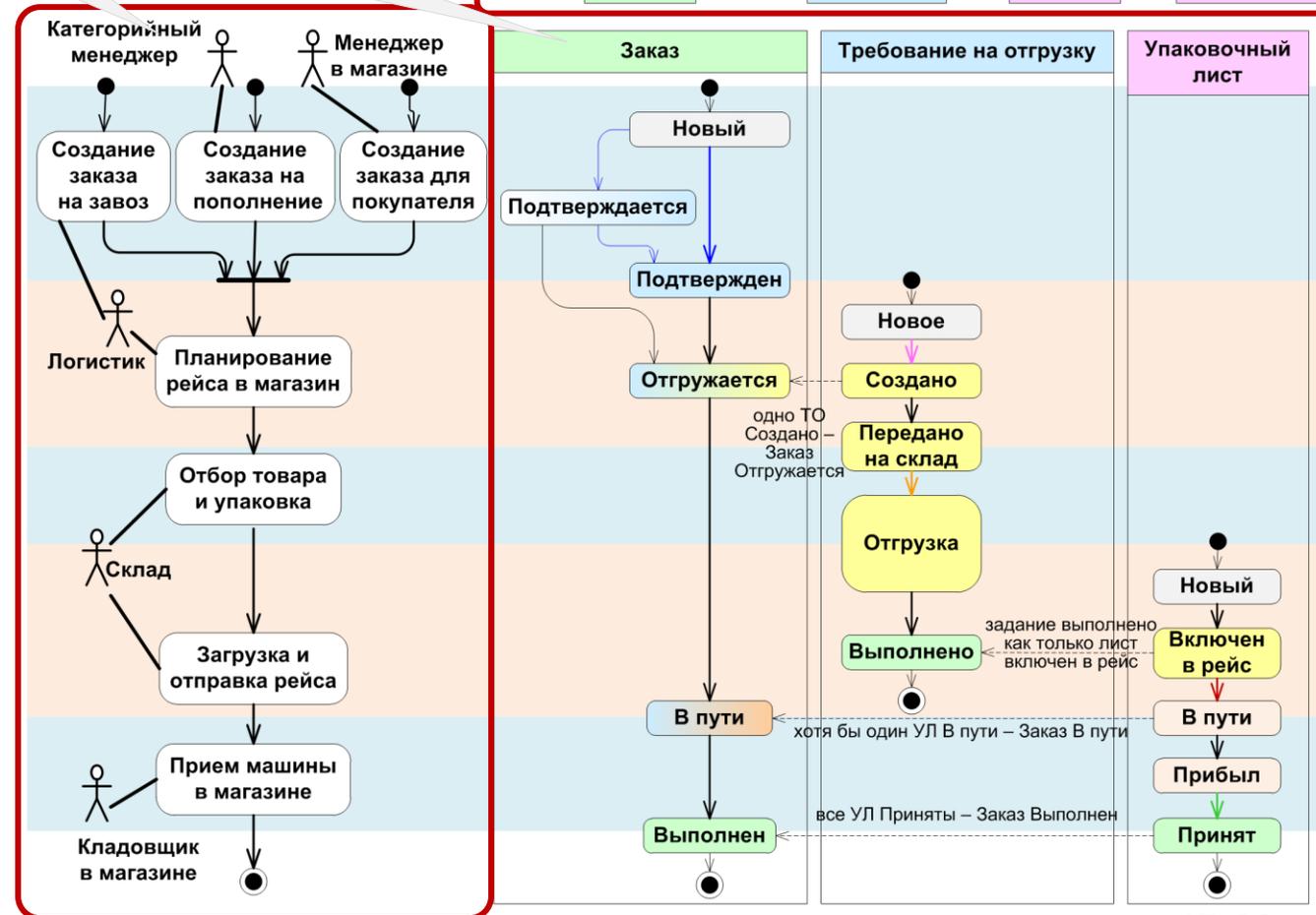
Состояния документов — State Diagram



Прозрачное соответствие между бизнес-моделью и реализацией в ИТ-системе:

- Классы системы соответствуют документам предметной области
- Workflow документов реализует бизнес-процесс

ArchiMate тоже ориентирован на прозрачное представление — через сервисы и объекты данных.

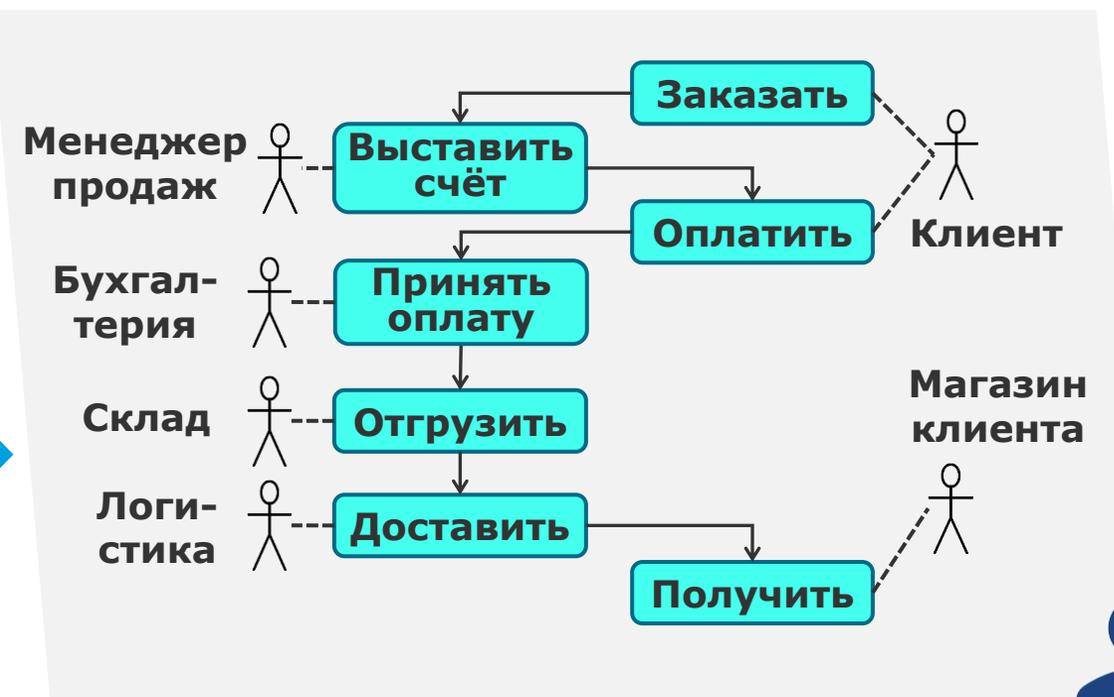




Бизнес-архитектура

Формализация бизнес-модели

Оптовые продажи магазинам и торговым сетям



Как осуществляется переход к формальной бизнес-модели от представлений о повседневной деятельности на интервью?

Процессы и потоки как объекты

Необходимо превратить натуральное представление повседневной деятельности в структурную модель.

Объектно-ориентированный подход хорошо выделяет статическую структуру:

- Структуру компании и окружения: подразделения, контрагенты и клиенты
- Описание предмета работы: товары, услуги, продукты
- Информационные объекты: договоры, накладные, платежи и другие документы

Для представления динамики деятельности используются другие модели:

- Бизнес-процессы и функции
- Отражение деятельности как потока ресурсов — учёт и бюджеты
- Расписания — увязывание деятельности и ресурсов

Для ряда из них разработаны эффективные представления в схемах: схемы бизнес-процессов, диаграммы потоков данных, диаграммы Ганта, диаграммы учёта.

Три уровня представления — обработка интернет-заказа



Деятельность



Схема бизнес-процесса



Объекты и их состояния

Формализация снабжения магазинов

Неформальная схема деятельности и её отражение в существующих системах

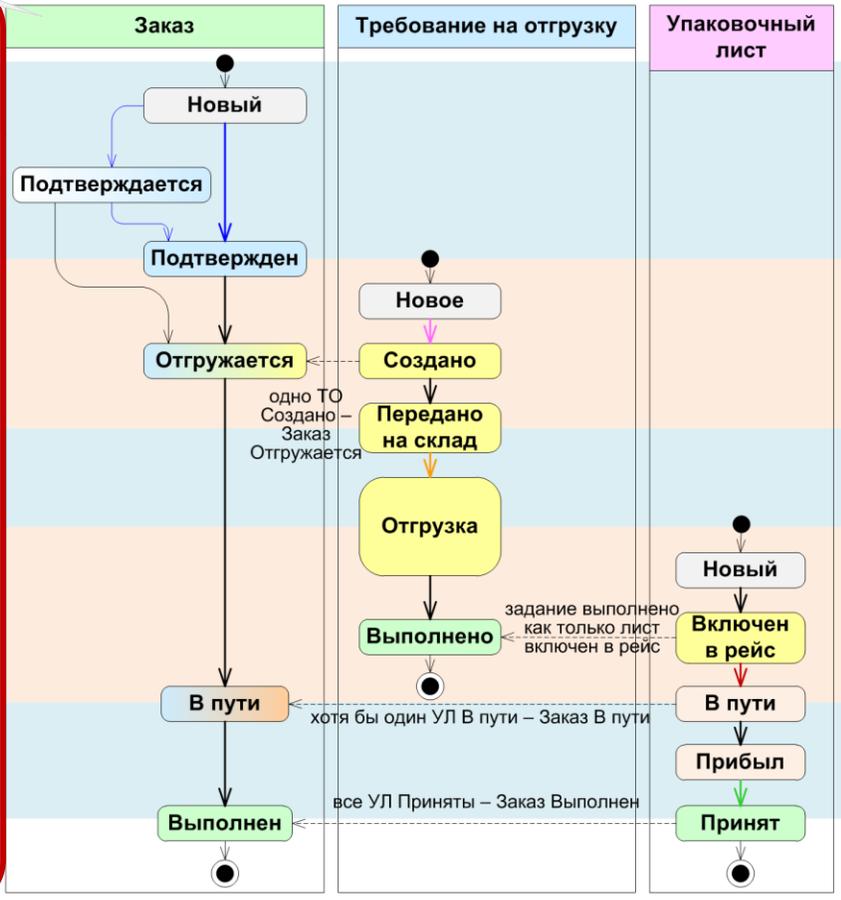
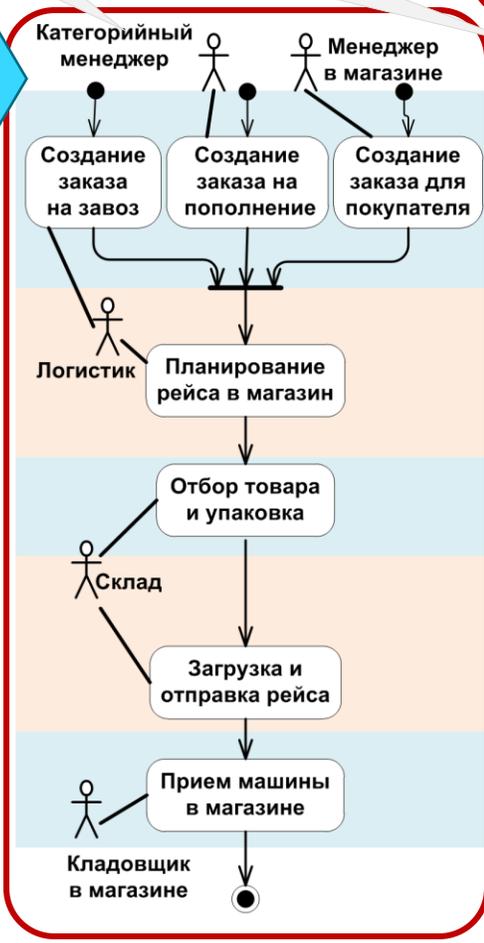
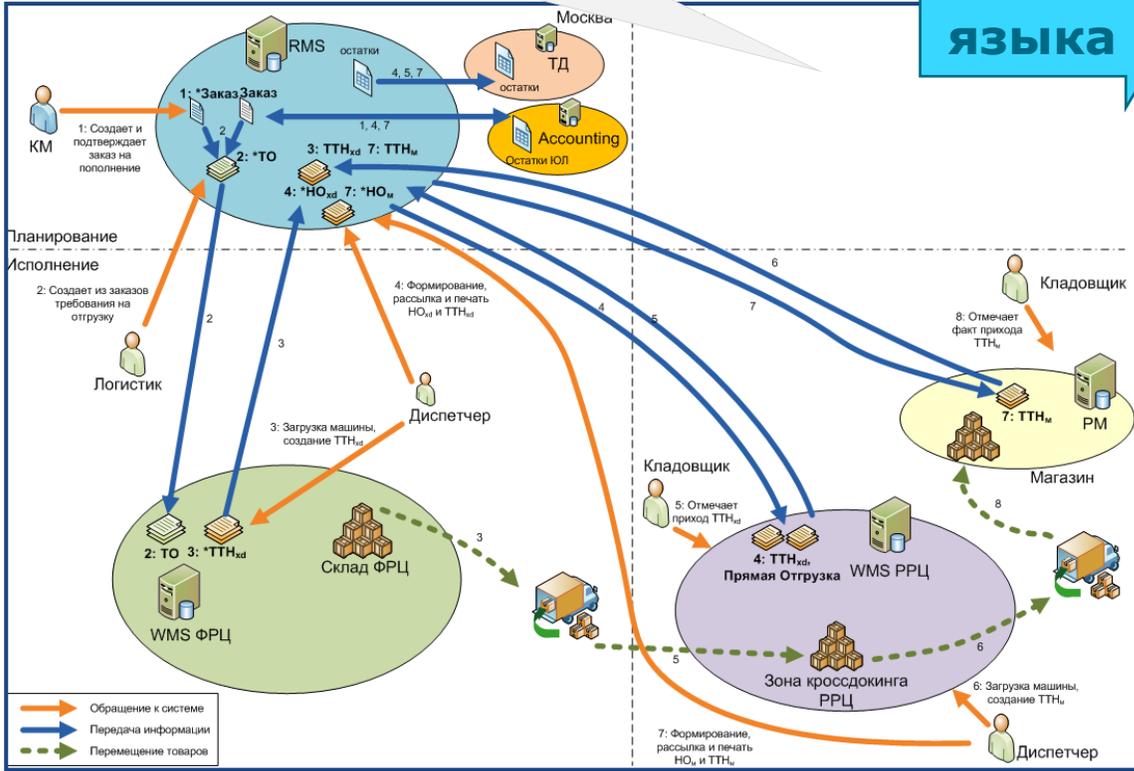
Бизнес-процесс — Activity Diagram

Объекты — Class Diagram

Состояния документов — State Diagram



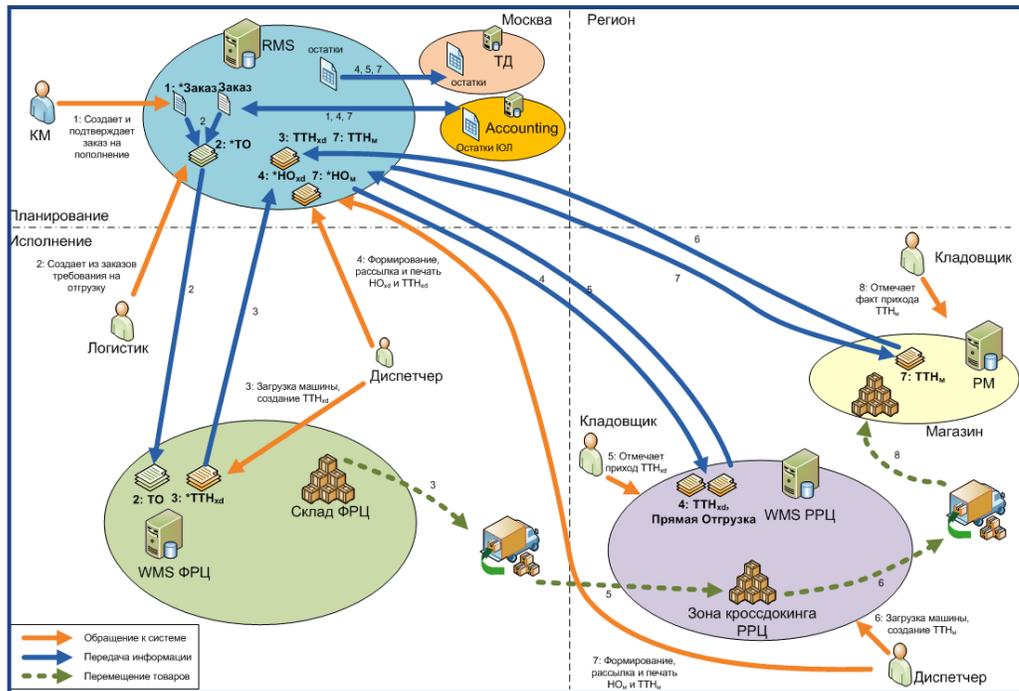
Смена языка



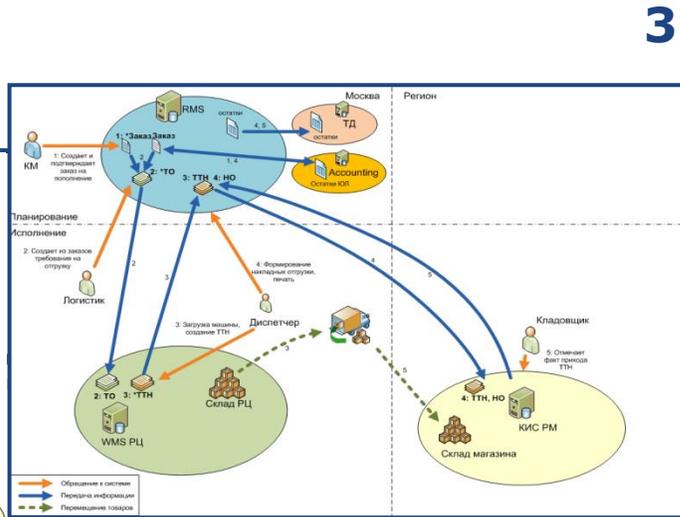
Проверка в неформальной модели

Мы можем построить формальную модель процесса и его реализации, но не всегда заказчик может её проверить. Часто он доверяет, а на внедрении вскрываются проблемы. Решение — вернуться в неформальную модель или показывать прототипы.

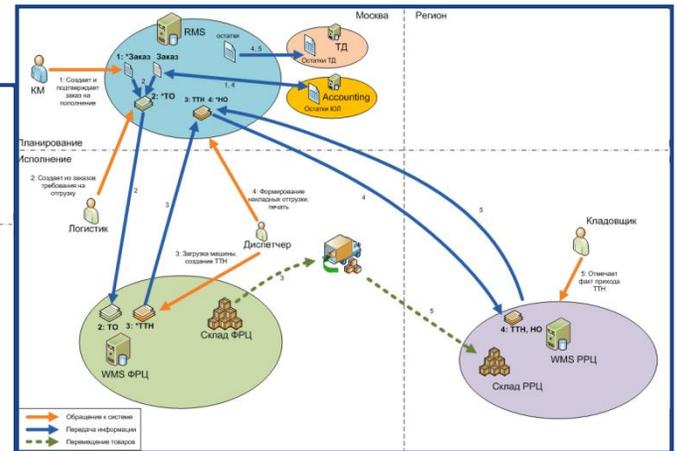
1



2



3



Снабжение магазинов: на неформальных моделях показывали кейсы работы будущей системы

Схема мыследеятельности: барьер понимания — системное явление

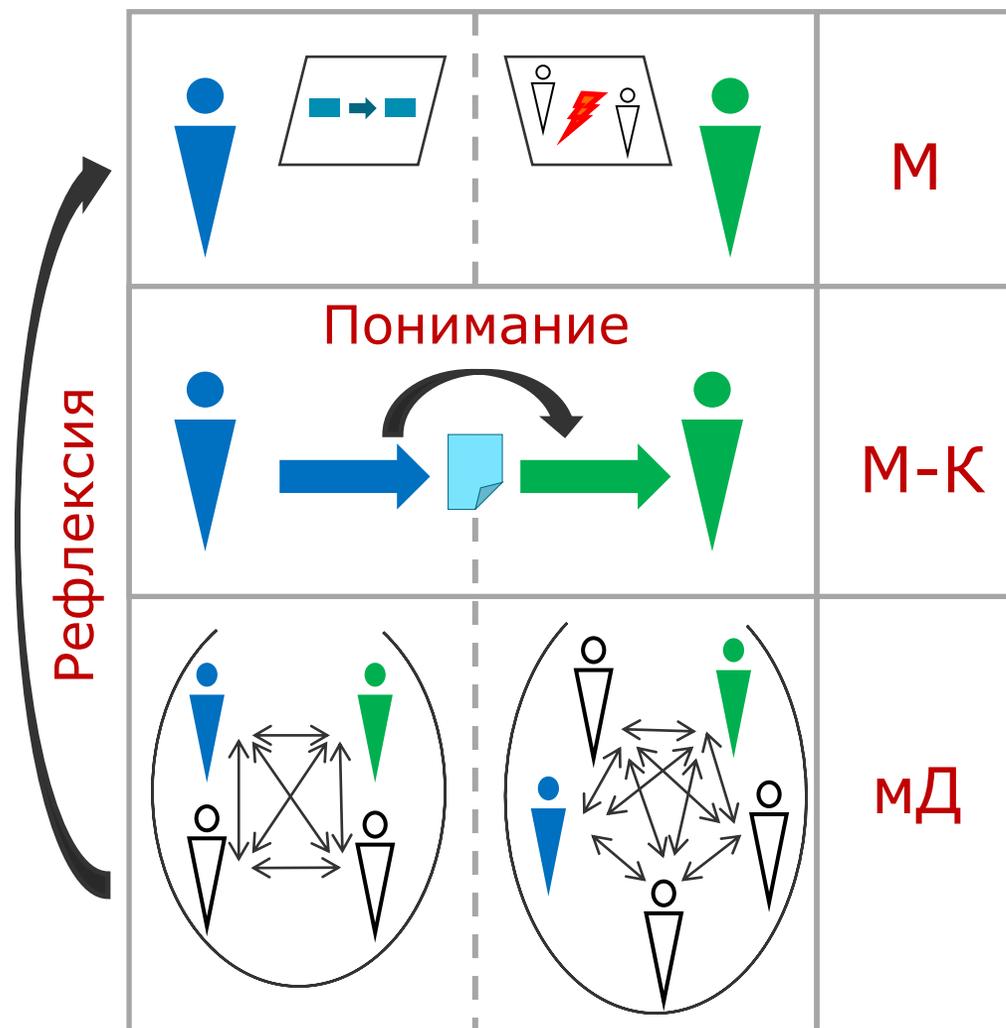


Схема из СМД-методологии

Три слоя:

- Чистое мышление
- Коммуникация
- Деятельность

Барьер понимания из-за разных представлений о мире:

- Разные картины мира
- Разный смысл слов и понятий

Позиции в деятельности с разными представлениями.

За схемой — реальный процесс



Если смотреть на схему, без содержания процесса, то его изменение может дать обратный эффект или привести к побочным

Кейс. Ремонтные бригады закрывали наряды на выполненную работу с задержкой, из-за этого не было адекватных остатков на складе — списание по закрытию наряда. От бригадиров потребовали закрывать наряды в день работы.

Ситуация ухудшилась: до выполнения работ в нарядах — плановые материалы, а не фактические. Вносить факт после работы не было времени и желания, поэтому закрывали с расходом, не соответствующим фактическому.

Кейс. Для оптимизации расходов на логистику перешли от доставки от подрядчиков на центральный склад к доставке на районные склады, исключили много операций. Но сохранили оприходование после поступления документов в бухгалтерию: срок оприходования возрос до 2-4 недель, так как поставщик собирает полный пакет.

Слабо структурированная деятельность

Есть довольно много примеров бизнес-функций, для которых сложно построить описание в виде процесса, при том что вовлечено много людей:

- Выверка бухгалтерской отчётности с решением обнаруженных проблем
- Планирование расписания при недостатке ресурсов с переговорами
- Поиск баланса между текущей работой, ремонтом (техдолг) и развитием
- А также обработка исключений и особых случаев

Не надо пытаться описать подобное языком бизнес-процессов!

Работает координация по целям, критерии продвижения к цели и её достижения.



Роль системного мышления — увидеть границы метода, и не пытаться применять его не по назначению. Апологеты методов часто выступают за повсеместное применение

Event Storming и бизнес-процессы

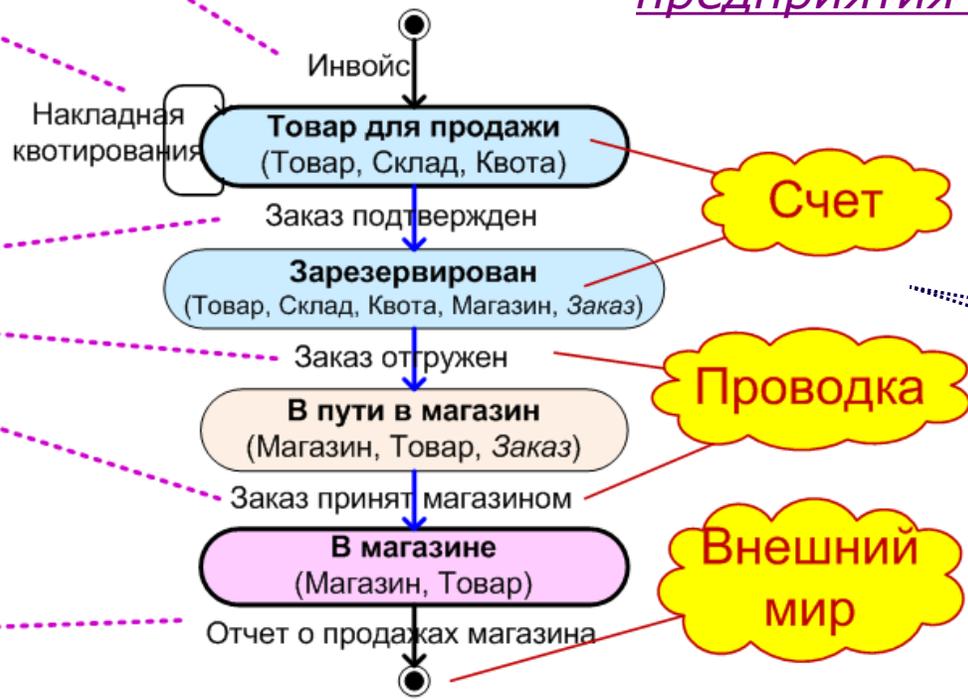
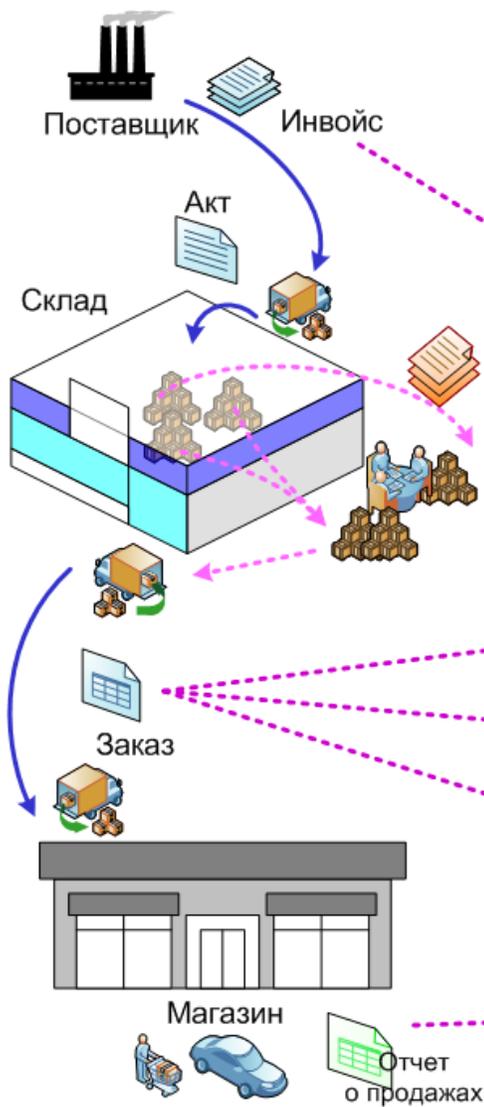
BPM — моделирование бизнес-процессов:

- Представляем деятельность как последовательность шагов с ветвлениями
- Хорошо подходит для основного потока операций, плохо — для исключений
- Есть работы, которые «состоят из исключений»: выверка отчётов, увязка планов, для них подходит Case Management, а не Process Management

Event Storming:

- Выдаёт событийную, а не процессную модель устройства деятельности
- Хорошо подходит для автоматизации «от событий» в сервисной архитектуре
- Помогает разобраться в бизнесе, но не даёт целостного представления
- Восстановить целостное представление, связав разные события, можно по-разному: через схему бизнес-процессов, через систему целей или через учёт

Учёт: бизнес как поток ресурсов



Когда всем понятно. Диаграммы учёта: мост между бухгалтером и разработчиком журнал «Бухгалтер и компьютер», №5-2011.

Комплексное представление — мой доклад «Целостное представление деятельности предприятия на диаграммах учёта».

Счет

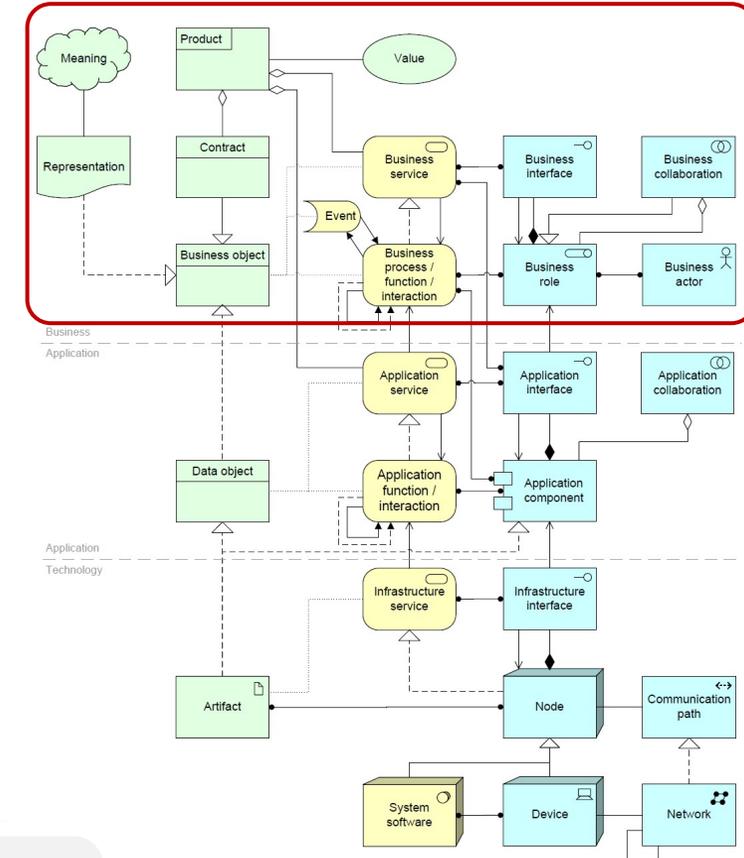
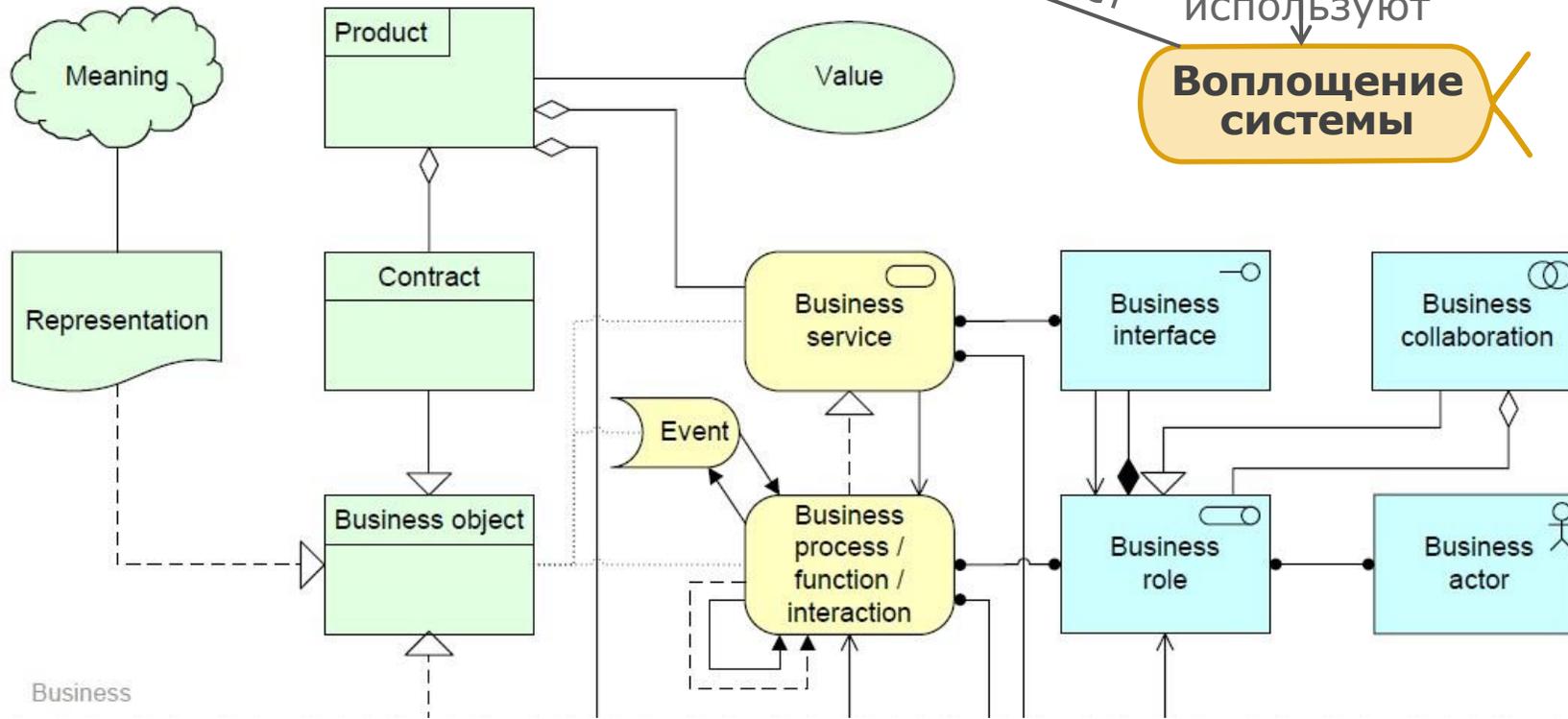
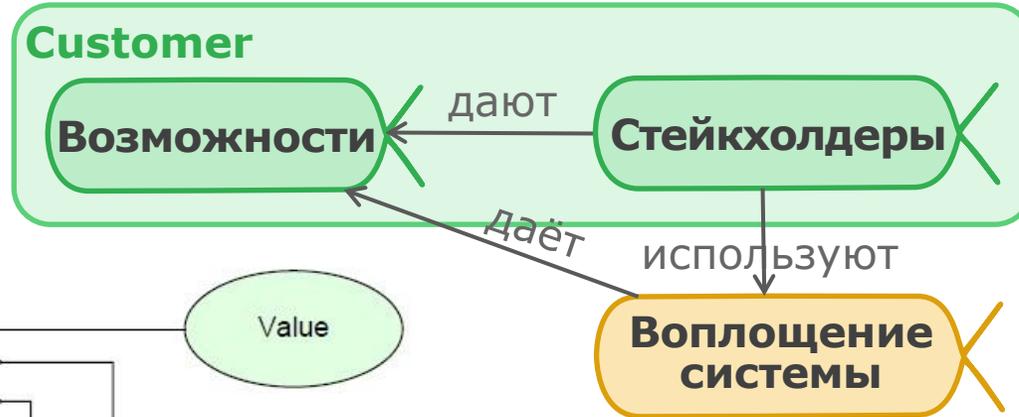
Проводка

Внешний мир

Движение товаров и других ресурсов отражается как проводки между счетами

Бизнес-уровень в ArchiMate

OMG Essence



В основе ArchiMate — продуктовая модель: продукты несут ценность для потребителя, их обеспечивают бизнес-сервисы, включающие процессы и функции, выполняемые агентами

Метамодель ArchiMate 1.0

Продукт как возможность

Пример. Бизнес требует поддержать услугу срочной доставки, так как без неё есть риск проиграть конкурентам, у которых она уже появляется

- Вопросы о возможности: кто будет пользоваться, для каких заказов, что значит «срочная», как мы будем проверять гипотезы и какова мощность в динамике?
 - Справится пеший курьер или нужна машина?
- Будет ли отдельная система срочной доставки или надо существующую систему доставки дополнить элементами и наделить способностью так доставлять?
- Какова целевая операционная стоимость доставки, какова стоимость в пилоте, каковы допустимые инвестиции, в том числе для создания ИТ-систем?
 - Если курьер будет брать самокат, стоимость останется допустимой?
- Какие возможны лёгкие решения на пилоте, справится ли Excel + мессенджер?
- Как будем реагировать, когда не справляемся, кого из клиентов обижать?

И так далее...

Деятельность ведут **агенты**: люди, команды, компании, сообщества.

Театральная метафора — **агент играет роль**, действуя в системе:

- Человек может совмещать роли, например руководитель и разработчик
- Компания тоже играет роли в надсистемах: на рынке, в государстве, в обществе

Роль предполагает конкретное взаимодействие с другими ролями, которое определяется конструкцией системы.

Агент имеет **цель**, для которой он встроился в работу системы, и она некоторым образом соотносится с **целью системы как агента**.

У агента есть **предпочтения** по поводу исполнения своей роли и **интересы**, касающиеся соучастия в результатах работы системы.

Влияние убеждений стейкхолдеров

У стейкхолдеров могут быть принципиальные различия убеждений, влияющие на проект:

- Одни выступают за тщательное планирование как залог успешной работы, и нестрашно, если процесс планирования будет долгим
- Другие говорят, что есть много факторов, которые невозможно предусмотреть, и надо быть гибким, быстро реагировать на изменения

Аналитику надо видеть за словами убеждения, и работать с ними.

Кейс. Дополнительные этапы согласования привели к удлинению процесса снабжения от полугода до года — с увеличением запасов и снижением точности.

Кейс. ERP-система была ориентирована на однократное планирование и исполнение плана без изменений. От новой системы бизнес хотел адаптивного изменения. Эту разницу необходимо было постоянно держать в фокусе при обсуждении архитектурных решений.

Система в развитии, а не в моменте

Система рассматривается в динамике **развития**, различаем функционирование системы и **изменения** этого функционирования.

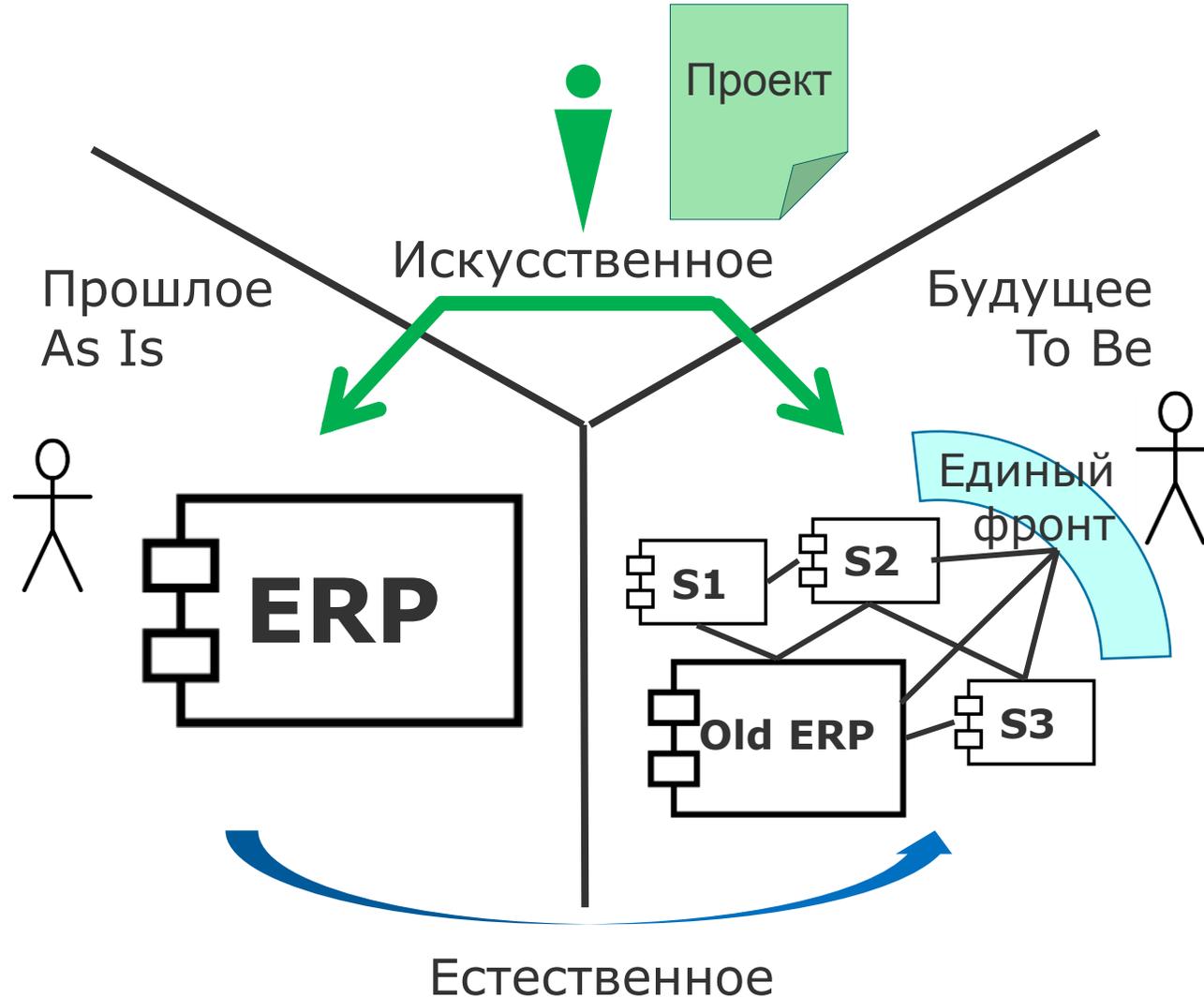
Системы **не стремятся** прийти к равновесию, гомеостазу с миром, у них есть **собственное движение развития**.

Источник развития — **неустроенности**: напряжения и противоречия между разными системными уровнями — баланс интересов системы и надсистемы

- Эволюция устраняет неустроенности экспериментально, методом проб и ошибок
- Люди могут сознательно менять системы для устранения неустроенностей, если знают их устройство, имеют хорошую модель; это **эффективнее** эксперимента

Надо отличать **неустроенности** от логических противоречий в описаниях и моделях, связанных с языком и ограничениями моделей.

Шаг развития: не забываем динамику



Бизнес меняется, ERP дорабатывается — в проекте надо показать, как догоним развитие.

Не всегда вектор развития ERP и нового проекта соответствуют

СМД-методология. Схема шага развития (одна из рисовок)

Подводя итоги...

Системное мышление необходимо:

- Для понимания границ применимости моделей
- Оценки адекватности парадигм моделирования его целям
- Оценки адекватности отражения моделью деятельности

Владение методами системного мышления обеспечивает работу со сложными моделями, понимание бизнеса в целом и места софта в нём.



<http://mtsepkov.org>



[@MaximTsepkov](https://t.me/MaximTsepkov)



На сайте много материалов по [анализу](#) и [архитектуре](#), [Agile](#), [ведению проектов](#), [управлению знаниями](#), мои [доклады](#), [статьи](#) и [конспекты книг](#)



Обсудить доклад



Вакансии

Пишите на hr@custis.ru,
подходите с вопросами