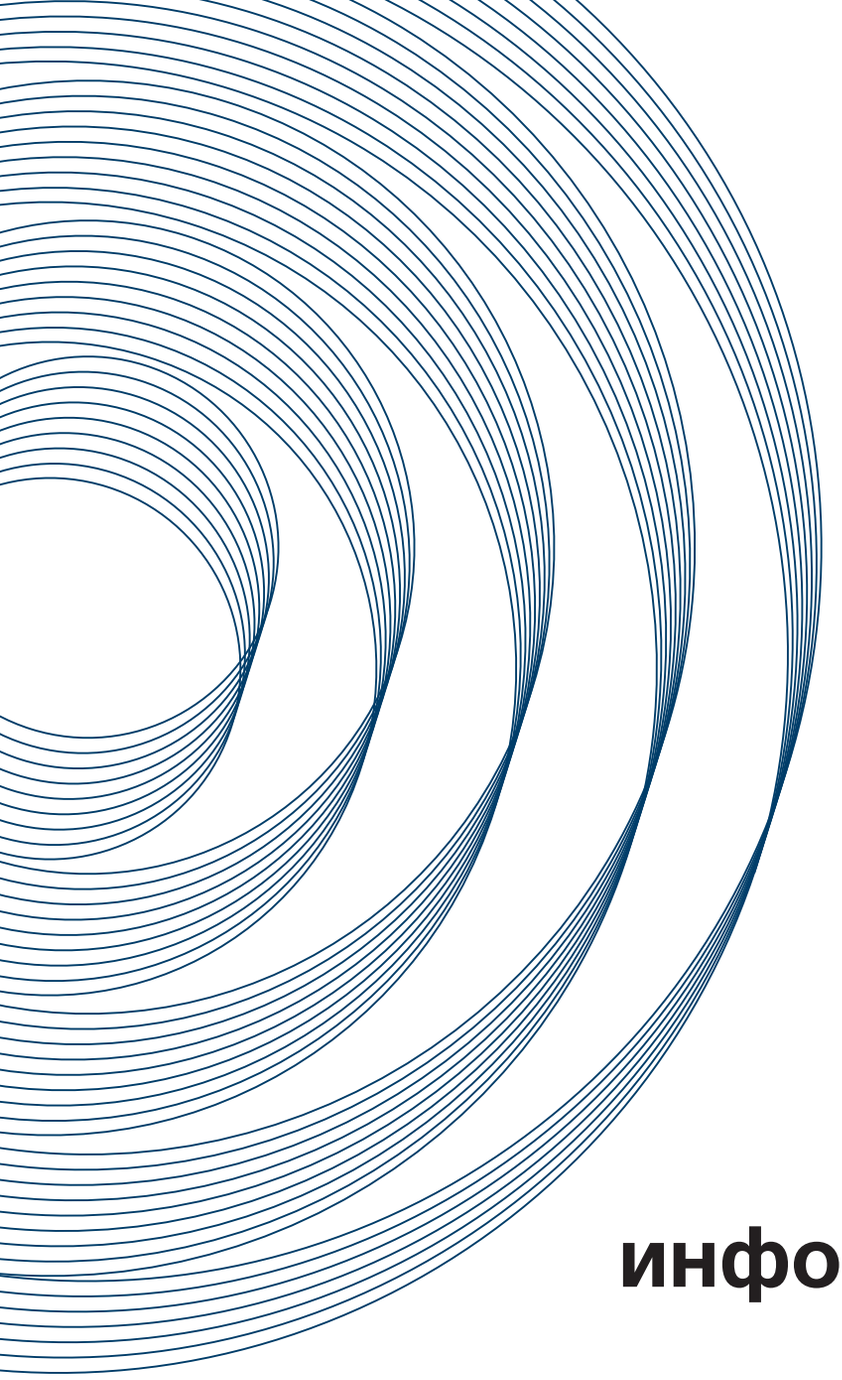


Самый лучший способ
предсказать будущее – это
изобрести его.

Алан Кей

Р.Д. Гимранов
И.Н. Холкин

Изобретая информационные системы будущего

Теория
и практика

Сургут
2017

УДК 519.711:331.1
ББК 22.18.65.054

Редактор: Тюрина Е.В.
Корректура: Тараканов С.Г.
Дизайн: Иванова Е.А.
Верстка: Тринда М.А.

Авторы: Гимранов Р.Д., Холкин И.Н.

Рецензенты: Галкин Валерий Алексеевич,
доктор физико-математических наук,
профессор, директор политехнического института
Сургутского государственного университета.

Лугачев Михаил Иванович,
доктор экономических наук, профессор,
заведующий кафедрой экономической
информатики экономического факультета МГУ
им М.В. Ломоносова.

Изобретая информационные системы будущего.
Теория и практика. — Сургут, 2017. — 192 с.

ISBN 978-5-906157-40-9

В настоящей книге представлен вариант архитектуры цифровой платформы, на которой можно будет создавать различные решения для цифровой экономики. Архитектура может быть использована как подход к обеспечению цифрового бизнеса и как постановка задачи для программирования непосредственного продукта – цифровой платформы.
Авторами описаны научные и методические решения, использованные в работе, некоторые промежуточные результаты творческого процесса, а также, как потенциально близкий вариант реализации, представлено укрупненное описание конкретной платформы.

Дизайн, верстка и подготовка к печати: студия «Аэроплан».
Санкт-Петербург, ул. Заозёрная, 8 «А», www.airoplan.ru.

Подписано в печать 10.09.2017.
Формат 215x280 мм. Тираж 200 экз.
Отпечатано в типографии «Моби Дик», Санкт-Петербург, Менделеевская ул., 9.

УДК 519.711:331.1
ББК 22.18.65.054

ISBN 978-5-906157-40-9

Содержание

Предисловие

Цифровизация сегодня у всех на устах, активно обсуждаются различные аспекты и технологии, ее составляющие. Есть множество авторских взглядов на то, что она должна собой представлять и из чего состоять. В настоящей книге предложен вариант архитектуры цифровой платформы, на которой можно будет создавать различные решения для цифровой экономики. Архитектура может быть использована как подход к обеспечению цифрового бизнеса и как постановка задачи для программирования непосредственного продукта — цифровой платформы.

Несмотря на доступность знаний «в любое время и из любого места», научные и практические сообщества остаются сегментированными вследствие того, что общий объем научных знаний очень велик, структура их разнообразна. Теоретики и практики, как и сто, и пятьдесят лет назад, объединяются вокруг одной парадигмы или совокупности нескольких парадигм — углубляют знания, накапливают умения, формируют портфель решений.

Авторы данной книги, представляя архитектуру онтологической цифровой платформы, также ограничены определенными парадигмами и сообществами. В рамках нашего круга взаимодействия мы не встретили решений, лежащих близко к нашему изобретению, и поэтому претендуем на определенную новизну. При этом существует множество концепций,

приблизившихся к решению, но все же не покрывающих весь спектр требований цифровизации[1].

Исходя из вышесказанного, в книге, кроме собственно архитектуры, описаны научные и методические решения, использованные в работе, некоторые промежуточные результаты творческого процесса, а также, как потенциально близкий вариант реализации, представлено укрупненное описание конкретной платформы.

Надеемся, что книга будет интересна широкому кругу специалистов в области системного подхода, информационных технологий, изобретательства. Кроме собственно архитектуры, представленной в главе 2, ученым будут интересны глава 1 и приложение 1, изобретателям, особенно поклонникам ТРИЗ,— глава 1, приложения 2 и 3, ИТ-специалистам — глава 3.

В силу того что архитектура развивается в деталях, появляются новые аспекты реализации, интересные кейсы, поэтому мы надеемся, что изложенное в книге существенно дополнится в будущем. Идеями и примерами мы будем делиться как через социальные сети (#уб-мангуст), так и в рамках участия в различных мероприятиях.

Игорь Холкин, Ринат Гимранов
Москва — Сургут, 2017

[1] На первый взгляд, контейнерная архитектура, микросервисы, мультиагентные сети, семантические приложения, онтологические СУБД и другие решения отвечают всем требованиям, но это только навскидку, без погружения в детали.

Глава 1.
Картина сегодняшнего мира информационных систем

Значение данной книги в контексте развития информационных систем

Без информационных технологий уже невозможно представить себе наш мир, ИТ стали органичной и неотъемлемой частью цивилизации, вошли в нашу повседневную жизнь, в науку, технику, бизнес. Они развиваются невероятными темпами, превращая идеи писателей-фантастов в привычную реальность. За какие-то пять-шесть десятков лет, на протяжении одной человеческой жизни, фантастическая идея о видеоразговоре между собеседниками на разных континентах превратилась в обыденную фразу: «Выходи в скайп!» И в наше время уже никого не удивляет, что бабушка из Иркутска обсуждает с внучкой, живущей в Праге, какое платье ей к лицу. Бизнесмен, находясь в командировке в Лондоне, просматривает учетные данные на своем складе, расположенном в Шанхае, а прилетев домой, еще в аэропорту включает отопление своего «умного дома» в Подмосковье.

Современные информационные системы и технологии изучаются во многих отраслях знаний. Существует немало исследований по данной теме. Читатель вправе спросить: что же нового хотят добавить авторы книги к этому многообразию?

Мы установили, что ни в одной научной работе по данному направлению не уделено внимание описанию практических действий, позволяющих

заглянуть в «завтрашний день» информационных технологий и создать информационную систему будущего. В многочисленных трудах в данной научной области нет руководства по разработке идеи, качественно преобразующей работу пользователя, и созданию информационной технологии на основе этой, возможно кажущейся современникам фантастической, идеи.

Мы надеемся, что наша книга позволит заполнить нишу практически-ориентированных работ в области информационных технологий. В работе над публикацией нам существенно помогли исследования: методология прогнозирования развития систем на основе ТРИЗ [1], [2], [3], [4], [5], [6], [7], [8], [9], эмержентная стратификация информационных систем [10], [11] и спиральная динамика, представленная применительно к развитию организаций в книге Фредерика Лалу «Открывая организации будущего» — Reinventing Organizations [12].

Опираясь на данные методологические труды, мы открыли свой подход к созданию концепций новых информационных систем (Reinventing Information Systems — перефразируя Фредерика Лалу).

Другими словами, книга, которую вы держите в руках, позволяет целенаправленно изобретать нацеленные в будущее информационные системы «по заказу».

Какие же «заказы от жизни» мы имеем на сегодняшний день? Давайте разберемся.

Взгляд ИТ-специалиста на современную ситуацию в области ИС

За два десятилетия активного развития информационные системы крупных предприятий достигли существенного уровня зрелости: обеспечено основное функциональное покрытие бизнес-требований, имеются и широкий арсенал средств для самых различных применений, и развитые механизмы межсистемного обмена, успешно проведены технологические изменения с целью снижения совокупной стоимости владения.

От набора простых программ, выполняющих расчеты, информационные системы (ИС) развились в сложные многокомпонентные программно-аппаратные комплексы. В составе такой системы на каждом среднем и крупном предприятии имеются наборы серверов и других аппаратных компонентов, большое количество баз данных и систем управления ими, многочисленные

прикладные ИТ-решения, связанные сквозными бизнес-процессами, средства мониторинга, анализа и управления.

Конечно, столь развитая совокупность разнообразных систем порождает проблемы, связанные с масштабностью и структурной сложностью. Например:

- чем больше систем, тем больше нормативно-справочных данных необходимо синхронизировать между ними;
- выполнение различных участков бизнес-процессов в различных системах требует развитой среды надежного обмена транзакционными данными;
- большое количество хранилищ данных и аналитических приложений нуждается в развернутом функционале очистки, загрузки, преобразования данных и так далее.

Различные попытки решить такие проблемы привели к появлению систем, вторичных по отношению к основным бизнес-функциям, ради которых создавались информационные системы. А для обеспечения слаженного взаимодействия всех систем, включая вторичные, были созданы дополнительные системы и методологии. То есть попытки справиться со сложностью привели к еще большему усложнению.

ИС крупного предприятия представляет собой сложную открытую иерархическую многоуровневую развивающуюся систему диффузного типа. Различные уровни описания информационных систем в реальности представляют собой иерархию моделей: моделей бизнес-процессов, приложений, данных, серверов, ролей, процедур, потоков и многого другого.

На сегодняшний день существует несколько подходов к формализованному представлению сложного устройства современных информационных многокомпонентных систем, помогающих управлять бизнесом и принимать решения. Современное крупное предприятие в процессе цифровой трансформации уже не может обойтись без выбора целенаправленного архитектурного подхода к развитию своей информационной системы [13].

Наиболее распространенная технологическая парадигма подобного рода — сервисно ориентированная архитектура (SOA). Ярким примером подхода методологического характера является TOGAF.

Выбор архитектурного подхода, адекватного особенностям конкретного предприятия или отрасли, весьма непрост. Поиск возможностей обеспечения пользователей достоверной информацией сталкивается с проблемой адаптации существующих архитектурных подходов (таких как TOGAF,

Zachmann, Strategic Architecture Model) и приводит к еще большему усложнению вариантов их применения [14].

Итак, современные системы стали слишком сложны, а существующим архитектурным подходам к их построению и развитию недостает научной основы, обеспечивающей связность различных моделей. Все сегодняшние архитектурные подходы созданы западными учеными и инженерами эмпирическим путем в результате попыток представить структуру системы с разных точек зрения (здесь можно вспомнить известную притчу про слона и шестерых слепых мудрецов). При этом использовались достаточно сложные метамодели описания ИС, отражающие многие ее аспекты, но не отражающие в полной мере реальность. Например, в упомянутой парадигме TOGAF на текущий момент не существует реализованной цельной модели архитектуры ИС крупного предприятия.

Мы видим, что современность «заказывает» нам решение трех задач:

- поиск научной основы, которая позволила бы справиться с формальным описанием современных ИС;
- создание архитектурного подхода, решающего проблему сложности и неудобства современных ИС;
- создание (изобретение) инновационной ИС как ответ на вызовы времени, связанные с цифровой трансформацией экономики, переходом общества к цифровому бизнесу.

«Заказ» на изобретение

Поиск научной основы

Первая задача в работе по выполнению «заказа», продиктованного жизнью,— это поиск научной основы, которая помогла бы справиться со сложностью описания (формализации) современных ИС.

В исследованиях по теории систем, в частности в работе Месаровича, Мако, Такахары [15], предложен научный подход к описанию любых сложных систем, названный стратификацией.

Стратифицированная система — это система, которая задается семейством моделей, а страта есть уровень абстрагирования, заданный конкретной моделью. Учитывая, что семейство моделей имеет иерархическую структуру страт, а каждая отдельная страта обладает своим системным свойством (эмержентностью), не сводящимся к сумме свойств ее элементов, мы имеем возможность разработать архитектурный подход,

основанный на научной базе стратификации. Он получил название эмержентной стратификации информационных систем. Данный подход описан в разделе «Научная основа — теория систем» и детально изложен в Приложении 1.

Упрощение ИС для удобства пользователя

Информационные системы организаций и предприятий, хранящие и обрабатывающие деловую информацию, состоят из аппаратного и программного обеспечения.

Мы рассматриваем здесь только программное обеспечение (ПО), «железо» остается за рамками этой книги.

В настоящее время ПО создается так, чтобы пользователь информационной системы мог выполнять формальный, четко структурированный перечень заранее предусмотренных действий по хранению и обработке информации. Умению так действовать нужно специально учиться, то есть требуется «компьютерная грамотность»: человек должен разбираться во множестве различных специализированных программ (компьютерных приложений), уметь быстро работать с клавиатурой и мышью, перемещая различные визуальные элементы на экране компьютера.

Даже самый поверхностный взгляд на количество приложений, систем и устройств, нас окружающих, заставляет задуматься: к чему все это приведет и как с этим справляться?

Увеличивается количество задач, ускоряется темп жизни, умножаются ошибки при вводе и обработке информации. «Человеческий фактор, что поделаешь»,— сетуют компьютерщики на пользователей. Пользователи же пеняют на неудобные программы и интерфейсы, на «глюки» и частые сбои в работе систем, на лишние системные настройки, только запутывающие понимание, на трудности «выуживания» нужной информации из множества модулей и процессов. Получать информацию, необходимую для управления предприятием, становится все более сложно, долго и неудобно.

К тому же жизнь подбрасывает организациям все новые задачи: то законодательство поменяется, то технологию надо улучшить, то новые требования к продукции появились, конечно, нельзя забывать и о конкурентах…

И все эти изменения надо учитывать в информационных системах, которые когда-то были спроектированы под прежние условия,— а перепроектировать их под сегодняшние запросы стоит большого труда, времени и денег. Быстро адаптировать информационные системы к изменяющимся

требованиям окружающей нас динамичной среды, увы, не получается. Это долгий и дорогой процесс.

Мир информационных систем все более усложняется вслед за ростом проблем современной жизни предприятий и организаций.

В общем, трудно и неудобно стало работать с информационными системами. Необходимо исправлять положение.

Таким образом, мы получаем второй «заказ», продиктованный жизнью: надо изобрести информационную систему, которая сможет легко приспосабливаться к быстро меняющимся реалиям современной деловой среды, а пользователю будет работать с ней гораздо проще и удобнее.

Цифровая трансформация бизнеса

Немного про сегодняшний тренд на цифровизацию.

Мировая консалтинговая компания «Гартнер» дает такие определения: «Digitalization is the use of digital technologies to change a business model and provide new revenue and value-producing opportunities; it is the process of moving to a digital business».

Цифровизация — это использование цифровых технологий для изменения бизнес-модели и предоставления новых возможностей получения доходов и создания стоимости; это процесс перехода к цифровому бизнесу.

«Digital business transformation is about doing things differently — creating new business designs by using digital technologies in combination to blur the boundary between the physical and the virtual worlds».

Цифровая трансформация бизнеса — это то, что позволит создавать новые модели бизнеса с использованием цифровых технологий, размывающих границу между физическим и виртуальным мирами.

В чем разница между информатизацией и цифровизацией? Или это только вопрос степени — цифровизация просто расширяет информатизацию? «Гартнер» использует термин «цифровизация», чтобы подчеркнуть то, что цель преобразований — создавать и приносить новую ценность потребителям, а не просто улучшать сделанное в рамках информатизации.

Цифровизация предполагает кардинальную смену парадигмы, переход на «полностью цифровой бизнес», формирование новой экономической реальности. В терминах эмержентной стратификации информационных систем цифровизация начинается с четвертой страты и направлена вверх.

На нижних стратах, с первой по третью, за последние 50 лет проведена «оцифровка», информатизация, *которая помогает человеку справляться с его текущей работой*. Цифровизация же вообще вытесняет человека

из *всех рутинных операций* и процессов как слабое звено (исключает пресловутый «человеческий фактор»), оставляя за человеком только творческие функции и принятие решений.

2 июня 2017 года на пленарном заседании Петербургского международного экономического форума президент РФ Владимир Путин сказал следующее: «Цифровая экономика — это основа, которая позволяет создавать качественно новые модели бизнеса, торговли, логистики, производства, изменяет формат образования, здравоохранения, государственного управления, коммуникаций между людьми, а следовательно, задает новую парадигму развития государства, экономики и всего общества».

И конечно же, цифровой экономике потребуются информационные системы нового поколения, которые будут отвечать этой новой парадигме государственного управления. Это еще один запрос современности.

Проблемное поле

Проблема низкой адаптивности к внешним изменениям

Основные проблемы сегодняшних ИС так или иначе связаны с их «врожденным недостатком»: они обеспечивают выполнение только формального, четко структурированного перечня заранее предусмотренных действий по хранению и обработке информации.

Пользователь не может получить от системы что-либо выходящее за это «прокрустово ложе».

Отсюда *возникает первая проблема*: в связи с постоянно происходящими изменениями в деловой среде предприятия реальные задачи обработки информации (например, учет, хранение и анализ данных о его деятельности) часто выходят за рамки перечня заранее предусмотренных действий.

Назовем это *низкой адаптивностью систем* к изменениям во внешнем мире. Современные ИС спроектированы жестко, их функции и структуры зафиксированы и не обладают должной гибкостью; негативно влияет как количественная сложность систем (они содержат множество объектов, процессов, подсистем, модулей), так и качественная — то есть высокая степень интегрированности всех этих объектов.

Требуются большие затраты денег, труда и времени на адаптацию ИС, так как приходится каждый раз дорабатывать или даже заново проектировать программное обеспечение систем.

Чтобы решить проблему, надо существенно повысить адаптивность ИС к меняющимся условиям функционирования, сделать систему максимально гибкой.

Наиболее близко к решению этой проблемы подошла концепция SOA, сервисно ориентированной архитектуры информационных систем.

Идея, использованная в SOA: для обеспечения большей гибкости ИС из бизнес-процессов предприятия выделяются обособленные самостоятельные повторяющиеся части, между которыми в информационной системе организуется стандартизованное взаимодействие. Подход SOA декларирует, что при изменении предметной области требуется перепроектировать только те части системы, на которые влияют произошедшие изменения, предполагая, что при этом не потребуется изменять всю ИС или ее значительную часть.

Тезисы SOA

Сервисы: SOA представляет необходимую пользователю функциональность в виде наборов «сервисов» — отдельных, самостоятельных, полноценных, но узкоспециализированных, логически обособленных программных модулей. Каждый из них выполняет конкретную повторяющуюся бизнес-функцию из всей совокупности бизнес-процессов предприятия. Сервисы могут быть использованы по своему узкому назначению: могут выполнять данную бизнес-функцию в любом бизнес-процессе.

Сервисные соглашения (интерфейсы). Доступ к сервисам осуществляется через четко определенные интерфейсы, которые являются платформо- и языковонезависимыми и используют открытые стандарты. Сервисы могут быть связаны между собой только посредством этих интерфейсов, и всё, что они знают друг про друга,— только то, что они существуют.

Автономность. Сервисы для внешнего мира выглядят как «черные ящики», то есть они скрывают свою внутреннюю реализацию и предоставляют только интерфейсы для обращений к себе извне (от пользовательских приложений); иными словами, сервисы заранее ничего не знают о пользовательском приложении, которое к ним обратилось, а это приложение не знает, каким образом сервисы выполняют свою бизнес-функцию; SOA предоставляет пользовательским приложениям возможность вызывать все сервисы, имеющиеся в рамках информационной системы, через сервисную шину (Enterprise Service Bus).

Свойства и методы. Сервисы работают по четко определенным методам и обладают четко определенными свойствами; для выполнения своих

задач они могут быть вызваны (инициированы) неким стандартным способом; при этом сервисы не хранят состояния между этими вызовами.

Архитектура. Архитектурно SOA объединяет в себе единый портал для доступа пользователей, композитные приложения, поддерживающие бизнес-процессы, репозиторий сервисов и интеграционную сервисную шину, выполняющую для сервисов функцию общей распределительной сети.

Недостаток SOA

Декларируется, что для обеспечения нужной адаптивности ИС к происходящим изменениям приложение в SOA не должно представлять собой монолитный код, который единожды инсталлируется, не требует и не допускает никаких доработок при изменении требований,— то есть приложение в SOA должно иметь возможность адаптации к изменениям.

Однако в реальности это не так: требованиям адаптивности удовлетворяют только так называемые «композитные приложения», которые можно разработать так, чтобы они не «скреплялись намертво», а менялись в соответствии с переменами в бизнесе, имея возможности для добавления, исключения или модификации отдельных функций.

Такое композитное приложение строится на базе существующих подсистем ИС (например, таких, как модули SAP MM, PM, FI…), как выделенное хранилище данных или общедоступный web-сервис. Они дополняются новой функциональностью и объединяются с помощью единого интерфейса. Декларируется, что композитные приложения обеспечивают пользователям возможность легко взаимодействовать в реальном времени с многочисленными системами и иметь прозрачный доступ к информации независимо от сложности среды ее хранения.

Но в реальности лишь малую часть всех бизнес-процессов можно реализовать композитными приложениями. Дело в том, что:

- во-первых, существующие корпоративные ИС к моменту появления SOA уже были очень сложными;
- во-вторых, традиционные системы типа ERP содержат в себе большой объем заранее настроенных бизнес-процессов, и лишь некоторая часть этих процессов доступна как сервисы;
- в-третьих, даже если автоматизировать предприятие «с нуля», то описать все его бизнес-процессы и реализовать их на SOA будет гораздо дороже, чем внедрить преднастроенное ERP-решение.

Именно поэтому ни одно крупное предприятие не имеет формализованной актуальной модели всех бизнес-процессов, а без этого SOA не может быть реализована в полном виде. Соответственно, в информационных

системах предприятий SOA используется лишь частично, в сочетании с традиционными жестко детерминированными приложениями.

Таким образом, попытка решить проблему адаптивности ИС средствами сервисно ориентированной архитектуры не удалась. Преимущества, потенциально заложенные в SOA, на практике работают лишь частично, и эта архитектура не обеспечивает нужной адаптивности ИС к изменениям в предметной области.

Нужен другой подход к повышению адаптивности систем.

Проблема внутренней ригидности ИС

Вторая проблема состоит в том, что не все возникающие информационные потребности пользователей ИС могут быть удовлетворены в традиционной архитектуре функциональных приложений. Далеко не всегда возможно получить из системы нужную пользователю информацию — например, в случае сложного, нетривиального запроса.

Назовем это проблемой *внутренней ригидности*.

Вот, например, попробуйте получить ответы от ИС предприятия (ERP-системы) на такие вполне обычные вопросы «из жизни»:

- выбрать из картотеки основных средств программные продукты и лицензии с ненулевой остаточной стоимостью и проанализировать изменение амортизации в случае применения экспертного коэффициента ускоренной амортизации 1.5; анализ надо провести с учетом следующих условий:
  - применить результаты за текущий период;
  - выполнить обратный расчет с корректировкой амортизации начиная с даты постановки основного средства на баланс;
- выбрать контрагентов с наличием кредиторской задолженности в объеме более миллиона рублей по договорам, с которыми были случаи нарушения сроков поставки или предоставления документов либо факты ненадлежащего качества продукции;
- рассчитать, сколько необходимо заказать отрезных кругов диаметром 200 мм для покрытия потребности в производстве деталей NN цехом ZZ во втором квартале с учетом прогноза остатков на производственных и центральных складах; а если это будут круги диаметром 100 мм, то учесть при расчетах еще и спецификацию материалов-заменителей.

Это совсем не тривиальные запросы для традиционной ERP-системы, получение ответов на которые требует обработки данных из разных и зачастую не связанных между собой источников,— и поэтому получить

ответ автоматически, «одним нажатием кнопки», невозможно, требуется подключать к выполнению запроса «человеческий фактор», специалистов-аналитиков.

Традиционные пользовательские приложения, обслуживающие ограниченный перечень функциональных задач (например, в области финансов, логистики, снабжения и в других сферах), не дают возможности работать с информацией настолько свободно, насколько это необходимо, и часто не позволяют пользователю получить ответ на сложные запросы, требующие взаимосвязанной информации из различных областей.

Суть проблемы в том, что традиционные приложения в ИС спроектированы так, что сами же и «мешают» сохранять информацию, организовывать ее и гибко извлекать. Они разрабатываются с расчетом на использование фиксированной схемы данных, определяющей типы сохраняемой информации, способы ее отображения и принципы управления. Пользователи, чья информация организована по другой схеме, не могут ее использовать.

Кроме того, фиксированные схемы данных не позволяют соединять информацию из нескольких приложений — например, связать контактные данные человека, указанные в смартфоне, с его авторством книги, представленной на Amazon.com. Поскольку соответствующие приложения «не знают» друг друга, они не могут работать с чужими данными и даже ссылаться на них.

Наиболее близка к решению этой проблемы (повышение гибкости ИС к изменениям информационных потребностей пользователей) концепция, основанная на понятии «семантическая паутина» — Semantic Web (ее наиболее известная реализация для бизнеса — международный стандарт деловой отчетности XBRL) [39].

Семантическая паутина (англ. Semantic Web) — это направление развития Всемирной паутины, целью которого является представление информации в виде, пригодном для машинной обработки.

В обычном Интернете, основанном на HTML-страницах, информация заложена в тексте страниц и извлекается человеком с помощью браузера. Семантическая же паутина предполагает запись информации в виде семантической сети (графа) с помощью онтологий. В этом случае программа-клиент может непосредственно извлекать из паутины факты и делать из них логические заключения.

Термин «семантическая паутина» был впервые введен сэром Тимом Бернерсом-Ли (изобретателем Всемирной паутины) в мае 2001 года в журнале Scientific American, это направление стало новым этапом в развитии

Интернета. Концепция семантической паутины была принята и продвигается Консорциумом Всемирной паутины (https://www.w3.org/).

Семантическая паутина декларирует способность предоставить пользователю большую гибкость, поскольку ее главное обещание — это поддержка «паутины» данных с многообразием схем.

Однако в действительности ситуация с реализацией этого подхода далека от реального решения задачи гибкого управления информацией.

Еще в 2006 году журнал IEEE Intelligent Systems опубликовал статью Тима Бернерса-Ли Semantic Web Revisited («Семантическая паутина: пересмотр»), в которой автор называет описанный им подход к организации информации в вебе «простой идеей, до сих пор в большой степени нереализованной». Наряду с видимыми преимуществами семантической паутины имеются различные препятствия к ее реализации, начиная с человеческого фактора (люди склонны избегать работы по поддержке документов с метаданными, открытыми остаются проблемы их истинности) и заканчивая общефилософскими аспектами. Например, есть сомнение в возможности описания мира некоторым конечным количеством концептов — то есть вряд ли можно разработать онтологию верхнего уровня, критическую для существования семантической паутины. В мире есть бесконечное количество концептов (понятий), из чего следует бесконечность числа классов, в которые их можно объединить.

Семантическая паутина потенциально улучшает возможности управления информацией, но полезных практических реализаций этих возможностей слишком мало. Например, можно назвать реально работающим во многих странах XBRL — формат деловой отчетности, претендующий на международный стандарт обмена бизнес-информацией. В нем реализовано несколько хороших принципов для обеспечения гибкости: отделение концептов от их связей, учет делового контекста, возможность гибкого расширения таксономий и другие. Но это пример специализированного структурирования информации в одной сфере — передача деловой отчетности от предприятий к контрольным органам и регуляторам.

Можно констатировать, что в настоящее время в Semantic Web нет возможности работать с информацией настолько свободно, насколько это необходимо, чтобы пользователь мог получить ответ на сложные запросы, требующие получения взаимосвязанной информации из различных функциональных аспектов деятельности предприятия.

Стоит отметить, что имеются некоторые разработки в области языков моделирования онтологий, направленные на улучшение гибкости систем,

но только в контексте поисковых запросов к массивам данных. Уровень зрелости этих решений пока низкий. Архитектур ИС, основанных на онтологических моделях, пока не существует.

Таким образом, проблема гибкости и адаптивности ИС к разнообразию изменяющихся информационных потребностей пользователей в традиционных архитектурах информационных систем пока не решена.

Проблема комфортности труда при использовании ПО

Следующая проблема связана с неудобством и низкой эргономичностью приложений, точнее — интерфейсов пользователя. Работать с клавиатурой и двигать мышкой по экрану приходится в очень сложной структуре не всегда хорошо продуманных разработчиками графических элементов, поэтому пользователю каждый раз требуется много времени на обучение и привыкание к ним, из-за чего снижается производительность труда, накапливаются раздражение и усталость.

Требуется обеспечить удобство использования ПО (улучшить взаимодействие между пользователем и системой). В идеале пользователь просто должен сказать системе, что он хочет от нее,— и система сама решит, как это сделать лучше.

Проблемы в управлении знаниями

В более широком контексте можно говорить о несовершенстве сложившейся на сегодняшний день парадигме управления знаниями. Ведь, в общем-то, пользователь (сотрудник предприятия) обращается к информационной системе, чтобы получить некоторые знания о конкретном участке деятельности предприятия, чтобы предпринять эффективные действия, но:

- либо не может получить их вообще, поскольку система не в состоянии автоматически собрать нужную информацию из разных источников; приходится привлекать на помощь других пользователей (аналитиков, ИТ-специалистов);
- либо, с трудом продираясь через сложные интерфейсы, получает их в неудобном виде и с большой задержкой.

Проблемы эти связаны с тем, что управление знаниями в большинстве информационных систем рассматривается как обычный процесс, аналогичный другим операциям (например, перемещения материальных ресурсов), и его надо лишь спроектировать и регламентировать, чтобы гарантировать определенный уровень его выходов.

Главные задачи управления знаниями при этом подходе сугубо механистические — надо инвентаризировать знания и распределить их:

- структурировать и классифицировать информацию должным образом;
- направить информационный поток к лицам, ответственным за принятие решений;
- автоматизировать процесс структурирования и передачи информации, а также обеспечить работу основных бизнес-приложений, в которые поступает эта информация.

Такой подход, существующий с восьмидесятых годов прошлого века, безусловно, устарел, доказательством чего является неэффективность «реинжиниринга бизнес-процессов». При создании новых информационных систем необходимо отказаться от взгляда на предприятие как на машину — жизнь оказалась гораздо сложнее.

«Информационная система будущего» как решение этих проблем

Как решить перечисленные выше проблемы, чтобы создать информационные системы нового поколения — гибкие, адаптивные, удобные?

Ведь получается, что для решения этих проблем надо создать такое программное обеспечение, которое:

- может легко и быстро изменяться адекватно произошедшим или предполагаемым изменениям в предметной области, обслуживаемой системой, например в деловой среде предприятия;
- предоставит пользователю возможность выполнять любые неструктурированные и не предусмотренные заранее действия с информацией, формировать любые запросы к системе, просто озвучив команду или давая текстовые задания компьютеру, и получать ответы в реальном времени.

Это весьма необычная задача, сама постановка которой вызывает недоумение: «Разве такое возможно? Это что-то из области фантастики…» Стоит вернуться к первым строкам нашей книги — о воплощении (когда-то «фантастической») идеи видеоразговора собеседников, находящихся на разных континентах, и мы сразу поймем, что речь идет не о фантастике, а о востребованном временем изобретении.

Как создать информационную систему будущего

Методологическая основа: Directed Evolution

- Чтобы создавать новые информационные системы «по заказу» сегодняшнего дня (то есть целенаправленно «изобретать будущее»), надо обратиться к методу управляемой эволюции (Directed Evolution) Б. Л. Злотина [4, 5] (см. Приложение № 2) и к методологии изобретательства и развития технических систем — ТРИЗ Г. С. Альтшуллера [2, 3] (см. Приложение № 3) и применить эти положения к области информационных технологий.

Для создания гибкой, адаптивной, удобной информационной системы нового поколения надо предпринять следующие шаги:

1. Спрогнозировать наилучшие направления развития информационных технологий в сторону повышения гибкости, адаптивности, удобства; для этого надо воспользоваться методом Directed Evolution.
2. Применить в процессе прогнозирования методологический инструментарий ТРИЗ для нахождения соответствующих изобретательских решений в области улучшения адаптивности и гибкости информационных систем (в нашем случае в качестве инструментов были использованы принципы «дробления», «перехода на микроуровень», понятие «идеальности» и закон полноты частей системы).
3. Совместить полученные идеи общего характера со спецификой информационных технологий (в нашем случае к специфике ИТ относятся эмержентная стратификация информационных систем, онтологическое моделирование предметной области, атомизация микросистем и концепция мгновенной адаптируемой онтологически направляемой группы умных бизнес-сервисов — МАНГУСТ).
4. Апробировать полученную инновационную ИТ-концепцию в рамках реально существующей системы, наиболее близкой по идеологии и по функциональности (мы использовали цифровую бизнес-платформу Comindware).

Научная основа: теория систем

Понятие эмержентной стратификации

Создавая концепцию новой информационной системы, мы исходили, как сказано выше, из того, что современные архитектурные подходы к построению ИС стали слишком сложны — они не обеспечивают необходимой адаптивности к изменяющимся условиям деловой среды и гибкости по отношению к потребностям пользователя.

Мы сочли, что основной причиной недостатков существующих архитектур является отсутствие в них опоры на научную концепцию. Ведь существующие архитектурные взгляды сформировались на Западе «явочным порядком»: и матрица Захмана, и TOGAF, и их различные клоны (DoDAF, SAM) были выработаны эмпирическим путем.

А ведь в нашей стране в недалеком прошлом (в советский период) при разработке вычислительных систем всегда опирались на научную основу, на математический аппарат. Во многом это связано с именами академиков АН СССР Л. В. Канторовича, А. Н. Тихонова, В. М. Глушкова и других. То есть впереди разработки всегда шла математика, а уже потом — алгоритмы и программы. Как многие еще помнят, в то время программное обеспечение называлось «математическим обеспечением».

Сейчас комплексной научной основы под архитектурой ИС нет, сложность предмета очень высока: построены сложнейшие информационные метамодели организаций и предприятий, которые в таком виде адекватно функционировать (адаптироваться к изменениям бизнеса) уже не способны. Поэтому в настоящее время не существует архитектуры информационной системы (и предприятия в целом) в полном соответствии, например, с TOGAF.

В поиске нужной научной основы — математического аппарата, на который можно опираться при структурировании сложных систем,— было бы логично обратиться в первую очередь к теории систем. Например, в работе Месаровича, Мако, Такахары [14] предложен достаточно четкий математический и логический подход к структурированному описанию любых сложных иерархических систем, известный как стратификация. В нем введено понятие страты, и любая система описывается иерархией этих страт.

Определение: «Стратифицированная система — это система, которая задается семейством моделей, а страта есть уровень абстрагирования, представленный одной конкретной моделью».

При этом авторы работы [14] не обозначили, чем каждая модель, описывающая одну страту, должна принципиально отличаться от другой модели, описывающей другую страту,— то есть не дано определение базового свойства, которое задавало бы структуру, иерархию стратифицированной системы.

Этот недостаток может быть преодолен посредством использования принципа эмержентности (иначе известного как «системное свойство»). Данный принцип был введен в кибернетике, в работах У. Р. Эшби [41]: «Чем больше система и чем больше различия в размерах между частью

Рисунок 1.

Пять уровней стратификации

и целым, тем выше вероятность того, что свойства целого могут сильно отличаться от свойств частей». Иными словами, система не является простой суммой своих частей.

Таким образом, учитывая, что стратифицированная система, заданная семейством моделей, имеет иерархическую структуру страт и при этом каждая отдельная страта (выраженная конкретной моделью) обладает своим системным свойством (эмержентностью), не сводящимся к сумме свойств ее элементов, мы получаем связный архитектурный подход, основанный на теории систем.

Данный подход получил название эмержентной стратификации ИС [9], [10], и в его основе лежит понятие «эмержентное свойство страты» (далее — ЭСС).

Пять уровней информационной системы

В эмержентной стратификации информационных систем выделено пять страт (см. рисунок 1).

1. На самом нижнем уровне находится страта машинных кодов и аппаратного обеспечения. Это сигналы, команды ассемблера, данные в виде битов и байтов. ЭСС на данном уровне — обработка данных, а потребитель информации — техник (электронщик или системщик).
2. Вторая снизу страта — структурированные данные, то есть информация как осмысленные данные: это информационные модели, описывающие структуру информации (например, ER-модель «сущность — связь»). Информация имеет транзакционный характер, живет в базах данных (БД) и в системах управления БД. ЭСС здесь — обработка информации (загрузка, преобразование, хранение, представление).
3. Третья страта — информационные сервисы, например, в форме программных приложений (software applications), содержащие деловую

логику и обеспечивающие нужную функциональность конечному пользователю сегодняшней ИС. Именно в рамках этой страты пользователи воспринимают информационную систему «своей», они работают в рамках составляющих ее приложений. ЭСС на этой ступени включает выполнение инфосервисов для бизнеса (бизнес-процессы, аналитика, поддержка принятия решений, в том числе через управление знаниями и методы искусственного интеллекта).

4. Четвертая страта — это модель бизнеса в целом, ведение текущей деятельности предприятия/организации. ЭСС здесь носит пока «неайтишный» характер, поскольку выражается в экономических понятиях (прибыль, доходность, доля рынка и т.п.). ИТ сюда только-только начинают проникать из области поддержки принятия решений, стараясь предложить цифровые бизнес-модели. На этой страте начинается «цифровизация», отличающаяся от «информатизации» на предыдущих стратах.

5. На пятой страте находятся модели развития бизнеса, затрагивающие цели существования предприятия. Эта страта — результирующая, отвечающая на вопрос: «Зачем нужны все остальные страты?» ЭСС пятого уровня — это миссия, видение, стратегия, цели. ИТ на этой страте пока совсем не представлены.

Пять уровней информации

Информация — основной продукт информационной системы. Понимание того, что такое информация в современных ИТ, отражено в рамках такой науки, как семиотика,— в ней выработано необходимое и достаточное на данный момент понимание и определение информации.

Семиотика рассматривает информацию в пяти аспектах (уровнях): статистическом, синтаксическом, семантическом, прагматическом, апобетическом [15].

Уровни информации точно соотносятся с пятью уровнями эмержентной стратификации. Каждый уровень информации нашел свое место на одной из пяти страт информационной системы (см. рисунок 2).

Эмержентная стратификация и семиотический подход как научная основа для построения новой архитектуры информационных систем подробно изложены в Приложении 1.

Практическая основа: новая архитектура

Опираясь на методологическую и научную основу создания системы нового поколения, необходимо выразить и практический аспект — архитектуру ИС. Особенности архитектуры информационной системы раскроем, ответив на вопросы:

Рисунок 2.

Пять аспектов информации и пять страт информационной системы

- Чем является каждая страта в архитектурном смысле?
- Какие могут быть архитектурные стили (варианты стратификации)?

Переход с третьей на четвертую страту

Прогресс на уровнях каждой страты дает прогресс в системе в целом.

Рассмотрим переход от третьей страты к четвертой. Прогресс именно на этом уровне нас сейчас интересует постольку, поскольку и мир, и наша страна вступили в процесс цифровизации[2] — цифровой трансформации экономики.

Развитие достаточно зрелой ИС, которая «плотно» освоила третью страту, основано на следующем принципе: для более полной «информатизации» необходимо минимизировать объем связей между нижележащими стратами.

Чем полнее, глубже и четче будет проведена стратификация в рамках каждой страты с первого по третий уровень, чем больше интегрированы элементы и связи в пределах одной страты (единая модель), чем меньше путаница связей между стратами (в том числе в проектах), тем больший эффект от «информатизации» будет получен.

Но «цифровизация», охватывающая четвертую страту (бизнес), требует совсем другого подхода. Здесь ИТ вносят свой вклад в ЭСС — к примеру, предлагая цифровые бизнес-модели, генерирующие гораздо большую прибыль. Цифровая бизнес-модель на четвертой страте должна содержать как минимум структурированную модель (онтологию) предметной области этого бизнеса, которая должна гибко и быстро транслировать изменения, происходящие в бизнес-процессах, реагировать на изменения в информационной системе (ERP), находящейся ниже, на третьем уровне. Для этого

[2] О понятии «цифровизации» и ее отличии от «информатизации» — см. выше в разделе «Цифровая трансформация бизнеса».

Страта	Наименование уровня	Передано	Получено
5	Апобетика, целевой уровень (зачем?)	Предполагаемый целевой результат	Достигнутый целевой результат
4	Прагматика, уровень исполнения (как?)	Ожидаемое действие	Выполненное действие
3	Семантика, уровень передачи смысла (о чем?)	Выраженная мысль	Осмысленное понятие
2	Синтаксис, уровень передачи данных (что?)	Созданный символьный код	Воспринятый символьный код
1	Статистика, сигнальный уровень (есть — нет)	Переданный сигнал	Полученный сигнал

нужно обеспечить множество гибких связей между четвертой и третьей стратами, между цифровой моделью бизнеса и информационной системой.

Исходя из этого, мы выдвинули гипотезу о доминировании двух главных архитектурных стилей: «Монокристалл» и «Атомизация».

Стиль «Монокристалл»

Архитектурный стиль «Монокристалл» сводит все связи в пределах каждой страты воедино. Например:

- внизу, на первой страте — «железо», мощный сервер, объем оперативной памяти;
- на второй страте — единая БД и СУБД;
- на третьей страте — единый заранее определенный набор приложений.

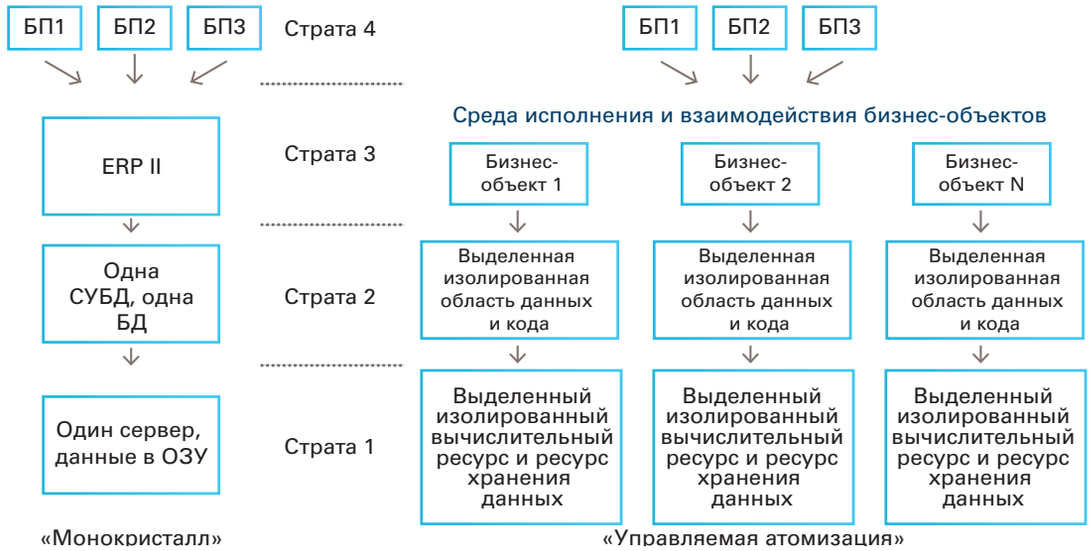
Это соответствует парадигме «системного свертывания», изложенной в [7], [8].

Основная идея свертывания состоит в том, чтобы устранить из системы часть ее элементов (вместе с их вредными функциями и другими уже известными и еще не выявленными недостатками), а их полезные функции распределить среди оставшихся элементов системы или передать их в надсистему. При свертывании системы технические противоречия, порождаемые экстенсивным ростом и усложнением структуры системы, могут разрешаться путем такого перераспределения функций структурных элементов.

Например, новые технологии, такие как обработка данных в оперативной памяти сервера (in-memory computing), позволяют отказаться в системе от нескольких специализированных подсистем в пользу одной универсальной, «свернутой», которая и выполняет все нужные функции.

Рисунок 3.

Два базовых архитектурных стиля



Подробней этот стиль описан во второй главе, в разделе «Архитектурные стили».

Стиль «Атомизация»

Архитектурный стиль «Управляемая атомизация» отвечает современным потребностям цифровизации, охватывает верхний уровень третьей страты (зрелые информационные системы) с переходом на четвертую страту (бизнес).

В этом архитектурном стиле присутствует множество вертикальных связей, которые идут сверху вниз от четвертой к нижележащим стратам.

Ниже (см. раздел «Архитектурный стиль «УБ-Мангуст») показано, как эти связи реализуют управление от бизнеса к информационной системе в форме «умных микросистем», то есть У-Блоков, если использовать новую терминологию.

Подробнее этот стиль описан в Приложении 1 (раздел «Архитектурные стили»).

Системное окружение: цифровая трансформация

Тренд, указанный в разделе «Цифровая трансформация бизнеса», в наше время становится основным драйвером развития информационных систем и технологических платформ. Изобретенной «системе будущего» придется работать в условиях проявления именно этой тенденции, поэтому рассмотрим цифровизацию подробней.

Целеполагание

Цель проведения преобразований в любой предметной области (от предприятия до государства) средствами цифровой трансформации — это достижение нового уровня развития рассматриваемой бизнес-системы в парадигме информационного общества и экономики знаний.

Этот уровень характеризуется формированием «цифрового мира», в котором появляются ранее не существовавшие возможности получения ценностей, недостижимых традиционными способами на предыдущих стадиях развития общества и экономики.

Информационное общество и экономика знаний определяются наличием на этом уровне общественно-экономического развития нового системного (эмержентного) свойства, обусловленного появлением ранее не существовавшего феномена «виртуального отражения реального мира», его цифровой модели. Это свойство обеспечивает получение «ценностей, недостижимых традиционными способами». Появляются новые бизнес-ценности (цифровые продукты и услуги), социальные ценности (признание, статус, самореализация через проявление человека в цифровом мире), общечеловеческие ценности (свобода и благополучие через возможность трудиться онлайн с любого места в удобное время) и другие, при этом резко снижаются барьеры для их получения.

Главное системное свойство «цифрового мира» метафорически можно сформулировать как «феномен волшебной палочки»: человек высказал желание — и сразу получил желаемое. Как верно подметил Артур Кларк в своем Третьем законе, «хорошо развитая технология неотличима от магии» [16].

Действительно, в цифровом мире удовлетворение потребностей человека или получение бизнес-сервиса предприятием происходит мгновенно, в любой географической точке, в любое время (при наличии ресурсов для оплаты).

Таким образом, новое системное свойство «цифрового мира» — это возможность выполнения желания, обеспеченного ресурсами, в любой момент, в любом месте, без задержек и требуемого качества.

При этом практическая польза от формирования «цифрового мира» проявляется в двух основных направлениях:

1. Системное свойство обуславливает появление новых ценностей, недостижимых ранее (в бизнесе, в социуме, в отношениях между людьми).
2. В процессе цифровых преобразований происходит увеличение эффективности бизнеса, недостижимое ранее (уменьшаются вредные

Рисунок 4.

Стиль «Управляемая атомизация»

факторы: затраты, издержки, проблемы; увеличиваются полезные факторы: время реакции, гибкость, адаптивность, производительность).

Общее понимание цифровизации экономики

Представим наше понимание цифровизации в сфере экономики с учетом изложенных аспектов.

Как сказано выше, цифровизация предполагает кардинальную смену парадигмы, переход на «полностью цифровой бизнес», формирование новой экономической реальности.

Цифровизация экономики подразумевает в первую очередь создание «цифрового продукта», ядра новой экономической парадигмы. Цифровой продукт создается, обрабатывается и распространяется цифровым образом.

В наше время в сфере цифровизации можно обнаружить как вполне традиционные, «нецифровые» организации, которые используют цифровой продукт на входе, так и полностью цифровизированные фирмы, использующие в том числе «интернет вещей».

Например, цифровой проектный институт с помощью специализированного ПО разрабатывает, просчитывает и моделирует объект и процедуру строительства, выдает в качестве продукта цифровизированный проект вместо бумажного. Виртуальный цифровой образ объекта уже построен.

Традиционная строительная компания его распечатывает или хранит на компьютере, но дальше все делает как обычно, по традиционной технологии, регулярно сверяясь с цифровым проектом; по нему же осуществляются архитектурный надзор и приемка.

А вот цифровизированная строительная компания загружает этот проект в исполнительские устройства, и они реально выполняют этот проект от начала до завершения. Цифровые бульдозеры выходят на объект и подготавливают строительную площадку, сразу разравнивая грунт по плану, без топографов, цифровые экскаваторы изготавливают идеальный котлован под фундамент, грунт вывозят цифровые самосвалы, роботы устанавливают блочную опалубку и арматуру. Причем это отнюдь не «образ будущего» — цифровые строительные машины, оснащенные электронно-цифровой системой управления и использующие цифровую модель результата, уже работают на российских стройках.

Таким образом, «цифровизированный продукт» на предыдущей, «доцифровой» стадии заменяется полностью цифровизированным процессом. Например, как в случае со строительной компанией, все бизнес-процессы регулируются электронно-цифровой системой управления, основанной на множестве датчиков «интернета вещей».

А следующий уровень цифровизации — изменение самой бизнес-модели. Когда в «бесцифровом» виде бизнес-процесс уже не может быть осуществлен, цифровизацию можно считать завершенной, потому что цифровизированный процесс невозможно отделить от бизнес-модели, от живой ткани бизнеса.

Еще один пример цифровизированного бизнес-процесса: компания получает оплату онлайн с каждого конечного потребителя (как юридического, так и физического лица) на основании информации от множества распределенных по стране приборов цифрового учета электроэнергии, автоматически передающих сведения в платежную систему на основе блокчейна и «умных контрактов», которая автоматически производит все финансовые транзакции без участия бухгалтерии предприятия и потребителя электроэнергии.

Конечно, информационная система такой компании должна быть очень гибкой, быстро и легко адаптирующейся к любым возможным изменениям.

Основные понятия цифровизации

В данном разделе приведены определения, выработанные практикой цифровой трансформации, на примере понятий «цифровой повестки» Евразийского экономического союза (ЕАЭС) [17], дополненных нашим видением.

Информация

Информация в контексте мироустройства — одно из базовых понятий Вселенной наряду с материей и энергией [15].

Информация в контексте ноосферы — виртуальное (нематериальное) отражение реального мира человеческой цивилизации, где каждый элемент (личность, объект, понятие, явление) разных аспектов ноосферы (техносфера, социосфера и др.) имеет своего информационного двойника, формируя «инфосферу» [31], [32]; информация инициируется только разумным началом — волевым или творческим актом человека; поскольку информация нематериальна, то она для перехода от передатчика к приемнику должна быть выражена в материальной (цифровой) форме на физических носителях — для обработки ее информационными системами.

Информация в контексте информационного общества; информация как субъект является носителем субстанциальных свойств и характеристик, определяющих качественные особенности информационного общества как объекта; иными словами, информация в цифровой форме, понимаемая и компьютерами, и людьми, в информационном обществе становится основным фактором, который влияет на развитие общества и в процессе развития формирует «цифровой мир».

Информация в контексте задач цифровой экономики является субъектом преобразований в странах, основным фактором цифровой трансформации экономики; информация служит основой для достижения уровня развития страны, соответствующего характеристикам информационного общества и экономики знаний; при этом формируется «цифровой мир», в котором с использованием цифровых моделей деятельности (бизнес-моделей) появляются возможности получения ценностей, недостижимых традиционными способами.

Информация в контексте информационных технологий (ИТ) является основным продуктом информационных систем (ИС), представленным в различных аспектах:

- по семиотическим уровням;
- по уровням эмержентной стратификации ИС;
- по целям управления самой информацией, в том числе в онтологических моделях, представляющих структуру понятий и связей в информации; в эпистемологических моделях, представляющих строение, структуру, функционирование и развитие знания как высшей формы информации.

Онтология

Форма представления информации о реальном мире, формализации некоторой предметной области знаний или ее части посредством схемы концептуализации, содержащей структуру релевантных классов

бизнес-объектов (концептов онтологии), их связи и правила (аксиомы, ограничения), принятые в этой области. Основные сферы применения — моделирование бизнес-процессов, семантическая паутина (Semantic Web), искусственный интеллект, цифровые платформы предприятий и организаций.

Информационное общество

Сегодняшняя стадия развития цивилизации, где основным продуктом экономических отношений становятся информация и ее высшая форма — знания. Информационное общество приходит на смену постиндустриальному обществу и характеризуется переходом экономических и общественных отношений в цифровую форму (см. «Цифровая трансформация»).

Цифровое преобразование

Комплекс действий по формированию и включению в онтологию (см. «Онтология») цифрового двойника материального объекта или субъекта.

Цифровая экономика

Экономическая деятельность, основанная на цифровых процессах, моделях, технологиях, цифровых товарах (сервисах), в том числе производимых электронным бизнесом.

Цифровая трансформация экономики

1. Смена экономического уклада, изменение традиционных рынков, социальных отношений, государственного управления, связанная с проникновением в них цифровых технологий.
2. Принципиальное изменение основного источника добавленной стоимости и структуры экономики за счет формирования более эффективных экономических процессов, обеспеченных цифровой инфраструктурой.
3. Переход функции лидирующего механизма развития экономики к институтам, основанным на цифровых моделях и процессах.
4. Составная часть и первая стадия цифрового развития общества (см. «Цифровое развитие»), характеризующаяся появлением нового системного (эмержентного) свойства (см. раздел «Новое системное свойство: результат цифровой трансформации»); эта стадия имеет среднесрочный горизонт прогнозирования и планирования (до 2025 года) на переходный период с целью создания необходимых и достаточных условий для формирования информационного общества.

Цифровое развитие

Вторая (основная) стадия цифровизации, следующая за первой стадией — цифровой трансформацией экономики; в отличие от первой

стадии, имеющей среднесрочный горизонт, цифровое развитие имеет долгосрочный горизонт с целью формирования информационного общества и экономики знаний/ценностей; на этой стадии завершается переход от парадигмы «цифровой трансформации» к парадигме «целенаправленного цифрового развития»; при этом проводится развертывание, масштабирование полученного на первой стадии системного свойства (см. раздел «Новое системное свойство: результат цифровой трансформации») по отраслям экономики, рынкам, странам; стадия опирается на принципы и механизмы целенаправленного развития, в том числе механизмы синтеза цифровых инноваций, встроенные в процессы функционирования информационного общества; индикатор зрелости стадии — появление цифровой модели общества, «виртуального отражения реального мира».

Цифровая инфраструктура

1. Инфраструктурный комплекс, обеспечивающий протекание процессов на основе цифровых технологий.
2. Комплекс технологий и построенных на их основе цифровых продуктов, обеспечивающих вычислительные, телекоммуникационные и сетевые мощности и работающих на цифровой основе.

Цифровое пространство

Совокупность социальных механизмов, деловых отношений и общих рынков, использующих цифровые технологии и цифровую инфраструктуру, формирующих и эксплуатирующих цифровые активы.

Цифровой актив

1. Систематизированный, индексированный контент (цифровые фотографии, анимация, видео, музыка и другое), доступный для применения.
2. Инкапсулированная в сети (Интернете или в иной) функциональность.
3. Специфическая форма собственности и ресурсов, инвестиции в которые повышают капитализацию физического актива и обеспечивают рост денежного потока.
4. Совокупность информации в цифровой форме (совокупность цифровых продуктов) о физическом или виртуальном объекте, процессе, субъекте деятельности, физическом лице, которая представляет ценность и может быть использована для извлечения добавленной стоимости.
5. Комплекс цифровых продуктов и инфраструктуры, процесс использования и изменения которых приводит к формированию добавленной стоимости и новой ценности, в том числе выраженной в денежной форме.

Цифровой продукт (услуга)

1. Продукт (услуга), производимый и/или предоставляемый в цифровом пространстве.
2. Одно из свойств продукта (услуги), возникающее при осуществлении цифровых процессов с образом продукта (услуги).
3. Ценная информация или доступ к электронному сервису, за который покупатели согласны платить деньги.

Цифровой образ

Совокупность характеристик субъекта или объекта реального мира, сущности в цифровой форме, однозначно характеризующая субъект или объект и его состояние.

Цифровой рынок

1. Совокупность экономических отношений, базирующихся на регулярных обменных операциях в электронном виде между производителями товаров (услуг) и потребителями.
2. Рынок данных и неструктурированной информации, создаваемый в цифровом пространстве.

Принципы цифровизации

Фокус на бизнес, а не на информатизацию

В терминах эмержентной стратификации информационных систем цифровизация начинается с четвертой страты, основывается на третьей и нижележащих стратах, но постепенно «свертывает», поглощает и интегрирует их в себе. Развитие направлено вверх, к пятой страте.

Информатизация — это «оцифровка» до третьей страты включительно, а цифровизация начинается с четвертой страты, с бизнеса.

На нижних стратах, с первой по третью, за последние полвека проведена «оцифровка», то есть информатизация, которая обеспечивает поддержку пользователя в текущей работе.

Цифровизация же вообще исключает «человеческий фактор», оставляя за человеком только творческие функции и принятие решений.

При этом одним из основных активов бизнеса становится информационная платформа, обеспечивающая управление и развитие этих процессов на основе виртуальной цифровой модели деятельности предприятия и его экосистемы.

Глобальная управляемая инновация

Цифровизация по своей природе — это глобальная управляемая инновация. Она, конечно, как всякая инновация, стимулируется и направляется выраженной

потребностью в изменениях к лучшему, но, чтобы сделать цифровизацию эффективной, необходимы целеполагание и методология в такой специфической области, как innovation on demand — «инновации по требованию».

В процессе цифровой трансформации экономики необходимо вырабатывать множество инновационных идей и решений, которые должны превратиться в новые бизнес-модели, цифровые продукты и услуги; и этот процесс должен быть систематическим, управляемым. Для этого прямо предназначены современные методы синтеза инноваций и целенаправленного прогнозирования, созданные на основе законов развития технических систем, методов инженерного творчества и сильного мышления [3], [4], [5], [6], [7], [8].

Один из них — *целенаправленное прогнозирование (Directed Evolution*).

Хочешь заглянуть в будущее — изобрети его. То есть прежде, чем начинать цифровизацию, надо провести прогноз развития (Directed Evolution) как бизнеса, так и составляющих его технологий — и получить «образ будущего».

Перевести емкое выражение «Directed Evolution» на русский язык можно как «управляемое инновациями развитие», «целенаправленное системное развертывание», «управляемое развитие информационных и бизнес-систем». В фокусе цифровизации этот метод может звучать как «направляемое инновациями цифровое развитие», или «инновационно ориентированная цифровая трансформация» (см. Приложение 2).

Внутри проекта Directed Evolution проводится целенаправленная генерация бизнес-идей, технологических идей, идей цифровизации, нужных для реализации этого «образа будущего». Иными словами, ведется целенаправленный *синтез цифровых инноваций*.

Эта работа осуществляется с использованием метода, рожденного в СССР и проверенного десятилетиями развития в России и за рубежом (см. Приложение 3).

С учетом этих понятий можно понимать цифровизацию как управляемую инновациями цифровую трансформацию (Digital Transformation Directed by Innovations — *DT/DI*).

Опора на концепцию «Индустрия 4.0»

Концепция «Индустрия 4.0» основана на кибермеханических системах, которые сами настраиваются на выполнение новых задач, сами себя обслуживают, анализируют и сами изменяют технологический процесс в зависимости от запросов, которые предъявляет пользователь.

Примером могут служить системы «цифровая фабрика» и «умная фабрика».

«Цифровая фабрика» — это перевод всех бумажных процессов, в том числе моделирования изделий, в виртуальное пространство. Например, 3D-принтеры, с помощью которых на заводах можно не вытачивать, а распечатывать детали. Добавленная стоимость перейдет от производителей к тем, кто владеет правом интеллектуальной собственности на чертежи, потому что их можно скачивать из онлайн-магазина.

«Умная фабрика» — это следующий этап после «цифровой фабрики». Например, детали распечатали на 3D-принтере, и надо, чтобы из них максимально быстро и эффективно роботы собрали автомобиль.

В идеале виртуальный завод будущего должен выглядеть так: вы заказываете товар в Интернете, а потом его печатают на ближайшем к вашему дому принтере.

Можно привести примеры из различных отраслей бизнеса. Например, ситуация из индустрии общепита: цифровая диет-сестра (инфоробот) разрабатывает цифровое меню, дальше оно либо изготавливается роботами-мультиповарами, либо поступает как технологическая карта в обычную кухню.

Цифровые модели бизнеса

Каждый объект реального мира предприятия (и его делового окружения) будет иметь своего информационного двойника в мире виртуальном, то есть будет иметь ту или иную цифровую модель.

В настоящее время это статичные геофизические модели, фиксированные информационные ER-модели, модели бизнес-процессов.

Завтра потребуются адаптивные онтологические модели предметных областей, модели адаптивного кейс-менеджмента, адаптивные производственные и динамические геофизические модели на основе «интернета вещей», гибкие нейросетевые модели и многие другие.

Для управления потребуются цифровая модель предприятия, а также «белые воротнички» — масса инженеров, программистов, аналитиков, онтологистов, которые будут с ней работать. Потребуются умные устройства (не просто датчики, но инфороботы, инфоагенты), которые смогут обмениваться нужной информацией друг с другом и с цифровой моделью.

Связь модели виртуального процесса с реальным (физическим), протекающим на производстве, обеспечивается бесшовно посредством «умных изделий» — например, с помощью получаемой от них информации о режимах работы, сбоях и замене устройств. Если в процессе участвует человек, то он также оснащается набором «умных устройств», фиксирующих его действия и результат этих действий. Некоторые части процессов уходят из физической реализации. Например, симулируя показания

«умных устройств», можно создать виртуальный объект, запрограммировать контроллеры как виртуальные объекты, подключить симулятор этого физического процесса и тестировать процесс «от и до», не выполнив ничего «в металле»,— отсюда возможность отладки процесса в виртуальном образце, а значит, экономия на монтаже и запуске в эксплуатацию.

Так, уже сегодня широко распространены системы виртуальных тренажеров, в которых эмулируется виртуальная производственная система, и персонал отрабатывает на виртуальных рабочих местах те действия, которые нужно предпринять в реальности.

Цифровые платформы

При цифровой трансформации происходит переход от парадигмы «бизнес как система» к более гибкой парадигме «бизнес как платформа».

В широком понимании платформа — это коммуникационная и транзакционная среда, участники которой извлекают выгоду из процесса взаимодействия друг с другом.

Вот некоторые определения цифровых платформ, используемые при цифровой трансформации.

1. Модель деятельности (в том числе бизнес-деятельности) заинтересованных лиц на общей платформе для функционирования на цифровых рынках.
2. Площадка, поддерживающая комплекс автоматизированных процессов и модельное потребление цифровых продуктов (услуг) значительным количеством потребителей.
3. Информационная система, ставшая одним из лидирующих решений в своей технологической нише по цифровизации (транзакционной, интеграционной).

Например, в рамках цифровой трансформации, проводимой Евразийским экономическим союзом (ЕАЭС), дается такое определение: «Цифровая платформа, реализующая доступ заинтересованных лиц к цифровым активам ЕАЭС, государственным и сертифицированным частным цифровым услугам в рамках цифрового пространства ЕАЭС, обеспечивающая функционирование отраслевых цифровых платформ, интеграцию с цифровыми платформами других стран и интеграционных образований, построенная на основе единой архитектуры (модели) ЕАЭС» [17].

Платформы эволюционируют далее до понятия «бизнес и ИТ как живой организм».

Архитектура или холархия?

Понятие «архитектура» выражает устаревший взгляд на ИС как на здание или сооружение — оно стало слишком жестким, неспособным раскрыть

такие свойства информационных систем, как гибкость и адаптивность, которые необходимы при цифровой трансформации.

Правильней ввести понятие «системная холархия» (см. главу 2, раздел «Системная холархия»). Понятие холархии изложено в [18].

Информация принадлежит всем слоям (уровням) холархии. Каждый уровень определяется наличием базового холона и обладает собственным системным свойством, то есть базовые холоны и системные свойства у каждого уровня холархии свои.

Цифровизация и ИТ-стратегия

ИТ-стратегия отражает состояние ИТ-систем компании на третьей страте и служит инфраструктурной основой, которую нужно учитывать, инициируя процесс цифровизации на четвертой страте.

Новые системы, создаваемые в парадигме цифровизации на четвертой страте, могут быть построены поверх старых систем третьей страты в режиме «выделить, отделить, изолировать» без нарушения течения бизнес-процессов.

Люди и культура — ключевой фактор успеха цифровой трансформации

В конечном итоге успешность трансформации будет зависеть не от специальных датчиков, алгоритмов или инструментов аналитики, а от более широкого набора факторов, связанных с человеческим ресурсом бизнеса.

Промышленным компаниям необходимо обеспечить условия для повышения цифровой культуры персонала, роста уровня заинтересованности в успешной трансформации со стороны административного состава, привлечения на предприятие специалистов цифровых технологий, обучения сотрудников, которые намного продуктивнее будут работать в динамичной цифровой экосистеме.

Вытеснение низкоквалифицированного персонала

Процесс цифровой трансформации на предприятии приведет к увеличению рабочих мест для специалистов, работающих с цифровыми технологиями (инженеров, программистов, аналитиков), для представителей новых профессий онтологистов, моделеров, и к сокращению низкоквалифицированного персонала, не востребованного вследствие автоматизации бизнеса (примером автоматизации могут служить безлюдные технологии для управления автомобилями, дроны, которые доставляют посылки, полностью роботизированное производство). Данная ситуация провоцирует социальную напряженность на предприятии, в обществе. Поэтому руководству компании в процессе его цифровой трансформации необходимо

учитывать социальный аспект и обеспечить условия для повышения квалификации специалистов и рабочего персонала.

Развитие отношений с клиентами через цифровые каналы

Все изменения в цепочке создания стоимости, продуктов и услуг будут клиентоориентированными, то есть направленными на точное соответствие индивидуальным потребностям клиента.

Нулевой барьер для пользователя: You Get What You Asked

Один из важнейших принципов цифровизации — удобство для пользователя. Необходимо обеспечить пользователю «нулевой барьер» для овладения цифровыми технологиями, устройствами и системами.

Цифровые сервисы должны иметь очень простые для пользователя интерфейсы по принципу «сказал, что хочу — мгновенно получил желаемое». Пользователь не должен думать о том, в каком именно функциональном приложении он должен работать (например, в системах SAP это BO, FI, CO, HR, SCM, CRM и подобные).

Пользователь в правильно построенном «цифровом мире» должен получить способ удобно, быстро и эффективно удовлетворять разнообразные информационные потребности, выполнять свои служебные обязанности и рабочие операции без необходимости изучать функции многообразных приложений и овладевать навыками работы в них. Он просто должен сказать системе: «Хочу получить вот это»,— и быстро получить адекватный ответ.

Поэтому очень важно еще раз акцентировать внимание на том, что уже было сказано выше, в разделе «Информационная система будущего как решение этих проблем»:

- Пользовательский интерфейс в цифровой платформе нового поколения должен иметь голосовой ввод или простое окно для ввода свободного текста, а не сложную систему навигации по разнообразным пунктам меню.
- Система должна обеспечить пользователю возможность выполнять любые неструктурированные и не предусмотренные заранее действия с информацией, включая формирование любых запросов к системе для получения от нее ответов в реальном времени — просто формулируя задачу на обычном языке или давая письменные задания компьютеру обычным текстом.

Такой подход назван нами You Get What You Asked (YGWYA) — «Игвия».

«Игвия» реализуется в предложенной нами новой системной архитектуре — онтологической цифровой платформе.

Новая системная архитектура для цифрового мира

Цифровая трансформация подразумевает создание и развитие «информационного виртуального мира», являющего собой полное цифровое представление всей цепочки создания стоимости (экономической ценности) и отражающего все необходимые аспекты основных составляющих экономики.

Этот «информационный виртуальный мир» технологически реализуется в форме некоторой цифровой платформы, обеспечивающей гибкое функционирование информационного пространства во взаимодействии с реальным миром.

Такая цифровая платформа может включать:

1. Информационную систему для функционирования «информационного виртуального мира» реальной экономической деятельности (программное обеспечение).
2. ИТ-инфраструктуру этого «информационного виртуального мира» (вычислительная техника и сети передачи данных).
3. Цифровую модель (комплекс бизнес-моделей) экономической деятельности.

На базе такой цифровой платформы с использованием единой проектной методологии строятся и развиваются предметно-ориентированные (функционально-специализированные) компоненты информационного пространства предприятия (организации, территории, государства, межгосударственных объединений).

Для этого необходимо в рамках определенной программы цифровой трансформации создать цифровую платформу, на которой будут строиться и развиваться компоненты (системы, проекты) «информационного виртуального мира», и обеспечить следующее:

1. Универсальность такой платформы для построения на ее базе любых необходимых функционально ориентированных информационных систем для нужд предприятия (организации, территории, государства, межгосударственных объединений).
2. Адаптивность цифровой платформы и развивающихся на ее базе информационных систем к постоянным изменениям в предметных областях.
3. Координацию разработки цифровой платформы с разработками специализированных (функциональных) информационных систем в рамках различных проектов программы цифровой трансформации.

Архитектура цифровой платформы, которая будет поддерживать цифровую трансформацию экономики, позволит управлять ее текущим функционированием, адаптироваться к изменяющимся условиям, развиваться (эволюционировать). Она должна быть построена в стиле «управляемой атомизации» (см. Приложение 1, раздел «Архитектурные стили»).

Вывод

В первой главе мы актуализировали потребность бизнеса в новой информационной системе, отвечающей условиям перехода на новый технологический уровень (обладающей свойствами универсальности для построения любых необходимых функционально ориентированных информационных систем и адаптивности к постоянным изменениям в предметных областях), которая станет основой для развития информационного пространства предприятия. Данный запрос требует изобретения «информационной системы будущего».

Для работы в условиях цифровой трансформации экономики, для повышения гибкости и адаптивности информационных систем, в частности для возможности выполнения «любых не предусмотренных заранее» действий с информацией, необходимо отразить в проектируемой системе следующее:

- общую онтологию предметной области информатизации, причем с возможностью быстрого, простого, многократного внесения в нее изменений;
- возможность работы с семантикой (смыслом) запросов пользователя в заданном контексте и во взаимосвязи с онтологией;
- способность в ответ на любое обращение пользователя к ИС найти в онтологии нужные концепты и связи, мгновенно создать из них нужный набор, отвечающий условиям запроса, и выполнить ряд операций, приводящих к результату (ответу);
- программный код, обрабатывающий данные запроса и реализующий нужные операции;
- интерфейс, принимающий запросы голосом и представляющий пользователю полученный от системы результат;
- механизм, который будет всем этим управлять.

То есть изобретаемый продукт, реализуемый в форме цифровой платформы, должен соответствовать названным выше требованиям. Давайте посмотрим, что из перечисленного удалось нам осуществить. Рассмотрим все аспекты нашего изобретения во второй главе.

Глава 2.
Продукт изобретения — онтологическая цифровая платформа

Вводная часть

Название

Цифровая платформа предприятия/организации построена в архитектурном стиле «управляемой атомизации» на основе «атомарных микросистем» (Smart-Brick, У-Блок) и «мгновенных инфосервисов» («Мангуст») под управлением семейства онтологических моделей.

Описание атомарных микросистем «У-Блок» приведено в разделе «Умный блок».

Описание мгновенных инфоспектаклей «Мангуст» (мгновенная адаптируемая онтологически направляемая группа умных бизнес-сервисов) дается в разделе «Понятие «Мангуст».

С учетом этого принято сокращенное название: онтологическая цифровая платформа «УБ-Мангуст».

Патент

Подана патентная заявка на изобретение:
U.S. Patent Application
Application No.: 15273333
Filed: September 22, 2016
For: SMART BRICKS INFORMATION SYSTEM

Область применения

Цифровая платформа «УБ-Мангуст» используется при построении и развитии информационных систем управления организациями/предприятиями, включая, но не ограничиваясь такими системами, как:

- учетные транзакционные системы (OLTP/ERP II);
- аналитические системы (BI, OLAP);
- системы поддержки принятия управленческих решений (DSS);
- системы технологического управления производством (SCADA).

С нашей точки зрения, «УБ-Мангуст» способна внести существенный вклад в цифровую трансформацию экономики. Как известно, цифровизация подразумевает создание и развитие «информационных виртуальных миров», цифровых моделей и цифровых платформ самого разного масштаба — от предприятия до государства (такого, как Россия) и надгосударственного объединения (такого, как ЕАЭС).

По нашему мнению, архитектура «УБ-Мангуст» сможет обеспечивать полное цифровое представление всей цепочки создания стоимости (экономической ценности) и отражать все необходимые аспекты основных составляющих экономики. Она сможет поддерживать гибкое (адаптивное) функционирование цифровой модели любого масштаба во взаимодействии с реальным миром.

Вот только некоторые примеры возможного использования «УБ-Мангуст» при цифровой трансформации экономики:

- «умный дом»;
- «умный город»;
- «адаптивная цифровая платформа для построения прикладных ИС (бизнес-конструктор)»;
- адаптивное обучение по индивидуальной траектории с использованием элементов микрообучения (microlearning);
- автоконструирование бизнес-процессов и информационной системы предприятия на основе адаптивного кейс-менеджмента (Adaptive Case Management).

Базовые понятия

У-Блок, Smart-Brick

Понятие Smart-Brick, использованное при патентовании данного изобретения,— это атомарная микросистема в составе макросистемы — цифровой платформы предприятия.

Это понятие, кроме его прямого значения «Умный блок», можно перевести также как «умница, славный парень, молодчина». Такая трактовка

английского Smart-Brick как нельзя лучше передает смысл и назначение базовой идеи новой цифровой платформы: по-умному и быстро отвечать на любые запросы пользователя, ловко и разумно приспосабливаться к изменениям ситуации — так, чтобы захотелось похвалить!

Для удобства далее используется сокращение «У-Блок» («Умный блок»).

У-Блок содержит минимально необходимые и достаточные системные компоненты для выполнения полного цикла получения, обработки и представления информации, а именно: данные, программный код, интерфейс и средства управления ими (метаданные).

«Мангуст»

Понятие «Мангуст» — это нечто вроде «специальной группы инфосервисов быстрого реагирования». Она мгновенно реагирует на запрос пользователя и исчезает.

Сама аббревиатура «МАНГУСТ» получилась от выражения «Мгновенно Адаптируемая и Направляемая Группа Умных Сервисов, Трансформируемая онтологически».

Другими словами, это собираемая «на лету» из У-Блоков группа «умных» информационных сервисов, выполняющая запрос пользователя под управлением онтологической модели «здесь и сейчас».

Метафорически «Мангуст» можно определить также как «мгновенное шоу», «мини-спектакль», «флэшмоб» с участием «артистов» — различных У-Блоков, быстро собранных для выполнения запроса пользователя.

«Мангуст» создается только для нужной операции (запроса, действия, обработки события) под управлением онтологической модели в ответ на запрос пользователя. Это «временная сборка по требованию», которая выполняет одноразовый сервис для предоставления одного ответа на один запрос.

«Мангуст» выполняет свою работу, а затем распадается обратно на составляющие его У-Блоки.

В случае если такой сервис оказывается регулярным, требует неоднократного повторения, то этот «Мангуст» остается «в сборе» для последующего использования. Далее мы покажем, что зафиксированные таким образом «Мангусты» могут выполнять функции приложений, кейсов, бизнес-процессов и инфороботов.

Онтология

Онтология (в информатике) [19] используется как метод всеобъемлющей и детальной формализации некоторой области знаний с помощью концептуальной схемы, содержащей иерархическую структуру из всех

релевантных классов объектов, их связей, правил и ограничений, принятых в предметной области.

Онтологии — это формально представленные знания на базе концептуализации, то есть описания множества объектов и понятий, знаний о них и связей между ними.

Онтологии в форме баз знаний могут быть прочитаны и поняты пользователем, могут отчуждаться от разработчика и/или физически разделяться пользователями, в том числе для совместной работы над какой-либо предметной областью.

Формально онтология определяется как

O = <X, R, F>, где:

X — конечное множество понятий (концептов) предметной области;

R — конечное множество отношений между понятиями;

F — конечное множество функций интерпретации, заданных на концептах и/или отношениях.

При R=0 и F=0 онтология представляет собой простой словарь. Пример — индексы поисковых машин.

Модель онтологической системы

Z = <O, P, M>, где:

O — онтология верхнего уровня (метаонтология), которая содержит общие понятия и отношения, не зависящие от предметной области: «объект», «свойство», «значение» и другие;

P — множество предметных онтологий и онтологий задач предметной области (с учетом предпочтений пользователя);

M — модель машины вывода данной онтологической системы (например, для изменения критериев релевантности поиска или критериев формирования репозитория).

Модель онтологической системы позволяет описывать взаимосвязь онтологий разных уровней.

Онтологическая модель предприятия

Онтологические понятия, указанные выше, используются в архитектуре «УБ-Мангуст». Для того чтобы микросистемы (У-Блоки) сами могли осмысленно собираться в мгновенный информационный сервис («Мангуст»), было использовано понятие онтологической модели (ОМ).

ОМ содержит строгое описание системы понятий, объектов, сущностей, отношений и всего другого, что составляет предметную область информатизации; иными словами, это виртуальное отражение реального мира предприятия с необходимой и достаточной степенью детали-

зации/упрощения; модель содержит описание структур, функций, взаимосвязей, свойств и характеристик — словом, всего внутреннего содержания предметной области информатизации.

В ОМ содержится семантический аспект, который обеспечивает соответствие бизнес-объектов понятиям естественного языка, отражая таким образом все смысловое многообразие предметной области (бизнеса предприятия) как на естественном языке (для пользователя), так и в формализованных нотациях (для ИС).

Фактически ОМ образует виртуальную модель предприятия. Это информационное отражение, виртуальный двойник реального мира предприятия. Здесь каждому материальному объекту реального предприятия соответствует его информационный виртуальный двойник. Каждая деталь, прибор, инструмент, станок, материал, изделие, а также и сотрудник — словом, буквально все, из чего состоит предприятие как бизнес-система для выпуска продукции или предоставления услуг, отражается в этой виртуальной составляющей.

Онтологическая модель поддерживает целостность данных и обеспечивает их качество при изменении внешних и внутренних условий работы системы.

Подробней см. ниже раздел «Управление» макроуровня».

Архитектурный стиль «УБ-Мангуст»

Архитектурный стиль «управляемой атомизации», в котором построена цифровая платформа «УБ-Мангуст», связывает в единое целое указанные понятия:

- атомарная микросистема У-Блок;
- «группа сервисов быстрого реагирования» «Мангуст»;
- адаптивная онтологическая модель.

Эту взаимосвязь можно увидеть на рисунке 5.

Более подробно см. Приложение 1 (раздел «Атомизация», рисунок 34).

Эти понятия вместе обеспечивают принципиальную возможность кардинального повышения адаптивности и гибкости информационных систем.

Соответственно, архитектура цифровой платформы «УБ-Мангуст», построенной в стиле «управляемой атомизации», существенно отличается от традиционной архитектуры ИС. Стратификация «УБ-Мангуст» также отличается от стратификации традиционных информационных систем. См. ниже в разделе «Отличия традиционной ИС и цифровой платформы «УБ-Мангуст».

Бимодальное управление

Для выполнения двуединой задачи повышения адаптивности и гибкости информационных систем в «УБ-Мангуст» предусмотрена бимодальная

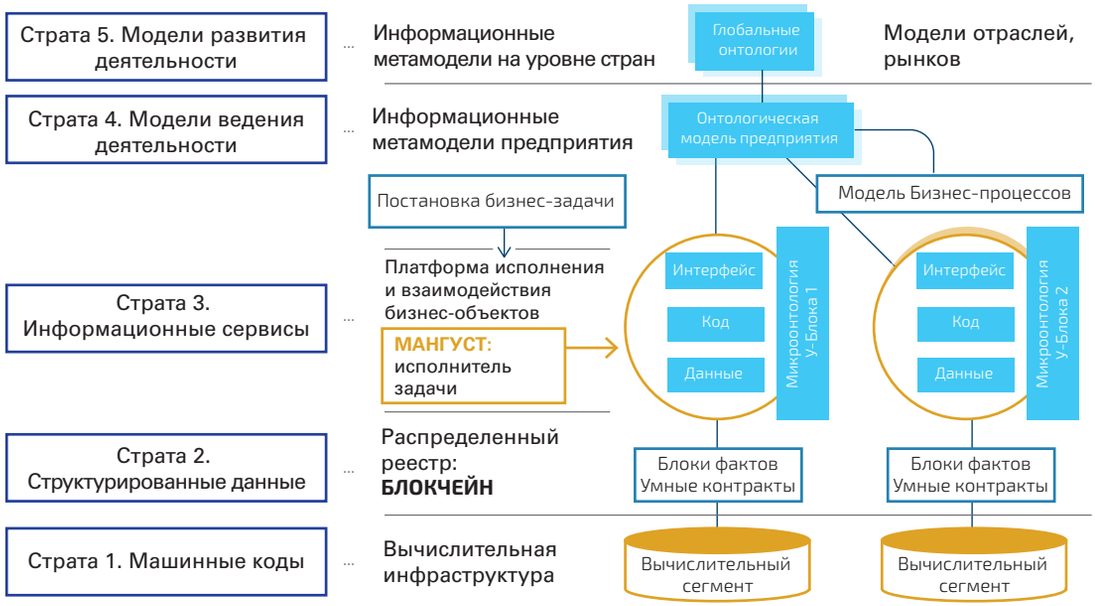


Рисунок 5.

Архитектурный стиль «управляемой атомизации»

система управления Run & Change, где Run относится непосредственно к функционированию, а Change отвечает за модернизацию и адаптацию к изменениям.

Эксплуатационный режим Run нужен для обеспечения гибкости ИС при решении задач управления предприятием; в этом режиме идет текущая работа пользователей с системой.

Режим развития Change нужен для обеспечения адаптивности ИС к изменившимся внешним условиям; в этом режиме изменяется онтологическая модель предметной области: «виртуальная модель» предприятия подстраивается под изменившуюся реальность, и адекватно изменившимся условиям внешней среды формируется новое состояние информационной системы, поддерживающее новые условия «эксплуатационного режима». Например, можно развертывать систему «с нуля» с учетом местных особенностей в разных городах или странах.

Основной процесс

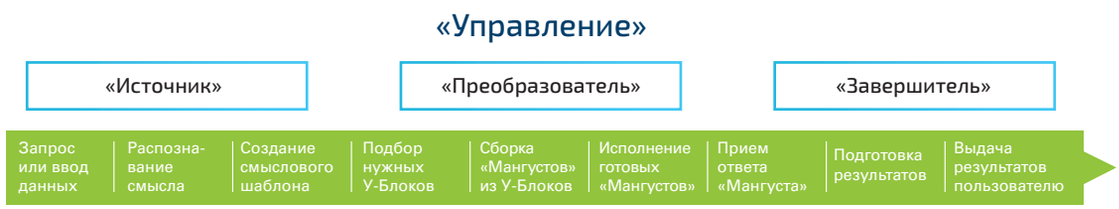
Основной процесс «УБ-Мангуст» подразумевает выполнение полного цикла операций от приема команды, запроса пользователя до представления ему результатов, ответа системы.

Основной процесс состоит из таких операций, как:

- прием запроса пользователя текстом или голосом;
- определение семантики данных (смысла запроса);
- выбор режима работы платформы (эксплуатация или развитие);

Рисунок 6.

Основной процесс



- выбор нужного семантического шаблона или запрос на создание нового;
- подбор необходимых атомарных У-Блоков;
- сборка «Мангуста»;
- выполнение «Мангуста»;
- получение готовых данных для ответа на запрос;
- формирование ответа пользователю в машиночитаемой форме;
- представление результатов для пользователя в удобной ему форме (графика, видео, аудио, текст, голограмма, распечатка на 3D-принтере).

Укрупненно основной процесс изображен на рисунке 6. Как видно из рисунка, процесс выполняется в структуре основных функциональных компонентов — «источник», «преобразователь» и «завершитель», построенной по принципу подобия по уровням иерархии (системной холархии, см. следующий раздел).

Системная холархия

Читатель может более подробно ознакомиться с понятиями системной холархии и эмержентной стратификации информационных систем, обратившись к Приложению 1.

Если кратко, то холархия — это системная иерархия, в которой каждый иерархический уровень характеризуется системным свойством (эмержентностью), а носителем этого свойства является холон — базовый компонент данного уровня. Холон — это целостность, которая есть часть других целостностей [18]. Например, целый атом — это часть целой молекулы; целая молекула — часть целой клетки; целая клетка — часть целого организма. Термин «холархия» происходит от слова «whole» (англ. «целый, завершенный»).

Каждый уровень холархии охватывает, превосходит и включает в себя нижележащий уровень, то есть является надсистемой для него. В свою очередь, этот нижележащий уровень охватывает, превосходит и включает в себя следующий уровень ниже.

Следуя этому подходу, мы говорим об архитектурной холархии стратифицированной информационной системы, где на каждой страте имеется

Рисунок 7.

Стратифицированная информационная система сегодня

свой базовый холон, который включает в себя и охватывает холон нижележащей страты, одновременно являясь включенным и охваченным в холон вышележащей страты.

Например, в этих терминах холон второй страты — это «структурированная информация».

Отличия традиционной ИС и цифровой платформы «УБ-Мангуст»

В архитектурной холархии «УБ-Мангуст» холон «пользовательское приложение» на третьей страте не существует, поскольку его назначение выполняют «Мангусты» (объединения У-Блоков), управляемые основной онтологической моделью предприятия и микроонтологическими моделями в составе У-Блоков.

Отличия хорошо видны, если сравнить два рисунка (7 и 8).

Обычная стратифицированная ИС представлена на рисунке 7.

Цифровая платформа «УБ-Мангуст» существенно отличается от сегодняшних информационных систем — см. рисунок 8.

Три уровня платформы с четырьмя компонентами на каждом

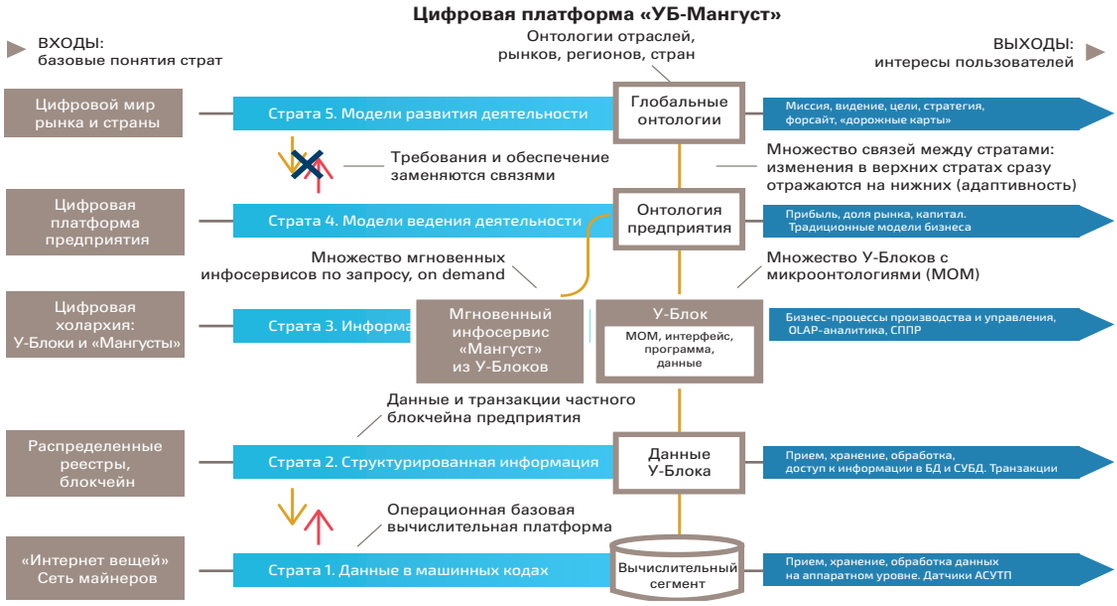
Далее мы рассмотрим три измененные по сравнению с традиционными страты «УБ-Мангуст»:

- страта 2 «Структурированная информация»;
- страта 3 «Информационные сервисы»;
- страта 4 «Модели ведения деятельности».

При построении архитектурной холархии на этих стратах необходимо учесть закон полноты частей системы, заложенный в «образ будущего»

Рисунок 8.

Стратифицированная цифровая платформа «УБ-Мангуст»



«УБ-Мангуст» (см. раздел «Закон полноты частей системы» в Приложении 2), который постулирует обязательное наличие в системе четырех функциональных частей, обобщенно называемых «источник», «преобразователь», «завершитель» и «управление».

Поэтому все уровни «УБ-Мангуст» построены по принципу подобия и каждый из них содержит четыре указанных функциональных компонента.

Таким образом, в стиле «управляемой атомизации», с учетом закона полноты частей системы, на трех измененных стратах выделяются три уровня цифровой платформы «УБ-Мангуст»:

- Макроуровень — это вся цифровая платформа целиком, содержащая макрокомпоненты «источник», «преобразователь», «завершитель» и «управление».
- Средний уровень — декомпозиция указанных выше макроуровневых компонентов на модули; каждый модуль на этом уровне также выполняет функцию или «источника», или «преобразователя», или «завершителя», или «управления».
- Микроуровень — множество атомарных микросистем (У-Блоков), каждая из которых также содержит свой «источник», «преобразователь», «завершитель» и «управление»; У-Блоки умеют объединяться в «Мангусты».

Макроуровень охватывает, превосходит и включает в себя средний уровень, то есть является надсистемой для него. В свою очередь, средний уровень охватывает, превосходит и включает в себя микроуровень, являясь для него надсистемой.

Таблица 1.

Изменения на стратах

Страта	Элементы ИС
Страта 1. «Данные в машинных кодах» Обработка и хранение данных аппаратного уровня	**Элементы** Физические серверы, системы хранения, сетевая инфраструктура. *Остаются прежними.* **Информация** Сигналы, биты, байты, коды ассемблера. *Остаются прежними.*
Страта 2. «Структурированная информация» Прием, хранение, обработка, доступ к информации в БД и СУБД	**Элементы** Серверы управления базами данных, серверы приложений, базы данных. **Информация** В составе баз данных — ER-модели, структура таблиц, запросы, «технические» роли. Данные имеют транзакционный характер. *Изменяются. Данные хранятся на микроуровне, в компонентах «Источник» каждого У-Блока (например, в технологии блокчейн). Управление данными ведется через OOM и MOM, их целостность поддерживается онтологией.*
Страта 3. «Информационные сервисы» Бизнес-процессы производства и управления, бизнес-аналитика, поддержка принятия решений	**Элементы** Пользовательские приложения и прикладные системы ERP. Пример SAP — CRM, SRM, аналитические системы и хранилища — BI, BW, системы обеспечения и интеграции — MDM, порталы, компоненты SOA, системы электронного документооборота. Базы знаний. **Информация** Бизнес-объекты, исполняемые модели бизнес-процессов, функциональные модули, функциональные роли, роли пользователей. *Изменяются кардинально. Функциональных прикладных систем и приложений в традиционном понимании не существует. Все задачи пользователя решаются через сборку мгновенных инфосервисов — «Мангустов».*
Страта 4. «Модели ведения деятельности» Прибыль, доля рынка, капитал и другие. Бизнес-модели	**Элементы** Бизнес-модели (модели ведения текущего бизнеса), бизнес-роли и функции, показатели бизнеса (прибыль, доля рынка, капитал и другие). **Информация** Частично обеспечивается из стратегических аспектов ППР и баз знаний. *Изменяется по принципам цифровой трансформации. Базы знаний превращаются в OOM и отражаются в MOM, они становятся органичной частью цифровой модели бизнеса.*
Страта 5. «Модели развития деятельности» Миссия, видение, стратегия, цели, «дорожные карты» развития	**Элементы** Набор стратегий (и стратагем), включающих дерево целей, миссию, видение, стратегические планы. Корпоративная культура. Корпоративные ценности. **Информация** *Изменяется незначительно — пока только вводится понятие глобальных онтологий (отрасли, продукты-услуги и так далее).*

Рисунок 9.

Три уровня платформы «УБ-Мангуст»

То есть архитектурная холархия «УБ-Мангуст» выстраивается в двух измерениях — по стратам и по уровням платформы, когда каждая из трех страт включает элементы из трех уровней платформы.

- Страта 2 «Структурированная информация» включает элементы микроуровня — компонент «данные» У-Блока.
- Страта 3 «Информационные сервисы» включает элементы микроуровня (У-Блоки и «Мангусты») целиком, компоненты макроуровня и среднего уровня «источник», «преобразователь» «завершитель», обрабатывающие У-Блоки и «Мангусты» в операциях основного процесса.
- Страта 4 «Модели ведения деятельности» включает элемент макроуровня «управление», ООМ.

На рисунке 9 изображена схема трех указанных уровней цифровой платформы (без указания страт, чтобы не усложнять рисунок).

На макроуровне изображены:

1. Компонент приема входных данных, «источник» макроуровня.
2. Компонент, формирующий «Мангусты», «преобразователь» макроуровня.
3. Компонент, представляющий ответ системы пользователю, «завершитель» макроуровня.

4. Основная онтологическая модель предприятия, ООМ — компонент «управление» макроуровня.

Средний уровень в качестве примера иллюстрирован декомпозицией компонента «источник» макроуровня:

2.1. Модуль ввода данных, «источник» среднего уровня.

2.2. Модуль семантического разбора данных, «преобразователь» среднего уровня.

2.3. Модуль генерации шаблонов «Мангустов», «завершитель» среднего уровня.

На микроуровне изображены У-Блоки и «Мангусты»:

5. У-Блок, состоящий из:

5.1. Данные («источник» микроуровня).

5.2. Программный код («преобразователь» микроуровня).

5.3. Интерфейс («завершитель» микроуровня).

5.4. Микроонтологическая модель, МОМ («управление» микроуровня).

6. «Мангуст» (простой пример из двух У-Блоков).

Дополнительно в позиции 7 указаны связи основной онтологической модели с МОМами У-Блоков.

Холархия стратифицированной цифровой платформы

Теперь сведем все изложенное выше в общую таблицу, включая такие аспекты, как:

- назначение информации по уровням семиотики;
- страты ИС, базовые характеристики информации;
- уровни холархии цифровой платформы «УБ-Мангуст»;
- базовый холон уровня холархии;
- описание холона и управление холоном и так далее.

Полученная картина архитектурной холархии цифровой онтологической платформы «УБ-Мангуст» в общем виде изложена в таблице 2.

Компоненты макроуровня

Макроуровень в целом

Верхний уровень платформы «УБ-Мангуст» имеет следующую функционально-компонентную структуру:

1. «Источник», Source — содержит исходные данные и механизм их ввода.

Таблица 2.

Холархия, общее представление

Назначение информации по уровням семиотики	Страты ИС, базовые характеристики информации	Уровни холархии цифровой платформы «Мангуст»	
5. Апобетика, целевое назначение информации	5-я страта. Модели развития предприятия	5. Уровень цифровой платформы предприятия, режим развития	
4. Прагматика, влияние информации на деятельность	4-я страта. Модели ведения деятельности предприятия (текущий бизнес)	4. Уровень цифровой платформы предприятия, режим эксплуатации	
3. Семантика, передача информации с деловым смыслом	3-я страта. Информационные сервисы	3. Уровень инфосервисов	
2. Синтаксис, передача структурированных данных	2-я страта. Структурированные данные	2. Уровень транзакций (блокчейн)	
1. Статистика, передача простых сигналов и инструкций	1-я страта. Машинные коды	1. Уровень сигналов и машинных кодов	

Базовый холон уровня	Описание холона	Управление холоном
Платформа «УБ-Мангуст» в режиме развития «яйцо»	Платформа умеет создавать новые платформы, расширяющие бизнес (то есть воспроизводить себя из «яйца» с иными характеристиками); создавать новые продукты и услуги, поддерживать их и развивать	Онтологическая модель «яйца» и связь с глобальными онтологиями (рынки, отрасли, продукты...)
Платформа «УБ-Мангуст» в режиме эксплуатации	Платформа умеет отвечать на любые запросы юзера или создавать гибкие «умные приложения», адаптивные кейсы, бизнес-процессы или инфороботов (Smart Application Set, Adaptive Case, Business Process, InfoRobot)	Основная онтологическая модель (ООМ) предприятия: концепты и связи, мета-информация предметной области; описывает объекты реального мира (предметной области) с их атрибутами и взаимосвязями
«Мангуст» On Demand	Временные «Мангусты» по вызову умеют выполнять любые запросы юзера и потом распадаются на составляющие их У-Блоки	Умный временный контракт (УмКо) и его связь с ООМ
«Мангуст» Smart Application Set, SmAppS	«Мангусты» типа SmAppS умеют выполнять взаимосвязанные комплексы повторяющихся функциональных задач юзера, как обычное функциональное приложение, но которое может гибко изменяться, изменяя состав и связи между входящими в него У-Блоками	УмКо по шаблону SmAppS и связь с отведенной ему частью ООМ
«Мангуст» Adaptive Case, A-Case	«Мангусты» типа A-Case содержат знание о некоторой части предметной области, умеют заключать между собой умные контракты на предмет этого знания и выполнять эти контракты в рамках кейса в Adaptive Case Management, нотация CMMN. В парадигме ACM они содержат компоненты кейса (business process, document/report, plan/schedule, etc.)	УмКо по шаблону Adaptive Case и связь с отведенной ему частью ООМ
«Мангуст» Business Process	«Мангусты» типа Business Process выполняют сколь угодно длинные последовательности операций в рамках бизнес-процесса в нотации BPMN	УмКо по шаблону Business Process и связь с отведенной ему частью ООМ
«Мангуст» инфоробот	Инфороботы умеют выполнять наборы типовых операций, совершать транзакции (цепочки транзакций), выполнять ограниченно интеллектуальную деятельность, то есть неинтерактивно генерируют ответственный результат вместо человека	УмКо по шаблону InfoRobot и связь с отведенной ему частью ООМ
У-Блок	У-Блоки содержат микрознания о некоторой части предметной области, процедуры обработки этих знаний и умеют заключать между собой умные контракты на предмет этого знания, образуя «Мангуст». Каждый У-Блок имеет связь с нижележащим уровнем (транзакционные данные) и вышележащим уровнем (ООМ)	МОМ, микроонтологическая модель в составе У-Блока, часть ООМ, относящаяся к компетенции конкретного У-Блока
Данные У-Блока		
Сигналы, машинные коды	Обработка и хранение данных аппаратного уровня	Программа на ассемблере

2. «Преобразователь», Transformer — получает данные от «источника» и команды/запросы к системе из блока управления и преобразует их в «Мангуст».
3. «Завершитель», Accomplisher — получает информацию от «Мангуста», превращает ее в желаемую форму и передает результат (информационный продукт, являющийся ответом системы) пользователю; тем самым компонент выполняет или завершает основную функцию «УБ-Мангуст».
4. «Блок управления», Control Unit — управляет вышеуказанными компонентами; это онтологическая модель, ООМ, которая содержит все концепты (объекты) и связи предметной области.

«Источник» макроуровня

Назначение компонента «источник» — прием исходных данных и подготовка этих данных для сборки или выполнения «Мангуста» на следующем шаге в компоненте «преобразователь».

Входные данные могут быть голосовым запросом пользователя, массивом транзакционных данных (таких, как данные из ERP-систем), потоком данных от датчиков или событий (таких, как данные из SCADA/АСУТП или от подсистем Complex Event Processing).

Важно понимать, что все данные, обрабатываемые в «УБ-Мангуст», существуют только на микроуровне, в составе У-Блоков. Данные находятся в каждом У-Блоке, в компоненте «источник». Возможна реализация этого подхода по технологии распределенного реестра, блокчейн.

Компонент состоит из трех модулей:

Входной модуль, который содержит, в дополнение к механизму получения входных данных, механизм распознавания естественного языка. Механизм ввода данных обеспечивает ввод пользователем с клавиатуры или голосом либо принимает машиночитаемые данные от электронных устройств и систем. В качестве опции вместо данных возможно разместить запрос (ссылку), чтобы получить данные из внешних источников.

Модуль распознавания семантики, который идентифицирует семантику (смысл) входных данных и сравнивает значение информации, содержащееся в этих данных, с семантическими единицами, присутствующими в основной онтологической модели предприятия (ООМ).

Модуль генерации шаблонов, который создает «шаблонную матрицу» из выбранных семантических единиц ООМ; этот шаблон необходим, чтобы собрать и выполнить «Мангуст» на следующем шаге.

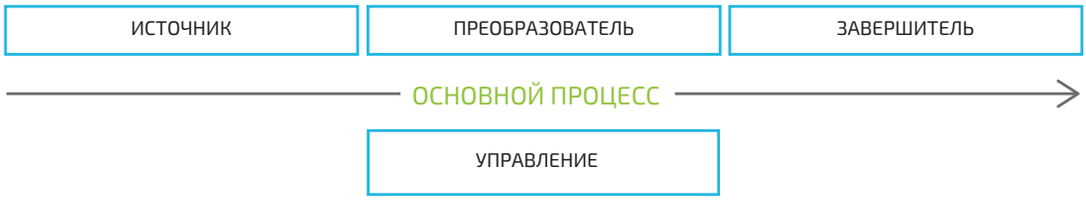


Рисунок 10.

Макроуровень

«Преобразователь» макроуровня

Компонент «преобразователь» содержит основной механизм описываемого изобретения, который обеспечивает гибкость и адаптивность цифровой платформы. Основное назначение этого компонента — формирование «Мангустов» из набора У-Блоков под управлением ООМ.

Здесь можно привести аналогию с процессом синтеза молекул белка из нуклеотидов при помощи шаблона ДНК.

В этом и состоит реализация идеи изобретения: чтобы обеспечить адаптивность и гибкость системы, надо разделить все информационное пространство на множество атомарных микросистем, У-Блоков (информационных атомов), которые по запросу пользователя умеют объединяться в «Мангусты» (информационные молекулы) и тем самым выполнять запрос пользователя или задачу какого-либо иного управляющего элемента системы.

«Преобразователь» получает данные и запросы от «источника», команды от «управления» и преобразовывает их в «Мангуст». Другими словами, этот компонент преобразовывает входные данные в значимую информацию — ответ от системы.

Он работает под управлением метаинформации, содержащейся в основной онтологической модели, компонент «управление» (см. описание ООМ ниже).

Компонент «Преобразователь» состоит из трех модулей:

- *модуля поиска У-Блоков*, который ищет подходящие к запросу У-Блоки, содержащие значения согласно семантическому шаблону, который был ранее сгенерирован модулем генерации шаблонов. Модуль поиска анализирует входные данные в «источнике», подготовленные на предыдущем шаге, распознает семантику этих данных (смысл входных команд и запросов пользователя) и готовит значимые данные для их обработки «Мангустом»;
- *модуля сборки «Мангустов»*, который собирает необходимые для ответа на запрос «Мангусты» из нескольких У-Блоков, содержащих требуемые значения информации, используя семантический шаблон и управляющую информацию от ООМ;

- *модуля исполнения «Мангустов»*, который их запускает и выполняет, отправляя затем результат в компонент «завершитель».

«Завершитель» макроуровня

Основная цель этого компонента состоит в том, чтобы завершить основную функцию «УБ-Мангуст» и передать результат (информационный продукт, готовый ответ) пользователю.

Компонент «завершитель» получает от «преобразователя» (из синтезированного экземпляра «Мангуста») информацию, содержащую ответ системы в формальной нотации (например, в XML). «Завершитель» преобразовывает его в законченную форму, удобную для восприятия пользователем (текст, речь, звук, графика или 3D-изображение), и передает его на периферийное устройство (экран, динамики, принтер, голографический проектор).

Компонент состоит из модулей «Приемник данных «Мангуста», «Синтезатор пользовательских данных», «Предъявитель результата» и «Управление» как сегмент ООМ.

«Управление» макроуровня

Указанными выше компонентами управляет основная онтологическая модель, ООМ — это компонент «управление».

ООМ содержит строгое описание понятий (концептов) предметной области, деловой среды предприятия. ООМ описывает совокупный набор объектов, структур, функций, отношений, правил, свойств, особенностей и всего остального, что формирует информационное пространство предприятия. Понятие онтологии дано выше, в разделе «Базовые понятия».

ООМ содержит семантические элементы, которые описывают все концепты предметной области с их атрибутами и отношениями, и обеспечивает соответствие между этими концептами и понятиями естественного языка. Таким образом, отражается все разнообразие семантики деловой среды предприятия как на естественном языке (для пользователя), так и в формальной машиночитаемой нотации (для системы).

Другими словами, ООМ — это виртуальное представление реального мира предприятия или его части с необходимой и достаточной степенью детализации. В ООМ каждый реальный материальный объект предприятия соответствует своему информационному виртуальному двойнику.

Каждая деталь, устройство, инструмент, машина, материал, продукт, сотрудник — практически все элементы, составляющее предприятие как бизнес-систему,— отражены в этом виртуальном компоненте цифровой платформы.

Каждый материальный объект имеет датчик и идентификатор «интернета вещей», который контролирует позицию объекта в технологическом цикле, цифровая платформа получает информацию с таких датчиков из различных источников, и в виртуальной цифровой модели предприятия постоянно отражаются (обновляются) наличие и статус технологических объектов.

В составе цифровой платформы реализуются онтологические модели разных типов.

Метаонтология содержит общие понятия и отношения, не зависящие от предметной области («объект», «свойство», «значение» и так далее). Этот тип глобальных онтологий используется на страте 5.

Предметная онтология включает понятия, описывающие конкретную предметную область и отношения, семантически значимые для данной предметной области. Этот тип онтологий используется в основной онтологической модели предприятия (предметной области) на четвертой страте «Модели ведения деятельности».

Онтология задач и методов рассматривает в качестве понятий типы решаемых задач, а отношения специфицируют декомпозицию задач на подзадачи. Используется на среднем уровне платформы, в составе компонента «управление» этого уровня, на третьей страте «Информационные сервисы».

Прикладная онтология (онтология приложения) описывает концепты, зависящие как от предметной области, так и от задач. Используется на микроуровне платформы на третьей страте «Информационные сервисы» и в микроонтологических моделях У-Блоков (компонент «управление» У-Блока) на второй страте «Структурированная информация».

Сетевая онтология используется для описания конечных результатов действий, выполняемых объектами предметной области или задачи. Применяется в компоненте «завершитель» макроуровня и управляет тремя его модулями, так как является компонентом «управление».

Таким образом, через онтологии разных типов, на разных уровнях системной холархии обеспечивается принцип гибкого управления компонентами цифровой платформы.

Компоненты среднего уровня

Средний уровень в целом

Средний уровень платформы представляет собой декомпозицию компонентов макроуровня и выглядит так (см. рисунок 11).

На рисунке видно, что макроуровень платформы декомпозирован до среднего уровня следующим образом:

1. «Источник» (Source) макроуровня является блоком постановки задачи и содержит:
   1.1. Модуль приема задачи («источник»).
   1.2. Модуль семантического разбора («преобразователь»).
   1.3. Модуль генерации шаблонов («завершитель»).
   1.4. Онтологическую модель для «источника» на среднем уровне («управление»).
2. «Преобразователь» (Transformer) макроуровня является блоком выполнения задачи и содержит:
   2.1. Модуль поиска У-Блоков («источник»).
   2.2. Модуль сборки «Мангустов» («преобразователь»).
   2.3. Модуль исполнения «Мангустов» («завершитель»).
   2.4. Онтологическую модель для «преобразователя» на среднем уровне («управление»).
3. «Завершитель» (Accomplisher) макроуровня является блоком представления результата и содержит:
   3.1. Модуль получения готовых данных из «Мангуста» («источник»).
   3.2. Модуль подготовки результатов для пользователя («преобразователь»).
   3.3. Модуль представления результатов пользователю («завершитель»).
   3.4. Онтологическую модель для «завершителя» на среднем уровне («управление»).
4. «Управление» (Control Unit) макроуровня управляет онтологическими моделями 1.4, 2.4, 3.4 «источника», «преобразователя» и «завершителя» на среднем уровне.

Онтологические модели, управляющие компонентами среднего уровня, далее по тексту называются ОМСУ.

1. Блок постановки задачи

Этот блок отвечает за получение задачи (команды, запроса) от пользователя и подготовку ее для обработки системой.

Блок состоит из четырех модулей:

1.1. Модуль приема задачи.
1.2. Модуль семантического разбора.
1.3. Модуль генерации шаблонов.
1.4. ОМСУ постановки задачи.

Рисунок 11.

Средний уровень

1.1. Модуль приема задачи

Входным элементом блока постановки задач является модуль приема задачи. Этот модуль выполняет следующие функции:

- принимает задачу пользователя, введенную текстом или голосом;
- преобразует текст или голос в форму машиночитаемых данных;
- передает эти данные для последующей обработки в модуль семантического разбора.

1.2. Модуль семантического разбора

Модуль семантического разбора выполняет следующие функции:

- принимает данные от модуля приема задачи в свой обработчик естественного языка (Natural Language Processor) для подготовки данных к семантическому анализу;
- проводит семантический анализ данных, выявляя смысл задачи во взаимодействии с семантическими элементами, содержащимися в ООМ;
- по результатам анализа запускает один из двух базовых режимов работы цифровой платформы — «Эксплуатация» или «Развитие»:
  - если модуль семантического разбора находит в ООМ все нужные концепты онтологии (то есть текущая версия ООМ содержит все элементы, необходимые для выполнения задачи), то фиксируется, что задача принадлежит к режиму «Эксплуатация»; передает этот факт в модуль генерации шаблонов;
  - если текущая версия ООМ не содержит всех элементов, необходимых для выполнения задачи, то фиксируется, что задача принадлежит к режиму «Развитие», и запускается соответствующая процедура (см. раздел «Два режима: бимодальное управление»).

1.3. Модуль генерации шаблонов

Модуль генерации шаблонов выполняет следующие функции:

- создание «семантического шаблона» на базе концептов онтологии (бизнес-объектов) ООМ, выявленных модулем семантического разбора;

- подготовка шаблона к работе в модулях блока выполнения задачи (компонент «преобразователь»).

1.4. ОМСУ постановки задачи

Локальная предметная область, охватываемая блоком постановки задачи, описывается соответствующей онтологической моделью среднего уровня, сегментом ООМ компонента «источник» макроуровня.

2. Блок выполнения задачи

Этот блок отвечает за исполнение задачи пользователя путем формирования из нужных У-Блоков и последующего выполнения «Мангустов», «умных инфосервисов быстрого реагирования».

Блок состоит из четырех модулей:

2.1. Модуль поиска У-Блоков.

2.2. Модуль сборки «Мангустов».

2.3. Модуль исполнения «Мангустов».

2.4. ОМСУ выполнения задачи.

2.1. Модуль поиска У-Блоков

Модуль поиска У-Блоков выполняет следующие функции:

- принимает семантический шаблон из модуля генерации шаблонов в составе блока постановки задачи;
- ведет поиск/подбор У-Блоков, содержащих нужные значения в соответствии с семантическим шаблоном. Для этого модуль:
  - обращается в ООМ с данными из семантического шаблона;
  - находит (или не находит) в ООМ соответствующие этому шаблону У-Блоки, подходящие для сборки «Мангуста» на следующем шаге;
  - если подходящие У-Блоки не находятся, то запускается режим «Развитие» по соответствующей процедуре (см. раздел «Два режима: бимодальное управление»).

2.2. Модуль сборки «Мангустов»

Модуль сборки «Мангустов» выполняет следующие функции:

- на основе полученного шаблона определяет, достаточно ли единственного У-Блока для выполнения задачи или требуется собрать нужный «Мангуст» из нескольких У-Блоков;
- обращается в ООМ за необходимыми дополнительными сведениями, такими как:
  - контекст выполнения задачи;
  - связи между У-Блоками (отношения подчиненности, взаимодействие и другие);

- бизнес-процесс или адаптивный кейс в рамках задачи и так далее;
- в случае возможности выполнения задачи единственным У-Блоком выбирает этот У-Блок из множества У-Блоков микроуровня;
- в случае необходимости привлечь несколько различных У-Блоков формирует из них «Мангуст», необходимый и достаточный для выполнения задачи пользователя.

2.3. Модуль исполнения «Мангустов»

Модуль исполнения «Мангустов» выполняет следующие функции:

- принимает сформированный «Мангуст» из модуля сборки «Мангустов»;
- запускает в «Мангусте» процедуру выполнения задачи пользователя, при этом:
  - извлекает из всех вошедших в состав «Мангуста» У-Блоков нужные для задачи данные, содержащиеся в «источниках» каждого У-Блока;
  - обрабатывает их программными кодами, содержащимися в «преобразователях» каждого У-Блока;
  - формирует получившийся ответ в компонентах «завершитель» каждого У-Блока под управлением ОМСУ выполнения задачи;
- интегрирует полученный из различных У-Блоков ответ системы (машиночитаемый «отчет «Мангуста») в качестве предварительного результата для последующей обработки в блоке представления результатов.

2.4. ОМСУ выполнения задачи

Локальная предметная область, охватываемая блоком выполнения задачи, описывается соответствующей онтологической моделью среднего уровня, сегментом ООМ компонента «преобразователь» макроуровня.

3. Блок представления результата

Этот блок — результирующий, он отвечает за завершение основного процесса и выдачу «информационного продукта» системы, то есть представляет пользователю ответ на его задачу.

Блок состоит из четырех модулей:

3.1. Модуль получения «отчета «Мангуста».

3.2. Модуль подготовки результата.

3.3. Модуль представления результата.

3.4. ОМСУ представления результата.

3.1. Модуль получения «отчета «Мангуста»

Модуль получения отчета «Мангуста» выполняет следующую функцию:

- принимает отчет «Мангуста» в формальной нотации (например, в XML) на обработку в рамках блока представления результата, переданный из компонента «завершитель» блока выполнения задачи.

3.2. Модуль подготовки результата

Модуль подготовки результата выполняет следующую функцию:

- преобразовывает экземпляр отчета «Мангуста» в формальной нотации к форме, удобной для восприятия пользователем (текст, речь, звук, графика или 3D-изображение — голограмма);
- создает электронный документ, информационную модель бизнес-объекта в принятой нотации или иной результат для представления его пользователю.

3.3. Модуль представления результата

Модуль представления результата передает подготовленный результат на периферийные устройства (экран, динамики, принтер, обычный или голографический проектор).

3.4. ОМСУ представления результата

Локальная предметная область, охватываемая блоком представления результата, описывается соответствующей онтологической моделью среднего уровня, сегментом ООМ компонента «завершитель» макроуровня.

Компоненты микроуровня

Микроуровень в целом

Микроуровень цифровой платформы состоит из множества атомарных микросистем Smart-Bricks — «Умных блоков».

Название этого компонента, кроме его прямого значения «Умный блок», можно перевести также как «умница, славный парень, молодчина».

На этом же уровне присутствуют «Мангусты» как «умные инфосервисы», мгновенные виртуальные объединения У-Блоков.

С использованием множества У-Блоков, находящихся на этом уровне, под управлением основной онтологической модели (ООМ), посредством модулей сборки и исполнения «Мангустов» на двух вышележащих уровнях платформы формируются нужные «Мангусты», которые дают ответы на запросы пользователей.

После выполнения запроса «Мангуст» может быть сохранен для повторного использования либо может распасться на составляющие его У-Блоки.

«Умный блок», У-Блок

Описание У-Блока

Понятие Smart-Brick (У-Блок) в данном изобретении — это атомарная микросистема в составе макросистемы (цифровой платформы), содержащая минимально необходимые и достаточные системные компоненты для выполнения полного цикла получения, обработки и представления информации, а именно: данные, программный код, интерфейс и средства управления ими («источник», «преобразователь», «завершитель» и «управление»).

Каждый У-Блок имеет функционально-компонентную структуру, подобную структуре вышележащих уровней, которая включает в себя упомянутые «источник», «преобразователь», «завершитель» и «управление» (см. рисунок 12):

- «источник» работает с входными данными (с голосом, текстом, данными из базы данных, с датчиков от SCADA или корпоративной «интрасети вещей» и так далее), хранит их и передает «преобразователю» по необходимости;
- «преобразователь» — это программный код, выполняющий функциональность конкретного У-Блока, он получает данные от «источника» и передает обработанные данные «завершителю»;
- «завершитель» — это интерфейс, который выполняет или завершает работу У-Блока и передает результат от конкретного У-Блока далее в систему для использования на следующих этапах обработки;
- «управление» является онтологическим описанием небольшой части основной онтологической модели (ООМ), называемой микроонтологической моделью (МОМ) в области ответственности (функциональности) конкретного У-Блока.

Иными словами, У-Блок — это информационная система в миниатюре, содержащая все необходимые и достаточные элементы (согласно закону полноты частей системы) для работы с определенной именно для этого У-Блока микрообластью информационного пространства предприятия.

У-Блок выполняет полный цикл приема, обработки и представления информации на микроуровне цифровой платформы «УБ-Мангуст». Совокупность всех У-Блоков и образует этот микроуровень.

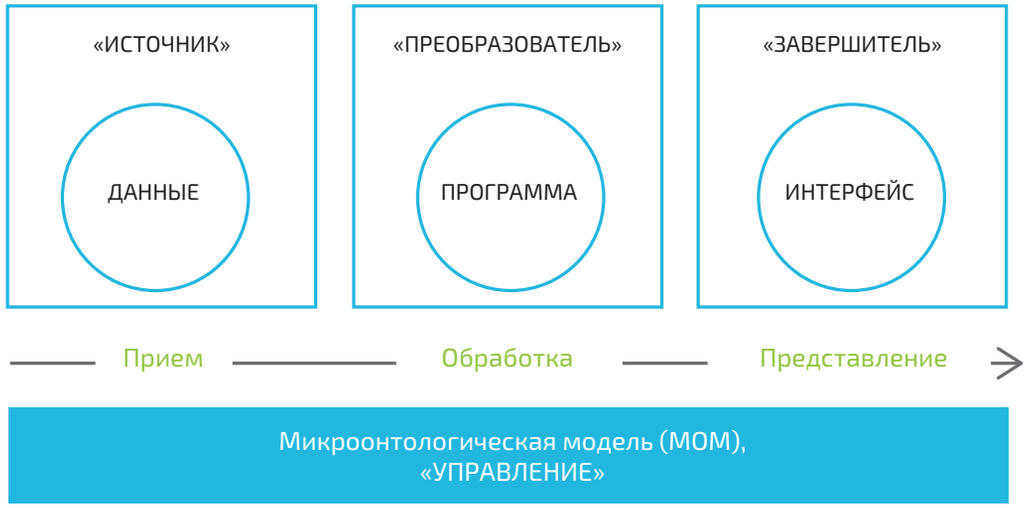
Некоторые свойства У-Блока

У-Блок может быть создан, изменен (модифицирован), удален в архив или окончательно стерт из системы, то есть имеет свой жизненный цикл.

Он может наследовать (копировать) свойства другого У-Блока, которые затем могут быть развиты или модифицированы.

Рисунок 12.

Микроуровень, У-Блок



Состав У-Блока

У-Блок состоит из:

- компонента «источник» У-Блока, содержащего данные;
- компонента «преобразователь» У-Блока, который представляет собой программный код;
- компонента «завершитель» У-Блока, который является интерфейсом;
- компонента «управление» У-Блока, который представляет собой микроонтологическую модель (МОМ).

Три компонента — «источник», «преобразователь» и «завершитель» — связаны через четвертый компонент — «управление» — с ООМ, онтологической моделью всего предприятия; компонент «управление» представляет собой онтологическую микромодель небольшой части предметной области (МОМ) для конкретного У-Блока.

Каждый из указанных выше компонентов У-Блока может адекватно изменяться при изменении ООМ предприятия, поскольку все МОМ всех У-Блоков связаны с ООМ.

Пример

Например, У-Блок «Трансформаторная подстанция» будет содержать:

- «источник», данные (перечень всех подстанций предприятия);
- «преобразователь», программный код, который работает с этими данными;
- «завершитель», интерфейс (графический/голосовой/семантический);
- «управление», микромодель онтологии «трансформаторное хозяйство предприятия».

Рассмотрим указанные выше основные компоненты У-Блока подробнее.

Компонент «источник» (данные)

Этот компонент работает с входными данными. Данные в У-Блоке имеют отношение только к конкретной микрообласти или микрособытию, которые описаны в рамках МОМ именно этого У-Блока.

Пример

Например, У-Блок «Трансформаторная подстанция» будет содержать данные «перечень всех подстанций предприятия», но не будет включать, к примеру, данные о ремонтных бригадах, которые обслуживают эти подстанции; такие данные будет содержать уже другой У-Блок, называемый «Ремонтные бригады».

В традиционной СУБД для подобной функциональной области «Управление энергетическим хозяйством» необходимо жестко указать в модели данных связь упомянутых объектов — «подстанция» и «бригада». В архитектуре «УБ-Мангуст» такой модели данных не существует, соответственно, нет и такой жесткой связи. Нужная связь временно создается только в рамках конкретного «мини-спектакля» «Мангуст», создаваемого, например, в ответ на запрос: «Какие бригады работали на подстанциях?» Этот «Мангуст», собранный «на лету», выполняет запрос и затем распадается за ненадобностью или сохраняется при необходимости повторения.

Если данные событийные (например, от датчиков SCADA/АСУТП), то У-Блок использует парадигму Complex Event Processing, CEP, которая вместо функции «сохрани и обработай» выполняет функцию «обработай на лету, а потом, если нужно, сохрани». У-Блок может подключиться к потоку событий в рамках какого-то «Мангуста», получить данные из этого потока, обработать запрос на лету в соответствии со своим программным кодом, сообщить полученный результат через интерфейс пользователю и, если надо, сохранить этот результат для последующего использования.

Все У-Блоки находятся на микроуровне, на третьей страте «Информационные сервисы», но их данные расположены на второй страте — например, в распределенном реестре (блокчейне).

Следует отметить, что для полностью цифрового предприятия при запросе (команде) пользователя, касающемся некоторого бизнес-объекта, происходит просто процесс получения МОМ этого объекта. Например, получают МОМ объекта «трансформаторная подстанция» как типа, плюс набор данных, описывающих конкретную трансформаторную подстанцию (экземпляр). При этом любой бизнес-объект на предприятии сразу поставляется в двух качествах: в физической форме (оборудование) и в цифровой (его У-Блок).

Компонент «преобразователь» (программный код)

Код описывает действия/процедуры, которые может выполнять У-Блок в рамках своей МОМ (онтологической микромодели) над входящими в нее концептами.

Пример

Например, У-Блок «Трансформаторная подстанция» содержит простой код, позволяющий выполнять действия над объектом «подстанция» с использованием данных «перечень подстанций предприятия»: создать в БД запись об объекте «Подстанция» (insert), изменить ее (update) или удалить (delete). Код, к примеру, может выполнить действия «создать новую подстанцию», «вывести подстанцию из эксплуатации».

В режиме «Развитие» код может быть создан пользователем-программистом, взят из библиотеки, скопирован из другого У-Блока и модифицирован. Для простых случав код создается автоматическими процедурами — скажем, из математических библиотек, содержащих процедуры вычислений (например: «выдай мне среднюю нагрузку на трансформатор»); это может быть записано в свойствах объекта микроонтологической модели У-Блока.

Компонент «завершитель» (интерфейс)

Этот компонент завершает выполнение базовой функции данного У-Блока и передает результат далее в систему по процедуре. Компонент может представлять собой графический, семантический, XML и другие интерфейсы. Может быть и только программный интерфейс, работающий без участия человека — например, обменивающийся данными со SCADA или корпоративным «интранетом вещей».

Компонент «управление» (мини-онтология)

Этот компонент управляет взаимодействием всех остальных компонентов У-Блока, представляя собой онтологическую микромодель небольшой части предметной области (МОМ) для конкретного У-Блока.

Пример

Рассматриваемый пример У-Блока «Трансформаторная подстанция» имеет компонент «управление», содержащий микромодель онтологии «трансформаторное хозяйство предприятия».

У-Блок с точки зрения стратификации

Каждый У-Блок — это микросистема, включающая вертикально небольшие объекты второй и третьей страт и имеющая связь по управлению с четвертой стратой.

Рассмотрим пример У-Блока «Трансформаторная подстанция» снизу вверх по уровням стратификации:

- Онтологическая модель «Управление энергетическим хозяйством предприятия» расположена на четвертой страте.
- Она связана с нижележащими объектами третьей страты — множеством У-Блоков («трансформаторной подстанцией», «ремонтной бригадой электриков» и другими).
- Бизнес-объект третьей страты, У-Блок «Трансформаторная подстанция», содержит микромодель онтологии «трансформаторное хозяйство предприятия», связанную:
  - с вышележащей онтологической моделью «Управление энергетическим хозяйством предприятия» четвертой страты;
  - со своими нижележащими частями (подсистемами У-Блока) — выделенными изолированными областями: данными, кодом, интерфейсом, то есть «источником», «преобразователем», «завершителем».
- На второй страте расположены подсистемы У-Блока, выполняющие основной процесс, через которые проходит «поток главного продукта» — получение, преобразование и представление информации. Здесь может быть использована технология блокчейн.
- На первой страте расположен аппаратный слой, ИТ-инфраструктура — выделенные изолированные вычислительные ресурсы и ресурсы хранения данных.

Таким образом, атомарная У-Блок-микросистема распределена между второй и третьей стратами по вертикали, а управляется с четвертой.

При этом горизонтальные связи между У-Блоками на третьей страте отсутствуют и реализуются только через онтологическую платформу (она расположена на стыке третьей и четвертой страт — как среда исполнения и взаимодействия бизнес-объектов).

На третьей страте расположена программная платформа исполнения и взаимодействия бизнес-объектов (У-Блоки и «Мангусты»).

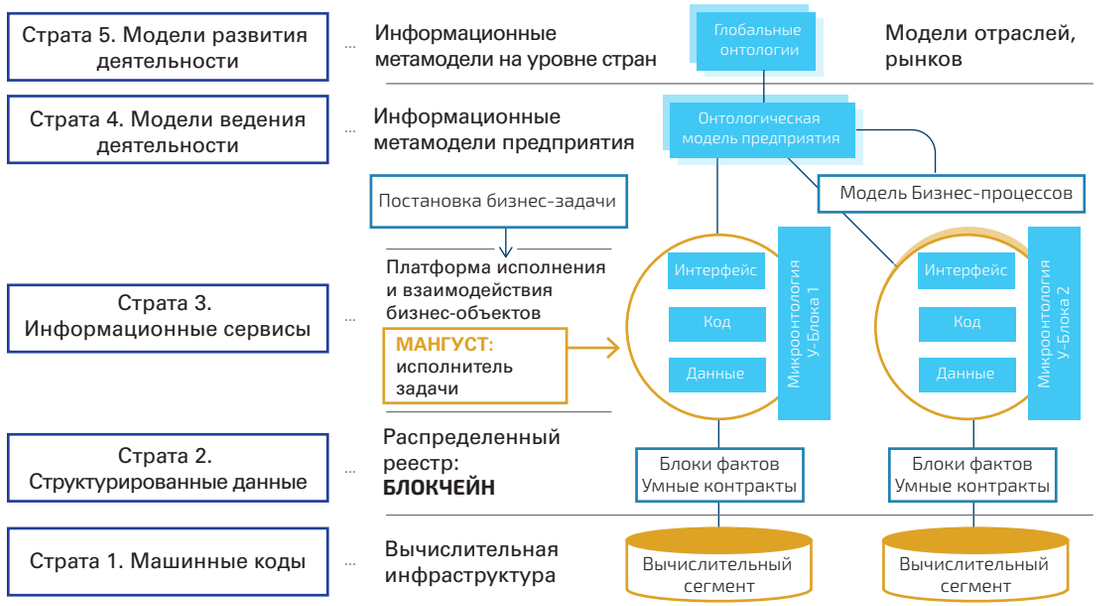
На четвертой страте расположена основная онтологическая модель.

Иллюстрацию этого подхода с указанием места У-Блоков и «Мангустов» смотрите на рисунке 13, который для удобства дублирует рисунок 5.

Связность модели данных при этом подходе обеспечивается онтологической платформой, работающей на языке онтологического программирования, а качество данных обеспечивает онтологическая модель, разработанная на языке онтологического моделирования.

Рисунок 13.

У-Блоки в стратифицированной системе



Внеуровневый элемент: инфосервисы «Мангуст»

Описание «Мангуста»

Как сказано выше в разделе «Базовые понятия», «Мангуст» — это нечто похожее на «специальную группу инфосервисов быстрого реагирования».

Более точно — это группа «умных» информационных сервисов, выполняющая запрос пользователя, которая мгновенно собирается из атомарных У-Блоков и управляется онтологической моделью.

«Мангуст» — динамический виртуальный временный элемент цифровой платформы, поэтому постоянного структурного представления в составе макро- или микрокомпонентов платформы не имеет, но с точки зрения стратификации находится на третьей страте «Информационные сервисы».

«Мангуст» — ключевой элемент онтологической цифровой платформы «УБ-Мангуст», формирующий ее основной информационный продукт — ответ на запрос пользователя.

«Мангуст» создается из правильно подобранных У-Блоков только для нужной операции (запроса, действия, обработки события) под управлением онтологической модели в ответ на запрос пользователя. Это «временная сборка по требованию», которая выполняет одноразовый процесс для предоставления одного ответа на один запрос.

«Мангуст» выполняет свою работу, а затем распадается обратно на составляющие его У-Блоки или остается собранным для повторного использования.

Состав «Мангуста»

Говоря о составе «Мангуста» как виртуального (внеуровневого) элемента цифровой платформы, можно указать такие его составляющие:

- несколько атомарных У-Блоков, собранных в набор, содержащий необходимые данные для ответа на запрос пользователя (У-Блоки находятся на второй страте);
- связи с каждой микроонтологической моделью — МОМ, блоком управления конкретного У-Блока, входящего в конкретную сборку «Мангуста» (МОМ, как и У-Блок, находится на второй страте, но связана с ООМ на четверной страте);
- связи с онтологическими моделями компонентов среднего уровня, содержащих модули, такие как модуль ввода данных, модуль генерации шаблонов, сборки и исполнения «Мангустов» и другие (данные модули находятся на третьей страте);
- связи с основной онтологической моделью предприятия (ООМ находится на четвертой страте).

Как работает цифровая платформа «УБ-Мангуст»

Пример функционирования цифровой платформы «УБ-Мангуст»

Рассмотрим пример работы цифровой платформы «УБ-Мангуст» с информационными объектами (бизнес-объектами) в нефтяной компании.

Для того чтобы выполнить такие процедуры с бизнес-объектами, как, например, «создать новую буровую вышку», «отремонтировать уже существующую» или «вывести из эксплуатации», можно использовать одиночный У-Блок или «Мангуст», собранный из нескольких У-Блоков.

Например, процедуры «создать новую буровую вышку» или «вывести из эксплуатации» могут быть выполнены средствами единственного У-Блока с названием «Буровая вышка»: надо просто выполнить последовательность операций, записанных в программном коде этого У-Блока, в компоненте «преобразователь».

А вот процедура «отремонтировать существующую буровую вышку» может быть выполнена «Мангустом» «Текущий ремонт нефтяных вышек», которые связывают У-Блок «Ремонтный цех» и У-Блок «Буровые вышки». В У-Блоке «Буровые вышки» наряду с другими есть конкретный объект «Буровая вышка № 001», которую требуется отремонтировать.

Другой вариант выполнения ремонтных работ — это когда один «Мангуст» связывает несколько У-Блоков: «Аварии», «Буровые вышки», «Ремонтный цех» и «Ремонтные бригады».

Если же имеется в виду не бизнес-объект в системе, а реальный объект — буровая вышка, которую надо установить на месторождении, то система уточнит это при вводе задачи, и процедура пошагово приведет к другому результату.

Например, таким результатом может быть найденный в системе бизнес-кейс по установке вышки на скважине, включающий процедуры, состав оборудования, стоимость, места возможного приобретения, контрагентов, логистику, персонал, сроки монтажа, известные проблемы монтажа, пути их решения и так далее.

Шаги основного процесса

Когда пользователь дает команду «Создать новую буровую вышку», то запускаются процедуры основного процесса «УБ-Мангуст» (см. раздел «Основной процесс») в рамках компонентов, указанных в разделах «Компоненты макроуровня», «Компоненты среднего уровня», «Компоненты микроуровня».

Повторим, что основной процесс состоит из следующих шагов:

- прием запроса пользователя текстом или голосом;
- определение семантики данных (смысла запроса);
- выбор режима работы платформы;
- выбор нужного семантического шаблона или запрос на создание нового;
- подбор необходимых атомарных У-Блоков;
- сборка «Мангуста»;
- выполнение «Мангуста»;
- получение готовых данных для ответа на запрос;
- формирование ответа пользователю в машиночитаемой форме;
- представление результатов для пользователя в удобной ему форме.

Рассмотрим процесс подробней.

Шаг 1. Прием задачи

Пользователь произносит или вводит текстом задачу «Создать новую буровую вышку», имея в виду свою работу с бизнес-объектом «Буровая вышка» в ее жизненном цикле: создание, модификация, удаление.

Задача принимается и обрабатывается блоком постановки задач.

Сначала задача принимается модулем приема данных (1.1), который отправляет данные в модуль семантического разбора (1.2), точнее, в его обработчик естественного языка (Natural Language Processor) для превращения голоса или текста в машиночитаемый формат, понятный системе.

Шаг 2. Определение семантики

После этого модуль семантического разбора (1.2) разбирает задачу в машинном формате и идентифицирует ее смысл, выделяя семантику элементов «создать», «буровая», «вышка» и сравнивая ее со значениями семантических элементов, содержащихся в ООМ. Результат этого семантического разбора переключает платформу «УБ-Мангуст» в один из двух базовых режимов: «Эксплуатация» или «Развитие».

Шаг 3. Выбор режима

Если определено, что задача принадлежит к режиму «Эксплуатация» (то есть текущая версия ООМ содержит все элементы, необходимые для выполнения команды «Создать новую буровую вышку»), то модуль 1.2 находит в ООМ нужный концепт «Буровая вышка» и отправляет этот факт в модуль генерации шаблонов (1.3).

Если определено, что задача принадлежит к режиму «Развитие» (то есть текущая версия ООМ не содержит всех элементов, необходимых для выполнения команды), то стартует описанная ниже процедура.

Например, дана задача «Создать новую подстанцию» — то есть команда содержит элемент «создать» и элемент «подстанция» вместо «буровая вышка». Предположим, что понятие «подстанция» не существует в текущей версии ООМ, поэтому должен быть запущен режим «Развитие». Он создает новый концепт «подстанция» в структуре ООМ и идентифицирует смысл этого концепта в диалоге с пользователем — что это именно трансформаторная подстанция, относящаяся к энергетическому хозяйству предприятия. Возможно дополнительное уточнение значения в диалоге, если есть несколько объектов, название которых содержит слово «станция» (ж/д станция, подстанция, радиостанция).

Созданный в ООМ новый концепт «подстанция» инициирует создание на микроуровне нового У-Блока «Трансформаторная подстанция» и запускает процесс заполнения нового У-Блока данными о конкретных экземплярах (подстанциях).

Шаг 5. Формирование семантического шаблона

Модуль генерации шаблонов (1.3) создает «семантический шаблон-матрицу» на базе выявленных на предыдущем шаге семантических элементов, для того чтобы запустить на следующем шаге нужный, адекватный смыслу задачи У-Блок или несколько, которые будут исполнены в блоке выполнения задачи (компонент «преобразователь»).

Шаг 6. Подбор нужных У-Блоков

В блоке выполнения задачи посредством модуля поиска У-Блоков (2.1) стартует поиск, точнее, подбор У-Блока, содержащего нужные значения

в соответствии с семантическим шаблоном. Для этого модуль 2.1 обращается в ООМ с данными из шаблона, содержащего значение семантических элементов (то есть смысл задачи), и находит там соответствующие этому шаблону У-Блоки, подходящие для сборки «Мангуста» на следующем шаге.

Шаг 7. Сборка «Мангуста»

Модуль сборки «Мангустов» (2.2) на основе полученного шаблона определяет, достаточно ли единственного У-Блока для выполнения задачи, или надо собрать нужный «Мангуст» из нескольких У-Блоков; при этом используются необходимые дополнительные сведения из ООМ.

ООМ, например, может дополнительно сообщить:

- контекст, в котором будет создаваться новая вышка (скажем, март 2018 года, месторождение А, буровая бригада Б);
- связи между У-Блоками (отношения подчиненности, взаимодействие и другие);
- бизнес-процесс или адаптивный кейс по созданию новой вышки и так далее.

Например, для задачи создания новой вышки нужен единственный У-Блок, а для задачи ремонта существующей потребуется собрать «Мангуст» из нескольких У-Блоков: «Аварии», «Буровые вышки», «Ремонтный цех» и «Ремонтные бригады».

То есть на шаге 7, как правило, формируется «Мангуст», необходимый и достаточный для выполнения задачи пользователя.

Шаг 8. Выполнение «Мангуста», получение ответа

Сформированный «Мангуст» попадает в модуль исполнения «Мангустов» (2.3). «Мангуст» запускает процедуру «создание новой нефтяной вышки», извлекая из всех вошедших в его состав У-Блоков требуемые для ее создания данные, находящиеся в «источниках» каждого У-Блока, обрабатывая их программными кодами, содержащимися в «преобразователях» У-Блока, и формируя получившийся ответ в компонентах «завершитель» каждого У-Блока под управлением ОМСУ выполнения задачи.

В результате этого шага машинный ответ системы (отчет «Мангуста») по созданию новой буровой вышки готов для отправки в блок представления результатов.

Шаг 9. Получение результата из «Мангуста»

На этом шаге компонент «завершитель» блока выполнения задачи передает информацию на обработку в блок представления результата, в его входной модуль. Эта информация, как сказано выше,— машиночитаемый ответ системы на команду пользователя «Создать буровую вышку», результат

выполнения «Мангуста» на предыдущем шаге. Он поступает в модуль получения результата из «Мангуста» (3.1) и представляет собой экземпляр отчета «Мангуста» в формальной нотации (например, в XML).

Шаг 10. Преобразование машинных данных

На этом шаге экземпляр отчета «Мангуста» в формальной нотации (например, в XML) преобразовывается в законченную форму, удобную для восприятия пользователем (текст, речь, звук, графика или 3D-изображение — голограмма). Работа выполняется в модуле подготовки результата (3.3). Результатом выполнения задачи по созданию бизнес-объекта «Новая буровая вышка» будет электронный документ или информационная модель бизнес-объекта в принятой на предприятии нотации.

Шаг 11. Представление результатов пользователю

Этот результат передается модулем представления результатов пользователю (3.3) на периферийные устройства (экран, динамики, принтер, обычный или голографический проектор).

Два режима: бимодальное управление

Чтобы выполнить двуединую задачу по повышению адаптивности и гибкости информационных систем, в «УБ-Мангуст» предусмотрена бимодальная система управления Run & Change, где Run относится непосредственно к функционированию, а Change отвечает за модернизацию и адаптацию к изменениям.

Эксплуатационный режим Run нужен для обеспечения гибкости ИС при решении актуальных задач управления предприятием; в этом режиме идет текущая работа пользователей в системе.

Режим развития Change нужен для обеспечения адаптивности ИС к изменившимся внешним условиям; в нем меняется онтологическая модель предметной области: «виртуальная модель» предприятия адаптируется к изменившейся реальности, и адекватно поменявшимся условиям внешней среды формируется новое состояние информационной системы, поддерживающее новые условия «эксплуатационного режима».

Итак, рассмотрим два режима подробнее:

- в режиме эксплуатации выполняются функции по предоставлению информации пользователю;
- в режиме развития выполняются функции по развитию системы.

Режим эксплуатации

Режим эксплуатации обеспечивает гибкость платформы «УБ-Мангуст» в текущих условиях управления предприятием (транзакции, аналитика

и поддержка принятия решений). В этом режиме выполняются все обычные (но как угодно сложные и разнообразные) запросы пользователей к системе, работающей при неизменных внешних условиях.

Иными словами, основная функция цифровой платформы «предоставлять информацию по запросу» выполняется в режиме эксплуатации; этот режим дает возможность пользователю вводить любые запросы и получать значимые адекватные ответы от цифровой платформы в условиях постоянной, неизменной окружающей деловой среды (предметной области); эти условия отражены в текущей версии основной онтологической модели (ООМ). «Виртуальная модель предприятия», ее ООМ не изменяется.

Режим развития

Второй режим, режим развития, необходим для адаптации «УБ-Мангуст» к изменяющимся внешним условиям. Этот режим изменяет ООМ, и «виртуальный образ» предприятия адаптируется к изменившейся реальности. Адекватно изменяющимся условиям окружающей среды формируется новый статус цифровой платформы, который поддерживает новую версию режима эксплуатации. При этом изменения в ООМ могут быть продиктованы как внутренними, так и внешними факторами.

Иначе говоря, вторая функция «УБ-Мангуст» — «совершать действия по развитию системы» — выполняется в режиме развития; такой режим предполагает реакцию на необходимость адаптации «УБ-Мангуст» к изменяющимся условиям деловой среды (предметной области); адаптация цифровой платформы производится изменениями в основной онтологической модели; в результате изменений создается новая версия ООМ и включается новая версия режима эксплуатации.

В качестве дополнительной возможности, при необходимости клонирования ООМ (или ее части) в информационную систему с иным назначением, ООМ можно использовать вне «УБ-Мангуст». Для этого предусмотрен режим развертывания цифровой платформы «с нуля», получивший название «режим яйца», или «цифровой зародыш».

«Режим яйца» обеспечивает способность клонировать цифровую платформу, выполняя процесс саморазвития от «цифрового зародыша». Это симуляция ускоренной эволюции системы — развертывание цифровой платформы из функционального ядра, состоящего из основных системных компонентов: «источника», «преобразователя», «завершителя» и «управления». При этом платформа развивается от ядра к периферии, формируя вокруг основных компонентов необходимое окружение из вспомогательных, дополнительных и обеспечивающих компонентов.

ООМ получает информацию из окружающей среды, которая отличается от той, в которой она была сформирована изначально (например, модель нефтеперерабатывающего завода, НПЗ), и по результатам анализа нового делового окружения адаптируется к новым условиям ведения бизнеса, подстраивается под них.

Например, можно запустить «режим яйца» в случае необходимости глубокой модернизации информационной системы упомянутого НПЗ, для развертывания такого же НПЗ в другом географическом регионе или даже для того, чтобы создать информационную систему «с нуля» для предприятия с другим назначением, например для завода по переработке отходов.

Вывод

Итак, читатель, давайте подведем некий итог и выясним, что из намеченного нам удалось осуществить?

Сумели ли мы найти пути преодоления проблемы ригидности и слабой адаптивности существующих информационных систем, то есть решить задачу, которую мы поставили в разделе «Информационная система будущего» как решение этих проблем», хотя бы теоретически, в рамках содержания этой книги?

Удалось ли нам придумать архитектуру цифровой платформы предприятия, которая:

- сможет легко и быстро изменяться адекватно произошедшим или предполагаемым изменениям в деловой среде предприятия;
- даст пользователю возможность выполнять любые неструктурированные и не предусмотренные заранее действия с информацией;
- позволит задавать любые вопросы к системе для получения от нее ответов в реальном времени, просто озвучивая их на родном языке или вводя их в виде обычного текста?

Мы считаем, что успешно справились с задачами и в итоге имеем плод изобретения — новую, удовлетворяющую запросам современного бизнеса цифровую платформу, которая способна во многом усовершенствовать информационную среду предприятия. Во всяком случае, начало «новой жизни» положено — ведь цифровая платформа будущего обрела свой дом в компании Comindware: https://www.comindware.com/ru/platform/

Давайте узнаем, как функционирует наше изобретение в реальной среде, и обратимся к главе 3, в которой описан опыт реализации новой цифровой платформы.

Глава 3.
Пример реализации новой цифровой платформы

Глава подготовлена в соавторстве с BPM-евангелистом Comindware Анатолием Белайчуком и техническим директором Петром Волынским.

Компания Comindware в настоящее время развивает свою цифровую платформу Comindware Business Application Platform с учетом архитектурных принципов онтологической платформы «УБ-Мангуст».

Вводная часть

Постановка проблемы

Цифровая платформа Comindware Business Application Platform возникла как ответ на возрастающую сложность практики управления бизнес-процессами в компании. Приведем цитату из [20]:

«Бизнес-процессы определяют то, как организации выполняют работу, создающую ценность для клиентов. Осознанное управление этими процессами ведет к совершенствованию методов ведения бизнеса, что, в свою очередь, выражается в более эффективной организации потоков работ, более высокой производительности, большей маневренности и в конечном итоге — к более высокой отдаче от инвестиций».

Приведем определение понятия «управление бизнес-процессами» из [20]:

«Управление бизнес-процессами (Business Process Management, BPM) — это концепция управления, увязывающая стратегию и цели организации с ожиданиями и потребностями клиентов путем соответствующей организации сквозных процессов. BPM сводит воедино стратегию, цели, культуру и организационную структуру, роли, политики, нормативы, методологии и программные средства для: а) анализа, проектирования, внедрения, управления и непрерывного улучшения сквозных процессов и б) регулирования отношений в области процессного управления».

Эта весьма обширная и сложная концепция в практическом применении невозможна без поддерживающего ее программного обеспечения (далее — ПО). Хотя такого ПО на рынке представлено достаточно много, надо сказать, что на текущий момент в этих ИТ-решениях не существует унифицированного механизма для удобного представления данных и контекста данных бизнес-процессов.

Как правило, большинство поставщиков программного обеспечения предоставляют одну часть «головоломки»: например, CRM (систему управления взаимоотношениями с клиентами), отслеживание счетов, отслеживание заказов, поддержку планирования с диаграммой Ганта. Многие поставщики в этой области являются относительно небольшими компаниями, и, как правило, программный продукт вырастает из их собственной начальной необходимости решить конкретную задачу, и этот продукт часто связан с конкретным «взглядом на мир» одного поставщика.

Типичным деловым предприятиям необходимо покупать и интегрировать несколько частей ПО, но при этом, например, ПО для управления проектами обычно не взаимодействует с бухгалтерским программным обеспечением, ПО для управления взаимоотношениями с клиентами не «разговаривает» с ПО заказа запчастей, и так далее. Кроме того, большая часть таких пакетов ПО обладают ограниченными возможностями смены способа предоставления данных пользователю.

Зачастую в качестве «двигателя» используются реляционные базы данных с присущими их природе ограниченными возможностями добавления полей в базу данных. Однако для большинства конечных пользователей необходимость смены «прямого» представления данных создает реальную сложность и поэтому часто требует найма консультантов или дополнительного ИТ-персонала.

Необходима унифицированная архитектура, которая позволяла бы деловым предприятиям с легкостью представлять контекст для бизнес-анализа в бизнес-процессах предприятия, при этом обеспечивая удобный механизм взаимодействия между пользователями, отражая различные аспекты бизнеса и нужные данные в единой гибкой бизнес-модели.

В ответ на эту потребность и была создана Comindware Business Application Platform; платформа относится к ПО для управления бизнес-процессами, основана на унифицированной архитектуре и единой синтаксической/семантической конструкции представления данных биз-

нес-процессов и обеспечивает удобное и простое управление бизнес-процессами предприятия.

Название и область применения

Comindware Business Application Platform — это современная платформа для адаптивного управления бизнесом и для быстрой цифровой трансформации бизнеса (https://www.comindware.com/ru/).

Это универсальная платформа для автоматизации бизнес-процессов и разработки корпоративных приложений под любые потребности бизнеса. Платформа предоставляет инструменты для работы с процессами, проектами, кейсами, для хранения данных и ведения коммуникаций между сотрудниками в единой рабочей среде.

Бизнес-логика корпоративных приложений на базе Comindware Business Application Platform не «бетонируется» в программный код, что делает их гораздо более гибкими по сравнению с коробочными системами ERP и CRM. На практике гибкость обеспечивается тем, что в ходе первоначального внедрения настраивается базовая бизнес-логика выбранного приложения, после чего можно менять бизнес-процессы и добавлять новую функциональность «на лету» в широких пределах, реагируя на новые требования бизнеса — внешние и внутренние.

Высокая скорость внедрения и адаптации достигается благодаря концепции минимального кодирования (low-code development). На практике это означает, что для внесения изменений достаточно переместить соответствующий элемент модели мышкой.

Платформа поддерживает такие аспекты бизнеса, как:

- управление отношениями с клиентами (CRM);
- электронный документооборот (СЭД);
- управление ИТ-услугами (ITSM-система);
- бюджетирование;
- планирование ресурсов;
- система управления логистикой;
- управление мастер-данными (MDM);
- управление отгрузками;
- выполнение заказов;
- управление закупками;
- кадровые сервисы.

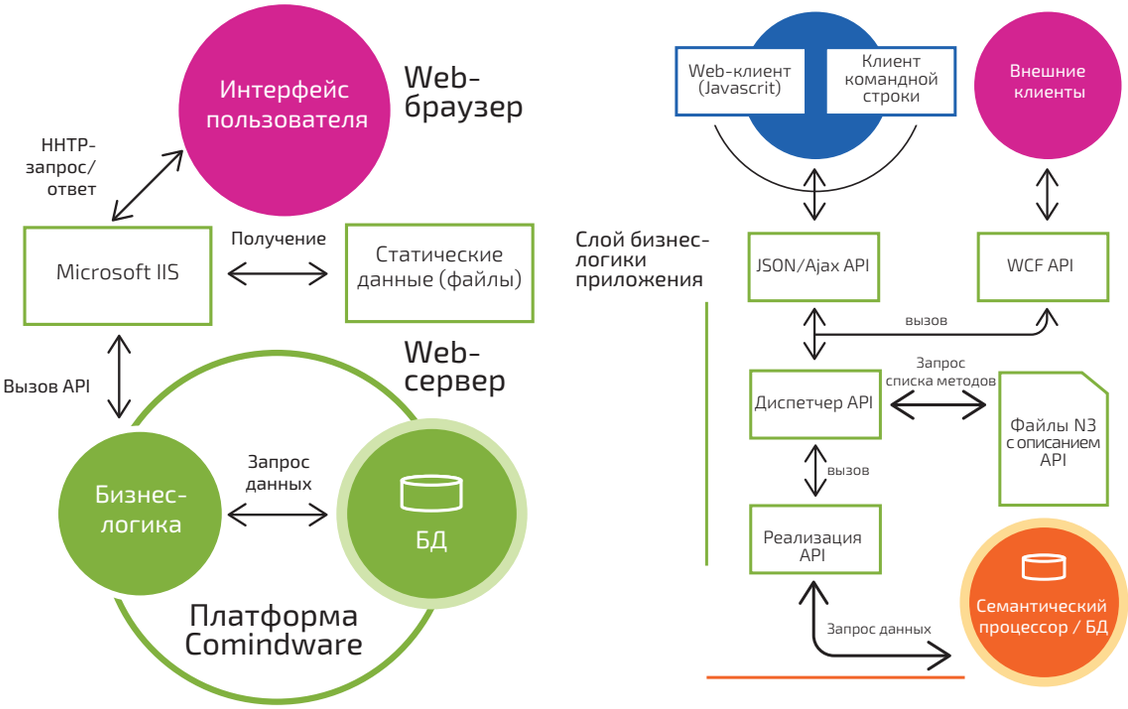


Рисунок 14.

Сервер приложений Comindware

Рисунок 15.

Обзорная диаграмма потока данных при вызове API

Архитектура Comindware Business Platform

Сервер приложений

Платформа Comindware при развертывании в режиме отдельно работающего сервера представляет собой web-сервер под управлением Microsoft IIS с установленным приложением Comindware Platform. Настройка сетевой инфраструктуры доступа к приложению осуществляется системным администратором с использованием стандартных инструментов Microsoft IIS.

Бизнес-логика платформы Comindware: операции и запросы

Общая схема запросов

Работа клиентов с сервером осуществляется через API. Общая структура обработки вызовов API представлена на рисунке 15.

При старте приложения менеджер API загружает из хранилища системных онтологий описание API платформы и модулей расширений. Данное описание может быть затребовано клиентом либо посредством специального ajax-запроса, либо в виде схемы в формате WSDL с использованием протокола SOAP. Далее клиентские приложения могут осуществлять вызовы с использованием ajax-запросов в формате пред-

ставления данных JSON либо по протоколу SOAP. Рассмотрим основные этапы обработки запроса:

1. Входящий запрос обрабатывается HTTP-сервером. Происходит сериализация представления JSON либо SOAP во внутренний формат.
2. Диспетчер API принимает входные данные, осуществляет валидацию входящих аргументов на соответствие описанию метода. На данном этапе также происходит первоначальная проверка безопасности, то есть проверка на соответствие вызывающего пользователя заданным требованиям.
3. Диспетчер API загружает и подготавливает требуемую модель данных и создает снимок модели для изоляции от других запросов и операций. В случае операции, которая меняет содержимое модели, происходит открытие транзакции на запись.
4. Вызывается код, реализующий метод.
5. В случае модифицирующей операции происходит закрытие транзакции, проверка изменений на безопасность, детекция конфликтов, обновление истории операций.
6. Результат сериализуется в формат, необходимый клиенту, и отдается в HTTP-ответе.

Описание API

Для описания API используется онтология на языке N3, представленная в файле sys_api.n3. Центральной частью данного описания является описание метода. В описании метода можно выделить следующие основные атрибуты:

publicNamespace — имя пространства имен для клиентского кода;

publicClass — имя класса для клиентского кода;

publicMethod — имя метода для клиентского кода;

history — признак того, что все изменения в БД, которые производит данный метод, должны сохраняться в истории операций;

operation — признак того, что метод осуществляет запись в БД;

resultRange — описание типа возвращаемого результата;

parameters — описание типов аргументов;

target — описание модели, с которой работает данный метод;

assemblyFile — имя файла модуля приложения, реализующего метод;

assemblyName — имя модуля приложения, реализующего метод;

codeClass — имя класса приложения, реализующего данный метод;

codeMethod — имя метода реализации;

requires — список пользователей либо групп, которым доступен вызов данного метода.

В настоящее время все методы в платформе Comindware разделены на два класса, находящихся в пространстве имен Comindware.Platform:

1. Класс Comindware.Platform.Core содержит набор методов для работы с объектами данных, такими как задачи, списки, объекты, контейнеры и другие.
2. Класс Comindware.Platform.Administration содержит методы для конфигурации системы. Большинство методов из этого класса доступно только администраторам.

Дополнительные классы и методы могут быть описаны в модулях-расширениях платформы. Для этого описание методов добавляется в файл api.n3, а реализация в виде модуля .NET подключается к платформе средствами IIS/ASP.Net.

Полный список методов API представлен в файле api.n3.

Изоляция операций и определение конфликтов

Операции и запросы изолируются с помощью транзакций. После открытия на запись либо чтение транзакция становится полностью изолированной от других. Любые изменения в модели данных, сделанные другими транзакциями, не отражаются.

При закрытии транзакции, открытой на запись, происходит выявление и разрешение конфликтов. При этом используется модель оптимистичного параллелизма (Optimistic concurrency).

Выявление конфликтов происходит на уровне отдельных семантических фактов. Конфликт возникает, когда один и тот же факт был модифицирован двумя транзакциями с момента создания снимка модели и до момента закрытия транзакции. При определении конфликта генерируется исключение, которое может быть обработано клиентом. При этом пользователю может быть предложено актуализировать сохраняемые изменения и повторить попытку.

Загрузка модели данных

В описании операции либо запроса указывается конфигурация модели, с которой будет осуществляться работа. Модель может состоять из объединения нескольких источников данных, таких как БД или файлы с онтологиями. При этом в рамках одной транзакции все изменения осуществляются только в одной базе данных из объединения. В настоящее время в платформе Comindware существует четыре основных источника данных (см. рис. 16):

1. Основная база данных с пользовательскими данными.
2. База данных истории операций.

Рисунок 16.

Обзорная диаграмма конфигурации БД

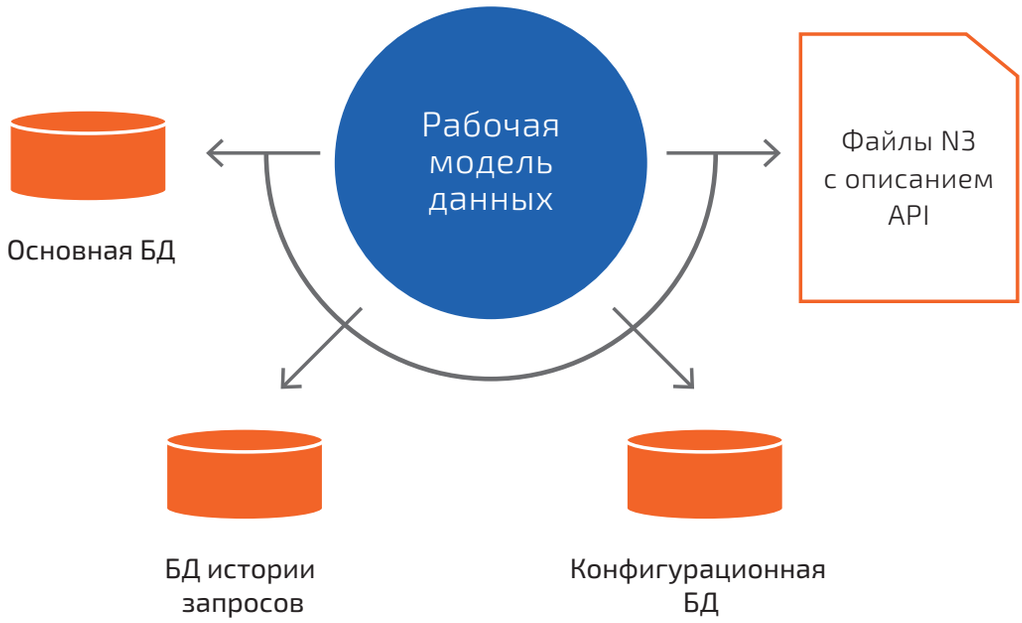
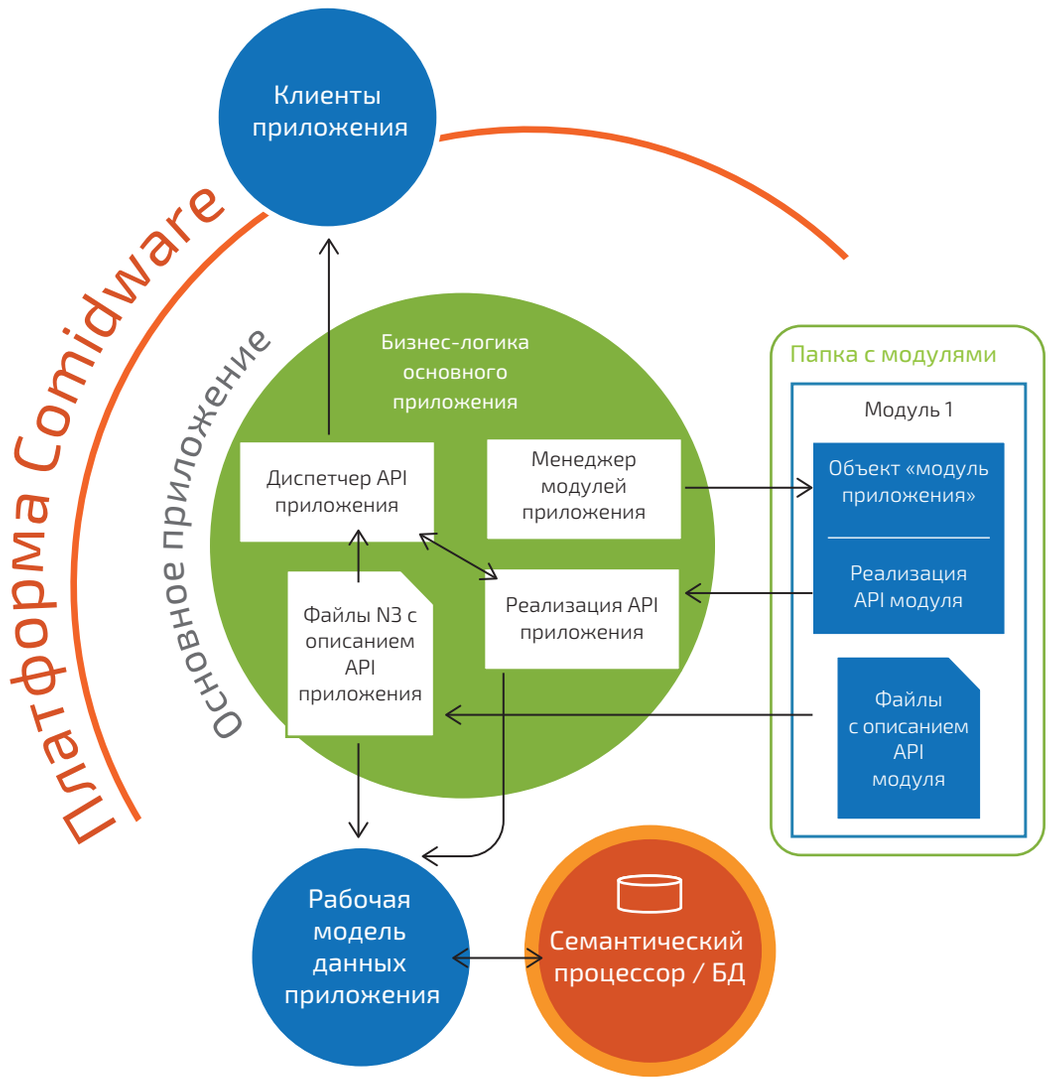


Рисунок 17.

Взаимодействие «приложение — модуль»



3. Конфигурационная база данных. Содержит настройки приложения.
4. Файлы онтологий в формате N3.

Модульная архитектура приложения

Приложение построено таким образом, что его функциональность может быть расширена за счет загрузки до установки новых модулей. Модуль представляет собой:

- двоичный файл формата.dll с реализацией API модуля;
- набор файлов формата N3, описывающих API модуля (файлы api.n3 и extension.n3).

Менеджер модулей приложения в процессе инициализации просматривает особую папку, в которую устанавливаются модули. Для каждого найденного модуля проводятся следующие операции:

- API приложения расширяется за счет API модуля, прочитанного из файлов api.n3 и extension.n3;
- производится инициализация кода модуля: в файле.dll модуля находится класс, реализующий специальный интерфейс и инстанцируется объект данного класса (объект «модуль»).

На рисунке 17 показаны основные компоненты модулей и приложения, участвующие во взаимодействии «приложение — модуль».

Модуль машины состояний и графовая БД

Ниже приведен пример использования графовой базы данных для описания модуля машины состояний.

Ключевые понятия модуля машины состояний

Рассмотрим ключевые понятия модуля машины состояний на примере процесса принятия нового сотрудника на работу (см. рисунок 18).

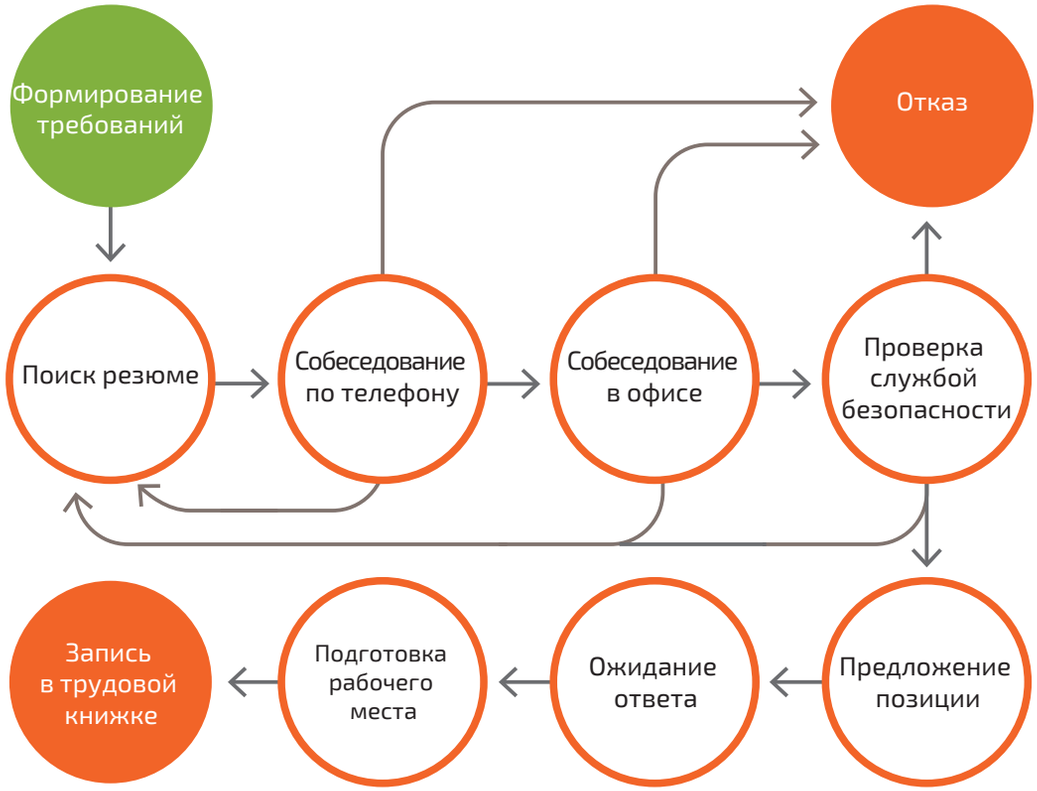
Машина состояний — описание состояний моделируемого процесса и переходов между этими состояниями. Пример: «принятие сотрудника на работу».

Автоматный объект — совокупность машины состояний и данных, необходимых для прохождения по ее состояниям. Пример: принятие Иванова Ивана Ивановича на должность генерального директора. В этом примере данные — это «Иванов Иван Иванович» и «генеральный директор», а машина состояний — это «принятие сотрудника на работу».

Начальное состояние — состояние в машине состояний, для которого нет входящих переходов. Всякий автоматный объект, реализующий машину состояний, возникает в начальном состоянии. Пример: «формирование требований» в машине состояний «принятие сотрудника на работу».

Рисунок 18.

Пример схемы машины состояний



Промежуточное состояние — состояние в машине состояний, для которого есть входящие и исходящие переходы. Пример: «поиск резюме», «предложение позиции» в машине состояний «принятие сотрудника на работу».

Конечное состояние — состояние в машине состояний, для которого нет исходящих переходов. Пример: «отказ», «запись в трудовой книжке» в машине состояний «принятие сотрудника на работу».

Переход — возможность перемещения автоматного объекта между двумя состояниями его машины состояний. По отношению к переходу состояния называются исходное и целевое состояния, причем по заданному переходу возможно перемещение объекта только из исходного в целевое состояние. Пример: «поиск резюме» (исходное состояние) и «собеседование по телефону» (целевое состояние) соединены переходом в машине состояний «принятие сотрудника на работу».

Модуль машины состояний — модуль, позволяющий создавать в системе произвольное количество машин состояний и связывать их с шаблонами пользовательских объектов. Таким образом, по таким шаблонам (будем называть их *автоматными шаблонами пользовательских объектов*) становится возможным создание автоматных объектов.

Рисунок 19.

Обзорная диаграмма архитектуры модуля машины состояний

Архитектура модуля

Модуль машин состояний не является самостоятельной программной единицей и работает только в составе основного приложения. При подключении к основному приложению он расширяет функциональность за счет добавления функций в API и расширения системных онтологий.

Модуль состоит из:

- менеджера машин состояний — набора API, позволяющего создавать автоматные шаблоны пользовательских объектов, редактировать и удалять их;
- менеджера автоматных объектов — набора API, позволяющего создавать, редактировать и удалять автоматные объекты. Отдельно стоит выделить API для выполнения перехода автоматного объекта между состояниями.

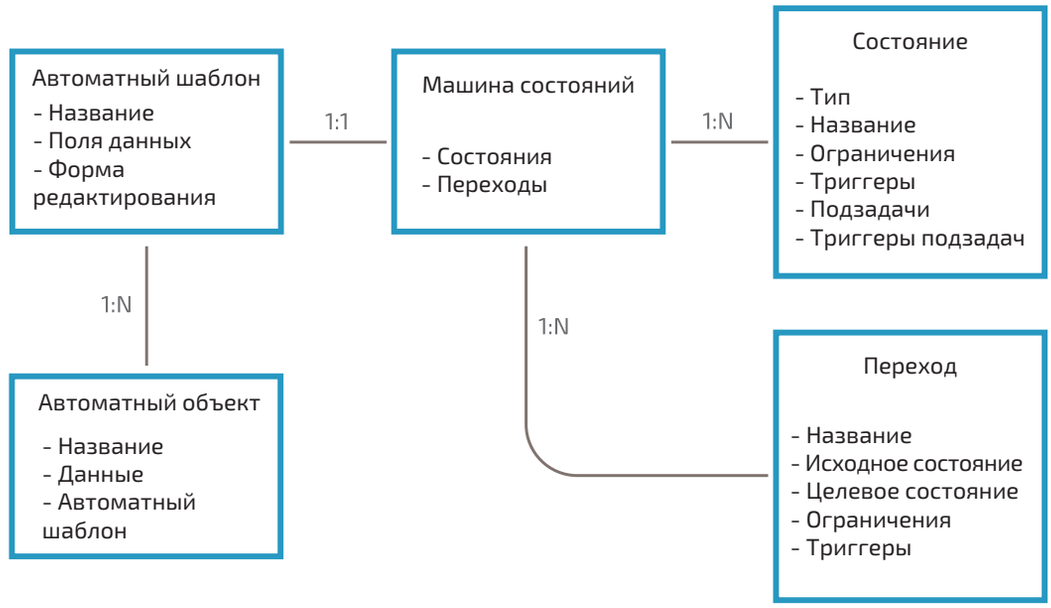
Реализация машин состояний в модуле

Машина состояний, ее состояние и переходы хранятся в модели в виде набора фактов. Упрощенная схема данных для хранения машины показана на рисунке 20.

Создание и редактирование машин состояний происходит через API менеджера машин состояний. Оно устроено таким образом, что любое

Рисунок 20.

Схема данных машины состояний



редактирование машины состояний приводит к ее валидации. Ниже перечислены основные этапы валидации машины состояний:

- В машине должно быть одно и только одно начальное состояние.
- В машине должно быть хотя бы одно конечное состояние.
- Начальное состояние должно иметь не менее одного исходящего перехода. Каждое конечное состояние должно иметь хотя бы один входящий переход.
- Каждое промежуточное состояние должно иметь минимум по одному входящему и исходящему переходу.
- Если в состоянии создается подзадача, то такое состояние должно иметь по меньшей мере один исходящий переход.

Как видно из схемы данных, переходы и состояния машины состояний могут иметь ограничения и триггеры:

- Ограничение — бизнес-правило, хранящееся в модели, целью которого является необходимость определить, соответствует ли автоматный объект используемому состоянию или переходу машины состояний. В случае если автоматный объект не будет соответствовать ограничению, система откажет в переводе объекта по этому переходу или в это состояние. Пример: в машине состояний на рисунке 20 можно добавить ограничение в состоянии «собеседование по телефону», указывающее, что при входе в это состояние поля «телефон» и «имя кандидата» являются обязательными.
- Триггер — бизнес-правило, хранящееся в модели, целью которого является изменение автоматного объекта при выполнении заданного

перехода или при входе в заданное состояние. Пример: в машине состояний на рисунке 20 можно добавить триггер в переходе между состояниями «проверка службой безопасности» и «предложение позиции», устанавливающие поле данных «собеседования закончены» автоматного объекта в значение «истина».

Переход в любое из состояний машины переходов, исключая конечное, может привести к созданию подзадачи автоматного объекта. Задача, как правило, необходима в случае, если ответственность за данное состояние автоматного объекта лежит на человеке и этот человек является пользователем приложения. В случае создания задачи ответственное лицо сможет узнать о необходимости своих активных действий через системные механизмы (уведомления, списки активных задач и другие). Так, в первом примере целесообразно создавать подзадачи в каждом из промежуточных состояний, исключая «ожидание ответа», так как ответственность за это состояние лежит на кандидате, который не является пользователем приложения. Действия в оставшихся состояниях, как правило, выполняются пользователями системы.

Результаты общения с потенциальными покупателями продукта, а также пользователями систем, основанных на BPMN, явно указывают на насущную потребность автоматизировать ряд несложных процессов с помощью систем, которые требуют гораздо меньше усилий по настройке.

Несмотря на ряд объективных ограничений, которые присущи логике машины состояний, прежде всего неспособность создания параллельных веток процесса, логика машины состояний доказала свою эффективность и простоту настройки для автоматизации таких процессов, как:

- обработка обращений в службу поддержки;
- наем сотрудников;
- управление маркетинговыми активностями;
- трекинг финансовой документации и так далее.

Выполнение перехода между состояниями машины

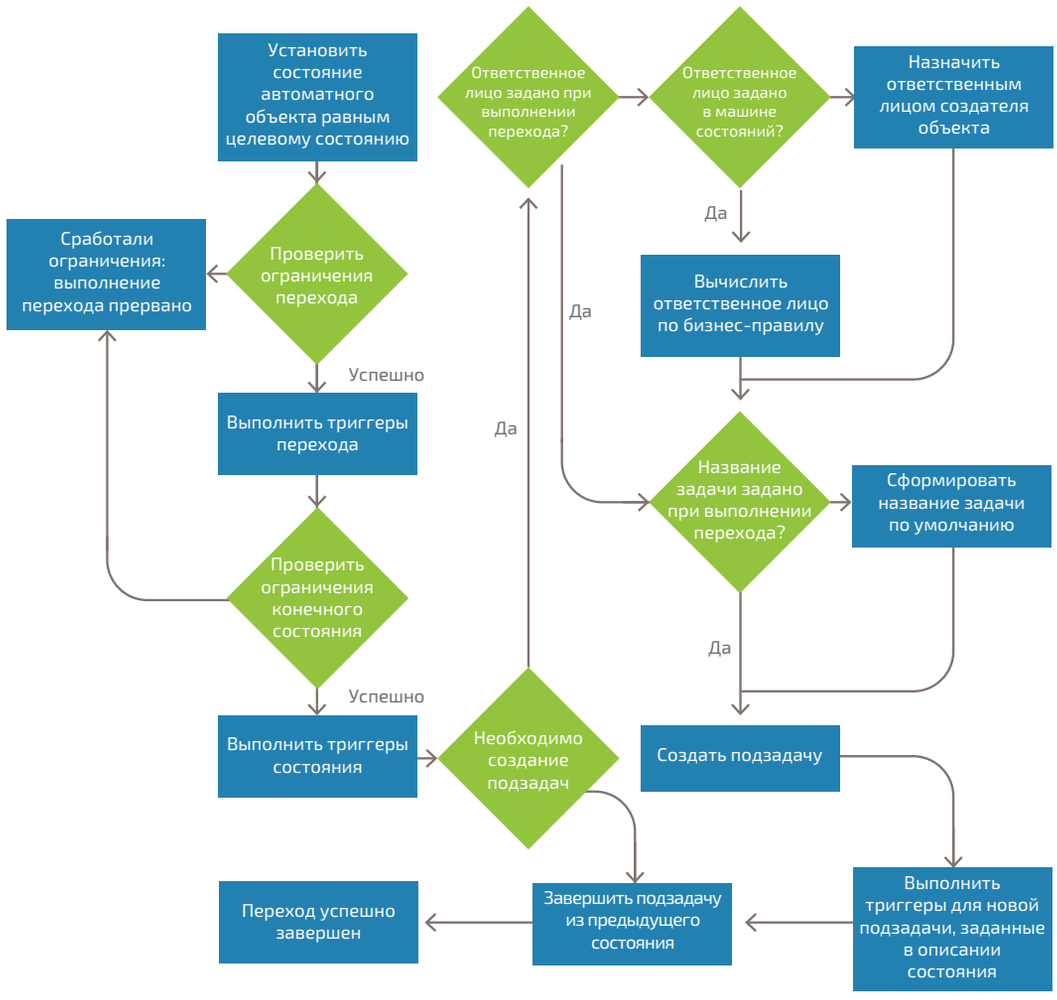
Выполнение перехода между двумя состояниями машины — комплексная задача, состоящая из нескольких этапов. Подробный алгоритм выполнения изображен в виде блок-схемы на рисунке 21.

В итоге после успешного выполнения перехода:

- изменяются свойства автоматного объекта в соответствии с триггерами использованного перехода и нового состояния;
- завершаются подзадачи автоматного объекта, относящиеся к предыдущему состоянию (если таковые существуют);

Рисунок 21.

Блок-схема алгоритма перехода по состояниям



- создаются новые подзадачи, соответствующие целевому состоянию перехода.

Модуль БД для хранения RDF-графов на основе BTree имплементации

Общее описание

Реализация на основе B-деревьев позволит хранить большое количество фактов в одном файле базы данных.

Количество фактов, хранящихся в файле, зависит от таких параметров, как размер страницы и степень связности (чем больше узлов графа связано между собой, тем больше фактов может быть сохранено в файле).

Рассмотрим предлагаемый способ хранения на основе B-деревьев.

Более подробно реализация на основе B-деревьев описана в [21].

Техническая реализация модуля баз данных

В модуле можно выделить три основных функциональных слоя:

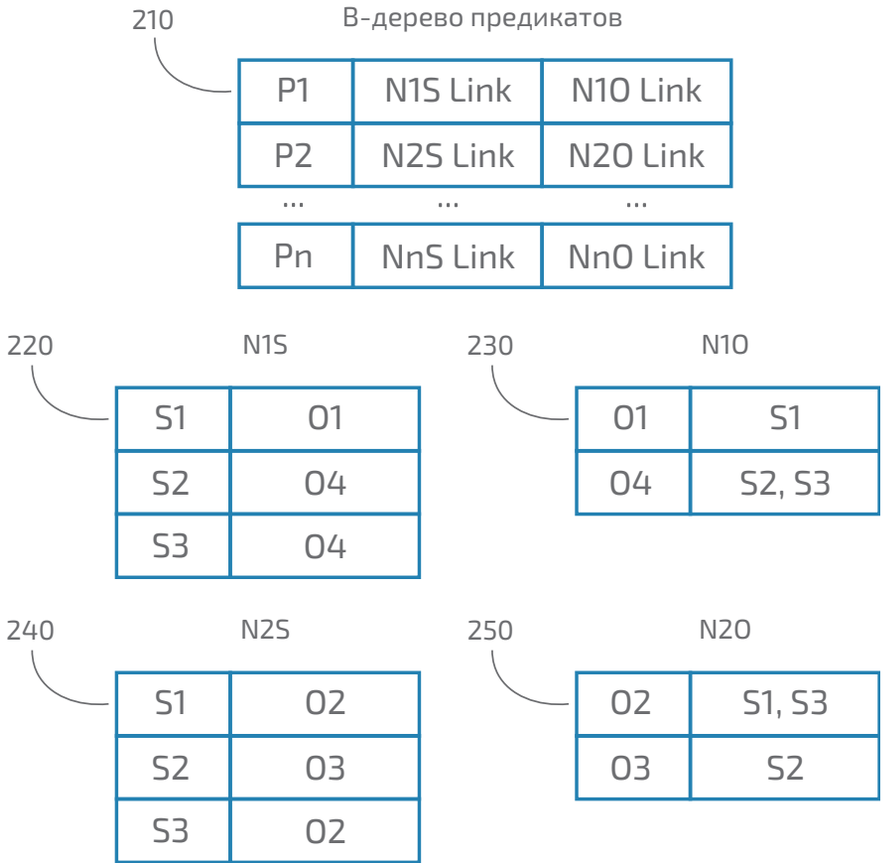


Рисунок 22.

База данных с B-древовидной структурой для хранения троек

1. Слой-обертка, реализованный на языке C#. Кроме предоставления C# интерфейса, данный слой отвечает за конвертирование данных в нужный формат для следующего слоя.
2. Слой доступа, реализованный на языке C++. Данный слой отвечает за организацию логических связей данных, хранимых в БД.
3. Слой BTree хранилища. Данный слой реализует непосредственное хранение данных в виде BTree.

Основные компоненты каждого слоя изображены на рисунке 23. Из диаграммы видно, что реализация нескольких компонентов разделена между слоями.

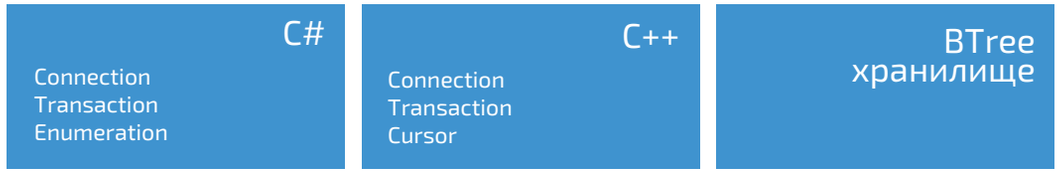
Краткое описание основных компонентов:

- Connection предоставляет snapshot базы данных для организации одновременной работы с ней нескольких клиентов.
- Transaction реализует безопасный транзакционный доступ к базе данных.
- Cursor предоставляет подмножество данных, выборку из базы данных.

Кроме основных компонентов, слои содержат набор второстепенных классов. Так как общая диаграмма классов чересчур громоздка, она не при-

Рисунок 23.

Логическая схема организации модуля базы данных



ведена в данном разделе. Взаимодействие компонентов также вынесено из раздела — см. [21].

Хранение N3 в клиентских СУБД

Описание

Система управления данными, позволяющая пользователям управлять данными на клиентской машине при наличии ограниченного доступа к серверу или при его отсутствии.

Цель

1. Уменьшить нагрузку на сервер.
2. Разрешить клиенту добавлять, изменять, извлекать и запрашивать данные «на лету».
3. Подготовка больших пакетов данных для передачи и обработки на сервере (механизм публикации).
4. Благодаря тому же формату данных, что и на стороне сервера, устранить необходимость преобразования данных.

Структура

Система состоит из трех основных частей:

1. Слой хранения.
2. Слой извлечения данных.
3. Слой логики.

1. Слой хранения подразделяется на:

1.1. Модель триплетов памяти — основной компонент, управляющий работой других подкомпонентов. Предоставляет базовые, низкоуровневые механизмы для добавления, редактирования, удаления или извлечения данных.

1.2. Словарь — простые списки так называемых «слов», которыми по существу являются любые константы, будь то «мой любимый фильм», или «27.03.2012», или «25,4». Каждое слово однозначно идентифицируется по его индексу в словаре. Доступ к словам предоставляется только по индексу.

1.3. Hash-словарь — обратная структура вышеупомянутого словаря, обеспечивает доступ к индексам по словам.

1.4. Счетчик ссылок — подкомпонент для хранения количества ссылок для каждого слова, то есть каждый раз, когда слово используется

в определенном контексте, счетчик ссылок на это слово увеличивается на единицу. После того как контент удален, счетчик ссылок уменьшается.

1.5. Subject Predicate Object Coordinate System — каждый «контекст», или в нашей терминологии «факт», внутренне представлен точкой в трехмерном пространстве. Например, рассмотрим три факта: «Петр любит автомобили», «Петр любит рыбалку», «Джейсон любит рыбалку». Предположим, что у нас уже имеются все слова, используемые в фактах в нашем словаре, и известны соответствующие им индексы, например: (Петр, 0), (любит, 1), (автомобили, 2), (рыбалку, 3), (Джейсон, 4), то эти факты будут записаны следующим образом: (0, 1, 2), (0, 1, 3), (4, 1, 3).

1.6. Object Predicate Subject Coordinate System представляет собой то же самое, но в данном случае факты записаны в обратном порядке для будущего доступа на основе объектов.

2. Слой извлечения данных состоит из следующих компонентов:

2.1. Brain Controller — компонент основного слоя, обеспечивает открытый интерфейс к функциям извлечения данных. Позволяет пользователю обрабатывать простые и упрощенные запросы через механизм Matcher Sequence Processor.

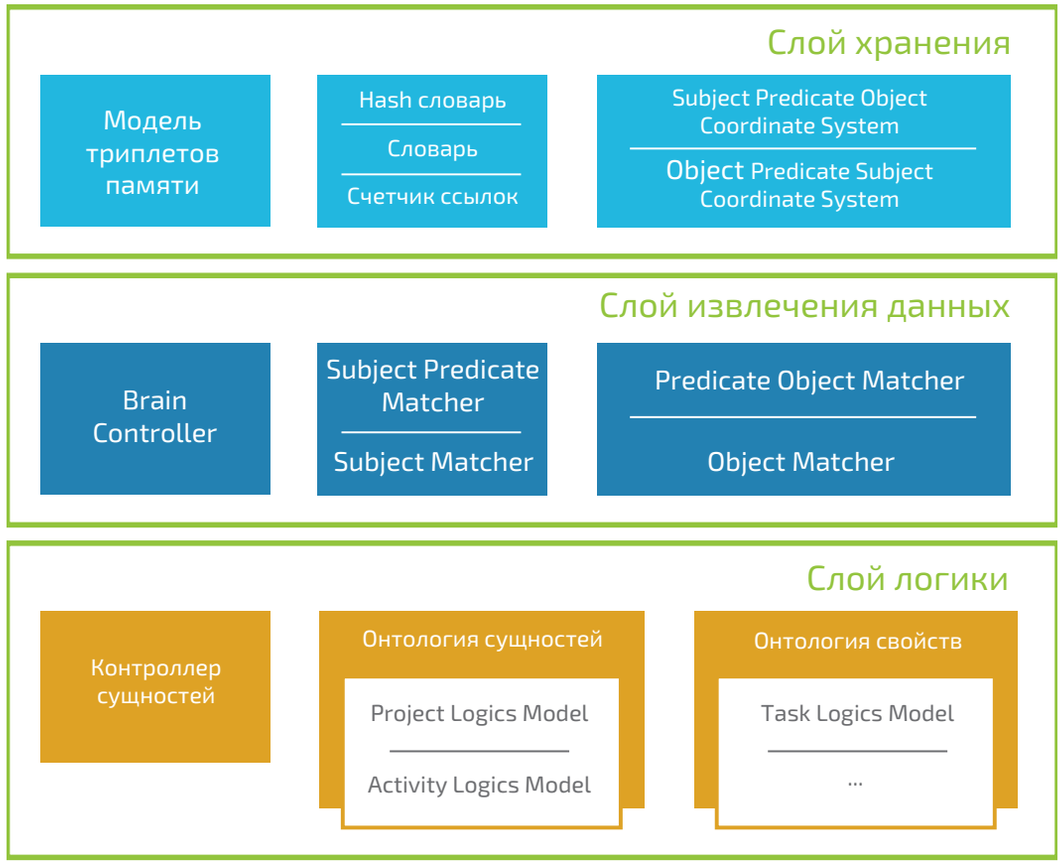
2.1.1. Matcher Sequence Processor — функционал, позволяющий обрабатывать совпадения последовательно, используя выходные параметры от предыдущих совпадений в последовательности в качестве входных параметров для следующих совпадений в последовательности. При этом поддерживаются рекурсивная обработка (например, на запрос «Что любит Петр из того, что Джейсон также любит?» результатом будет «рыбалка») и фильтрация (то есть мы можем добавлять дополнительные условия для совпадений, например такие: «Кто любит рыбалку и имеет номер телефона, начинающийся с кода «12»?»).

2.2. Subject Predicate Matcher — позволяет пользователю извлекать данные, основанные на известных подлежащем и сказуемом. Например, на основе фактов из пункта 1.5 мы можем спросить Brain Controller: «Что Петр любит?» Или, если говорить более формально, передать параметры: «Петр» и «любит» в Subject Predicate Matcher, который будет собирать все факты, содержащие «Петр любит», в качестве подлежащего и сказуемого. В нашем случае результат будет «Петр любит автомобили и рыбалку».

2.3. Predicate Object Matcher — другие типы совпадений, отвечающие на такие вопросы, как «Кто любит рыбалку?». В данном случае сказуемое «любит» и дополнение «рыбалка» известны, а подлежащее неизвестно. Результатом будет «Петр и Джейсон любят рыбалку».

Рисунок 24.

N3 клиентская СУБД



2.4. Subject Matcher — компонент, сходный с вышеперечисленными, но в качестве входного параметра использует только подлежащее.

2.5. Object Matcher — то же, что и Subject Matcher, но для дополнений.

3. Слой логики содержит логику приложения, а именно: правила, ограничения, описания бизнес-объектов и свойства. Основными компонентами являются:

3.1. Контроллер сущностей — предоставляет открытый интерфейс для создания, обновления, удаления и извлечения данных для бизнес-объектов. Работает с онтологией сущностей, онтологией свойств, моделью триплетов памяти, Brain Controller, а также с различными моделями бизнес-объектов.

3.2. Онтология сущностей — древовидная структура, которая описывает свойства бизнес-объектов и отношения между объектами. Например, бизнес-объект «Проект» также является основной сущностью бизнес-объекта, таким образом, наследует все его свойства, имеющие особые, определенные только для него свойства.

3.3. Онтология свойств — каталог всех доступных свойств в системе с соответствующими им типами, форматами и так далее. Например: «Дата начала» — свойство типа «дата», имеющее формат «месяц.день.год».

3.4. Task Logics Model — структура, содержащая правила и ограничения для установки и получения свойств задач бизнес-объекта. Например, правило может быть следующим: «Если дата завершения задачи отодвигается, при этом задача является частью проекта, то, соответственно, следует отодвинуть дату окончания проекта».

3.5. Project Logics Model, Activity Logics Model и так далее аналогичны описанной выше Task Logics Model.

Обзор системных объектов

Модель системных объектов полностью отражает модель компонентов, описанную выше.

При получении запроса к одному из методов назначенный контроллер выполняет требуемое действие и выдает необходимый результат. Результаты могут быть представлены в любой форме, необходимой для дальнейшей обработки: например, в виде деревьев, графиков, простых списков, таблиц, JSON-объектов или любого произвольного формата, описанного пользователем.

Запуск приложения

Описание

Приложение — это объект системы, который включает в себя модули и позволяет им общаться друг с другом через обмен сообщениями. На одной странице может быть одно или несколько приложений. В системе существуют следующие приложения:

- рабочий стол;
- администрирование;
- рабочая область.

Модуль — это объект системы, предоставляющий пользователям системы специальные функциональные возможности. Модули могут содержаться в приложениях системы.

Панель — визуальная часть модуля или приложения, обычно содержащая один или несколько элементов управления. Например, в приложении слева находится панель навигации, которая содержит элементы приложения: списки, панели мониторинга, специальные группы и рабочие области.

У всех модулей, доступных пользователям, есть собственная панель управления, которая находится вверху приложения и отображает доступные команды.

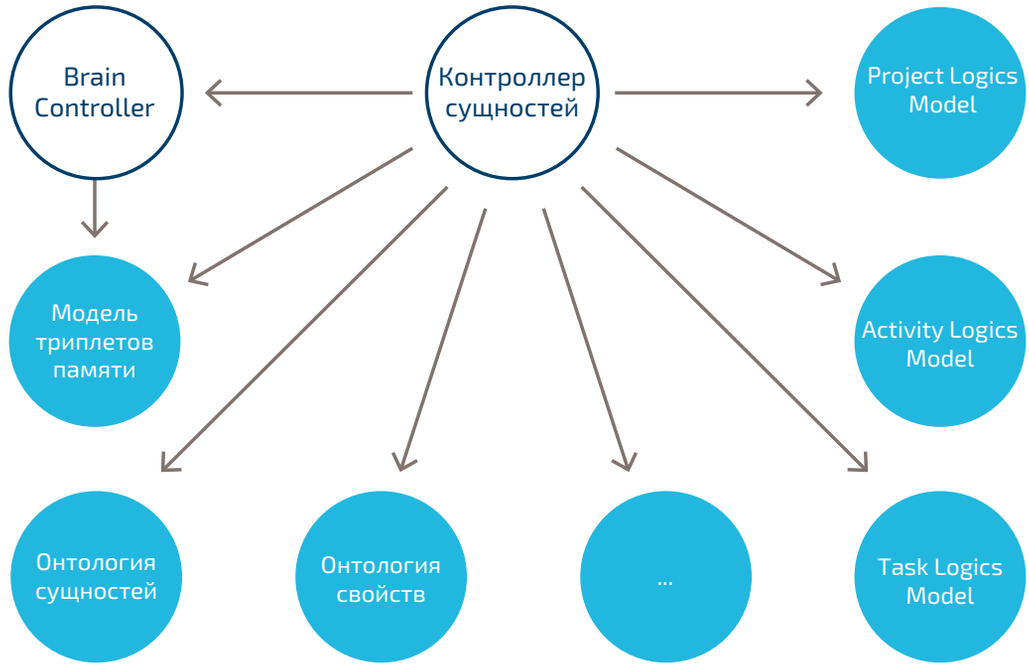
Application Manager — центральный объект системы, который подготавливает данные и инициализирует приложения и модули на стороне клиента.

Цели:

1. Реализация общих функций веб-приложения в одностраничном режиме (история, переход по ссылкам и другие).

Рисунок 25.

Обзор системных объектов



2. Подготовка и предоставление общей функциональности для модулей.
3. Организация функциональности на стороне клиента в повторно используемых модулях.

Последовательность

Запуск приложения состоит из трех основных этапов, каждый из которых исполняет Application Manager:

1. Инициализация.
2. Подготовка области просмотра.
3. Запуск приложения.

Инициализация — общий процесс, состоящий из следующих частей:

1. Общие утилиты — загружаются модули и функциональные возможности, необходимые для общего пользования.
2. Локализация — с сервера запрашивается текущий словарь локализации для дальнейшего использования в генерации HTML-кода.
3. Server API — загружается конфигурация AJAX методов, доступных в приложении, добавляются проверки и возможные ограничения. После этого Application Manager по конфигурации методов генерирует классы, используемые различными модулями.

Подготовка области просмотра — на этом этапе по полученной с сервера конфигурации подготавливается общий вид приложения:

1. Запрос данных — на этом этапе Application Manager получает общие данные с сервера: данные о текущем пользователе и его роли,

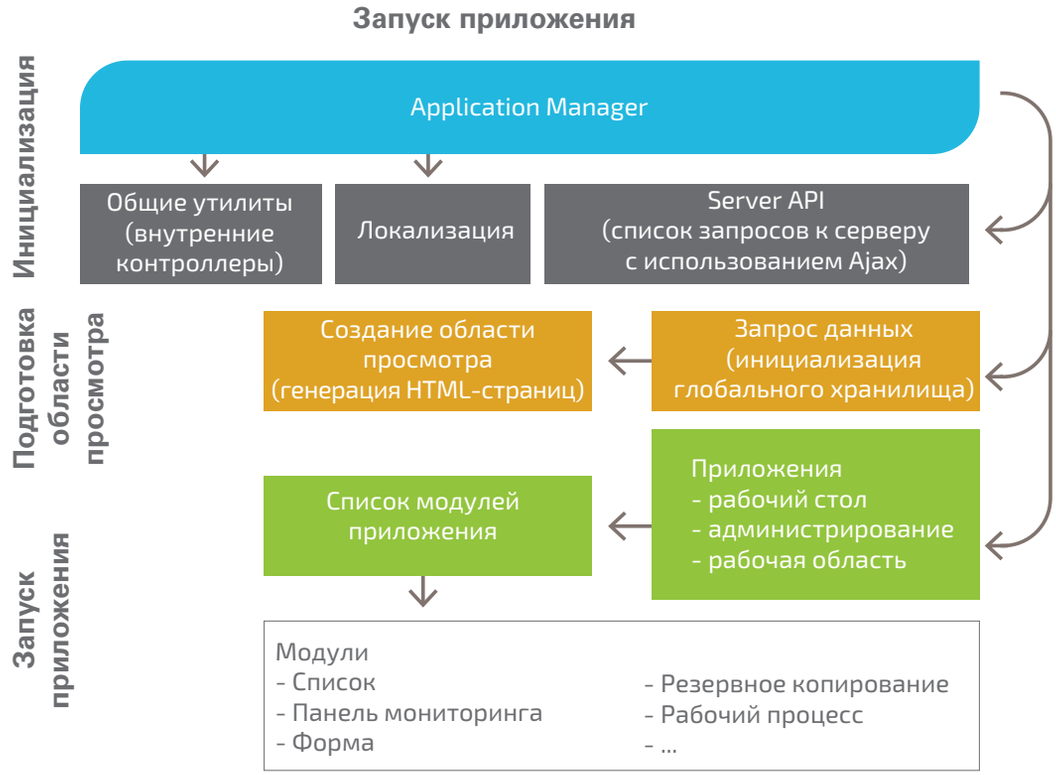


Рисунок 26.

Запуск приложения

конфигурацию системы, информацию об установленных продуктах и другие.

2. Создание области просмотра — применяются ранее полученные данные, генерируется HTML-код и добавляется на страницу.

Запуск приложения — на данном шаге Application Manager инициализирует доступные приложения. В зависимости от конфигурации сервера, роли пользователя и адреса страницы могут быть загружены различные приложения: рабочий стол, администрирование, рабочая область и другие. Каждое приложение может содержать различный список модулей. Когда приложение готово для выполнения, Application Manager запускает его.

Перспективы развития платформы

Благодаря полной поддержке онтологии во всех архитектурных компонентах платформы реализацию хранения в графовой СУБД становится возможным осуществить посредством архитектурного решения «УБ-Мангуст». Разработка конкретных решений для компонентов «УБ-Мангуст» является перспективной задачей для компании Comindware.

Ода онтологии. Вместо заключения

Итак, мы познакомили вас с нашим видением текущих проблем и предложениями по созданию архитектуры цифровой платформы для онтологических информационных систем завтрашнего дня. Мы старались поделиться не только результатом нашей изобретательской активности, но и его составляющими: научной и методической основой, промежуточными результатами и находками. В заключение хотим особо выделить онтологию как ключевое звено и одно из базовых понятий информационных технологий будущего.

Онтология как портал

Онтология, на наш взгляд, это «портал», обеспечивающий беспрепятственный переход между миром реальным и цифровым. Как только цифровые платформы и построенные на их основе информационные системы «научатся программировать» онтологии, взаимосвязь между бизнесом и ИТ, которой сегодня уделяется особое внимание, будет отдана из человеческих рук в «руки» онтологических ИС. В результате существенно изменится весь ИТ-мир.

Онтология и ИТ

Что представляет собой традиционный подход к созданию информационных систем для управления предприятием? Составляется модель бизнес-процессов, подбираются (зачастую вместе с консультантами) соответствующие программные компоненты, они поставляются в частный ЦОД или предоставляются в «облаке» с набором преднастроенных лучших

практик, которые затем адаптируются к нуждам конкретного предприятия и дополняются объемом программного кода для тех особенностей, которые не входили в набор преднастроенных практик.

Процесс развития такой системы — это программирование, закупка новых модулей и систем с набором лучших практик и так далее. Если платформа «понимает» онтологии, то для создания ИС нужен только первый шаг — составление модели бизнес-процессов[3]. Описываемый в онтологической платформе бизнес-процесс сразу «создает» требуемую информационную систему.

Предварительное программирование лучших практик заменяется на составление онтологических шаблонов, а развертывание и подготовка к внедрению системы — на загрузку в платформу конкретной онтологической модели, по которой будут созданы необходимые У-Блоки и репозиторий предварительно созданных «Мангустов»[4].

Затем наступает период органического роста, когда информационная система развивается вместе с процессами на предприятии, в дальнейшем потребуется оптимизация, реинжиниринг, но не программ и приложений, а онтологических моделей и созданных на ее основе У-Блоков и «Мангустов». Таким образом, мир ИТ сильно изменится, а кто из сегодняшних ИТ-гигантов будет способен на кардинальные перемены, покажет время.

Онтология и рынок труда

Чтобы грамотно и профессионально работать с онтологическими моделями (создавать, оптимизировать), необходима новая профессия — онтологист. Он будет работать в тесной связке с «традиционным» ИТ-специалистом, который должен ориентироваться в процессе того, каким образом онтология создает и модифицирует информационную систему. Следовательно, значение и роль ИТ-специалиста понижается до уровня техника, обслуживающего «механизм».

Онтологист как профессия будет иметь несколько специализаций: по отраслям промышленности, сферы услуг и госуправления, по областям взаимодействия («интернет вещей», инфороботы и другие системы искусственного интеллекта в процессе взаимодействия с человеком), по специализации бизнес-сервисов (операционные процессы и управление, системы принятия решений и поддержки принятия решений, экспертные системы). Благодаря тесной связи бизнеса и ИТ в последние годы появление новой профессии

[3] Модель бизнес-процесса здесь предполагается в самом полном виде: связывающей производственные объекты, функции, исполнителей и оргструктуру, среду реализации, обрабатываемые документы и так далее.

[4] Кстати, идею развития системы из универсального зародыша (платформы) по специфичным моделям мы «подсмотрели» у братьев Стругацких в романе «Полдень, XXII век» (глава 4 «Поражение»).

не будет болезненным для рынка труда: произойдет обособление и развитие частично имеющихся компетенций у ИТ-специалистов и менеджеров.

Онтология и образование

В связи с появлением новой профессии и областей специализации сфере образования потребуется освоить предметную область, создать соответствующую модель компетенций и образовательные программы, подготовить методические и учебные материалы, организовать переподготовку преподавателей. Начало уже положено: существуют значительные наработки в программах по моделированию бизнес-процессов, в курсах системной инженерии.

Другой аспект влияния онтологии на образование — использование образовательных платформ, построенных на онтологических моделях, которые способны обеспечить максимальную адаптивность индивидуальных образовательных траекторий при сохранении целостности результата и соблюдении дидактических принципов. Такой подход требует совершенно иного процесса организации образовательной деятельности и управления ею, а также новых методических решений. Поэтому сферу образования ждут большие перемены.

Онтология и роботизация

Роботизация как ведущая тенденция века имеет свои особенности. Создается множество различных роботов для многих сфер жизни, в том числе бестелесных инфороботов с разным уровнем искусственного интеллекта.

Одна из существенных проблем роботизации — создание роботов, обладающих способностью «осознавать» контекст. В настоящее время роботу недостаточно выполнять лишь основную функцию, ради которой он создан. Он должен «осознавать», как его действия вписываются в окружающую действительность, то есть знать и понимать контекст действия.

Робот-пылесос хорошо умеет собирать пыль с определенной области пола под своим корпусом, но для успешной уборки квартиры ему надо знать расположение комнат, рельеф пола, распознавать домашних животных и лежащие на полу предметы, определять, открыты ли двери, своевременна ли уборка и многое другое.

Аналогичные примеры можно привести про роботов-водителей (именно «непонимание» контекста сдерживает их активное использование), а также инфороботов, которые при неспособности «понимать» контекст, например при осуществлении учетных операций, будут требовать слишком частого человеческого вмешательства. Проблема состоит в том, что контекст является постоянно изменяющейся частью действительности, в отличие от основных действий роботов. Именно онтологические системы способны обеспечить робота достаточным уровнем понимания контекста.

Онтологические системы могут обмениваться онтологиями и даже сформировать единую распределенную онтологию как отражение действительности (по аналогии с фантастическим Скайнетом), предоставляя роботам необходимую в данном конкретном случае контекстную информацию — результат выполнения соответствующего «Мангуста». Робот-пылесос получает контекст от онтологической системы, управляющей «умным домом», а робот-водитель, например, от системы управления траффиком (переключения светофоров и отслеживания загрузки улиц)[5].

Онтология и человек

Мы верим в светлое будущее, в то, что порабощающее воздействие техники на человека и общество будет успешно преодолено, а жизнь существенно улучшится. В то, что онтологические цифровые платформы будут способствовать тому, чтобы человек был меньше занят рутинной и ручной работой, люди были образованнее, получили возможность больше общаться друг с другом, трудиться созидательно, эффективно и интересно, жить и отдыхать с комфортом.

Мы видим существенный вклад онтологии в появление нового системного свойства цифровой трансформации экономики, обусловленного возникновением ранее не существовавшего феномена «виртуального отражения реального мира». Это свойство обеспечивает получение ценностей, недостижимых традиционными способами.

Появляются новые бизнес-ценности (цифровые продукты и услуги), социальные ценности (признание, статус, самореализация через возможность проявить себя в цифровом мире), общечеловеческие (свобода и благополучие благодаря возможности трудиться онлайн в любом месте в удобное время) и многие другие, а барьеры на пути их достижения существенно снижаются.

Главное системное свойство «цифрового мира» и онтологии как его части можно метафорически сформулировать как «феномен волшебной палочки»: человек высказал желание — и мгновенно получил желаемое, оплатив его тоже мгновенно.

Удивительное будущее открывает перед нами онтология, не так ли?

[5] Интересные размышления касательно роботов и контекста можно найти в статье антрополога Ильи Утехина «Взаимодействие с «умными вещами»: введение в проблематику», журнал «Антропологический форум», № 17.

Приложение 1. Эмержентная стратификация информационных систем

Здесь описывается научно обоснованный подход к построению информационных систем, получивший название эмержентной стратификации информационных систем (ЭСИС).

Этот подход был опробован в программе трансформации ИС крупного нефтегазового предприятия. В рамках данной программы с опорой на указанный подход были определены направления трансформации, выбраны нужные архитектурные стили, разработана программа проектных мероприятий по переходу предприятия на новую технологическую платформу. Программа успешно выполняется.

Описанный научный подход и практический опыт были использованы нами при разработке концепции информационной системы нового поколения.

Математические и логические основы стратификации

Читатели, которых не интересуют математические аспекты этого подхода, могут без ущерба для общего понимания сразу перейти к разделу, следующему за этим.

Формализация стратифицированных систем вводится в [14] следующим образом.

Система S преобразует множество внешних стимулов X в множество откликов Y:

$$S : X \to Y.$$

Множество стимулов X и множество откликов Y задаются декартовыми произведениями множеств X_i и Y_i, которые представляют собой множества стимулов и откликов для i-й страты ($1 \le i \le n$, где n — количество страт):

$$X = X_1 \times \ldots \times X_n \text{ и } Y = Y_1 \times \ldots \times Y_n.$$

Каждая пара (X_i, Y_i) приписывается определенной страте, i-я страта системы S — это система, представленная как отображение S_i:

$$1)\ S_i : X_i \times W_i \to Y_i, \text{ если } i = n,$$
$$2)\ S_i : X_i \times E_i \times W_i \to Y_i, \text{ если } 1 < i < n,$$
$$3)\ S_i : X_i \times E_i \to Y_i, \text{ если } i = 1.$$

E_i и W_i представляют собой множества стимулов, исходящих от страт, и примыкающих к страте соответственно сверху и снизу. Они задаются отношениями h_i и c_i — это соответственно информационная и распределительная функции i-й страты.

$$h_i : Y_i \to W_{(i+1)},\ c_i : Y_i \to E_{(i-1)},\ 1 \le i \le n.$$

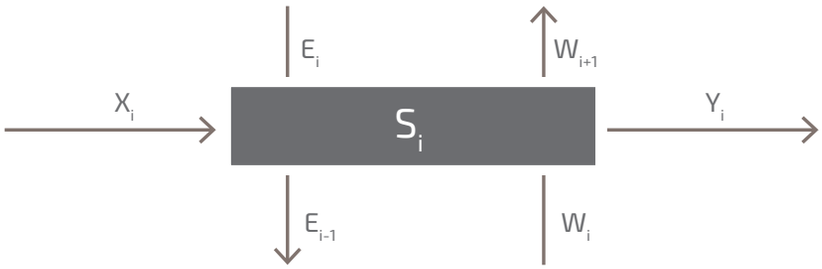


Рисунок 27.

Стимулы и отклики i-й страты

Схематически страта с ее стимулами и откликами изображена на рисунке 27.

На основании такого определения можно применить качественно различные подходы к стратифицированию систем, поэтому [14] предложен математический аппарат для нахождения степени стратификации.

Определено, что система может быть полностью стратифицированной и устойчиво стратифицированной.

В полностью стратифицированной системе S каждая страта S_i, $1 \le i \le n$, такова, что для любой пары (e_i, w_i) из E_i x W_i и любых двух элементов x_i и x'_i из X_i:

$$h_i (S_i (x_i, y_i, w_{ij}) = h_i (S_i (x'_i, y_i, w_i)),$$
$$c_i (S_i (x_i, y_i, e_{ij}) = c_i (S_i (x'_i, y_i, e_i)).$$

А для устойчиво стратифицированной системы не требуется независимость страт для любой пары «воздействие — обратная связь», но считается

необходимым существование некоторых состояний, для которых отклики оказываются локализованными в соответствующих стратах.

Также в работе предложены следующие характеристики стратифицированного описания систем:

1. Выбор страт, в терминах которых описывается данная система, зависит от наблюдателя, его знаний и заинтересованности в деятельности системы.
2. Аспекты описания функционирования системы на различных стратах в общем случае не связаны между собой, поэтому принципы и законы, используемые для характеристики системы на любой страте, в общем случае не могут быть выведены из принципов, используемых на других стратах.
3. Существует асимметричная зависимость между условиями функционирования системы на различных стратах. Требования, предъявляемые к работе системы на любой страте, выступают как условия или ограничения деятельности на нижестоящих стратах.
4. На каждой страте имеется свой собственный набор терминов, концепций и принципов. То, что является объектом рассмотрения на данной страте, более подробно раскрывается на страте, расположенной ниже; элемент становится набором; подсистема данной страты является системой для нижестоящей (рисунок 29; [14], с. 61).
5. Понимание назначения системы возрастает при последовательном переходе от одной страты к другой: чем ниже мы спускаемся по иерархии, тем более детальным становится раскрытие системы; чем выше поднимаемся, тем яснее становится смысл и значение всей системы.

Предложенные характеристики можно рассматривать как принципы стратификации:

- принцип целенаправленности описания;
- принцип независимости описания;
- принцип зависимости требований сверху вниз;
- принцип иерархической вложенности;
- принцип повышения детализации сверху вниз.

Исходя из принципа иерархической вложенности и независимости описания, можно сделать предположение, что наиболее эффективным подходом к построению полностью стратифицированной модели системы будет определение эмержентных свойств по подсистемам и разделение на уровни абстрагирования (страты) соответственно группировке наиболее четко дифференцируемых эмержентных свойств.

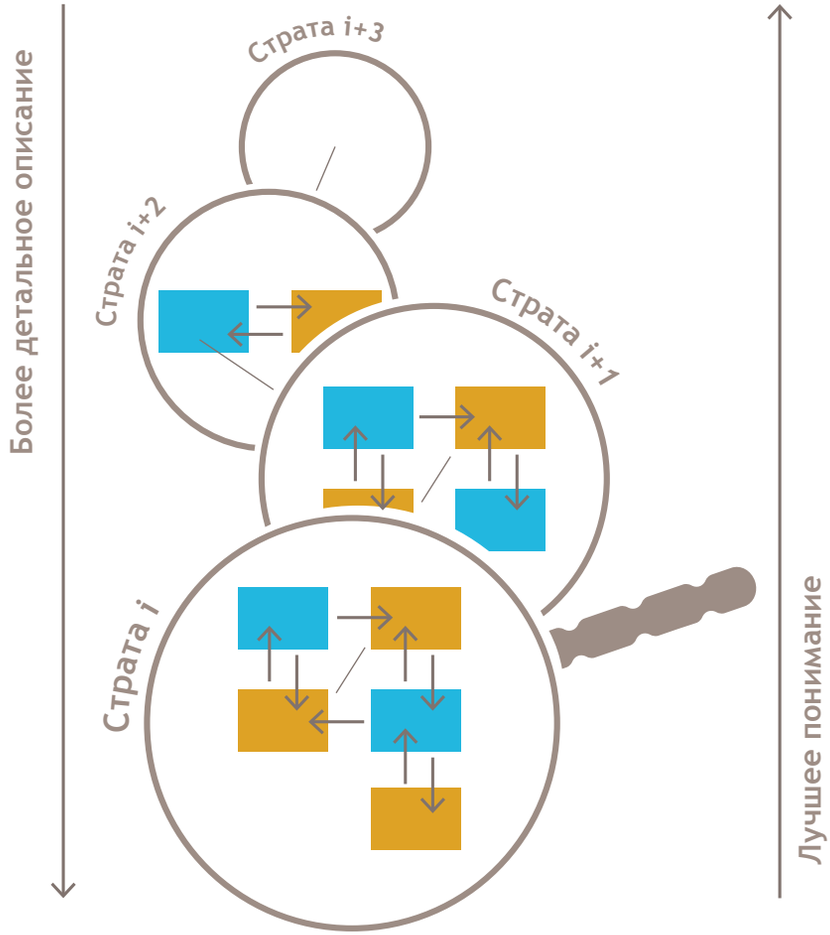


Рисунок 28.

Взаимосвязь между стратами

Говоря о понятии «эмержентность», следует обратиться к известному определению: «Система — это совокупность материальных и информационных объектов, взаимодействующих между собой для достижения общей цели; она обладает системным свойством, которыми отдельные объекты системы не обладают. Оно появляется только в результате целевого объединения (взаимодействия) этих объектов. Это свойство *эмержентности* (от англ. «emerge» — «возникать, появляться»)».

Можно привести другие определения этого свойства:

- Эмержентность — степень несводимости свойств системы к свойствам элементов, из которых она состоит.
- Эмержентность — свойство систем, обусловливающее появление новых свойств и качеств, не присущих элементам, входящим в состав системы.
- Эмержентность — принцип, противоположный редукционизму (редукционизм утверждает, что целое можно изучать, расчленив его на части, и затем, определяя их свойства, определить свойства целого).

Назовем это эмержентным подходом к стратификации.

Существующие подходы к стратификации

Для информационных систем до сих пор не было предложено четких критериев, по которым необходимо определять страты, и, соответственно, невозможно было определить степень стратификации.

По цели создания информационных систем (управление бизнесом) верхние страты должны отражать экономическую модель предприятия. По физической реализации нижние страты представляют собой программно-аппаратную инфраструктуру.

Рассмотрим некоторые из имеющихся в научной литературе моделей, которые можно рассматривать как стратифицированное описание.

Различные варианты структурирования по экономическим стратам предлагаются в работах В. М. Глушкова [25], И. Н. Синицына [26], А. Остервальдера [27].

По стратам, имеющим непосредственное отношение к информационной системе, в отечественной и зарубежной литературе существуют различные подходы, например:

- экономические факторы, обработка информации и управление, физические процессы;
- бизнес-архитектура, логическая архитектура, технологическая архитектура;
- бизнес-архитектура, ИТ-архитектура, архитектура данных, программная архитектура, техническая архитектура.

Структурирование в вышеприведенных примерах выполнено эмпирически, путем фиксации сложившейся к определенному моменту практики, поскольку при этом не стояла задача выполнить стратификацию.

Таким образом, до сих пор не существует стройной стратифицированной модели информационной системы крупного предприятия с обозначенными входами и выходами каждой страты, с взаимосвязями между стратами, с оценкой степени стратификации.

Опираясь на описанный выше подход к стратификации, можно предложить научно обоснованную базу для построения моделей информационных систем — эмержентную стратификацию информационных систем (ЭСИС).

Пять уровней стратифицированной ИС

Ниже предлагается вариант стратификации ИС с использованием эмержентного подхода. При этом следует учесть, что построение детальной стратифицированной модели с поэлементным описанием каждого уровня каждой страты — отдельная сложная задача, с учетом многообразия архитектурных подходов.

Порядок уровней стратификации указан по мере развития ИС и усложнения обрабатываемой информации от простого к сложному, от сигналов к стратегии бизнеса.

1. На самом нижнем уровне находится страта машинных кодов и аппаратного обеспечения. Это сигналы, команды ассемблера, данные в виде битов и байтов. Эмержентное свойство страты (ЭСС) здесь — обработка данных, а потребитель информации — техник (электронщик или системщик).
2. Вторая снизу страта — информация как осмысленные данные: это информационные модели, описывающие структуру информации (например, ER-модель «сущность — связь»), информация имеет транзакционный характер, живет в базах данных (БД) и в системах управления этими БД. ЭСС в данном случае — это обработка информации (загрузка, преобразование, хранение, представление).
3. Третья страта — информационные сервисы, например в форме программных приложений (software applications), содержащие деловую логику и обеспечивающие нужную функциональность конечному пользователю сегодняшней ИС. Именно в рамках этой страты пользователи воспринимают информационную систему «своей», они работают в рамках составляющих ее приложений. ЭСС здесь — это выполнение инфосервисов для бизнеса (бизнес-процессов, аналитики, поддержки принятия решений, в том числе через управление знаниями и методы искусственного интеллекта).
4. Четвертая страта — это модель бизнеса в целом, ведение текущей деятельности предприятия/организации. ЭСС здесь носит пока «неайтишный» характер, поскольку выражается в экономических понятиях (прибыль, доходность, доля рынка). ИТ сюда только-только начинают проникать из области поддержки принятия решений, стараясь предложить цифровые бизнес-модели. На этой страте начинается «цифровизация», отличающаяся от «информатизации» на предыдущих стратах.
5. На пятой страте находятся модели развития бизнеса, затрагивающие цели существования предприятия/организации. Эта страта —

Рисунок 29.

Обычная стратифицированная ИС

результирующая, отвечающая на вопрос: зачем нужны все остальные страты? ЭСС — миссия, видение, стратегия, цели. ИТ на этой страте пока совсем не представлены.

Иллюстрация уровней стратификации дана на рисунке 29.

Принципы стратификации ИС

При эмержентной стратификации ИС принципы стратификации выполняются следующим образом:

1. Принцип целенаправленности описания. Стратифицированное описание ИС выполнено в основном с айтишной точки зрения, содержит как назначение системы на пятой страте, так и последовательное детальное ее описание до аппаратных компонентов.
2. Принцип независимости описания. На каждой страте действуют свои законы и принципы: физические на первой страте, информационные на второй, процессные на третьей, экономические на четвертой, стратегические на пятой. То же самое можно сказать о применении различных типов моделей, информационных и математических, для каждой страты в отдельности.
3. Принцип зависимости требований сверху вниз. Объемные показатели эмержентных свойств каждой страты являются главными входными требованиями для определения конкретной модели нижестоящей страты. Для четвертой страты цели деятельности компании определяют

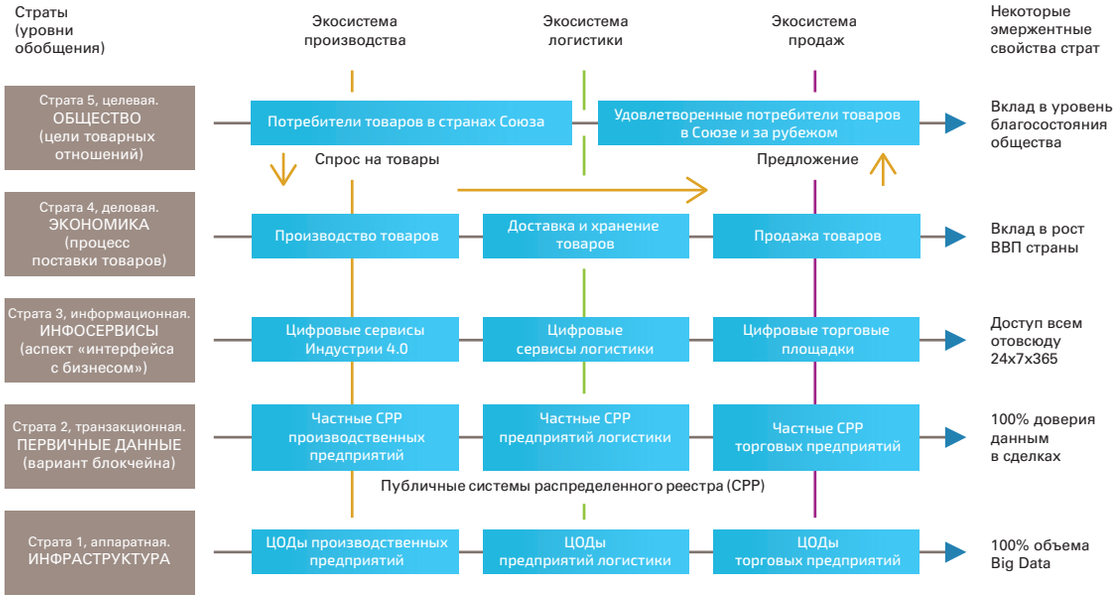


Рисунок 30.

Эмержентная стратификация цифровой торговли (пример)

бизнес-модели; для третьей входными требованиями являются бизнес-процессы предприятия четвертой страты; для второй входными требованиями являются взаимосвязанные ИТ-решения; в свою очередь, количество, объем и скорость работы баз данных второй страты являются основанием для определения конфигурации и расчета производительности оборудования первой страты.

4. Принцип иерархической вложенности. Строгое следование этому принципу возможно, но не обязательно будет реализовано при эмержентном подходе. Для сложных на всех стратах моделях однозначного отношения 1:n вряд ли удастся достигнуть, однако общие подходы к реализации принципа сохраняются: цели определяют бизнес-модели, конкретные бизнес-процессы реализуются с помощью ИТ-решений, которые определяют объекты данных и программные процедуры, реализуемые конкретной СУБД, которая, в свою очередь, запускается на конкретном составе оборудования.

5. Принцип повышения детализации сверху вниз. Количество объектов, описывающих систему в целом, на каждой нижестоящей страте до первой увеличивается: если на пятой страте десятки целей определяют около тысячи бизнес-процессов четвертой страты, то на третьей страте мы получим несколько тысяч приложений, на второй — более миллиона объектов данных и процедур; однако на первой страте будет меньшее число аппаратных составляющих — около тысячи, что свидетельствует о том, что это граница стратификации, являющаяся верхним уровнем другого стратифицированного описания (от кластеров и серверов

до транзисторных переходов). Аналогично следующей за пятой стратой в сторону укрупнения будет страта отрасли или рынка, содержащая миллионы предприятий, что также говорит о достигнутой границе.

Таким образом, для эмержентной стратификации информационных систем наиболее актуальными являются первые три принципа: целенаправленности, независимости описания, зависимости требований сверху вниз.

Интеграция эмержентного подхода и семиотики

Для того чтобы убедиться, работает ли такой подход в реальной жизни, давайте проследим эволюцию понятия «информация» на протяжении последних пятидесяти лет, учитывая предположение, что страты развиваются снизу вверх.

Как известно, в настоящее время существует множество определений, что же такое информация, и они часто не стыкуются между собой.

Например, традиционные специалисты по вычислительной технике (воспитанные в свое время на больших ЭВМ) воспринимают данные как «записанную информацию».

В современной информатике, наоборот, считают, что данные — это еще не информация, а просто сигналы, биты и байты, не передающие смысл информации. Информация сейчас, на второй страте,— это «осмысленные данные», причем следующий, более высокий уровень информации — это уже «знания». Наиболее часто используется такое определение: «Информация — это осознанные сведения об окружающем мире, которые являются объектом хранения, преобразования, передачи и использования».

Это говорит о том, что общество в целом поднялось в понимании информации выше по стратам, а технические специалисты, работающие с данными нижнего уровня, трактуют ее по-прежнему в понятиях первой страты.

В ТРИЗ описан «закон вытеснения человека», в соответствии с которым человек постепенно становится лишним звеном в любой развитой системе; «человека нет, а его функции выполняются». Такое вытеснение происходит снизу вверх: сначала человек вытесняется из простых рабочих операций (например, роботизированным конвейером), затем из бизнес-процессов (например, автоматизацией документооборота) и наконец из системы управления (например, аналитическими системами с поддержкой принятия управленческих решений).

То есть по мере развития ИТ человек стал замещать часть своих функций информационной системой.

В наше время на первой (аппаратной) страте замещение составляет 99%, там практически все автоматизировано: можно сказать, что в мире осталось менее одного процента программистов на ассемблере.

Развитие шло снизу вверх, и со второй страты появлялись языки все более высокого уровня, на третьей страте уже созданы языки программирования бизнес-объектов (например, ABAP). Для четвертой соответствующих языков пока нет (например, есть языки онтологического моделирования, но нет языков онтологического программирования и соответствующих компиляторов или интерпретаторов).

Поиски современного научного понимания, что же такое информация на сегодняшнем уровне развития ИТ, привели нас к семиотике. Казалось бы, семиотика — это наука о знаках, но для нее информация — безусловная основа, и оказалось, что в рамках этой науки было выработано вполне адекватное современным реалиям понимание и определение информации.

Семиотика рассматривает информацию в пяти аспектах (уровнях): статистическом, синтаксическом, семантическом, прагматическом, апобетическом [15]. Более подробно они рассмотрены в следующем разделе «Пять уровней информации».

Оказалось, что эти уровни информации точно соотносятся с пятью уровнями эмержентной стратификации! Каждый уровень информации нашел свое явное место на одной из пяти страт. То есть различные определения информации могут быть правильны, но в пределах одной, «своей» страты — и каждый из специалистов по-своему прав в своем определении, но в рамках одной страты.

Например, признанные «зубры» вычислительной техники правы в пределах первой страты — там информация носит статистический (числовой) характер; программисты, работающие с базами данных, уверены в правильности синтаксического характера информации на второй страте, и так далее.

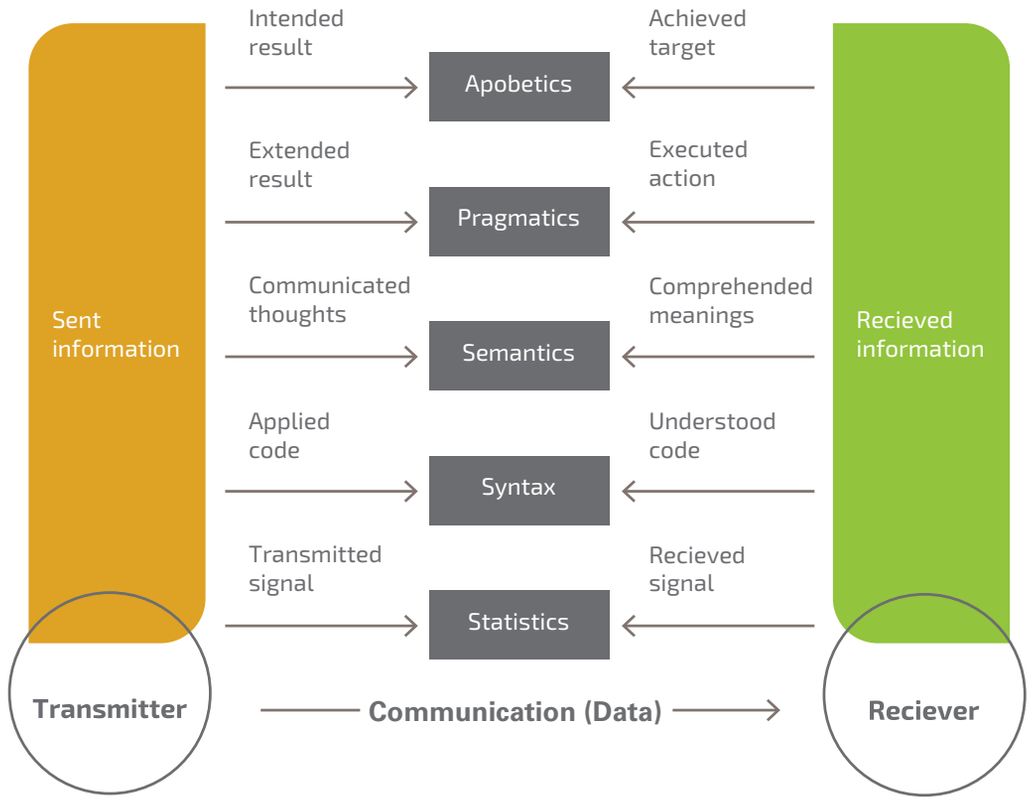
При этом подтверждается работоспособность принципа «независимости описаний»: на каждой страте можно применять свои термины и определения, и они не будут противоречить терминам, используемым на других стратах.

А в целом архитектура стратифицированной системы описывается некоторым множеством моделей, которые в совокупности и содержат интегральное понимание того, что есть информация.

Архитектурный подход, основанный на ЭСИС, гармонично сводит все разнообразие пониманий термина «информация» к единой логике. Каждое определение может найти свое место на соответствующей страте, в рамках той точки зрения, того холона и той модели, которые требуются конкретному заинтересованному лицу.

Рисунок 31.

Пять уровней информации [6]



Таким образом, научную основу нового архитектурного подхода можно свести к интеграции двух взаимодополняющих понятий:

- эмержентная стратификация (пять уровней систем);
- семиотический подход (пять уровней информации).

Семиотический подход, описывающий пять уровней информации, излагается ниже.

Пять уровней информации

Сложность понятия информации и многообразие использования этого термина в настоящее время наиболее полно отражены в семиотическом подходе, который рассматривает информацию в пяти аспектах (уровнях): статистическом, синтаксическом, семантическом, прагматическом, апобетическом [15].

Семиотический подход используется не только лингвистами, но с успехом применяется для рассмотрения информации в экономических и информационных системах [28].

Рассмотрим особенности каждого уровня снизу вверх, от простого к сложному.

[6] Источник: http://www.matthias-dorn.de/tl_files/pdfs/dorn_knowledge_transfer_via_the_internet.pdf

Рисунок 32.

Пять уровней информации

Передача информации

На нижнем, статистическом уровне значение (смысл) полностью игнорируется. На этом уровне информация выражена в виде последовательности сигналов (да — нет, ноль — один, сильный — слабый). Мерой информации являются биты и байты, измеренные как объем всего потока данных.

На синтаксическом уровне передаются лексические и синтаксические аспекты. Информация выражена в виде набора символов, имеющих логическую структуру и синтаксические правила. Мерой информации также являются биты и байты, но не всего потока, а только в объеме деловой информации этого потока.

На семантическом уровне информация выражает значение, несет смысл. Каждый фрагмент информации связан с источником смысла (интеллектом отправителя, базой знаний, семантической моделью и так далее). Для обеспечения правильного понимания необходимо наличие у приемника словаря (например, глоссария, тезауруса, таксономии). Мерой информации на этом уровне также выступают биты и байты, но измеренные в объеме, воспринятом получателем, с учетом его тезауруса.

Приложение 1. Эмержентная стратификация информационных систем

№	Наименование уровня	Передано	Получено
5	Апобетика, целевой уровень (зачем?)	Предполагаемый целевой результат	Достигнутый целевой результат
4	Прагматика, уровень исполнения (как?)	Ожидаемое действие	Выполненное действие
3	Семантика, уровень передачи смысла (о чем?)	Выраженная мысль	Осмысленное понятие
2	Синтаксис, уровень передачи данных (что?)	Созданный символьный код	Воспринятый символьный код
1	Статистика, сигнальный уровень (есть — нет)	Переданный сигнал	Полученный сигнал

Таблица 3.

Пять уровней информации

На прагматическом уровне принимается во внимание то, как передаваемая информация используется на практике — например, какие действия выполнит получатель после ее получения. На этом уровне учитывается вклад контекста в смысловое значение. Мерой прагматического аспекта является уже не информация, а единицы измерения ценности, получаемой по окончании процесса на этом уровне. Часто прагматической мерой информации выступают меры стоимости. Прагматическая мера информации существенным образом зависит от действий получателя информации (например, лица, принимающего решение).

Апобетический уровень

Он описывает конечную цель передачи информации, то есть эффект, который отправитель ждет от получателя. Это похоже на прагматику, но имеет другую цель. Это касается уже не просто действий, а скорее мотивов, на которых основаны действия. Первые четыре уровня описывают передачу информации, а апобетический уровень объясняет, *почему* эта информация должна быть передана (зачем она нужна).

Апобетический уровень информации является наиболее важным, так как он отражает намерения отправителя. Информация, циркулирующая на четырех нижних уровнях, необходима как средство осуществления этого намерения.

Пример

Для примера рассмотрим вариант получения менеджером отчета о расположении товара на складе.

Статистически получено 10 килобайт информации.

С синтаксической точки зрения получен документ, содержащий объем деловой информации размером 5 килобайт: наименование материала, количество, стоимость, размещение.

С семантической точки зрения возможны следующие крайние ситуации:

а) менеджер не имеет знаний и опыта в области управления складом, тогда объем полученной семантической информации равен нулю;

б) менеджер уже знает точно и помнит, где и что у него лежит, тогда объем полученной информации также равен нулю;

в) менеджер умеет управлять складом и ожидает информацию о расположении товара, тогда объем семантической информации равен пяти килобайтам.

Конечно, возможны промежуточные ситуации — скажем, частичного знания о расположении товара,— тогда объем полученной семантической информации будет больше нуля, но меньше пяти килобайт.

С прагматической точки зрения можно подсчитать, сколько товара потенциально возможно разместить дополнительно в результате решения, принятого на основании полученного отчета, и тогда измерить прагматическую информацию количеством денег, сэкономленных на аренде определенной площади склада. Либо можно рассчитать вероятность принятия оптимального решения без полученной информации и с ее учетом вычислить ценность по формуле Харкевича — Бонгарда. Уточнить эти расчеты возможно на апобетическом уровне, потому что, по определению, мы должны знать, для какой цели информация была запрошена, и отследить, достигнута ли цель после получения информации.

Из примера видно, насколько трудно достоверно измерить прагматический и апобетический уровни информации: во-первых, потому, что они сильно зависят от предпринятых действий и их результативности, а во-вторых, потому, что трудно выбрать адекватную метрику и методику измерения. При этом статистически и синтаксически информация измеряется относительно легко, а для семантического аспекта необходимо каким-то образом учесть тезаурус приемника.

Информация и энтропия

Чтобы определить применимость эмержентной стратификации информационных систем, необходимо выбрать адекватные меры и способы измерения результативности. Для этого предлагается использовать меры неопределенности при принятии управленческих решений с помощью информационной системы.

В классических работах по кибернетике [29] и последующих исследованиях информации [30] подробно рассматривается энтропия как мера неопределенности, мера недостатка информации о рассматриваемой системе. Количество информация определяется по формуле Хартли или чаще Шеннона:

$$I = -\sum_{i=1}^{N} p_i \log_2 p_i,$$

где I — количество информации, N —- количество возможных событий, p_i — вероятность i-го события.

Это дает нам возможность рассматривать изменение количества информации как меру изменения неопределенности.

Для информационных систем большое значение имеет рассмотрение неопределенности при принятии управленческих решений, потому что ИС созданы для того, чтобы эту неопределенность снижать.

Другими словами, ИС нужна для того, чтобы дать информацию об объекте управления таким образом, чтобы неопределенность при принятии бизнес-решений уменьшилась.

Энтропия, как мера неопределенности в различии между двумя мирами — материальным миром и его виртуальным отражением, информационным миром,— должна снижаться по мере развития, в процессе эволюции.

Это коррелирует с подходом «Интернет как все более полная виртуальная копия реального мира» [31], [32], в котором изложена гипотеза о том, что каждый объект реального мира вскоре будет иметь своего «информационного двойника» в мире виртуальном. Это «интернет вещей», «интернет всего» и сопутствующие тренды виртуализации, которые в процессе конвергенции технологий постепенно эволюционируют, приближаясь к концепции «ноосферы». Можно трактовать это как формирование все более полной (и развивающейся) виртуальной копии реальности, информационного, цифрового аспекта материального мира.

Развитие информационных систем по стратам должно соответствовать этой тенденции.

Для того чтобы прогресс в развитии измерить количественно, надо понять, в чем же его измерять. В нашем случае четкие метрики по измерению степени неопределенности предоставляет семиотический подход к информации.

Семиотический подход при эмержентной стратификации

Нетрудно заметить, что стратификация информационных систем по их эмержентным свойствам образует страты, которые точно соответствуют различным уровням семиотического подхода к информации.

Так, *первой страте «Аппаратные ресурсы» соответствует статистический аспект*, мерой информации на котором являются биты, байты. Возможными метриками могут быть, с одной стороны, поток данных, поступающих в страту, например, по локальной сети. Можно измерить отдельно по каждому стимулу множества X1. С другой стороны, можно измерить объем обрабатываемых и хранимых данных, которые возможно измерить по каждому отклику из Y1, либо, если детализация не нужна, можно также измерить поток данных исходящего трафика по локальной сети.

Второй страте «Обработка информации» соответствует *синтаксический аспект*. Измерять здесь необходимо не данные вообще, а конкретную

деловую информацию, поступающую, хранящуюся, обрабатываемую системами управления базами данных. Метриками могут быть объемы занятого табличного пространства без учета системных, вспомогательных таблиц и отведенного под таблицу пустого объема, а также объем деловой информации, запрашиваемой и потребляемой приложениями,— например, результаты выполнения SQL-запросов.

Третьей страте «Приложения» соответствует *семантический аспект*. При этом для внедренных и используемых информационных систем можно принять коэффициент соответствия тезаурусу получателя равным 1, так как классический подход при создании приложений предполагает, что реализованная функциональность и размещаемые данные имеют для предприятия важное значение и производственный смысл. Когда сотрудник предприятия запускает приложение или запрашивает отчет, он компетентно выполняет должностные обязанности, следовательно, нуждается в запрашиваемой информации и способен ее понять и квалифицированно воспользоваться. Метриками могут быть выбраны как объем информации, вносимой в первичные и другие документы, так и объем информации в получаемых отчетах, выводимой на монитор (не монитор как устройство, а монитор как приложение, визуализирующее показатели).

Четвертой страте «Бизнес» соответствует *прагматический аспект*. И здесь для измерения информации необходимо разрабатывать или использовать готовые специализированные методики расчета экономической эффективности, применять ключевые показатели и другие подходы. Поскольку информационные технологии на этой страте еще используются слабо, предложить для прямого измерения адекватную и подготовленную информацию в информационных системах не представляется возможным. Поэтому наиболее применимым будет метод определения вероятностей решения.

Пятой страте «Стратегия» соответствует *апобетический аспект*.

Сочетание подходов и возможные метрики изложены в таблице 4.

Понятие холона и холархии

С эмержентной стратификацией хорошо согласуется общефилософское понятие *холона*. Термин был впервые введен Артуром Кёстлером (Arthur Koestler) в 1967 году в книге «Дух в машине» (The Ghost in the Machine). Термин «холон» был им сконструирован из греческого слова «холос», что означает «целое», с добавлением суффикса «-он», означающего «часть».

Таблица 4.

Страта	Эмержентные свойства	Уровень	Информация	Мера	Метрики (обобщенно)
Стратегия	**Достижение целей**	**Апобетический**	**Достигнутый результат**	**—**	**Результативность в достижении цели**
Бизнес	**Получение прибыли**	**Прагматический**	**Использованная информация**	**Рубли, натуральные единицы измерения**	**Ценность, вероятность оптимального результата**
Приложения	**Выполнение бизнес-сервисов (операций бизнес-процессов и поддержки принятия решений)**	**Семантический**	**Выраженное и воспринятое значение. Полученная информация, сопоставленная с тезаурусом**	**Биты, байты**	**Объем деловой информации, используемой в бизнес-сервисах**
Обработка информации	**Обработка, передача, хранение информации**	**Синтаксический**	**Символы, осмысленные данные — информация**	**Биты, байты**	**Объем информации в СУБД**
Аппаратные ресурсы	**Обработка, передача, хранение данных**	**Статистический**	**Сигналы без смысла, данные**	**Биты, байты**	**Объем данных**

Понятие холона и холархии было развито Кеном Уилбером (Kenneth Earl Wilber II) в его философском интегральном подходе [18].

В терминах современного системного подхода речь идет о вложенной системной иерархии «подсистема — система — надсистема».

Приведем развернутую цитату из работы [18] (с. 63–64), поскольку это будет важно для дальнейшего понимания нашего архитектурного подхода.

Цитата: «Ингредиентами иерархий являются холоны. Холон — это целостность, которая есть часть других целостностей. Например, целый атом — это часть целой молекулы; целая молекула — часть целой клетки; целая клетка — часть целого организма. Или опять же целая буква — часть целого слова, которое является частью целого предложения, которое является частью целого абзаца, и так далее. Реальность не состоит только лишь из целостностей или из частей, она состоит из целостностей/частей, или *холонов*. Реальность во всех измерениях, по существу, состоит из холонов.

Вот почему, как отметил Артур Кёстлер, иерархия развития — это на самом деле холархия, ведь она состоит из холонов: «Например, можно говорить о холархии от атомов до молекул, клеток и организмов… вот почему холоны являются основой холизма; они превращают груды чего-либо в целостности, которые являются частями других целостностей, и так до бесконечности… Космос есть последовательность «гнезд», вложенных в «гнезда» без конца и края, которая отражает в себе все более

холистический охват (холархии холонов повсюду). Вот почему у всех своя собственная холархия и почему любая холархия взаимосочетается и согласуется со всеми остальными».

Принцип холархии является универсальным для мироздания, включая как живые, так и искусственные системы.

Например, уровень живой клетки охватывает, превосходит и включает в себя нижележащий уровень клеточных подсистем (ядро, ядрышко, рибосому, цитоскелет, аппарат Гольджи, митохондрию и так далее), которые состоят из молекул, состоящих, в свою очередь, из атомов, и так далее по уровням материи.

Следуя этому подходу, мы говорим об архитектурной холархии стратифицированной информационной системы, где на каждой страте имеется свой базовый холон, который включает в себя и охватывает холон нижележащей страты, одновременно являясь включенным и охваченным в холон вышележащей страты. Холархия — это системная иерархия, в которой каждый иерархический уровень характеризуется системным свойством (эмержентностью), а носителем этого свойства является холон — базовый компонент данного уровня. Иными словами, в каждой страте имеется базовый холон, и его наличием определяется эмержентное свойство страты (ЭСС).

Например, в этих терминах холон второй страты — «Структурированная информация». Информация включает и охватывает холон нижележащей страты — «Данные в машинных кодах» — и одновременно является включенной и охваченной холоном вышележащей страты — «Информационные сервисы».

Холархия страт — это полная система. Каждая вышестоящая страта включает в себя, охватывает и превосходит нижестоящую страту.

Практически при разработке архитектуры стратифицированной ИС на каждой страте необходимо выделять базовый холон этой страты и его ЭСС.

Трансформация корпоративных ИС по стратам

Элементы ИТ-архитектуры в стратифицированной ИС

При построении модели информационной системы на основе эмержентной стратификации в соответствии с описанием страт и принципами стратификации для информационной системы типа ERP II можно распределить элементы ИТ-архитектуры следующим образом (см. таблицу 5).

Таблица 5.

Страты	Элементы
Страта 5. «Стратегия»	Дерево целей, миссия, набор стратегий (и стратагем), корпоративная культура [7]
Страта 4. «Бизнес»	Бизнес-модели, бизнес-процессы, бизнес-роли, функции, документы
Страта 3. «Приложения»	Прикладные системы — ERP, CRM, SRM, аналитические системы и хранилища — BI, BW, системы обеспечения и интеграции — MDM, порталы, компоненты обеспечения SOA. В составе прикладных систем — бизнес-объекты, функциональные модули, функциональные роли, пользователи
Страта 2. «Обработка информации»	Серверы управления базами данных, серверы приложений, базы данных. В составе баз данных — структура таблиц, запросы, «технические» роли
Страта 1. «Аппаратные ресурсы»	Физические серверы, системы хранения, сетевая инфраструктура

Подобное разделение позволяет без потери целостности рассматривать отдельные модели по стратам, при этом связи между стратами являются обеспечением нижестоящей стратой требований вышестоящей. Например, оценка видов и количества документов, шаги и интенсивность их обработки на страте «Бизнес» требует наличия определенных приложений в составе прикладных систем. Расчеты количества пользователей, объемов бизнес-объектов, функциональных модулей, их взаимодействия на страте «Приложения» предъявляют требования к размеру БД и производительности серверов БД и серверов приложений.

Свертывание в ИТ-архитектуре

При рассмотрении конкретных моделей будем ориентироваться на передовые решения, находящиеся в стадии массового продуктивного использования, то есть на этапе «Плато продуктивности» в модели цикла зрелости технологий Gartner Hype Cycle.

При поиске вариантов исключения задержек и выбора направлений для изменения моделей будем использовать результаты прогнозирования по ТРИЗ-методологии Directed Evolution [7]. Основной вывод прогноза заключается в том, что на данном этапе развития корпоративных информационных систем весьма результативно использование принципа свертывания.

Обозначим подходы к свертыванию с точки зрения ИТ-архитектуры.

Взаимодействие пользователя с корпоративной ИС базируется на ожидании быстрого ввода учетной информации и быстрого получения максимально актуальной информации, в том числе вычисляемой на основе только что

[7] Большинство практически реализованных ИТ-архитектур обходятся без элементов этой страты.

введенных данных. Временные задержки вызваны ожиданием ответа от системы, занятой обработкой и передачей данных между ее компонентами.

Можно выделить два основных аспекта архитектуры ИС, которые влияют на скорость обработки данных:

- Вертикальный: наличие разных звеньев, для которых необходимо организовывать взаимодействие и выделение вычислительных ресурсов. Например, ОС — СУБД — сервер приложений. Зачастую СУБД и серверы приложений намеренно разделяют для балансировки нагрузки, что приводит к необходимости выделения дополнительных вычислительных мощностей и сетевого оборудования.
- Горизонтальный: наличие различных компонентов, выполняющих функции передачи, хранения и расчета данных. Система сбора данных из исходных систем. Хранилища данных, используемые для построения отчетности на основе большого объема данных (в том числе исторических) или сложных расчетов, для которых необходимо выполнять предварительную подготовку данных. Эти решения позволяют исключить некоторые ограничения на уровне аппаратной среды.

Крупные компании для учета и управления деятельностью используют КИС, которые функционируют в следующих условиях [2]:

- Объем корпоративных БД, отражающих ключевую информацию о деятельности компании, составляет от одного-двух до десятков терабайт.
- Количество пользователей варьируется от пяти тысяч до пятидесяти тысяч, но количество активных пользователей намного ниже — от пятисот до десяти тысяч работников.
- Характер процессов: автоматизированы основные учетные операции, ключевые отчетные и аналитические формы, предоставление анализа данных. Режим работы: объем данных и частота обращений, связанных с извлечением данных и расчетом, во много раз больше объемов и интенсивности ввода данных (с учетом автоматизации и исключения дублирования данных).
- Инфраструктура: как правило, выполняется централизация в части размещения вычислительных мощностей и каналов данных.

Рассмотрим варианты свертывания в вертикальном и горизонтальном направлениях по стратам.

Страта 1. «Аппаратные ресурсы»

Виртуализированная среда распределения вычислительных ресурсов позволяет разделять и распределять ресурсы пула физических серверов

между виртуальными ресурсами практически неограниченно. Причем имеющийся физический ресурс может быть выделен как на множество виртуальных серверов, так и на один сервер. Поэтому с точки зрения задержек в обработке и передаче информации можно рассматривать классическую модель одного сервера.

Наиболее узким местом с точки зрения задержек является обращение к системе хранения (дисковому массиву). Таким образом, при построении систем класса RTE2.0 надо обращать внимание на решения, которые размещают как можно больше данных в оперативной памяти сервера.

В итоге на первой страте горизонтальное свертывание уже реализовано на виртуальной инфраструктуре, а вертикальное свертывание заключается в том, чтобы в идеале все операции по обработке, хранению данных производились только в оперативной памяти сервера, исключая необходимость обращаться к дисковому массиву. Дисковый массив при этом превращается в архивное и резервное хранилище (до массового использования энергонезависимой оперативной памяти).

Страта 2. «Информация»

Наиболее распространена в наше время в корпоративных информационных системах трехзвенная архитектура «клиент — сервер приложений — сервер БД». При этом уровень сервера приложений был специально выделен из сервера БД, чтобы обеспечить выполнение процедур обработки информации отдельно от процедур извлечения и изменения данных в БД. Исторически это было обусловлено ростом требований к производительности, с которыми не справлялись имеющиеся на массовом рынке вычислительные ресурсы. Таким образом, вертикальное свертывание может быть выполнено путем исключения сервера приложений и возврата всей обработки на уровень сервера БД.

Реляционная СУБД, положенная в основу практически каждого сервера, в текущий момент не справляется с решением задач различного назначения (с аналитической обработкой, обработкой потока событий, обработкой и хранением неструктурированных данных и так далее). Это привело к тому, что в ландшафте серверов предприятия появились отдельные системы по аналитической обработке информации (OLAP), по транзакционной обработке (OLTP), системы реального времени и другие. Наличие большого количества серверов БД и серверов приложений привело к необходимости специальных решений, обеспечивающих очистку, передачу и загрузку данных, поддерживающих интеграцию систем и целостность

данных. Таким образом, для идеального горизонтального свертывания требуется создание мощной универсальной СУБД, которая сможет заменить множество специализированных, исключив дублирование данных и все интеграционные процедуры.

Страта 3. «Приложения»

Функциональное развитие моделей корпоративных информационных систем на этой страте шло от простых MRP-систем до ERP и ERP II. Классическая модель ERP II предполагает наличие ERP-ядра и специализированных систем: взаимоотношений с покупателями (CRM), поставщиками (SRM), управления жизненным циклом продукта (PLM), управления логистическими цепочками поставок (SCM) и других. По горизонтали свертывание подсказывает объединение всех этих систем в одну. При этом модель ERP II сохраняется, но бизнес-функции реализованы не отдельными системами, а функциональными модулями внутри одной системы.

По вертикали специального свертывания в этом случае можно не производить, так как все вспомогательные системы, обслуживающие и обеспечивающие составной ландшафт приложений, при горизонтальном свертывании становятся ненужными, кроме тех, что обеспечивают внешние связи, они должны войти в состав единой системы.

Страта 4. «Бизнес»

Данная страта является, с одной стороны, потребителем возможностей предыдущих страт, с другой — именно с этой страты выставляются временные требования по исполнению бизнес-процессов. Поэтому страта 4 не исполняет, а обуславливает информационную систему. Можно предположить, что изменения коснутся и моделей этой страты, но они будут сделаны не для исключения задержек, а благодаря исключению [33]. Поэтому в рамках данной работы свертывание на четвертой и пятой стратах не применяем.

Использование результатов свертывания

Результаты свертывания по стратам можно использовать в качестве ориентиров при развитии информационных систем крупных предприятий ERP II. Можно составить комплексные требования к архитектуре информационной системы и на их основе выработать критерии оценки различных вариантов реализации. В качестве эталона при проектировании конкретных ландшафтов можно установить реализацию полного комплекта бизнес-функций ERP II в одной системе, в одной СУБД, на одном сервере

и в оперативной памяти. Важно оценить, насколько возможно сокращение задержек при полном или частичном свертывании.

Предварительные оценки показывают, что вполне реально свести задержки к уровню, который определил академик В. М. Глушков применительно к задачам экономического планирования и управления: «Реальный масштаб времени… это такая организация человеко-машинной системы, при которой ответы машины не задерживают естественный ход человеческой мысли, не требуют переключения человека на другую работу, пока производятся машинные расчеты в связи с его очередным заданием» [34, с. 286]. То есть речь идет о приемлемой задержке в единицы секунд от момента подачи задания до представления результата. Различные проекты по апробации результатов свертывания [35] показывают, что это достижимый показатель. Реальные же задержки в сегодняшних информационных системах измеряются десятками минут и даже часов.

Архитектурные стили

Понятие «Архитектура» отражает устаревший взгляд на ИС как на здание или сооружение — и оно стало слишком жестким, не отвечающим потребностям в таких свойствах ИС, как гибкость и адаптивность. Правильней ввести понятие «Системная холархия», как у живых организмов.

Страты, модели, связи

«Централизация» всей системы на одном сервере не является единственным возможным архитектурным решением свертывания, учитывающим диалектику страт информационных систем.

Первая и вторая страты существуют давно и в значительной степени насыщены проработанными моделями и зрелыми решениями, третья страта активно развивается последние двадцать лет, и развитие мощных бизнес-ориентированных архитектурных подходов типа TOGAF является признаком того, что достигнут определенный уровень зрелости.

В настоящее время развитие информационных технологий нацелено на четвертую страту [36]. Начинаются попытки ее освоения. Объектами четвертой страты являются бизнес-модели, бизнес-процессы, бизнес-роли, функции, документы. Адекватные этим объектам модели — это модели типа шаблона А. Остервальдера [27].

Чтобы облегчить постановку задачи для проектировщиков и программистов, разработаны различные подходы к онтологическому моделированию бизнес-моделей — например, язык моделирования бизнес-моделей Остервальдера или моделирование онтологии корпорации Jan L. G. Dietz [19].

Исследования диалектики страт показывают, что рано или поздно будут созданы языки онтологического «программирования» и платформы (компиляторы или интерпретаторы) для создания онтологических «программ». Назовем такие информационные системы онтологически управляемыми.

При этом взаимосвязи в онтологически управляемых информационных системах могут быть реализованы по-разному.

В первом варианте онтологическое описание является постановочным и реализуется (жестко программируется) на второй страте как отношения между таблицами БД, SQL-запросы и так далее, на третьей — как бизнес-объекты, как интеграция модулей и приложений.

Во втором варианте онтологические связи реализуются в конкретном запросе на четвертой страте и исполняются на нижестоящих стратах. Здесь используются новые перспективные возможности онтологически управляемых информационных систем.

«Монокристалл»

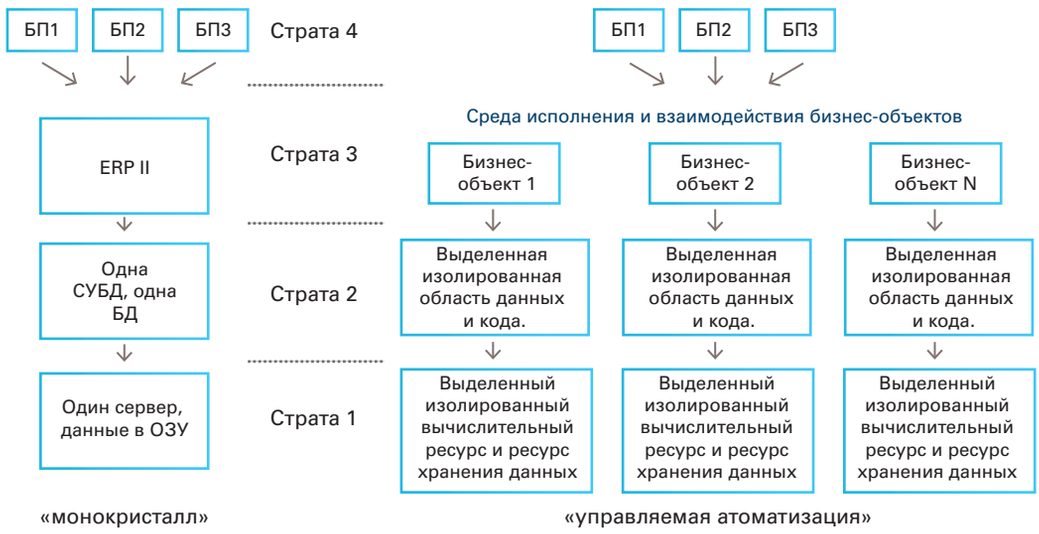
Первый вариант задает архитектурный стиль «Монокристалл». С одной стороны, это наиболее близкий к реализации подход, так как не требует наличия принципиально новых технологий и решений на четвертой страте, а с другой стороны, он входит в сильное противоречие с существующими информационными системами и для реализации потребует полного перевнедрения корпоративных информационных систем, что в большинстве случаев на практике крайне затруднительно.

Примером реализации этого подхода является программа ОРБИТА, принятая в ОАО «Сургутнефтегаз» для комплексного улучшения управляемости предприятия на базе перехода на платформу SAP HANA с кардинальным сокращением числа СУБД.

Этот стиль достаточно тяжел на практике — «монокристалл» получается массивный. Конечно, степень информатизации в каждой страте весьма высока, общий уровень информатизации предприятия близок к требующемуся, но надежность при таком подходе все же снижается. Например, отказ единственной СУБД приведет к полной остановке системы, поэтому приходится принимать дополнительные меры по обеспечению надежности.

Рисунок 33.

Архитектурные стили



К тому же это не соответствует современному мировому вектору, общей линии развития ИТ в мире по принципу «дробления» и перехода на микроуровень для обеспечения гибкости систем.

Но, несмотря на это, пока такой подход вполне отвечает интересам крупных компаний на горизонте стратегического планирования в три, максимум пять лет.

«Атомизация»

Второй вариант задает архитектурный стиль «Атомизация». Данный стиль предполагает реализацию каждой атомарной бизнес-сущности в виде «микрокристалла», при этом горизонтальные связи между ними устанавливаются на четвертой страте в виде онтологических связей, определяющих реализацию бизнес-запросов непосредственно в момент их исполнения.

С точки зрения практики стиль является более перспективным, так как позволяет снижать задержки как при получении необходимых данных, так и при адаптации системы к бизнес-среде (внесении изменений в программный код). Однако пока не разработаны ИТ-решения (информационные системы), реализующие такой стиль на практике.

Именно в рамках этого архитектурного стиля могут создаваться гибкие адаптивные цифровые платформы предприятий и организаций — онтологические информационные системы.

Стоит заметить, что уже существуют сигнальные технологические решения, существование которых позволяет утверждать, что стиль «Атомизация» является реализуемым и наиболее перспективным.

Среди примеров таких сигнальных подходов можно упомянуть:

a) понятие Semantic Web, язык Web Ontology Language (OWL);

Рисунок 34.

«Атомизация»

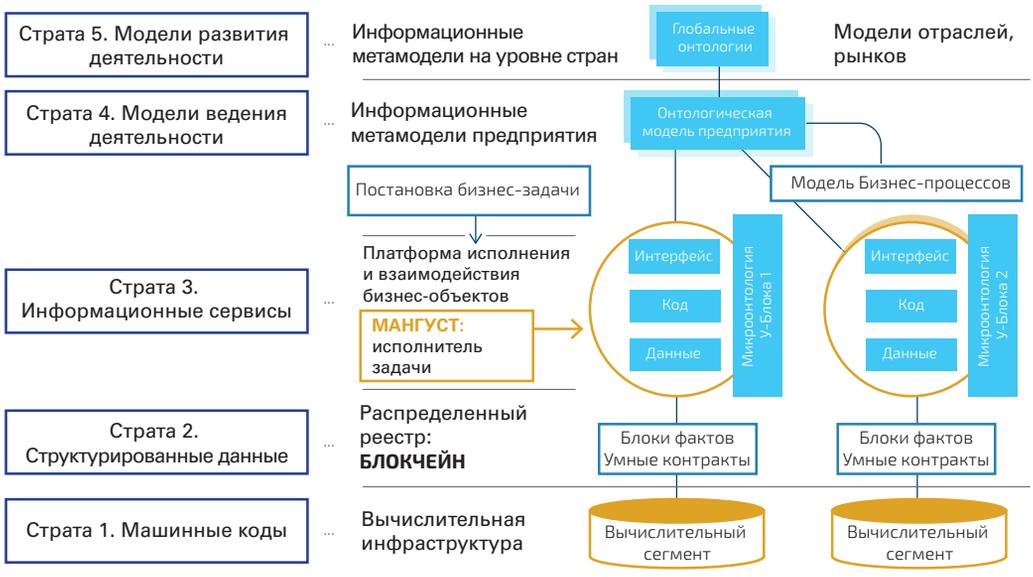


Рисунок 35.

Пример стиля «Атомизация» в цифровой платформе «УБ-Мангуст»

б) графовые СУБД, которые «на лету» реализуют взаимосвязи по онтологическим (пока слабо формализованным) описаниям, например Comindware ElasticData;

в) микросервисная архитектура, являющаяся развитием SOA;

г) технология представления деловых отчетов на XBRL [38], в которой каждый отчет являет собой законченную совокупность его XML-схемы, деловых данных, форматов их представления, связей.

У данного архитектурного стиля больше перспектив еще и потому, что:

- он позволяет гибко интегрировать различные информационные системы и среды исполнения (например, облако или частный ЦОД);
- имеет потенциально более широкую область применения (например, «умный дом», системы контроля производства, управление объектами), чем стиль «Монокристалл», который подходит в основном для корпоративных систем управления предприятием (ERP II).

Диалектическое развитие стратифицированных систем подводит нас к тому, что системы будут дробиться на маленькие, управляемые с верхних страт элементы. Уже стало понятно, что не нужно строить единую гигантскую модель данных, интегрировать между собой сотни СУБД, обеспечивать фиксированные потоки данных между ними — это становится неуправляемым с четвертой страты, от модели бизнес-процессов.

Идея состоит в том, чтобы разделить все информационное пространство на независимые кусочки (микросистемки), которые будут взаимодействовать:

- между собой по горизонтали (собираясь вместе, чтобы дать ответ на запрос пользователя);
- между онтологиями по вертикали (для поддержания целостного взгляда на предметную область и самоуправления).

Ближайшее будущее ИТ-архитектуры

На сегодняшний день ИТ-индустрия не дает целевой архитектуры для построения систем, способных управляться с четвертой страты. Это можно сделать только в новой, онтологически ориентированной архитектуре (холархии).

Можно предположить, что в ближайшем будущем будут реализованы проекты в основном в стиле «Монокристалл» — в силу большей технологической готовности и инерции унаследованных систем.

Однако в дальнейшем, в начавшемся процессе цифровой трансформации экономики, больше перспектив для реализации имеют «Атомизация», а также различные комбинации этих стилей, когда некоторые «атомы» могут быть «мини-кристаллами».

Вывод

Предложенный эмержентный подход к стратификации, апробированный в проекте архитектурной трансформации крупного предприятия, позволяет:

- разрабатывать целевую стратифицированную архитектуру информационных систем, в том числе архитектуру ИС нового поколения — онтологических ИС;
- определять адекватные каждой страте модели и инструменты;
- детализировать цели и задачи по каждой страте, соответствующие нуждам бизнеса;
- формировать требования к компетенциям членов проектной команды.

При моделировании и проектировании по конкретным стратам, при достижении достаточной степени стратификации обеспечивается адекватность и точность математических и информационных моделей, отражающих бизнес-реальность.

На основании эмержентной стратификации информационных систем:

- определены направления для построения ИТ-архитектуры по каждой страте, позволяющие сократить задержки в обработке информации и выполнении бизнес-процессов;
- сформулированы и описаны два доминирующих архитектурных стиля построения корпоративных информационных систем, имеющих целью развитие в направлении цифровой трансформации бизнеса.

Приложение 2.
Метод Directed Evolution

Введение в Directed Evolution

На первом шаге на пути к будущей системе перед нами стоит задача спрогнозировать наиболее перспективные направления развития информационных технологий в сторону повышения гибкости, адаптивности, удобства. Цель такой работы — минимизация рисков внедрения ошибочных и «тупиковых» инженерных решений, максимизация доходности при «попадании в точку», то есть создание востребованной рынком, гибкой, адаптивной, удобной информационной системы.

Эта задача решается в рамках метода управляемой эволюции (Directed Evolution), который заменяет прогнозирование результатов на целенаправленное создание новых путей и способов развития систем.

Метод основан на закономерностях технической эволюции, изложенных в ТРИЗ, содержит в себе способы интенсификации инженерного творчества и организован в соответствии с принципами управления проектами.

Он хорошо работает при концептуальном проектировании новых информационных систем, когда необходим выбор наиболее правильного из возможных направлений усовершенствования.

Целью работы является создание «образа будущего» (проектируемой информационной системы), содержащего как можно более полный перечень свойств будущей системы с указанием конкретных технических и организационных решений, направленных на достижение желательных результатов и исключение нежелательных.

При проведении Directed Evolution выполняются следующие задачи:

- Выявление возможных позитивных вариантов развития системы.
- Оценка рисков, которые могут быть связаны с развитием (с использованием метода «инверсионного анализа»).
- Выбор среди возможных вариантов развития одного, наиболее легко реализуемого имеющимися внутренними ресурсами системы.
- Формулирование изобретательских задач, необходимых для реализации выбранного варианта на практике, решение этих задач методами ТРИЗ и разработка необходимых инновационных ИТ-решений, обеспечивающих реальную возможность желаемого развития.
- Создание сценария желательного развития системы на базе выбранного варианта развития и инновационных ИТ-решений.
- Управление реализацией сценария: план, координация, контроль, анализ, оповещение (отчетность) и обратная связь.

ТРИЗ-прогнозирование как предшественник метода

Далеко не все существующие методы прогнозирования обеспечивают нужные результаты. Например, экспертно-интуитивные методы основаны на использовании знаний, опыта и профессиональной интуиции экспертов, а формализованные расчетные методы основаны на математической экстраполяции существующих тенденций. И те и другие имеют свои специализированные области применения, присущие им достоинства и недостатки.

История и методология технологического прогнозирования в России изложена в сборнике И. В. Бестужева-Лады [1]. В нем ученый отмечает, что прогноз явлений не должен сводиться только к предсказанию, но должен включать и оптимизацию наблюдаемых явлений, то есть фактически предлагает «управлять будущим», просчитывая возможные последствия планируемых решений и выбирая наиболее оптимальные из них.

Начиная с конца 1980-х годов в России начали развиваться методы прогнозирования, основанные на теории решения изобретательских задач (ТРИЗ)[8]. Основная суть ТРИЗ — выявление и использование законов, закономерностей и тенденций развития технических систем [2]. Используя ТРИЗ, можно вести прогнозирование, основанное на комплексном анализе развития конкретной технической системы с позиций ее соответствия законам развития данных систем, формулировании и решении перспективных задач по развитию данной системы или ее замене другой,

[8] ТРИЗ — теория решения изобретательских задач — область знаний, исследующая механизмы развития технических систем с целью создания практических методов решения изобретательских задач. Автор ТРИЗ — Генрих Альтшуллер.

более перспективной. Причем в отличие от указанных выше экспертных и формализованных прогнозов, предсказывающих изменение тех или иных характеристик системы, но не определяющих, каким образом изменение будет достигнуто, ТРИЗ-прогнозирование дает конкретные технические решения, позволяет сформулировать целостную и обоснованную концептуальную модель системы или технологии нового поколения [3].

Суть метода: чтобы предсказать будущее, надо изобрести его.

Однако в большинстве случаев такого рода прогноз неприменим на практике. Это связано с двумя принципиальными моментами [4]:

- В любой реальной системе может быть сформулировано огромное множество разных задач по улучшению и дальнейшему развитию разных подсистем, параметров и так далее. Решать все задачи сразу — просто бессмысленная затрата сил и времени. То есть *главное для прогноза — не решение, а выбор задач, которые стоит решать и которые будут решены.*
- ТРИЗ-прогнозирование концентрируется главным образом вокруг прогнозирования изобретений высокого уровня. Однако это не всегда так. *Нередко новое поколение систем возникает не из одного или нескольких высокоуровневых изобретений, а из накопления опыта, появления большого количества мелких изобретений или инженерных решений, которые в сумме и приводят к качественному скачку в развитии.* Так, например, нельзя назвать какие-то отдельные изобретения высокого уровня, которые бы привели к таким крупным «техническим революциям», как переход от паровозов к тепловозной и электрической тяге или появление нового экономичного автомобиля после нефтяного кризиса семидесятых годов.

Развитием ТРИЗ-прогнозирования стал метод Directed Evolution. Он нацелен не на поиск решений высокого уровня, а на предсказание следующих, эволюционных шагов развития системы. Важнее не предсказать, что случится, а дать рекомендации, что делать: как управлять развитием системы, чтобы добиться победы в бизнесе, конкурентной борьбе, получении заказов и так далее.

Теория эволюции нелинейных систем

Помимо методов ТРИЗ-прогнозирования, важнейшей теоретической базой Directed Evolution стала теория эволюции нелинейных систем. Большинство систем, с которыми мы сталкиваемся, от ИС предприятия до комплекса систем целой отрасли, являются нелинейными, а потому подчиняющимися общим закономерностям эволюции нелинейных систем. Главные утверждения этой теории можно свести к нескольким тезисам:

- Эволюция происходит только в системах так называемого «детерминированного хаоса», то есть таких, в которых взаимодействуют как четко определенные, так и хаотические элементы. В процессе развития самое главное — сохранение баланса между детерминированностью и хаосом, когда усиление одной составляющей должно быть скомпенсировано усилением противоположной; в противном случае система рискует быть разрушенной.
- Эволюция есть комбинация периодов спокойного и хорошо предсказуемого развития и кризисов, которые неизбежны[9] и результаты которых практически непредсказуемы, так как зависят от случайных флуктуаций.
- Эволюция системы ведет к специализации подсистем, в том числе к возникновению и развитию управляющих подсистем, формированию иерархических и сетевых систем, дроблению, объединению ранее независимых систем — свертыванию (см. ниже).
- Эволюция приводит к возникновению и развитию в системах сложных обратных связей, основанных на получении и обработке информации.
- Нелинейные системы исключительно богаты потенциальными ресурсами, включая многочисленные явные и скрытые эффекты, позволяющие реализовать множество разнообразных полезных функций или вызывающих неожиданные вредные последствия; в процессе эволюции система начинает использовать все большее количество потенциальных ресурсов и использует их все эффективнее.
- Направление эволюции определяется некоторыми аттракторами — состояниями, к достижению которых система стремится, и репеллерами — состояниями, которых система стремится избежать. Стремление системы к аттракторам как раз и описывается законами и закономерностями развития, сформулированными в ТРИЗ.

Главным практическим выводом теории эволюции нелинейных систем является понимание возможностей целенаправленного управления развитием любых систем путем:

- предсказания наиболее эффективных из принципиально возможных вариантов развития;
- выбора желаемого варианта развития из нескольких возможных и наиболее эффективных;
- целенаправленных воздействий на систему (преимущественно в точках кризисов) для управления ее развитием.

[9] Неизбежность возникновения кризисов вытекает из теории самоорганизованной критичности и свойств динамических систем с точками бифуркации.

Состав метода Directed Evolution

Начиная с середины девяностых годов на базе метода ТРИЗ-прогнозирования Борисом Злотиным и его коллегами была разработана технология управления эволюционными процессами, получившая название Directed Evolution [5]. Метод Directed Evolution — это технология прогнозирования развития технических и информационных систем на базе объективных закономерностей технической эволюции. Цель Directed Evolution — предсказание как можно более полного набора конкретных событий в развитии системы с указанием конкретных технических или организационных решений, как нужно действовать, чтобы обеспечить достижение желательных результатов и блокировать нежелательные.

Ключевой посыл: нужно спрогнозировать группу обоснованных законами системного развития «образов будущего» и, целенаправленно управляя их эволюцией (делая в нужное время нужные изобретения и «патентные заборы», планируя, продвигая и направленно инвестируя в проекты), обеспечить конкурентные преимущества для заранее выбранного желательного варианта развития.

Таким образом, Directed Evolution относится к методам управляемой эволюции и заменяет прогнозирование на планируемое, целенаправленное создание новых путей и способов развития систем. В основе этого метода лежат закономерности технической эволюции, выявленные в рамках классической ТРИЗ и развитые современными исследователями в России и за рубежом [6].

Ниже изложены основные положения этого метода.

Закон повышения идеальности систем

Это первый, один из наиболее важных законов, говорящий о направлении развития систем:

Все системы развиваются в направлении повышения своей «идеальности».

Базовое понятие технической эволюции, «идеальность» систем, означает следующее:

$$\text{Идеальность} = \frac{\sum \text{Полезных факторов}}{\sum \text{Факторов риска} + \sum \text{Факторов затрат}}$$

Идеальность системы — это отношение суммы полезных факторов в системе к сумме ее негативных факторов (рисков и затрат).

Полезные факторы — это полезные характеристики системы: функциональность, производительность, эргономичность и другие.

Факторы риска — это отрицательные характеристики системы: сбои, недостатки, проблемы, аварии и прочие негативные явления.

Факторы затрат применительно к ИТ — это совокупная стоимость владения информационной системой (Total Cost of Ownership).

Видно, что существует два вида увеличения степени идеальности системы: увеличение полезных факторов и уменьшение факторов риска и затрат (см. рисунок 36). Следствием из первого закона является понятие «идеальной системы», сформулированное Генрихом Альтшуллером:

Идеальная техническая система — это система, вес, объем и площадь которой стремятся к нулю, хотя ее способность выполнять работу при этом не уменьшается. Иначе говоря, идеальная система — это когда системы нет, а функция ее сохраняется и выполняется.

Теоретически «максимально идеальная» система имеет бесконечное количество полезных функций в числителе и полное отсутствие вредных факторов в знаменателе. Иными словами, это система, которая «умеет всё», делает это мгновенно, не стоит ни копейки и сделана «из ничего».

Как этого достичь?

Например, в рамках функционально-идеального моделирования — целенаправленным свертыванием, удалением ненужных функций и компонентов, упрощением процессов, процедур и операций (о свертывании читайте ниже). Изобретатель танка Т-34 Михаил Ильич Кошкин говорил: «Самая лучшая, самая непоражаемая деталь танка — это та, которой нет». При построении функционально-идеальной модели, свертывании систем, ставится цель ликвидировать компоненты, процедуры и операции, выполняющие ненужные, дублированные, вспомогательные, а по возможности и некоторые основные функции. При этом вспомогательные функции свернутых (ликвидируемых) операций исключают, а основные функции передают другим компонентам системы.

Применяя этот закон к информационным системам, мы получаем:

- полезный продукт (результат работы системы) — это информация; полезными являются все функции системы, направленные на получение информационного результата;
- затратный аспект системы — это ПО и оборудование, необходимое для получения полезного результата; все характеристики системы, связанные с ними, являются обеспечивающими (неполезными, затратными).

Несколько парадоксально, да? Тот предмет, которым мы привыкли профессионально заниматься,— компьютерные платформы и системы — оказываются неполезным аспектом! В дальнейшем мы увидим, в чем сила

и результативность этого подхода. Для информационных систем этот основной закон диктует развитие по направлению к выполнению все большего количества полезных функций со все большей производительностью и с использованием все меньших вычислительных и программных ресурсов (к уменьшению размеров и количества подсистем, оптимизации программного кода, переходу от программирования к сборке из программных модулей, самоорганизации систем и самообслуживанию, уменьшению совокупной стоимости владения системой).

Закон развертывания-свертывания

Системы развиваются от функционального центра к периферии, проходя фазы развертывания и свертывания. Соответствующий закон развертывания-свертывания гласит:

Система, возникнув и начав захватывать ресурсы, увеличивает как полезные факторы, так и затраты и риски (развертывается), а достигнув некоторого предела в факторах затрат и рисков, начинает их уменьшать (свертывается).

Таким образом, вводится два понятия:

Развертывание — это увеличение числителя (суммы полезных факторов) с одновременным увеличением знаменателя (суммы негативных факторов — рисков и затрат).

$$\text{Идеальность} = \frac{\uparrow \sum \text{Полезных факторов}}{\uparrow \sum \text{Факторов риска} + \sum \text{Факторов затрат}}$$

Свертывание — это увеличение или сохранение числителя с одновременным уменьшением знаменателя.

$$\text{Идеальность} = \frac{\uparrow \sum \text{Полезных факторов}}{\downarrow \sum \text{Факторов риска} + \sum \text{Факторов затрат}}$$

Согласно законам ТРИЗ, развертывание системы начинается с момента появления ее функционального центра и продолжается к периферии системы. К функциональному центру добавляются другие части (см. ниже закон полноты частей системы), улучшающие выполнение полезных функций системы,— и структура системы усложняется. При первоначальном экстенсивном развитии системы подключение и использование тех или иных ресурсов происходит неравномерно, что вызывает возникновение технических противоречий в системе, которые разрешаются созданием все новых подсистем. Система развертывается.

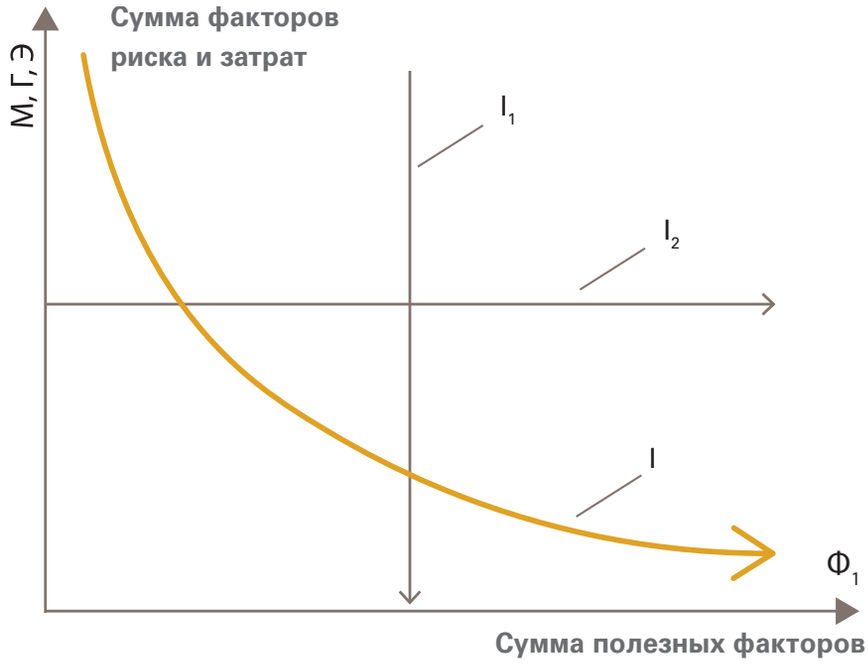


Рисунок 36.

Два вида повышения степени идеальности системы

Примечание:
I1 — уменьшение факторов риска и затрат при сохранении полезных факторов системы;
I2 — увеличение полезных факторов системы при сохранении факторов риска и затрат;
I — общий вид идеализации систем отражает оба вида повышения степени идеальности системы.

Однако это не происходит до бесконечности, существует точка максимального развертывания системы (рисунок 37). И после достижения этой точки система обязательно начинает свертывание — новый этап преобразований, связанный с уменьшением факторов риска и затрат.

Приложение 2. Метод Directed Evolution

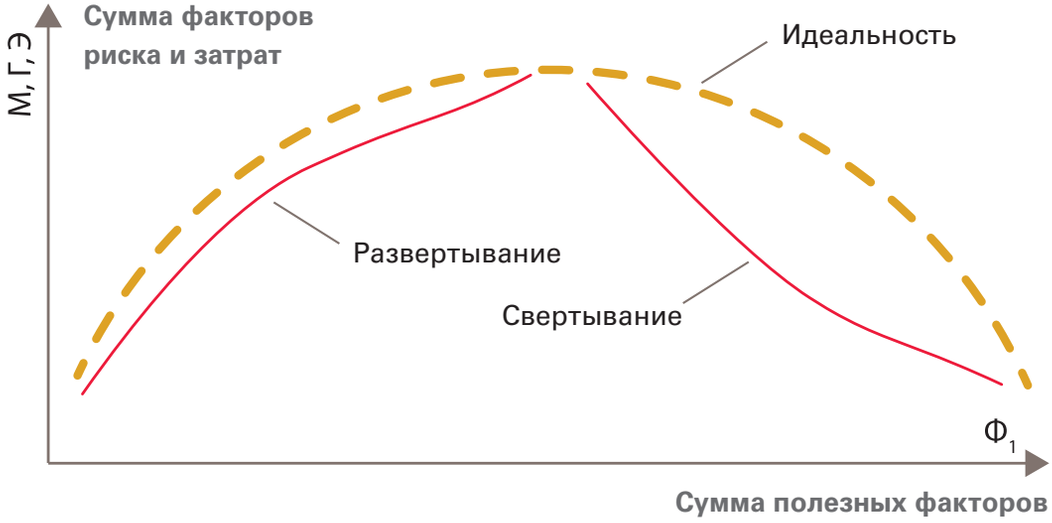
Основная идея свертывания, как уже упомянуто выше,— устранить из системы часть ее элементов (вместе с их вредными функциями и другими известными и неизвестными недостатками), а их полезные функции распределить среди оставшихся элементов системы или ее надсистемы. При свертывании системы противоречия, порождаемые экстенсивным ростом и усложнением структуры системы, могут разрешаться путем передачи функций систем и подсистем соседним системам. Например, новые достижения науки и техники позволяют отказаться в системе от нескольких специализированных подсистем в пользу одной универсальной, которая и выполняет все нужные функции.

Вот три возможных направления повышения идеальности функции при проведении свертывания систем:

- функцию выполняет другая подсистема, надсистема или окружающая среда (подсистемы нет, а главная функция системы сохраняется) (рисунок 38);
- функция выполняется сама собой (компонента нет, а его функция выполняется);

Рисунок 37.

Закон свертывания-развертывания системы



- функция не нужна (не нужны и функция, и компонент, который ее выполнял).

Другой пример — свертывание операций (процедур) проходит по следующим направлениям:

- операцию выполняют другие компоненты системы;
- операции (процедуры) нет, а ее функция все равно выполняется (результат достигается);
- операция выполняется «сама собой»;
- операция не нужна.

Для каждой операции возможность ее свертывания формулируют в виде требований «МНВЕ» — операцию «Можно Не Выполнять, Если» возможно ее осуществление за счет предшествующих или за счет последующих операций. Например, вот формулировки направлений свертывания для *обеспечивающих и вспомогательных* операций:

- нет операции, которую «обеспечивает» или которой «помогает» ликвидируемая операция;
- обеспечиваемая операция сама себя обеспечивает;
- обеспечение производится на других операциях (обеспечивающих или создающих), предшествующих ликвидируемой.

Формулировки направлений свертывания для *создающих* операций следующие:

- не нужен объект, обрабатываемый в данной операции;
- объект получен из другой операции и не изменяется в ликвидируемой;
- функцию этой операции выполняют последующие (или другие) операции, вплоть до операций других процессов.

То есть, анализируя возможности улучшения систем, мы стараемся двигаться к «идеальному конечному результату» — когда максимальная польза достигается минимальными средствами.

Отметим, что в развитии реальных систем чаще всего идут смешанные процессы: развивается и повышает степень идеальности то одна, то другая часть системы, тот или иной уровень иерархии. Одна часть системы может развертываться, при этом другая — свертываться. Кажущаяся бессистемность такого развития обуславливается внезапностью появления и обострения противоречия в какой-либо части системы, то есть процесс соответствует закону неравномерности развития частей системы.

Закон трехэтапного развития системы по S-кривой

Идеальность систем изменяется по S-образной кривой, охватывающей три этапа развития систем (рисунок 39):

- Первый этап — «рождение и детство», эксперименты, start-up, выход на рынок (именно этот этап развития системы описывает известный Hype Cycle компании Gartner).
- Второй этап — «рост и взросление», промышленная эксплуатация, широкое признание рынком.
- Третий этап — «угасание и старость», «дотягивание» характеристик, неизбежный уход с рынка.

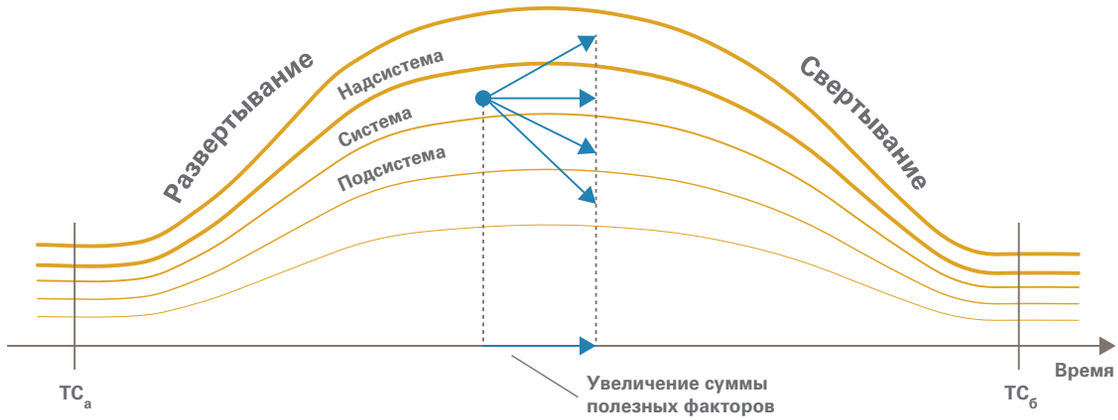
Закон трехэтапного развития диктует необходимость мониторинга уровня «идеальности системы» и прогнозирования развития технической системы, для того чтобы вовремя заметить наступление третьего этапа и начать новый инновационный цикл, избежав потерь на поддержание функционирования изжившей себя системы. Это особенно актуально для информационных систем — с учетом очень высоких темпов развития информационных технологий по сравнению с другими технологическими областями, в том числе с учетом действия закона Мура.

Практическое применение этого закона очевидно: это основа для прогнозирования и планирования развития системы в рамках ее жизненного цикла, будь это бизнес-система или информационная система предприятия. Но S-кривая дает только понимание тенденции, а планирование требует измерений, в том числе измерения значений достигнутой системой идеальности в качестве комплексного показателя уровня совершенства (зрелости) системы. Как же можно измерить идеальность?

Для этого используется метод System's Life Cycle Analysis (SLCA), описанный в [38]. Метод SLCA основан на том, что внедрение любых инноваций, в том числе нововведений информационно-технологического характера,

Рисунок 38.

Возможные пути развертывания-свертывания систем



изменяет идеальность системы, которая, в свою очередь, описывается положением системы на S-кривой развития. Измеряя значение идеальности системы в разных точках жизненного цикла на S-кривой развития, можно оценивать динамику прироста идеальности системы вследствие внедрения инновации (рисунок 40).

На старте проекта развития ИС производится оценка идеальности системы на данный момент, фиксирующая начальное значение идеальности, с тем чтобы задать точку отсчета для мониторинга идеальности в процессе развития системы. Затем, в процессе развития информационной системы предприятия, необходимо вести мониторинг значений идеальности как самой информационной системы, так и бизнес-системы (самого предприятия). Это позволяет оценивать эффективность предпринимаемых действий по совершенствованию ИС и определить ту точку, в которой прирост идеальности начнет снижаться. Последнее будет означать необходимость перехода на новую систему, новый виток развития, новую S-кривую.

Закон полноты частей системы

Необходимым условием принципиальной жизнеспособности системы является наличие и минимальная работоспособность ее четырех основных частей: «источника», «преобразователя», «завершителя» и «управления» (рисунок 41). Они выполняют следующие функции:

- *«источник»* — формирует ресурсы для выполнения главной полезной функции системы и выработки продукта системы;
- *«преобразователь»* — превращает ресурсы в продукт;
- *«завершитель»* — завершает выполнение главной полезной функции системы и передает продукт потребителю;
- *«управление»* — прогнозирует, планирует, координирует, контролирует, анализирует и оповещает о работе системы.

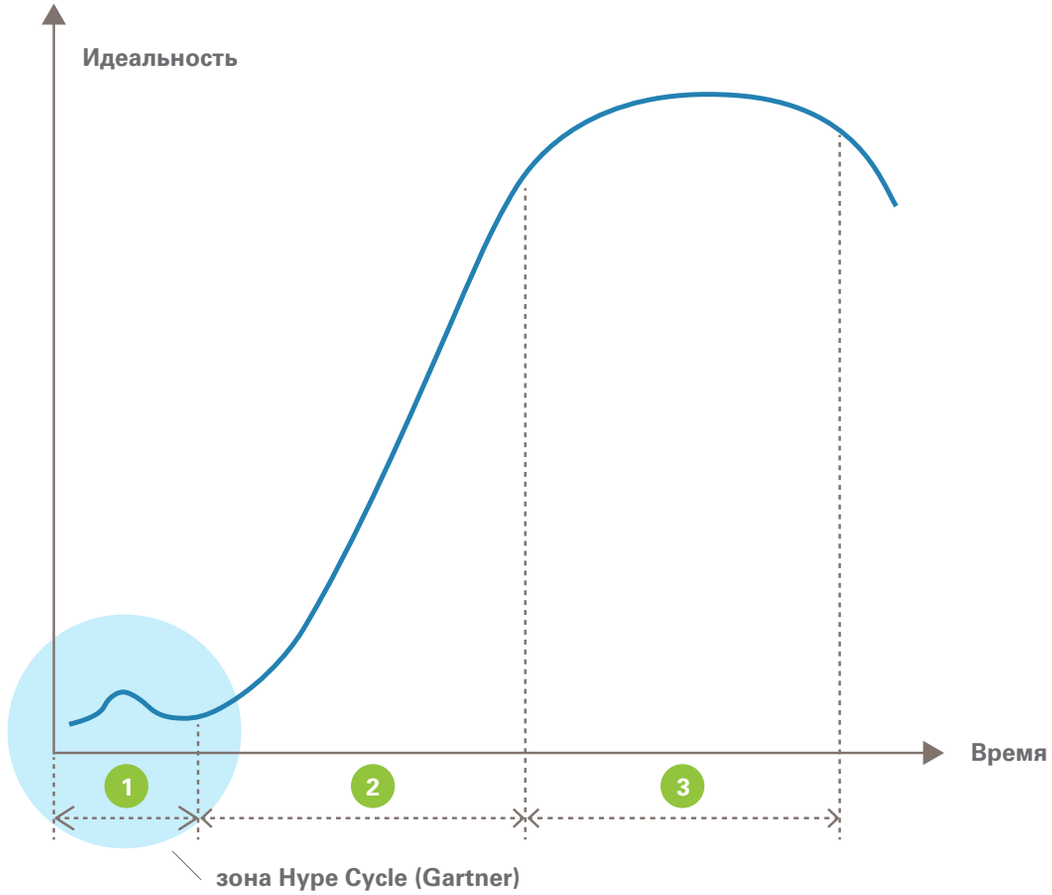


Рисунок 39.

Закон трехэтапного развития системы по S-кривой

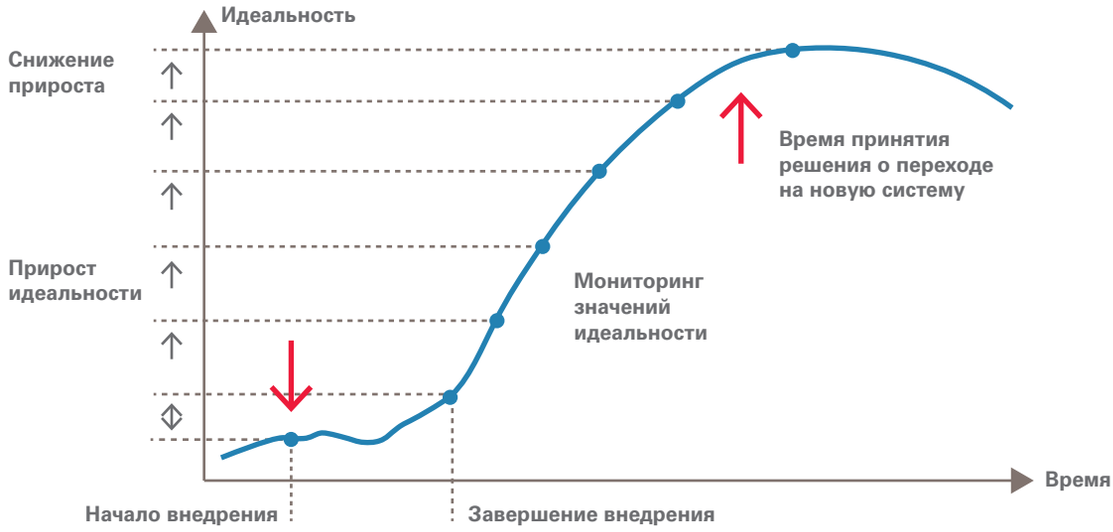
Приложение 2. Метод Directed Evolution

Для информационной системы этот закон означает, что ее архитектура должна включать все четыре указанные части. Если в системе отсутствует какая-либо из этих частей, то ее функцию выполняет человек или окружающая среда. И если система по каким-то причинам неполна, в ней отсутствует или не развита та или иная часть, то система будет развиваться неправильно, с множеством проблем.

Следует указать важный нюанс: в полноценной системе приоритет в развитии должен быть отдан «завершителю», рабочему органу. Именно на его развитие необходимо направлять основные ресурсы, ведь он является как бы «полномочным представителем» системы перед потребителем. Но инженеры часто увлекаются развитием других частей (например, «преобразователя»), оставляя «завершитель» без должного внимания. Например, в ИТ «завершитель» — это пользовательский интерфейс, и ярким примером пренебрежения этим правилом может служить преувеличенное внимание к разработке разнообразных внутренних «фишек» в системе при оставлении вне поля зрения неудобных, устаревших интерфейсов пользователя.

Рисунок 40.

Мониторинг значений идеальности системы в ходе ее развития



Линии развития систем

Всего в классической ТРИЗ сформулированы восемь законов развития технических систем. Выше мы подробно рассмотрели четыре:

1. Закон повышения идеальности.
2. Закон свертывания-развертывания.
3. Закон трехэтапного развития по S-кривой.
4. Закон полноты частей системы.

Помимо них, сформулированы такие законы:

- *Закон повышения динамичности и управляемости.* В процессе развития системы повышаются ее способности к целенаправленным изменениям, обеспечивающим возможность адаптации к меняющимся требованиям со стороны пользователя, других систем, внешней среды. Развитие, а значит, и жизнеспособность системы определяется показателем «степень динамичности», то есть способностью быть подвижной, гибкой, приспосабливаемой к внешней среде. Чем выше степень динамичности, тем шире диапазон условий, при которых система сохраняет свою функцию. Этот закон описывает также переход систем к самоуправлению и самоорганизации.
- *Закон противоречий в развитии.* Данный закон описывает возникновение, обострение и разрешение противоречий в процессе развития системы. Развитие частей системы идет неравномерно; чем сложнее система, тем неравномернее развитие ее частей. Неравномерность развития частей системы является причиной возникновения технических и физических противоречий и, следовательно, изобретательских задач.

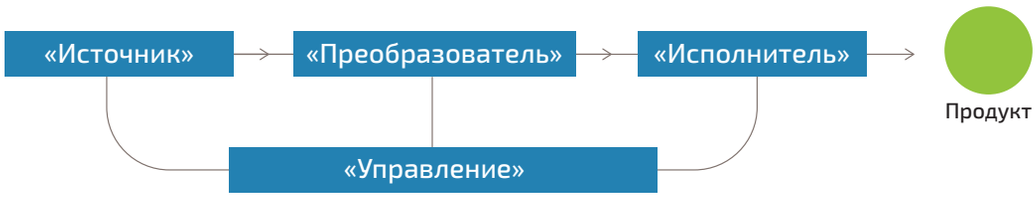


Рисунок 41.

- *Закон согласования/рассогласования.* Необходимым условием принципиальной жизнеспособности технической системы является согласование ритмики (частоты колебаний, периодичности) всех частей системы. Он описывает развитие системы в следующих направлениях:
  - последовательное согласование системы с другими системами;
  - рассогласование, обеспечивающее уменьшение и прекращение прохождения ненужных потоков (материальных, информации и других);
  - сдвиг согласования, обеспечивающий отбор части полезного или вредного потока для выполнения дополнительных полезных функций.
- *Закон перехода на микроуровень.* Переход с макроуровня на микроуровень — одна из основных тенденций развития всех современных технических и информационных систем.

Приложение 2.
Метод
Directed Evolution

Эти законы за прошедшие двадцать лет были подробно декомпозированы на линии развития, которые используются в методе Directed Evolution. Для прогнозирования используются сотни линий развития систем — например, такие, как:

1. Свертывание системы через:
   - перенос части функций системы в надсистему;
   - объединение подсистем в составе данной системы и сокращение количества подсистем.
2. Повышение использования системой ресурсов посредством:
   - увеличения числа типов используемых ресурсов;
   - перехода к использованию производных ресурсов[10];
   - перехода к использованию ресурсов окружающей среды и надсистемы.
3. Повышение адаптивности системы путем:
   - повышения динамичности;

[10] Развитие систем нередко сдерживается недостатком необходимых для развития ресурсов. Чаще всего это означает, что нужных ресурсов нет в готовом, нужном для применения виде, но при этом есть «предресурсы», то есть те, из которых можно создать нужные. Благодаря методам преобразования ресурсов в ТРИЗ получаются производные ресурсы, необходимые для развития системы.

- повышения управляемости;
- увеличения степени согласования или, наоборот, рассогласования частей системы с другими системами.

4. Выявление и разрешение противоречий в системе.
5. Вытеснение человека из системы.

И многие другие.

В целом метод Directed Evolution включает около шестисот линий развития систем, а также около семисот операторов для решения инновационных задач и проведения инверсионного анализа.

Каждая из линий развития систем имеет свое начало и конец, и весь путь системы от зарождения до «идеального состояния» по конкретной линии можно принять за 100%. Анализируя существующий этап развития системы по каждой линии, эксперт может дать каждой линии количественную экспертную оценку. Таким образом, можно построить диаграмму идеальности системы по выбранным линиям ее развития (рисунок 42).

Общая схема проведения Directed Evolution

Метод Directed Evolution можно использовать на всех этапах жизненного цикла инноваций (рисунок 43). Более того, он является одним из компонентов метода синтеза инноваций. Процесс проведения Directed Evolution состоит из следующих логических шагов:

- Прогнозирование развития рассматриваемой системы:
  - выявление возможных и вероятных позитивных вариантов развития;
  - выявление негативных последствий — опасностей и рисков, которые могут быть связаны с развитием (использование метода «инверсионного анализа»).
- Выбор среди возможных вариантов развития одного, наиболее реализуемого имеющимися ресурсами системы и окружающих ее систем.
- Формулирование и синтез необходимых инноваций в системе, обеспечивающих реальную возможность желаемого развития.
- Разработка сценария желательного развития системы на базе выполненного прогноза и синтезированных инноваций.
- Управление реализацией сценария: планирование, координация, контроль, анализ, оповещение (отчетность) и обратная связь.

Рисунок 42.

Диаграмма идеальности системы

Этот процесс можно проиллюстрировать схемой, показанной на рисунке 44. Видно, что этот процесс, начинаясь в сегодняшнем дне, изучает прошлое системы, на его основе прогнозирует будущее, а потом управляет его построением [5].

В состав проекта по проведению Directed Evolution входят следующие крупные этапы:

- подготовка проекта;
- экспресс-прогноз;
- анализ эволюции и истории развития системы;
- анализ ресурсов и ограничений развития;
- причинно-следственный анализ истории развития системы;
- прогноз по различным линиям развития системы;
- выработка прогнозных гипотез;
- оценка результатов прогнозирования;
- разработка сценария и плана желательного развития системы.

Весь процесс проведения Directed Evolution поддерживается соответствующим ПО, указанные выше этапы подробно декомпозируются на шаги процесса. Проект Directed Evolution идет итеративно, со многими циклами, включая большие циклы итераций и малые (вспомогательные).

Большие циклы включают:

- экспресс-прогноз Directed Evolution как подготовку к основной работе;

Рисунок 43.

Применение метода Directed Evolution в жизненном цикле инноваций

- основной цикл работ;
- вспомогательные циклы, проводимые по мере необходимости.

Малые (вспомогательные) циклы на каждом шаге работ включают:

- сбор специфической информации по данному шагу;
- анализ собранной информации;
- творческий процесс поиска новых идей;
- предварительную интеграцию идей данного шага с идеями, найденными ранее.

Определяющие особенности метода

Отметим три значимые особенности Directed Evolution, которые во многом определяют успех прогнозирования.

1. *Правило «надсистемного входа» при прогнозировании.* Главной ошибкой классического технологического прогнозирования была опора на тенденции развития рассматриваемой системы. Это давало хороший прогноз в «ближайшей перспективе», где развитие определяется в основном собственными ресурсами системы, но приводило к провалам при среднесрочном и долгосрочном прогнозировании. Появление новых поколений системы, переход на новые принципы и подходы

в большей степени зависит от общего развития общества, техники, рынка и создаваемых этим развитием ресурсов, чем от собственных ресурсов системы. Поэтому попытки чисто технического прогноза принципиально ограничены и неизбежно несут в себе ошибки.

Для адекватного дальнего прогноза развития системы требуется изучение надсистемы или нескольких надсистем разных уровней, формирующих условия развития системы — движущие силы развития и ограничения. Они приходят с надсистемного уровня, и без осознания их значения и оценки прогноз имеет малую ценность.

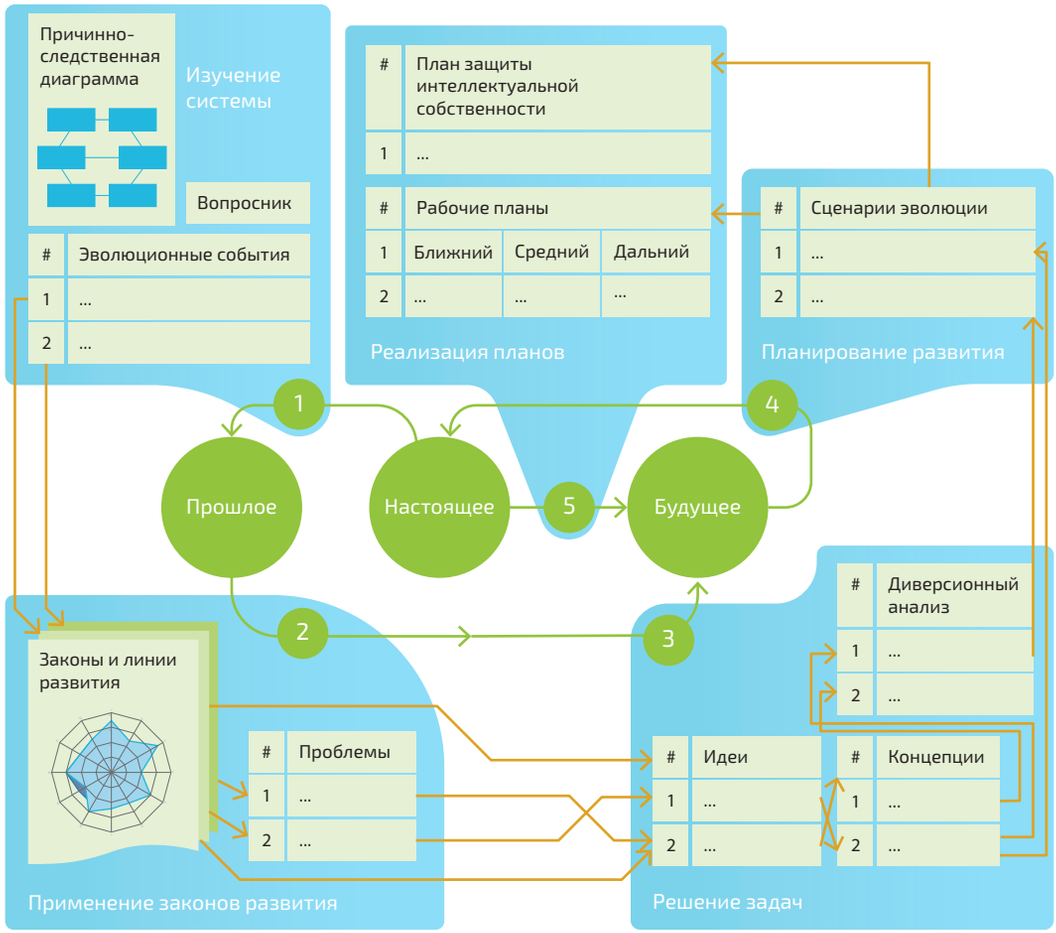
2. Для проведения Directed Evolution необходимо глубокое изучение самой системы, ее надсистем и истории их развития. Лучший способ изучения системы — рассмотрение целенаправленных попыток решения тех или иных задач, связанных с ней, совершенствования тех или иных функций, параметров. Результат — глубокое понимание системы, ее функционирования и специфических особенностей. Это не заменяет другие способы изучения системы, но очень хорошо с ними совместимо. Таким образом, происходит как бы слияние, взаимопроникновение разных стадий работы — изучения системы и выполнения прогноза, возникает итеративный процесс с положительной обратной связью: шаги изучения системы активизируют шаги прогнозирования, а те требуют дополнительного изучения системы.

Для улучшения этого итеративного процесса был разработан список типовых «контрольных вопросов», обеспечивающих выявление и документирование наиболее важной информации о системе и ее развитии:

1. Разрез «надсистемы — система — подсистемы»:
   - структура системы, подсистемы;
   - надсистемы и окружение системы;
   - системы с аналогичными проблемами.
2. Разрез «входы — процессы — выходы»:
   - входы в систему;
   - функционал системы, процессы;
   - выходы из системы.
3. Разрез «причины — проблемы — результаты»:
   - задачи, которые нужно решить, проблемы;
   - механизмы, порождающие проблемы;
   - нежелательные результаты нерешенных проблем;
   - другие задачи, решение которых может дать желаемый результат.
4. Разрез «прошлое — настоящее — будущее»:

Рисунок 44.

Общая схема процесса Directed Evolution



- история развития системы, появления и решения ее проблем;
- период до появления системы;
- период, который наступит после того, как система завершит свой жизненный цикл;

5. Разрез «ресурсы и ограничения»:
   - доступные ресурсы для работы системы;
   - допустимые изменения системы;
   - запреты и ограничения, связанные с системой;
   - критерии для оценки результатов развития системы.

Схема этого итеративного процесса получила название «системное яйцо» (рисунок 45).

Основные преимущества метода

Метод Directed Evolution может быть положен в основу проектирования информационных систем нового поколения, имеющих принципиальные

Рисунок 45.

«Системное яйцо»

отличия в лучшую сторону по производительности, стоимости и иным ключевым характеристикам.

Кратко резюмируя, можно сказать, что метод Directed Evolution имеет следующие преимущества:

- «просчитывает ходы» технической эволюции и помогает пройти самым коротким путем к выигрышу в конкуренции информационных систем;
- используется для целенаправленного совершенствования любых типов информационных систем и технологий;
- обеспечивает высокую точность прогноза в любой предметной области на среднесрочный и долгосрочный период;

Directed Evolution обеспечивает принципиальное улучшение ключевых характеристик, устранение проблем и тупиков в развитии систем. Недостаток — относительная сложность и дефицит квалифицированных кадров — специалистов, обученных данному методу.

Приложение 3. Синтез инноваций: цифровая платформа будущего

Здесь мы приведем пример пошагового процесса изобретения, то есть целенаправленного синтеза конкретной инновации под названием «цифровая онтологическая платформа нового поколения «УБ-Мангуст».

Такой процесс содержит следующие шаги:

Шаг 1. Прогнозирование направлений развития системы или технологии.

Шаг 2. Выбор приоритетного варианта развития для конкретной системы.

Шаг 3. Синтез необходимых инноваций для выбранного варианта развития.

Шаг 4. Реализация полученных инновационных идей в ИТ-архитектуре.

Рассмотрим каждый шаг подробно.

Шаг 1. Прогнозирование направлений развития системы или технологии

На этом шаге проводится выявление возможных и вероятных позитивных вариантов развития информационной системы. Работа ведется с использованием метода Directed Evolution [8].

Ниже приводится пример такой работы — экспресс-прогноз по Directed Evolution, который был сделан нами в нефтяной компании, внедряющей в своей информационной системе технологию in-memory computing. Результаты, полученные в ходе этой работы [7], легли в основу изобретения цифровой платформы «УБ-Мангуст».

Путь технологии in-memory от идеи до промышленной реализации занял восемнадцать лет, с 1998 (Bell Labs, Dali Main-Memory Storage Manager)

по 2011 год (выпуск SAP HANA). Мы задались целью спрогнозировать дальнейшее развитие этой технологии по направлению к реализации концепции Real-Time Enterprise (RTE) [7], [30], но в ходе работы были получены более широкие результаты, послужившие основой для создания концепции «УБ-Мангуст» и написания этой книги.

База и инструменты прогнозирования

База для экспресс-прогноза — существующая корпоративная ИС нефтяной компании. Компания традиционно опирается на ИТ-решения компании SAP, ее операционная деятельность поддерживается ERP-системой SAP R/3 (сейчас SAP ECC). В настоящее время в компании идет внедрение информационно-аналитической системы, основанной на SAP HANA, использующей технологию in-memory. Технология in-memory, в свою очередь, является существенным шагом к RTE и служит одной из опорных точек для развития будущей цифровой платформы предприятия.

Это направление развития ИС наметилось еще до проведения экспресс-прогноза, поэтому работа проводилась в заранее определенном направлении к ИС класса RTE2.0 и в конкретных обстоятельствах — для максимальной реализации инновационного потенциала SAP HANA, с учетом наличия в компании глубокой экспертизы в системах SAP.

Инструмент проведения экспресс-прогноза развития ИС — это серия проектных мастерских с использованием подходов, входящих в метод Directed Evolution. Проектная мастерская — это управляемый мозговой штурм со структурированием и последующим анализом полученных идей. Такая форма работы используется на старте сложных комплексных проектов с целью стимулирования свободной генерации продуктивных идей, которые могут быть положены в основу проектной концепции. Одна из ключевых идей проектной мастерской заключается в том, что процесс генерации идей отделен от процесса их обсуждения и анализа. Каждая сессия мастерской содержит три фазы:

- индивидуальный мозговой штурм;
- обсуждение в малых группах;
- управляемое структурирование полученных идей.

Помимо генерации идей, в рамках проектной мастерской происходит визуальное структурирование полученных предложений (рисунок 46), а также совместное обсуждение и выработка согласованного мнения участников по поводу результирующих идей. Вот базовые правила проектной мастерской:

Рисунок 46.

Рабочая визуализация идей, полученных во время одной из сессий проектных мастерских

- высказываются любые, даже самые «дикие» идеи, поощряется «подхватывание» и развитие высказанных идей;
- критика высказываемых идей запрещена, никаких обсуждений, оценок, возражений, полная свобода для полета мысли, бурный «поток сознания»: пишется все подряд — первое, что приходит в голову.

Экспресс-прогноз был проведен на старте проекта по созданию аналитической системы, где в рамках проектной мастерской был сформирован «образ будущего» — портрет идеальной системы предприятия, использующей технологию in-memory и другие связанные с ней идеи и технологии.

«Идеальный образ», основанный на знаниях экспертов предметной области и ИТ-специалистов, был дополнен прогнозными гипотезами, вытекающими из закономерностей технической эволюции — основы Directed Evolution.

Результаты прогноза легли в основу концепции построения новой аналитической системы, дали импульс дальнейшего развития информационной системы компании в целом и были использованы для построения методологии создания информационных систем будущего.

Понятно, что прогнозы развития системы должны отталкиваться от ИТ-стратегии компании, сопряженной, в свою очередь, с бизнес-стратегией. Поэтому, прежде чем приступать к прогнозированию, необходимо сформулировать миссию, видение и цели развития ИС (если это не сделано ранее при формулировании ИТ-стратегии). Причем впоследствии результаты прогноза могут либо подтвердить правильность выбранных целей, либо потребовать их корректировки или даже пересмотра. Например, казалось бы, правильная цель — «нарастить мощность аналитических подсистем посредством внедрения дополнительных модулей от вендора X» — может оказаться совершенно неправильной в части «внедрения дополнительных модулей». Ниже будет изложено понятие свертывания, объясняющее причины этого.

Миссия и видение информационной системы

Миссия ИС:

Миссия ИС определяет предназначение системы для предприятия, верхнеуровневую цель ее деятельности на обозначенную перспективу, ее «жизненное призвание». В нашем случае миссия ИС заключается в том, чтобы стать для компании главной опорой устойчивого развития по направлению к «предприятию реального времени» на базе высокопроизводительной гибкой ИТ-архитектуры.

Такая миссия подразумевает, что:

1. Мы знаем, в каком направлении необходимо развиваться и как осуществить этот процесс. Управление ИТ обеспечивает устойчивое предсказуемое развитие ИС посредством создания и внедрения точных (целенаправленных) пошаговых ИТ-инноваций, при этом производя глубокую экспертизу в существующих системах и технологиях.
2. У нас есть все необходимое для ИТ-инноваций: команда, знания, методология и неограниченный доступ к современным практическим наработкам мировых специалистов по синтезу инноваций.

Видение: Real-Time Enterprise 2.0

Стратегическое видение — это взгляды руководителя и ИТ-менеджеров на то, какими направлениями надо заниматься и каков при этом долгосрочный курс.

Видение содержит ответы на основные вопросы: какой мы видим свою систему, что мы собираемся делать и чего хотим достичь?

Связь видения с миссией: если миссия системы ориентируется на потребителя (пользователя), то видение акцентирует внимание на деятельности, позволяющей реализовать эту миссию.

Желаемая цель — повышение полезных характеристик транзакционных и аналитических систем, в том числе систем бизнес-аналитики и поддержки принятия решений (Business Intelligence & Decision Support Systems), и их развитие в направлении реализации концепции Real-Time Enterprise (RTE).

Обратимся к понятию RTE. Еще в 2002 году при появлении конвергентных аппаратных решений эксперты Gartner предложили определение предприятия, управляемого в реальном времени:

The RTE is an enterprise that competes by using up-to-date information to progressively remove delays to the management and execution of its critical business processes.

Важно отметить, что в определении делается акцент на последовательном (progressively) исключении задержек в предоставлении данных о работе предприятия, которые обусловлены архитектурными и функциональными ограничениями прежде всего со стороны СУБД.

Рисунок 47.

Цели системы и разных фокусных групп, влияющих на развитие системы

Однако инновационная технология in-memory дает новые возможности, которые принципиально меняют ситуацию. Например, в аналитических приложениях стало возможным оперировать свежими данными непосредственно из транзакционной БД, строить аналитические запросы тысячекратно более сложные, чем раньше, и даже реализовать «несбыточную мечту» руководителя — производить «закрытие предприятия» (иметь актуальный баланс) даже после каждой транзакции. Таким образом, мы приближаемся к управлению предприятием в режиме реального времени, когда информационная система помогает руководителям принимать управленческие решения на основе «мгновенного среза» деятельности по самой актуальной, «горячей» информации[11].

Такую информационную систему можно определить как Real-Time Enterprise 2.0, изложив определение Gartner 2002 года в следующей редакции [13]:

RTE2.0 is an enterprise that competes by using up-to-date information to completely remove delays to the management and execution of its critical business processes.

То есть «предприятие реального времени версии 2.0» — это то, которое конкурирует на рынке за счет использования самой актуальной информации, тем самым полностью устраняя задержки в управлении предприятием.

[11] Подробнее об этом читайте в статье «На пути к Real-Time Enterprise 2.0. Изменения корпоративных информационных систем при использовании технологии in-memory Data Management. // Information Management, 07–08, 2014.

Технология in-memory позволяет нам уже сейчас вплотную подойти к реализации парадигмы RTE2.0. И это видение стало основным для нашего прогноза[12].

Целеполагание: иерархия целей развития системы

Сразу после формулировки видения системы надо определить цели ее развития.

Для целостного представления корпоративной информационной системы и ее окружения предлагается графическая метафора четырех сегментов и двух слоев (верхнего и нижнего), которая отражает общее видение взаимодействующих целей разных фокусных групп, влияющих на развитие рассматриваемой информационной системы (рисунок 47).

Верхний слой состоит из целей надсистемного окружения корпоративной системы — сообществ, объединений. Они отражают совокупные потребности, намерения и тенденции в общественных, информационных, технологических системах, влияющих на развитие ИС. Выделены две категории:

- потребители услуг ИС (социум, участники рынка);
- поставщики услуг ИС/ИТ.

Нижний слой — это индивидуальные цели. Они отражают личностно ориентированные точки зрения и потребности людей, так или иначе вовлеченных в развитие ИС (владельцы бизнеса, руководители, функциональные менеджеры, специалисты и другие). Рассматриваются две категории:

- пользователи (потребители услуг ИС);
- ИТ-специалисты (поставщики услуг ИС).

В центре находятся цели ИС, учитывающие влияние всех четырех сегментов. Полученная в результате иерархия целей развития ИС представлена в форме диаграммы Ишикавы (рисунок 48).

На ней отражены главная бизнес-цель развития ИС и четыре ее составляющие:

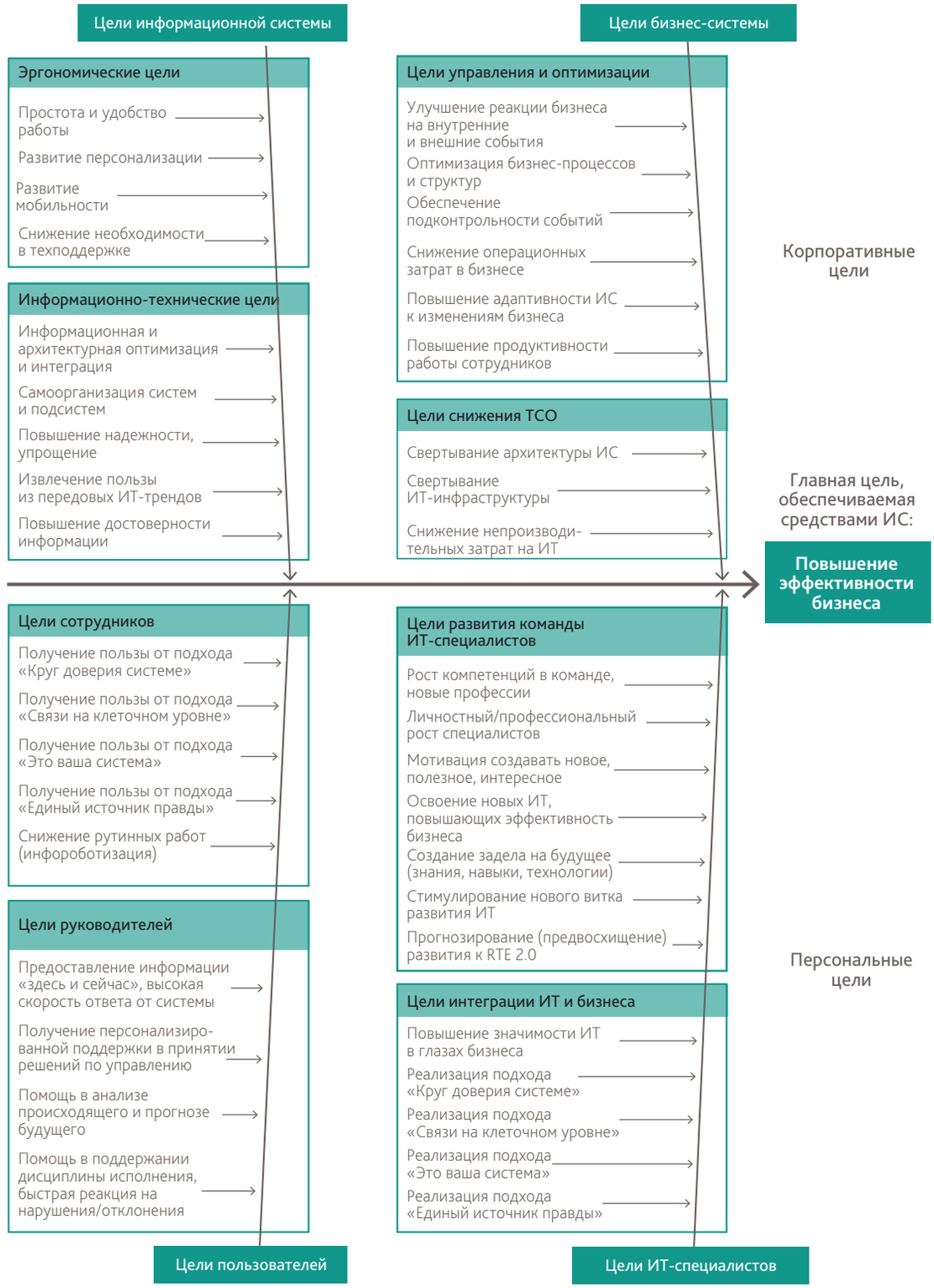
- корпоративные цели поставщиков ИТ-услуг (фокус на информационную систему предприятия);
- корпоративные цели потребителей ИТ-услуг (фокус на предприятие как бизнес-систему);
- индивидуальные цели поставщиков ИТ-услуг;
- индивидуальные цели потребителей ИТ-услуг.

Цели бизнеса и поставщиков услуг группируются на уровне корпоративных целей (указаны вверху диаграммы), а цели пользователей и ИТ-специ-

[12] Отметим, что, когда мы стали формулировать идеальную систему — «образ будущего» для ИС, стало понятно, что многие спрогнозированные свойства такой системы касаются не только парадигмы RTE2.0, но могут быть основой для более общего понятия «цифровой платформы предприятия».

Рисунок 48.

Иерархия целей развития ИС в форме диаграммы Ишикавы



алистов сгруппированы в персональные цели (внизу диаграммы). Таким образом, мы получаем иерархию целей, наиболее полно отражающую интересы как самого предприятия, так и его руководителей, сотрудников, специалистов, заинтересованных в адекватном продуктивном взаимодействии с ИС.

Ограниченные рамки книги не позволяют привести детализацию всех целей — полная иерархия достаточно обширна и к тому же привязана к условиям конкретного предприятия. Акцентируем внимание лишь на том, что прогноз развития системы должен быть связан с целями.

Отметим также, что при разработке целеполагания стоит учитывать тенденции в надсистемном окружении, такие как современные технологические тренды:

1. Тенденции, значительно изменяющие бизнес и рынок:

- дальнейшее развитие технологии in-memory и реализация на ее базе, кроме СУБД, еще и серверов приложений, систем бизнес-аналитики, бизнес-моделирования, а также ERP, CRM, SCM (в нашем случае на базе развития решений SAP);
- взрывное распространение мобильных устройств и мобильных приложений;
- «интернет вещей»;
- гибридные ИТ и облачные вычисления;
- «большие данные»;
- аналитика «на лету» в реальном времени;
- инфороботизация — выполнение рутинных бизнес-операций инфороботами.

2. Тенденции, значительно изменяющие пользовательское поведение:

- планшетные устройства (tablets);
- новые приложения и интерфейсы для мобильных устройств;
- BYOD («принеси свой девайс»);
- интерфейсы, изменяемые в зависимости от контекста;
- развитие социальных медиа и их интерфейсов;
- индивидуальный интернет-бизнес, удаленная работа, «отвязка» работника от офиса и работодателя;
- персональные киберпомощники.

В нашем случае практически все эти тенденции были учтены, то есть результирующие прогнозные гипотезы и конструктивные решения на их основе проверялись на соответствие этим тенденциям.

Закономерности технической эволюции для технологии in-memory

Закон трехэтапного развития системы по S-кривой

Системы развиваются по S-образной кривой, охватывающей три этапа (рисунок 49):

- Первый этап — «рождение и детство», эксперименты, start-up, выход на рынок (именно этот этап развития системы описывает известный Hype Cycle компании Gartner).
- Второй этап — «рост и взросление», промышленная эксплуатация, широкое признание рынком.
- Третий этап — «угасание и старость», «дотягивание» характеристик, неизбежный уход с рынка.

Технология in-memory находится на выходе из первого отрезка S-кривой, то есть в ближайшее время ожидается переход с первого этапа («рождение и детство») на второй этап развития («рост и взросление»).

На рисунке 50 приводится оценка, сделанная компанией Gartner в 2012 году, которая детализирует положение различных применений технологий in-memory. На рисунке указаны некоторые применения технологии in-memory, находящиеся на разных этапах развития.

На расстоянии от двух до пяти лет до достижения «плато продуктивности» находятся следующие продукты:

- аналитические СУБД in-memory;
- инфраструктурные приложения, обеспечивающие работу технологии in-memory;
- распределенные сети данных in-memory;
- бизнес-аналитика на базе in-memory;
- технология DDR4 DRAM.

Из этого следует, что основанные на технологии in-memory информационно-аналитические системы уже через два-три года будут широко внедряться многими компаниями.

На расстоянии от пяти до десяти лет до достижения «плато продуктивности» (начала масштабного признания рынком) находятся следующие продукты:

- серверы приложений in-memory;
- СУБД in-memory для OLTP;
- серверы, использующие флэш-память как дополнительную;
- инфраструктура для высокоскоростного обмена сообщениями;
- магниторезистивная RAM;
- твердотельные устройства;
- обработка сложных событий «на лету».

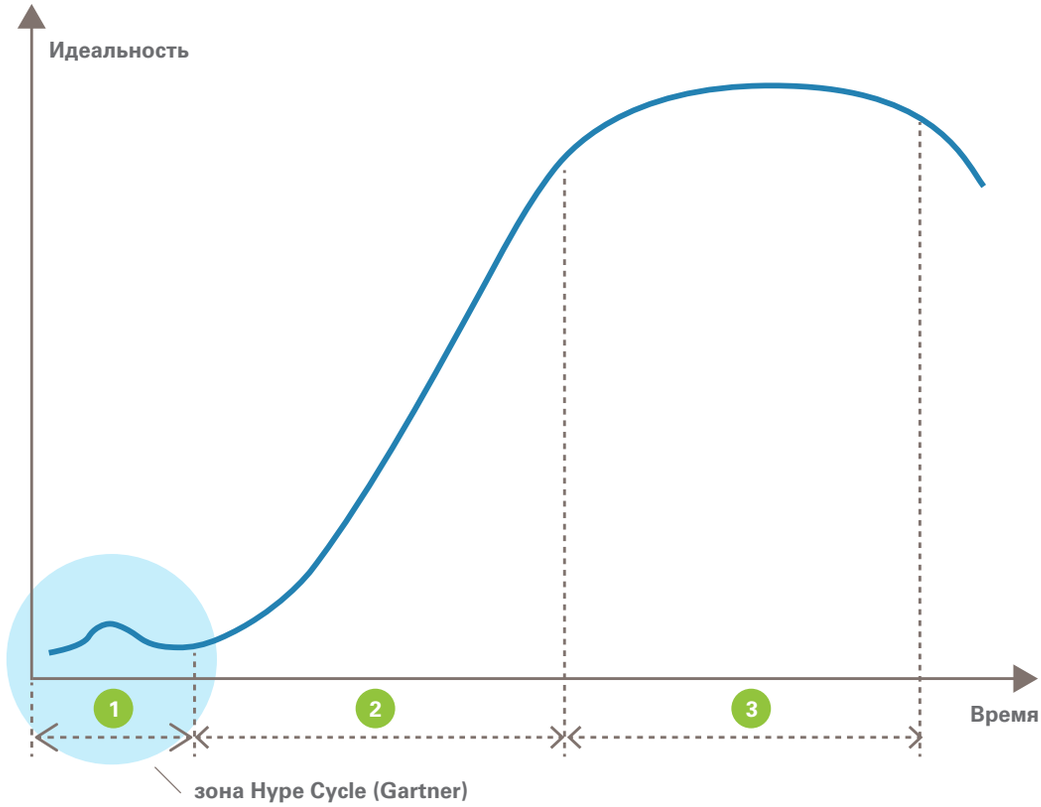


Рисунок 49.

Закон трехэтапного развития систем

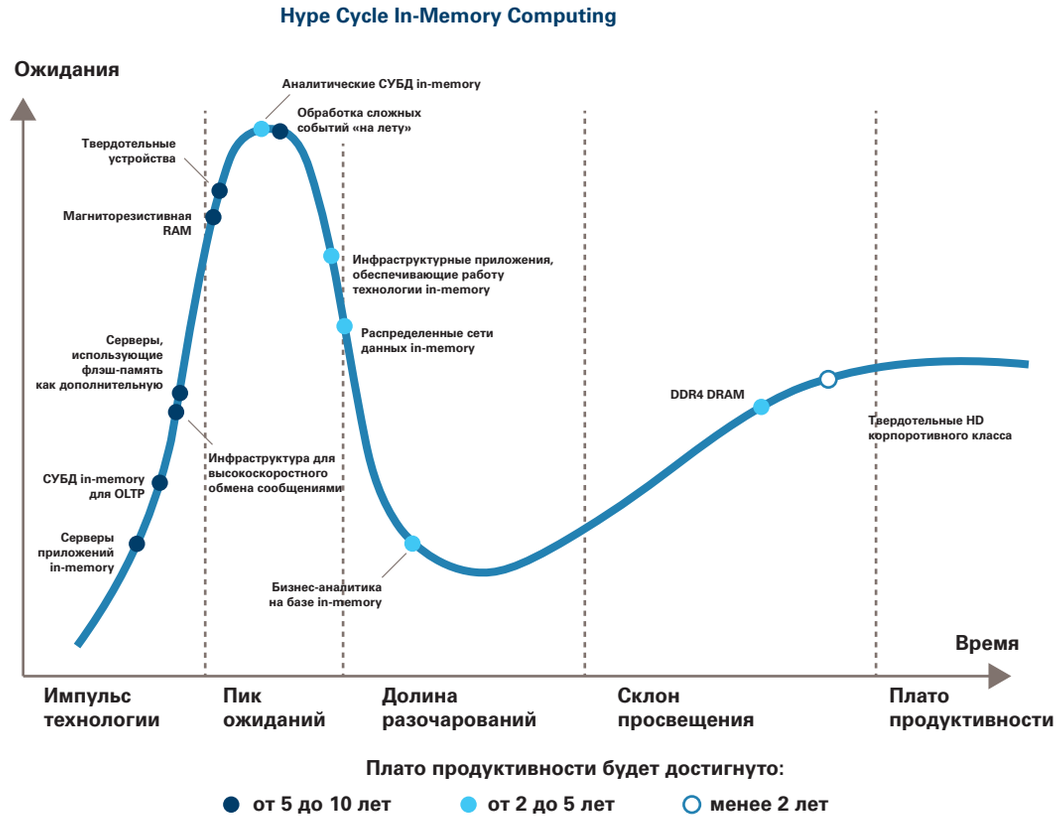


Рисунок 50.

Hype Cycle

Закон полноты частей системы

Этот закон означает, что архитектура информационной системы должна включать четыре части: «источник», «преобразователь», «завершитель» и «управление». В информационно-аналитической системе, которая использует технологию in-memory, эти компоненты представлены следующим образом (рисунок 51):

1. *«Источник»* формирует ресурсы для выполнения главной полезной функции системы и выработки ее продукта. Главной полезной функцией информационно-аналитической системы является поддержка бизнес-аналитики, продуктом — аналитические отчеты, а ресурсами — данные из транзакционных систем. Источником являются системы SAP ECC, а также компонент SLT, реплицирующий данные в «преобразователь».
2. *«Преобразователь»* превращает ресурсы в продукт. В информационно-аналитической системе преобразование транзакционных данных в аналитические модели для последующего преобразования в продукт (аналитические отчеты) происходит в SAP HANA и BW on HANA.
3. *«Завершитель»* после окончания выполнения главной полезной функции системы передает продукт потребителю. В информационно-аналитической системе «завершитель» — SAP BO — на основе данных и правил, содержащихся в аналитических моделях преобразователя, формирует продукт — аналитические отчеты, запрашиваемые пользователем.
4. *«Управление»* планирует, координирует, контролирует, анализирует, оповещает и дает обратную связь. В информационно-аналитической системе функции управления выполняет «Репозиторий бизнес-объектов».

В развитой системе, включающей все четыре части, приоритет в развитии должен быть отдан «завершителю». Именно на его развитие необходимо направлять основные ресурсы, ведь он является как бы «полномочным представителем» системы перед потребителем. В ИТ-системе «завершитель» — это пользовательский интерфейс, и ярким примером пренебрежения этим правилом служит преувеличенное внимание к разработке разнообразных внутренних «фишек» в системе, в то время как интерфейсы пользователей остаются неудобными и устаревшими.

Таким образом, одно из важнейших направлений развития рассматриваемой ИС — совершенствование интерфейсной части.

Закон развертывания-свертывания

Технология in-memory — это отличный пример свертывания системы. Можно сказать, что предшествующая ей традиционная технология СУБД

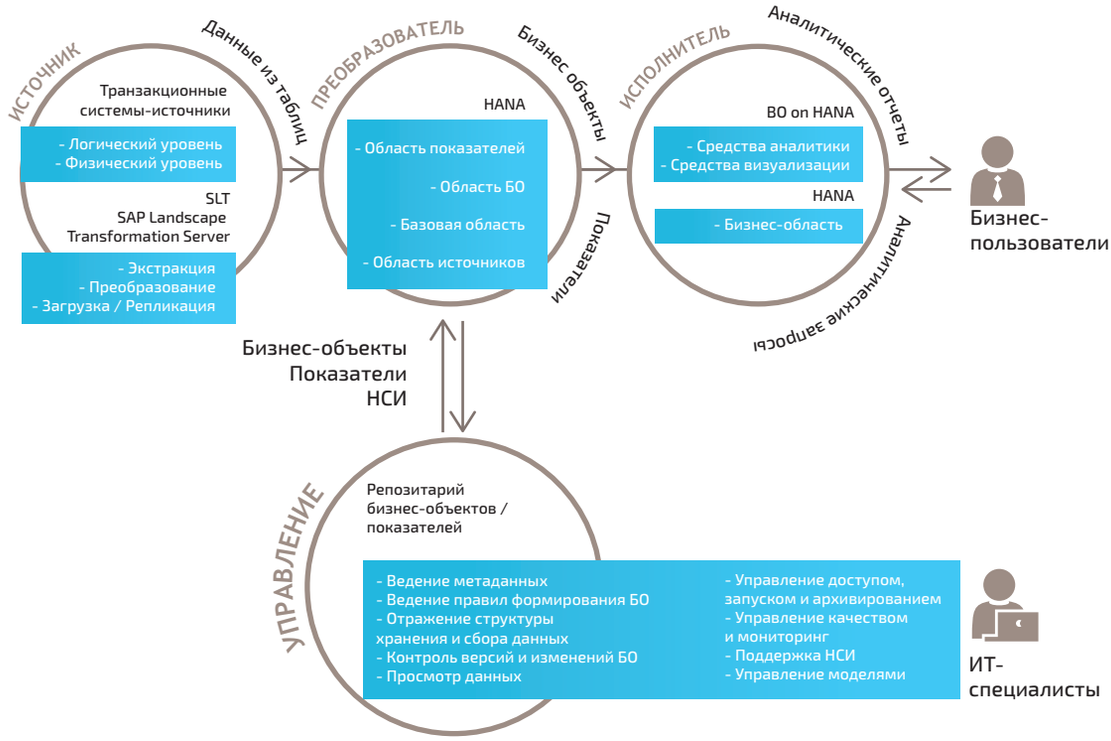


Рисунок 51.

Полнота частей системы

«свернулась»: устройства дисковой памяти, чтения/записи на диск и ввода/вывода объединились, слились в единое целое. При этом произошло:

- увеличение полезных факторов: на два порядка возросло быстродействие, улучшилась энергоэффективность;
- уменьшение негативных факторов: исчезли «лишние» подсистемы, обеспечивающие функции дисковой памяти, чтения/записи на диск, ввода/вывода.

Свертывание произошло за счет использования обработки в оперативной памяти, поколоночных структур данных и параллелизма для поддержки многоядерных архитектур. In-memory обеспечивает нужное для параллелизма расположение данных близко к ядрам в локальной памяти, колоночные структуры данных эффективны при вводе/выводе и являются необходимым условием для сжатия: их можно сжимать гораздо более эффективно, чем строчные данные. Отметим, что свертывание с помощью технологии in-memory — это реализация только одной из линий развития, одного из направлений повышения идеальности ИС.

Но не всякое свертывание и упрощение, ведущее к снижению сложности системы, дает эффект. Приведем пример частичного свертывания системы, которое не дало существенных результатов. В 2008–2011 годах мы были свидетелями такой попытки свертывания ИС. Речь идет об интегрированных (конвергентных) программно-аппаратных системах, например Oracle Exadata Database Machine и IBM PureSystems. Основная их идея — объединение в единой системе оборудования,

ПО и вспомогательных сервисов, которые заранее «плотно подгоняются» друг к другу поставщиком в расчете на определенный профиль загрузки. Это должно было обеспечить сокращение затрат и времени на развертывание и обслуживание систем, а также увеличение производительности. Такая идея казалась многообещающей, и в 2012 году аналитики Gartner даже включили интегрированные системы в топ-10 стратегических технологий. Тем не менее никакого существенного влияния эти технологии на сокращение затрат и времени не оказали. И тому есть объяснение.

Свертывание — это кардинальное улучшение системы путем отказа от ненужно-избыточных компонентов с переносом их функций на другие компоненты, в том числе за границы системы. При свертывании (особенно в частном случае объединения альтернативных или даже конкурирующих систем) получаются «прорывные» бескомпромиссные решения, резко увеличивающие основные характеристики системы и обеспечивающие новое системное качество.

Интегрированные (конвергентные) программно-аппаратные системы — это максимально оптимизированная и четко сфокусированная на определенных задачах комбинация ПО и оборудования. Произошло ли свертывание системы? Да, произошло, но частичное.

Интегрированные программно-аппаратные системы — это двухслойная полисистема. Нижний аппаратный слой был свернут, однако верхний — программное обеспечение — остался «несвернутым», количество подсистем не сократилось. А поскольку главную функцию, непосредственно направленную на пользователя, выполняет именно верхний слой, то свертывание нижнего слоя практически никак не сказалось на положительных свойствах системы, кроме некоторого сокращения затрат на аппаратную составляющую. Новое системное качество не было получено, поэтому и эффект от интегрированных систем оказался весьма скромным.

Также если заменить традиционную СУБД на in-memory СУБД в уже имеющихся ERP-системах, без их глубокой переработки, то мы получим лишь частичное свертывание, и эффект от этого также будет минимальный. Необходима радикальная переработка обеих систем — СУБД и прикладной системы, работающей на ее базе, только тогда мы получим полноценное свертывание.

Это позволит вывести эффект от свертывания нижнего слоя на уровень всей системы и получить новое системное качество.

Надо сказать, что такое свертывание кое-где произошло, но не в области бизнес-приложений, а на периферии — в системах управления доступом и входа/выхода в Интернет. В наше время они уже практически не поставляются в виде отдельного управляющего сервера и специального оборудования, а имеют вид интегрированного программно-аппаратного комплекса.

Развитие ИС в направлении к RTE2.0

При исходных условиях, имея цель развития ИС, образ RTE2.0 и используя законы технической эволюции, можно сделать экспресс-прогноз развития ИС.

Прежде всего, он опирается на формулировку идеальной системы.

Теоретически «самая идеальная» система имеет бесконечное количество полезных функций в числителе и полное отсутствие вредных факторов в знаменателе. Имея в мысленном фокусе такое представление о будущей системе RTE2.0 и анализируя систему по «линиям развития», можно спрогнозировать некоторые ее будущие свойства. В нашем случае мы опирались главным образом на линию свертывания, но были и другие линии развития.

На наш взгляд, многие виды корпоративных ИС приблизились к точке максимального развертывания, системы достаточно развились и сильно усложнились, затраты и риски возросли многократно. Это дает нам право предполагать, что теперь целый ряд систем должны начать свертывание — новый этап преобразований, связанный с уменьшением факторов риска и затрат.

Вместо вопроса, какой новый компонент нужно создать для выполнения данной функции, разработчики все чаще должны будут задаваться вопросом: какой уже существующий компонент или сервис может взять на себя данную функцию? При этом увеличится многократное использование одних и тех же функциональных элементов ИС, удельная функциональная нагрузка на них, а значит, повысится эффективность функционирования. Поэтому фокус в нашем прогнозе был направлен прежде всего на линию свертывания.

Но развитие различных частей ИС идет неравномерно, и другие части системы находятся в состоянии развертывания, поэтому часть прогнозов сделана и в этом ключе. Кроме того, важными для реализации парадигмы RTE2.0 представляются и другие линии развития: изменения в человекоориентированности, увеличение степени динамичности и управляемости системы и изменения в системном окружении.

Прогноз по линии свертывания

1. Свертывание вычислительных ресурсов.

В настоящее время мы наблюдаем тенденцию свертывания именно аппаратной части, причем не только в виде технологии in-memory, но и в многочисленных попытках создания интегрированных устройств (appliances). Вероятнее всего, вычислительные ресурсы будут продолжать сворачиваться. Дальнейшее развитие технологии in-memory приведет к резкому снижению потребности в дисковых ресурсах (за счет размещения данных в оперативной памяти и их сжатия), исключению дорогостоящих высокопроизводительных дисковых массивов (остаются

флэш-память для энергонезависимой копии и простые дисковые массивы для исторических данных). За счет снижения удельной вычислительной мощности на аналитический запрос уменьшаются требования к производительности серверов баз данных. Так как вычисления производятся непосредственно на сервере СУБД, требуется существенно меньшая производительность серверов приложений.

Количество требуемых систем/серверов также уменьшается, так как исключаются элементы ИТ-архитектуры: хранилища данных, системы загрузки, трансформации и очистки данных и прочие. Соответственно, снижаются требования к сетевой инфраструктуре, объему потребляемой электроэнергии, мощности систем кондиционирования. С уменьшением количества серверов исчезает необходимость виртуализации вычислительных ресурсов, которая на данный момент «съедает» от пяти до двадцати процентов производительности оборудования. В идеале инфраструктура корпоративной информационной системы сворачивается до двух серверов, в реальности потребуется чуть больше — от четырех до шести взамен существующих десятков серверов и нескольких дисковых массивов класса hi-end. Кроме того, вычислительные ресурсы сворачиваются за счет частичного перехода в надсистему — приложения могут выполняться в мощных облачных дата-центрах за пределами ЦОД предприятия.

2. Свертывание компонентов ИТ-архитектуры.

Свертывание различных компонентов систем, вероятнее всего, будет продолжаться. Системы будут эволюционировать путем свертывания подсистем. Вероятно, будут сворачиваться некоторые функциональные компоненты архитектуры, то есть произойдет перенос функций с одних компонентов на другие (некоторые компоненты становятся ненужными, некоторые объединяют свои функции с другими компонентами — в результате система упрощается). Например, операционная система и СУБД могут быть свернуты в единый компонент. Сегодняшние ОС разработаны для решения очень широкого круга задач (не только задач СУБД), и если создать специализированный комплекс ОС + СУБД (ограничить ОС только обслуживанием задач СУБД in-memory), то сложность такого решения снизится на порядок. К тому же в процессе свертывания происходит многократное использование одних и тех же функциональных элементов: повышается эффективность функционирования, ликвидируется дублирование функций, происходит унификация типовых операционных процессов.

3. Свертывание систем, связанных с бизнес-аналитикой.

Свертывание произойдет и на уровне бизнес-аналитики — как отдельный комплекс приложений этот уровень, скорее всего, перестанет существовать. В частности, становится ненужным разделение на транзакционные

и аналитические БД, можно исключить хранилища данных, системы загрузки и трансформации (ETL), системы обработки цепочки сервисов.

Уровень бизнес-аналитики свернется в единую с транзакционным уровнем платформу in-memory, где расчеты и аналитика ведутся «на лету». Кроме ускорения аналитической обработки данных в сотни раз (за секунды вместо часов) и уменьшения объемов хранимой информации в пять-десять раз, для управления предприятием появляются совершенно новые возможности, которые описаны в следующих пунктах.

4. Свертывание приложений.

Свертывание продолжится и на уровне прикладных систем. Вероятно, отдельных функциональных приложений в «свернутой» ИС не будет. В одной системе будут реализованы вся необходимая функциональность и все необходимые технологии корпоративного управления. Развитие парадигмы «ПО как услуга» (SaaS) предположительно также будет происходить в направлении свертывания отдельных сервисов в интегрированный набор. Уже сейчас имеется технология предоставления гибких информационных сервисов вместе с обычными жестко детерминированными функциональными компонентами (например, сервисно ориентированная архитектура, SOA), и эта технология будет развиваться, преодолевая существующие недостатки SOA.

Возможно, обособленные функционально ориентированные приложения свернутся в самоорганизующийся конгломерат неких мгновенно собираемых по запросу пользователя информационных сервисов, обеспечивающих получение любых необходимых ему данных и выполнение нужных действий по управлению предприятием.

5. Свертывание систем обработки потоков событий.

По всей видимости, свернутся и системы, связанные с обработкой потоков событий. Обработка будет вестись по данным, поступающим в единую транзакционную СУБД прямо с датчиков на технологическом оборудовании. Часть функций систем АСУТП/SCADA в сегодняшнем понимании становятся ненужными, сворачиваются; анализ и управление технологическими процессами будут вестись непосредственно на оперативных технологических данных без необходимости использования отдельных СУБД, обеспечивая мгновенную обратную связь. Исчезает громоздкая инфраструктура для ввода данных — они поступают напрямую от любых источников, очищаются и проверяются непосредственно во время работы. В результате мы получим:

- практически мгновенную регистрацию и обработку любых событий — как простых, так и сложных;

- поступление данных с приборов автоматически, без вмешательства человека;
- регистрацию большинства (в идеале всех) событий в ходе бизнес-процессов;
- единый источник исходных данных для всей деятельности.

6. Свертывание систем поддержки принятия решений.

Свертывание систем, связанных с бизнес-аналитикой, вероятно, приведет к тому, что системы поддержки принятия решений перестанут быть отдельным классом ПО, их функциональность будет встраиваться в системы, поддерживающие каждый критически важный бизнес-процесс. Эти локальные подсистемы поддержки решений будут анализировать не только текущую информацию, но и события в бизнес-процессах, осуществлять как мониторинг выполняемых бизнес-процессов, так и прогноз возможных событий. Станет возможным вести мониторинг истории и траектории управленческих решений: какое именно решение было предложено, какое принято, как оно выполнялось, каковы результаты и последствия.

7. Сокращение стоимости владения системой.

Стоимость владения системой существенно уменьшится за счет свертывания ставших ненужными подсистем. В частности, исключение хранилищ данных, систем загрузки и трансформации, сокращение числа СУБД ведет к снижению стоимости владения ИС.

Прогноз по линии развертывания

1. Сложные запросы и «мгновенный снимок» бизнеса.

Радикальное улучшение систем, связанных с бизнес-аналитикой, дает принципиально новые возможности, качественно улучшающие полезные свойства ИС — функциональность систем развертывается.

Станет возможным иметь единый оперативно-исторический отчет о бизнесе, без ограничений по глубине и объему аналитических выборок по номенклатуре продукции, складам, датам, любым другим параметрам. Такие аналитические запросы могут тысячекратно усложниться в сравнении с «эпохой хранилищ данных» и разделения на OLAP/OLTP.

«Закрытие периода» (получение актуального баланса) на предприятии может выполняться даже после каждой транзакции, то есть предприятие сможет практически в любой момент иметь «мгновенный снимок» своего бизнеса.

2. Развертывание корпоративного «интранета вещей».

Радикальное улучшение систем обработки потоков событий и возросшая скорость обработки данных дадут возможность на порядки

увеличить количество данных, которые собираются в информационной системе.

Технологическое оборудование предприятия и потребительские устройства будут интегрироваться в корпоративный интранет через развернутую сеть встроенных сенсоров по технологии RFID (см. выше «Свертывание систем обработки потоков событий»).

Вероятно, это развертывание будет основываться на системе передачи данных нового поколения — полностью беспроводной, высокоскоростной, с пропускной способностью, близкой к real time, поскольку требуется мгновенная передача данных между всеми объектами. Здесь возникают действительно «большие данные».

3. Развертывание виртуальной модели предприятия.

Дальнейшее развертывание корпоративного «интранета вещей» приведет к тому, что в идеале каждому материальному объекту реального предприятия будет соответствовать его виртуальный двойник. Иначе говоря, все информационные бизнес-объекты в системе (понятия, явления, предметы транзакций, учета, аналитики) будут точно отражать свойства и поведение реальных соответствующих объектов предприятия — например, продуктов, заказов, сотрудников, договоров — и связи между ними. Каждый объект, существующий в реальном мире предприятия, будет иметь в виртуальном пространстве своего «информационного дублера», в точности отражающего его поведение.

Возникает действующая виртуальная модель предприятия, в которой будут содержаться и пополняться все знания о предприятии, и каждый сможет оперировать этими знаниями в своих интересах и на своем уровне, в том числе моделировать ситуации, анализировать и прогнозировать развитие событий. В результате:

- каждое событие будет сразу отражаться во всех нужных ракурсах (в производстве, в финансах, на материально-технических ресурсах, в работе персонала и в других сферах);
- появится возможность практически мгновенного доступа к любому произошедшему событию в любом интервале времени и в любом ракурсе;
- станет возможным поиск, исправление и предупреждение любых негативных явлений в системе (ошибок, сбоев, аварий, проблем и тому подобного);
- станет возможной работа по принципу «я хочу делать это (указать рабочий процесс)», а не «я хочу воспользоваться таким-то приложением».

По всей видимости, такая детализация позволит заметно улучшить качество принимаемых решений. Корпоративный «интранет вещей» позволит сформировать виртуальную копию реального мира предприятия

в реальном времени, в которой каждый сотрудник сможет мгновенно и из любой географической точки получить любую запрошенную информацию о предприятии в удобной ему форме (в соответствии с правами доступа).

4. Виртуальные помощники.

Мгновенная аналитика, корпоративный «интранет вещей», виртуальная модель предприятия и другие технологические изменения дадут возможность создавать новые полезные функциональные компоненты — виртуальных помощников, ассистентов-инфокиберов. Ими могут быть, например, «виртуальный аудитор» финансового департамента, «инспектор ТБ», «контролер качества», «риск-аналитик» и другие. В результате появляются возможности:

- прогнозирования событий и моделирования возможных ситуаций, предложения вариантов решений;
- использования искусственного интеллекта для аналитики, прогнозирования, СППР и экспертных подсистем, возможности сценариев «а что, если?».

Прогноз изменений в человекоориентированности

Как мы писали выше, одно из важнейших направлений развития рассматриваемой ИС — совершенствование интерфейсной части. Здесь, вероятнее всего, произойдут следующие изменения:

- максимальное освобождение человека от исполнения рутинных функций, интеллектуальная обработка информации (все рутинные операции будут высокоавтоматизированными, что повлияет на изменение ролей пользователей — например, бухгалтеры становятся аудиторами);
- ИС будет проста и понятна для пользователя, станет обладать удобным интуитивным интерфейсом (операционные инструкции будут не нужны);
- ИС будет вызывать желание с ней работать, станет эстетичной и эргономичной;
- пользователю будет выдаваться только необходимая информация, без лишних данных; 90% информации будет обрабатываться автоматически инфороботами;
- появится возможность двустороннего общения с системой на естественном языке (речь, текст), самообучения системы (лексический анализ);
- ИС будет уметь оптимально распределять полномочия пользователей;
- ИС будет предугадывать потребности пользователя и в опережающем порядке предоставлять соответствующие функциональные возможности; на любой вопрос пользователя в системе будет адекватный ответ;
- ИС будет поддерживать инженерное творчество, позволять функциональному пользователю и ИТ-специалисту развиваться профессионально.

Прогноз изменений в системном окружении и в надсистеме

При развитии информационных систем в рассмотренных направлениях, когда аналогичные изменения происходят на многих других предприятиях и в организациях, ИС конкретного предприятия становится более интегрированной с бизнес-партнерами, государством и рынком. Например:

- информационные системы поставщиков, работающие на тех же принципах и интегрированные с ИС предприятия, вероятно, смогут предоставлять мгновенную прозрачную информацию о состоянии процесса поставок, возможных задержках и сбоях в логистике, мониторинге движения грузов и многом другом;
- потребители, скорее всего, смогут общаться с ИС предприятия через мобильные устройства (делая заказы, осуществляя закупки, получая отчеты и консультации);
- ИС налоговых и регулирующих органов будут иметь возможность получать статистику и регламентированную отчетность через Интернет, проводить быстрые налоговые и иные контрольные проверки и так далее.

Прогноз увеличения динамичности и управляемости системы

Это важнейшее направление развития ИС. Сейчас лишь по некоторым технологическим новшествам можно судить о тех изменениях, которые произойдут в этом направлении. Прежде всего, это технологии анализа контекста работы пользователей — вероятно, появятся системы, которые смогут «общаться» с окружением, посылая и принимая информационные сообщения от корпоративного «интранета вещей». Они смогут распознавать и понимать окружение, в котором они находятся, самообучаться и функционировать автономно.

В ходе проектной мастерской были получены некоторые идеи, которые мы сочли полезным включить в прогноз.

1. Адаптивность:

- вероятно, произойдет существенное повышение трансформируемости и настраиваемости системы, при необходимости все параметры будут быстро и просто меняться;
- пользователь будет вносить все необходимые ему изменения в поведение системы без программирования, с помощью простого, но гибко настраиваемого персонализированного интерфейса.

2. Самоорганизация:

- будут кардинально улучшены функции самообслуживания и самодиагностики для выявления/устранения сбоев;

- будет повышена степень самоуправления и самооптимизации системы: она будет автоматически себя изменять — подстраиваться под изменения законодательства, получит возможность самомодификации по требованиям пользователей для обеспечения функционирования при изменениях как внешних факторов (решаемых бизнес-задач), так и внутренних факторов (недостаточный объем ресурсов, рост объема данных).

Итоги прогнозирования

Подводя итог экспресс-прогноза развития систем по направлению к «предприятию реального времени» с использованием проектной мастерской и ряда положений метода Directed Evolution, можно выделить ряд ключевых предположений:

1. Приложения превращаются в «свернутое ПО». Пользователи смогут в реальном времени получить любую аналитику по предприятию, включая «мгновенный снимок бизнеса», прогноз показателей деятельности, предупреждения о возможных событиях или о возникающих рисках.
2. Существенно упрощается и удешевляется вычислительная инфраструктура, но одновременно развертывается инфраструктура съема данных с датчиков для корпоративного «интранета вещей», образующего действующую в реальном времени виртуальную модель предприятия.
3. На предприятии появляются информационные роботы-киберпомощники: «виртуальный аудитор», «виртуальный инспектор», «виртуальный контролер качества» и другие.

Следующим важным шагом к идеальной системе станет полная смена существующей парадигмы вычислений — переход от архитектуры фон Неймана к искусственным нейронным сетям. Нейронные сети уже выходят из «детского возраста», их основная проблема — низкая скорость «обучения» — уже решается, и практическое применение нейронных сетей в корпоративных системах возможно уже в ближайшем будущем.

Но это уже следующая S-кривая развития информационных систем.

Шаг 2. Выбор варианта развития для конкретной системы

На этом шаге среди возможных вариантов развития производится выбор одного, наиболее реализуемого имеющимися ресурсами системы и ее окружения в рамках выбранного горизонта планирования (в нашем случае — пять лет).

Обзор выявленных направлений развития

На предыдущем шаге выявлено более десятка возможных вариантов развития по направлению к «идеальной системе»:

1. Свертывание вычислительных ресурсов.
2. Свертывание компонентов ИТ-архитектуры.
3. Свертывание систем, связанных с бизнес-аналитикой.
4. Свертывание приложений.
5. Свертывание систем обработки потоков событий.
6. Свертывание систем поддержки принятия решений.
7. Сокращение стоимости владения системой.
8. Появление возможности «мгновенного снимка бизнеса».
9. Развертывание корпоративного «интранета вещей».
10. Развертывание виртуальной модели предприятия.
11. Появление «виртуальных помощников».
12. Изменения в человекоориентированности:
    - максимальное освобождение человека от исполнения рутинных функций;
    - простота и понятность для пользователя, удобный интуитивный интерфейс;
    - желание работать с системой: она эстетична, удобна и эргономична;
    - пользователю выдается только необходимая и достаточная информация, без лишних данных; 90% информации обрабатывается автоматически;
    - возможность двустороннего общения с системой на естественном языке (речь, текст);
    - возможность самообучения системы;
    - возможность удовлетворять все информационные потребности пользователя и предоставлять соответствующие функциональные возможности; на любой вопрос пользователя в системе будет адекватный ответ.
13. Увеличение динамичности и управляемости системы:

Адаптивность, гибкость:

- существенное повышение трансформируемости и настраиваемости системы: при необходимости все параметры меняются быстро и просто;
- пользователь вносит все необходимые ему изменения в поведение системы без программирования, с помощью простого, настраиваемого персонализированного интерфейса.

Самоорганизация:

- самообслуживание и самодиагностика;
- самоуправление и самооптимизация: система автоматически подстраивается под изменения внешней среды и по требованиям пользователей — при изменениях бизнес-задач и технологических факторов вроде недостатка системных ресурсов или роста объема данных.

Выбранный приоритетный вариант

Проанализировав все перечисленные направления, возможности и насущные потребности, мы решили создать новую цифровую платформу предприятия и для этого сосредоточиться на варианте, указанном первым пунктом в разделе «Итоги прогнозирования»: *«Приложения превращаются в «свернутое ПО».*

Этот вариант позволяет решить обозначенные проблемы повышения адаптивности и удобства информационных систем, не изменяя их аппаратную часть, но кардинально изменяя подход к созданию и функционированию ПО. В этом варианте интегрируются пункты: 4 «Свертывание приложений», 10 «Развертывание виртуальной модели предприятия», 12 «Изменения в человекоориентированности», 13 «Увеличение динамичности и управляемости системы».

«Свернутое ПО» должно обладать такой структурой и свойствами самоорганизации, которые обеспечивали бы следующие способности:

- мгновенно выполнять любой запрос пользователя к системе вне зависимости от того, насколько он сложен, касается ли он учетных/транзакционных процедур или же аналитики и принятия управленческих решений;
- легко изменять функциональные возможности информационной системы при изменении условий/требований внешней среды (предметной области информатизации, бизнес-окружения);
- воспринимать задачи пользователя в форме устной речи или свободного текста.

Принимаем этот вариант и переходим к следующему шагу — начинаем генерировать идеи инноваций, обеспечивающие решение проблем и получение работоспособной архитектуры системы нового поколения.

Шаг 3. Синтез необходимых инноваций для выбранного варианта развития

На этом шаге производится формулирование и синтез необходимых инноваций в системе, обеспечивающих реальную возможность желаемого

развития (в нашем случае — развития системы в сторону новой цифровой платформы предприятия со «свернутым ПО» в парадигме RTE2.0).

Для этого используем методологию синтеза цифровых инноваций на базе инструментария ТРИЗ, предназначенную для создания изобретений в области цифровизации и информационных технологий.

Идеальная система: «сказал — получил»

Идеальная система должна работать по принципу «сказал, что требуется,— мгновенно получил желаемое». Пользователь не должен думать о том, в каком именно функциональном приложении он должен работать (например, в системах SAP, таких как BO, FI, CO, HR, SCM, CRM и подобные). Пользователь должен получить способ удобно, быстро и эффективно выполнять свои служебные обязанности и рабочие операции без необходимости изучать функции многообразных приложений и овладевать навыками работы в них. Он просто должен сказать системе: «Хочу получить это»,— поэтому пользовательский интерфейс в новой системе должен иметь голосовой ввод или простое окно для ввода свободного текста.

Такой подход назван нами You Get What You Asked (YGWYA) — ИГВИЯ.

Поэтому в идеальной системе надо отказаться от прежней концепции «приложений» и перейти к концепции «мгновенных инфоспектаклей», возникающих в тот самый момент и только для той операции, которая нужна пользователю «здесь и сейчас».

Идеальным интерфейсом для системы должен быть речевой ввод, как основной и преимущественный способ диалога «человек — машина».

Принцип перехода на микроуровень: «атомарная микросистема»

Прием «дробления» в ТРИЗ. У-Блок как мини-АРМ.

Каким образом могут быть «свернуты» существующие сегодня приложения с детерминированным набором функций? Нужен самоорганизующийся конгломерат неких мгновенно собираемых по запросу пользователя информационных сервисов, обеспечивающих любые необходимые ему действия в рамках деятельности предприятия, а также обеспечивающих возможность ИТ-специалистам легко изменять ПО при изменении внешних условий. Итак, необходим механизм, обеспечивающий функционирование ПО при постоянно меняющихся задачах пользователей и изменяющихся условиях деятельности предприятия.

Используем для получения базовой идеи решения тризовский принцип «дробления», рекомендующий «увеличить степень дробления объекта».

Этот принцип нам подсказывает, что информационное пространство предприятия должно быть раздроблено на атомарные информационные микросистемы-«кирпичики».

назовем такой кирпичик Умным блоком, У-Блоком.

У-Блок — это микросистема в составе «большой системы», некий микро-АРМ, атомарная ячейка. Множество У-Блоков составляют микроуровень «большой системы», компонентную основу нужного нам самоорганизующегося конгломерата мгновенно собираемых по запросу пользователя информационных сервисов.

Из чего будет состоять и как будет работать такой У-Блок, является предметом наших дальнейших рассуждений. В них нам поможет закон технической эволюции, обобщенно описывающий компонентный состав любой системы в форме требования соблюдения «полноты частей системы».

Закон полноты частей системы: трехуровневая иерархия взаимоподобных компонентов архитектуры

Формулировка этого закона звучит следующим образом:

«Необходимым условием принципиальной жизнеспособности системы является наличие и минимальная работоспособность ее четырех основных частей: «источника», «преобразователя», «завершителя» (иногда называемого «рабочим органом») и «управления»:

- «Источник» — формирует ресурсы для выполнения главной функции системы и выработки продукта системы.
- «Преобразователь» — превращает ресурсы в продукт.
- «Завершитель» — завершает выполнение главной функции системы и передает продукт потребителю.
- «Управление» — прогнозирует, планирует, координирует, контролирует, анализирует, оповещает и дает обратную связь о работе системы».

Для информационной системы этот закон означает, что ее архитектура должна включать все четыре указанных структурных компонента. Причем эта компонентная структура может повторяться по принципу подобия на разных уровнях системной иерархии: на верхнем (макроуровне), на среднем уровне и на нижнем (микроуровне).

Макроуровень — это информационная система в целом (рисунок 52).

«Источник» — содержит исходные данные и средства их ввода в систему.

«Преобразователь» — получает данные от «источника», команды/запросы системе от пользователя и от «управления» и формирует на этой основе нужную информацию (например, ответ системы на запрос пользователя).

«Завершитель» — получает информацию от «преобразователя», превращает его в понятную пользователю форму и передает результат (информационный продукт) пользователю; этим самым он заканчивает главную функцию системы.

«Управление» — управляет всеми вышеупомянутыми компонентами.

Средний уровень — это каждый из указанных выше компонентов.

«Источник», «преобразователь» и «завершитель» состоят каждый в свою очередь из своего «источника», «преобразователя», «завершителя» и «управления» (рисунок 53). Надо сказать, что сам компонент «управление» не подчиняется принципу подобия — его внутренняя структура иная, чем у трех других.

Подробно их состав прояснится в дальнейших рассуждениях, а пока лишь примем во внимание тот факт, что такая структура в системе должна быть «по определению». Этот подход дает опору для размышлений при последующем проектировании.

Микроуровень — это множество атомарных микросистем, У-Блоков.

Каждый У-Блок имеет унифицированную структуру, состоящую из следующих компонентов (рисунок 54):

- данные (из СУБД, от технологических датчиков и многие другие);
- программный код, формирующий функциональность сервиса («свойства» и «методы» в терминологии SOA);
- интерфейс (вход — голос и текст, выход — синтезированная речь, текст, графика, видео);
- средство управления.

Это микроаналог АРМ (автоматизированного рабочего места), в котором содержатся все необходимые и достаточные части для выполнения процесса «ввод данных — обработка данных — вывод результата».

Заметим, что при таком подходе остается нерешенной проблема, как обеспечить целостность системы на уровне предприятия. Рассмотрим ее позже, в последующих рассуждениях.

ММЧ и самоорганизация: «мгновенный спектакль микросистем»

Теперь надо решить, как организовать нужные У-Блоки, чтобы они мгновенно выполняли любой запрос пользователя.

Для поиска решения мы использовали три способа:

- метод ММЧ, предложенный Г. С. Альтшуллером, для снятия психологической инерции и активизации творческого воображения;
- требование идеальности «само собой, без ничего…»;
- стандарт из вепольного анализа[13] на самоорганизацию.

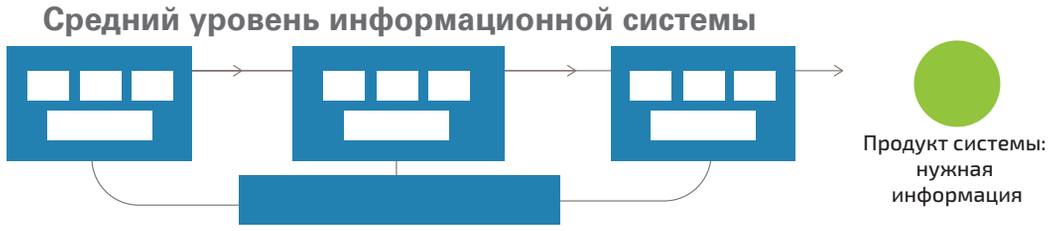
[13] Структурный вещественно-полевой (вепольный) анализ — раздел ТРИЗ, изучающий и преобразующий структуру технических систем.

Рисунок 52.

Макроуровень системы

Рисунок 53.

Средний уровень системы



Метод ММЧ (моделирование маленькими человечками)

В ТРИЗ одним из методов снижения психологической инерции мышления является метод моделирования «маленькими человечками» — ММЧ. На первый взгляд он может показаться слишком детским, несерьезным, ненаучным, но это не так. Метод вызывает яркие образы и ассоциации, уводя от стереотипных решений и привычных действий. Используя образ толпы маленьких человечков, легче представить себе модель системы или процесса. Замена «серьезных» инженерных компонентов «несерьезными человечками» раскрепощает творческое мышление, делает его более свободным. Еще до появления ТРИЗ подобные методы использовались многими исследователями.

Например, в эксперименте при разработке динамической теории газов известные «демоны Максвелла» открывали дверцу для горячих быстрых частиц газа и закрывали ее перед охлажденными, медленными. Структурную формулу бензола Фридрих Кекуле увидел в виде кольца, образованного из группы обезьян, ухватившихся за лапы и хвосты друг друга.

В нашем случае представим себе пользователя системы в виде зрителя спектакля-импровизации, который может давать любую тему для импровизации труппе маленьких человечков-актеров.

Человечки-актеры (то есть атомарные У-Блоки) принимают пожелание зрителя (запрос пользователя к системе), выходят на сцену и исполняют импровизированный мини-спектакль, раскрывающий заданную тему, таким образом отвечая на запрос пользователя, после чего уходят со сцены (отправляются в место хранения У-Блоков в системе).

Близкий аналог такого мини-спектакля — явление «флэшмоб» (мгновенная толпа), когда группа людей, договорившись через мобильные устройства, собирается в общественном месте и выполняет кратковременное согласо-

Рисунок 54.

Микроуровень системы

ванное действо (например, танец или пантомиму), после чего люди одновременно быстро расходятся в разные стороны, как ни в чем не бывало.

Вот, например, пользователь дает этой труппе человечков-актеров свой запрос: «Найдите сведения про контрагентов с наличием кредиторской задолженности в сумме более миллиона рублей по договорам, с которыми были случаи нарушения сроков поставки».

На этот запрос отзовутся актеры, исполняющие роли «знаток контрагентов» и «знаток договоров», и разыграют между собой импровизацию, описывающую, какие из известных им контрагентов имеют такие задолженности.

Мы сначала назвали такую импровизацию Smart-Brick Instant Performance, а потом переименовали в «Мангуст».

Например, У-Блок «Контрагенты» будет содержать данные «перечень всех контрагентов предприятия», но не будет содержать данные о договорах, которые заключены с ними. Такие данные будет содержать уже другой У-Блок, называемый «Договоры».

В традиционной СУБД для подобной функциональной области «управление договорными отношениями» необходимо жестко указать в модели данных связь упомянутых объектов — «контрагент» и «договор». Здесь такой модели данных не существует, соответственно, нет и такой жесткой связи. Нужная связь временно создается только в рамках конкретного «Мангуста», создаваемого в ответ на указанный выше запрос про задолженности и сроки поставки. Этот «Мангуст», собранный мгновенно, выполняет запрос и затем распадается за ненадобностью (или сохраняется при необходимости повторения).

Формулирование инновационной идеи для новой системы

Таким образом, для обеспечения большей адаптивности и гибкости информационных систем требуется разделить все информационное пространство предприятия на атомарные ячейки-микросистемы, которые имеют способность самособираться в мгновенный информационный сервис, обрабатывающий запрос пользователя и затем распадающийся обратно на составляющие.

На эту же идею выводит и тризовское требование «идеальности» — необходимые действия должны происходить «сами собой, без ничего», а также стандарт на самоорганизацию из вепольного анализа.

Такие мгновенно выполняемые по запросу «блиц-спектакли» с участием атомарных микросистем позволяют вообще отказаться от понятия «приложение» (software application).

Мы назвали их МАНГУСТ — от выражения «Мгновенно Адаптируемая и Направляемая Группа Умных Сервисов, Трансформируемая онтологически».

Их краткое описание дано в разделе «Базовые понятия», в подразделе «Мангуст», а более подробно они рассмотрены в разделе «Компоненты микроуровня», в подразделе «Внеуровневый элемент: инфосервисы «Мангуст».

Шаг 4. Реализация инновационных идей в ИТ-архитектуре

На этом шаге решается, как реализовать в архитектуре ИС найденную инновационную идею.

Полученное на этом этапе решение в форме онтологической цифровой платформы предприятия кратко описано выше, в главе 2, раздел «Базовые понятия», и подробно детализировано в последующих разделах главы 2.

Что получилось

Устройство и способ обработки информации в информационных системах предприятий с целью учета, аналитики и поддержки принятия управленческих решений на основе этой информации, посредством мгновенно формируемых по запросу пользователя информационных конгломератов микросистем во взаимодействии с онтологической, динамической виртуальной моделью предприятия, функционирующей в реальном времени.

Данные устройство и способ реализованы в форме цифровой платформы организации, построенной в архитектурном стиле «управляемой атомизации» на основе «атомарных микросистем» (У-Блоков) и «мгновенных инфосервисов» («Мангустов») под управлением семейства онтологических моделей.

Такое решение позволяет повысить:

- адаптивность информационных систем к изменениям во внешней среде (в предметной области информатизации, в бизнес-окружении и так далее);
- гибкость информационных систем при выполнении задач учета, аналитики и принятия управленческих решений на предприятии;
- простоту и удобство работы с информационной системой для конечного пользователя.

Преимущества изобретения

Изобретение позволяет отказаться от жесткой архитектуры монолитных приложений и преодолеть недостатки SOA с помощью гибкой онтологической архитектуры в стиле «управляемой атомизации», обеспечивающей возможность мгновенного исполнения информационных бизнес-сервисов, формирующихся в нужное время для конкретной операции (запроса, действия, события).

Это обеспечивает:

- возможность пользователю выполнять любые, не формализованные заранее действия в информационной системе и в текущих бизнес-процессах предприятия — другими словами, можно мгновенно выполнить любой запрос к системе, независимо от его сложности, от того, относится он к учетным или транзакционным процедурам, к бизнес-аналитике или поддержке принятия решений;
- возможность ИТ-службе предприятия легко адаптировать функциональность информационной системы к внешним изменениям деловой среды; система может быть адаптирована как к уже произошедшим, так и к предполагаемым изменениям;
- эффективное функционирование системы при постоянно меняющихся потребностях пользователя и при любом изменении условий деловой среды предприятия.

Эта система предоставляет предприятию значительные конкурентные преимущества:

- устранение материальных, финансовых и временных затрат, связанных с необходимостью постоянного перепроектирования системы при изменении условий;
- быструю адаптацию системы к любым изменениям в предметной области (в деловой среде);
- рост производительности труда персонала и возможность расширения функций пользователей, повышение универсальности сотрудников.

Литература

[1] Бестужев-Лада И. В. Социальное прогнозирование. Курс лекций. Часть II «Концепция технологического прогнозирования и ее сущность». М.: Педагогическое общество России, 2002. 392 с. ISBN5–93134–152–8.

[2] Альтшуллер Г. С. Творчество как точная наука. Теория решения изобретательских задач. М.: Советское радио, 1979.

[3] Петров В. М., Злотина Э. С. Теория решения изобретательских задач — основа прогнозирования развития технических систем. Прага: ЧДНТО, 1989.

[4] Злотин Б., Зусман А. ТРИЗ-прогнозирование: вчера, сегодня, завтра. Выход за парадигму. Ideation International, 2002.

[5] Zlotin Boris, Zusman Alla. Directed Evolution: Philosophy, Theory and Practice. Ideation.

[6] Mizrachi Yonatan, Ph D. Comments on intellectual history, the logical and applicative visibility, and the underlying assumption of Directed Evolution (DE). // The BRM Institute for Technology and Society, The Graduate School of Business Administration, Tel Aviv University, 2007.

[7] Гимранов Р., Холкин И., Зимин К. Прогнозирование развития КИС в направлении Real-Time Enterprise. Пример экспресс-прогноза. Information Management, № 7–8, 2014, с. 74–89.

[8] Холкин И. Н. Прогнозирование развития ИС с помощью метода Directed Evolution. Information Management, № 7–8, 2014

[9] Гимранов Р. Д., Лугачев М. И. Подходы к построению цифрового предприятия на основе эмержентной стратификации информационных систем. Вестник кибернетики, № 2, 2016, с. 165–168.

[10] Гимранов Р. Д. Стратификация информационных систем. Вестник кибернетики, № 1, 2016, с. 57–62.

[11] Лалу Фредерик. Открывая организации будущего. М.: Манн, Иванов и Фербер, 2016. ISBN978–5–00057–786–8.

[12] Агиевич В. А., Гимранов Р. Д. Архитектурный подход в управлении развитием ИТ и средства его реализации. // Управление жизненным циклом информационных бизнес-систем. Архитектура предприятия: сборник статей под редакцией Р. Д. Гимранова, В. А. Агиевича. Сургут, РИИЦ «Нефть Приобья» ОАО «Сургутнефтегаз», 2014, с. 6–14.

[13] Гимранов Р. Д. Проблема сложности информационных систем предприятия и ее преодоление для построения Real-Time Enterprise. Вестник кибернетики, № 3 (19), 2015, с. 181–185.

[14] Месарович М., Мако Д., Такахара И. Теория иерархических многоуровневых систем. М.: Мир, 1973.

[15] Gitt, W. In the Beginning was Information: A Scientist Explains the Incredible Design in Nature. Green Forest, AR: Master Books, 2005.

[16] Кларк А.Ч. Черты будущего (Profiles of the Future, 1962). Перевод с английского Я. Берлина и В. Колтового. Под общей редакцией Я. Берлина. М.: Мир. Редакция научно-фантастической и научно-популярной литературы, 1966.

[17] Глоссарий цифровой повестки ЕАЭС [Электронный ресурс]. Available: http://www.eurasiancommission.org/ru/act/dmi/workgroup/Pages/glossary.aspx/ [Дата обращения: 12.07.2017]

[18] Уилбер, Кен. Теория всего. Интегральный подход к бизнесу, политике, науке и духовности / К. Уилбер; пер. с англ. Е. Пустошкина. М.: Постум, 2013. ISBN978–5–91478–027–9.

[19] Dietz, J.L. Enterprise Ontology. Theory and methodology. Berlin Heidelberg: Springer-Verlag, 2006.

[20] Свод знаний по управлению бизнес-процессами: BPM CBOK 3.0 / Под ред. А.А. Белайчука, В.Г. Елиферова. Пер. с англ. М.: Альпина паблишер, 2016. ISBN978–5–9614–5455–0.

[21] Техническое описание. Архитектура платформы Comindware Business Application Platform. Компания Comindware, 2016.

[25] Глушков В.М. Макроэкономические модели и принципы построения ОГАС. Москва: Статистика, 1975.

[26] Синицын И.Н., Шаламов А.С. Методическое обеспечение информационных технологий стохастического моделирования процессов организационно-технико-экономических систем на «малых» рынках финансов, товаров и услуг // Исследования по экономике информационных систем: Материалы научно-практической конференции «Экономическая эффективность информационных бизнес-систем». Под ред. В.Л. Макарова, М.И. Лугачева, К.Г. Скрипкина. Москва, Экономический факультет МГУ имени М.В. Ломоносова, 2015.

[27] Остервальдер А. Построение бизнес-моделей: Настольная книга стратега и новатора. Москва: Альпина паблишер, 2016.

[28] Лугачев М.И. Экономическая информатика. Введение в экономический анализ информационных систем. ИНФРА-М, 2005.

[29] Винер Н. Кибернетика, или Управление и связь в животном и машине. 2-е издание. М.: Наука, главная редакция изданий для зарубежных стран, 1983.

[30] Волькенштейн М.В. Энтропия и информация. М.: Наука, 1986.

[31] Холкина М.М., Холкин И.Н. Феномен инфосоциосреды: системные отношения и прогноз развития. В рамках гранта РФФИ № 09–06–00274 «Трансформации общественного сознания и картины мира личности в условиях формирования информационного общества». Институт системных исследований и КСП. Москва, 2010.

[32] Холкина М.М., Холкин И.Н. Интернет как часть феномена инфосоциосреды в информационном обществе. Институт системных исследований и КСП. Москва, 2010.

[33] Гимранов Р.Д. Информационные модели предприятия в реализации технологии In-Memory Data Management. Современные информационные технологии и ИТ-образование, № 10, 2014, с. 75–79.

[34] Глушков В.М. Кибернетика. Вопросы теории и практики. М: Наука, 1986.

[35] Управление жизненным циклом информационных бизнес-систем. Real-time Enterprise 2.0.: сборник статей под редакцией Р.Д. Гимранова. СПб, Сургут, 2014.

[36] Управление архитектурой предприятия. Опыт применения для развития ИТ. Сборник статей. Под ред. Чалей И.В., Гимранова Р.Д. Москва: ЗАО «Открытые системы», 2013.

[37] Гимранов Р.Д., Агиевич В.А. Обеспечение достоверной информации в информационной системе крупного предприятия на основе архитектурного подхода. Нефтяное хозяйство, № 4, 2013, с. 116–119.

[38] Холкин И.Н. Оценка эффективности внедрения информационных систем методом анализа жизненного цикла (SLCA) // Управление архитектурой предприятия. Опыт применения для развития ИТ. Под общ. ред. И.В. Чалей, Р.Д. Гимранова. М.: Открытые системы, 2013.

[39] Intelligent XBRL-based Digital Financial Reporting [Электронный ресурс], Available: http://xbrl.squarespace.com/intelligent-xbrl/. [Дата обращения: 04.07.2017]

[40] Гимранов Р.Д. На пути к Real-Time Enterprise 2.0. Изменения корпоративных информационных систем при использовании технологии in-memory Data Management. Information Management, № 7–8, 2014.

[41] Эшби У. Введение в кибернетику. М: Иностранная литература, 1959.

Для заметок

Для заметок

Для заметок

Для заметок

Для заметок

Для заметок