РАЗДЕЛ V. СОВЕРШЕНСТВОВАНИЕ ИНСТРУМЕНТАРИЯ НА БАЗЕ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Ильина О.П., Барабанова М.И.

Принципы системного проектирования ИТ-решений

(СПбГУЭФ, Санкт-Петербург)

Информационные системы (ИС) и/или системы информационных технологий (ИТ) рассматриваются как «ИТ-решения», для создания которых применяются различные методологии ведения проектных работ, CASE-ттехнологии. При проектировании современных и качественных ИТ-решений необходимо учитывать ряд принципов системного проектирования.

I. Согласование бизнес-стратегии и ИТ-стратегии

ИТ-стратегии (ИТС) выражают концептуальные основы информатизации, нацелены на долгосрочные изменения в ИС и ИТ, необходимый для эффективного функционирования бизнес-системы функционал корпоративной ИС, соответствующую организацию информационных ресурсов, выбор программной платформы, обеспечивающих перспективность применения ИТ-решений, соответствие растущим информационным потребностям системы управления бизнес-системы.

ИТ-стратегии создаются на этапе *архитектурного проектирования ИС*:

- архитектура вычислительной системы определяет состав и характеристику средств вычислительной техники, периферийного оборудования, компьютерной сети и т.п.;
- архитектура программного обеспечения определяет платформу, системное и прикладное программное обеспечение для автоматизации функций управления и поддержки бизнес-процессов;
- архитектура информационного обеспечения задает параметры структуру данных интегрированной базы данных, схему информационных потоков (системы документации и схему документооборота), применяемые системы классификации и кодирования информации системы управления.

Принимаемые и реализуемые ИТ-решения в соответствии с принципами и целевыми установками ИТ-стратегий ориентированы на увеличение времени жизненного цикла в части этапа эксплуатации и обеспечение качества функционирования ИС.

ИТ-стратегии разрабатываются с учетом принятых бизнес-стратегий, представленных в виде стратегических бизнес-целей и системы сбалансированных показателей и реализуются через соответствующие ИТ-решения. С помощью ИТ-решений создаются специально организованные информационные управляющие потоки (DataFlow), поддерживающие управление материальными потоками в составе бизнес-процессов и рабочих операций (Work Flow). Для бизнес-целей ИТ-решения становятся критическими факторами успеха.

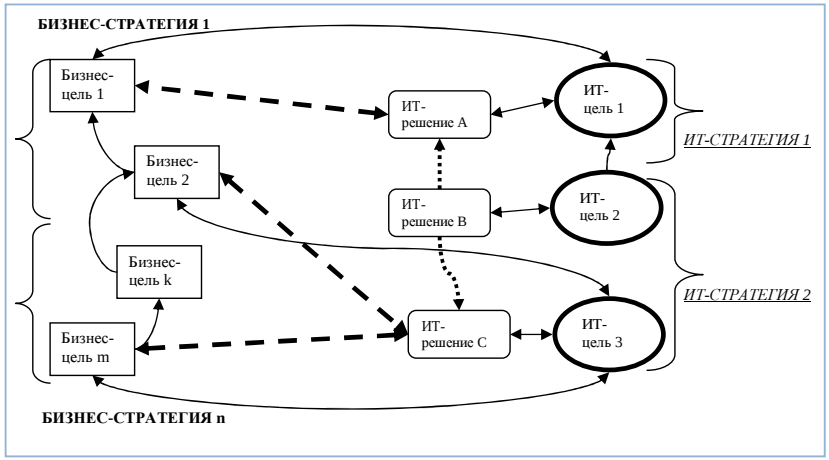

Рис. 1

На рис. 1 показана специфика проектирования ИТ-решений.

Задана упорядоченность целей бизнес-стратегий и ИТ-стратегий – отношения типа «предыдущая-следующая». Реализация бизнес-целей возможна при достижении определенных ИТ-целей (например, бизнес-цель 1 зависит от реализации ИТ-цели 1, бизнес-целей 2 и m – от ИТ-цели 3 и т.п. – см. рис. 1). В свою очередь, ИТ-стратегии реализуются с помощью ИТ-решений, между которыми также существует связь, обусловленная технологическими особенностями, целесообразной последовательностью действий. ИТ-решения служат определенным целям ИТ-стратегий и являются средством реализации бизнес-целей, а также должны обеспечивать:

- перспективность, долговечность, трансформируемость информационных технологий и информационно-вычислительных ресурсов;
- высокое качество формируемой управляющей информации (полнота, достоверность, точность, актуальность, своевременность);
- экономическую эффективность (низкую полную стоимость владения ИТ-решением в течение жизненного цикла) и др.

Для согласования стратегических целей бизнеса и ИТ следует:

1. Получить описание бизнес-стратегий на вербальном уровне.
2. Представить состав бизнес-целей, их связи между собой для выражения бизнес-стратегий с помощью измеримых экономических показателей.

3. Установить взаимосвязь бизнес-целей, отражающих последовательность их достижения во времени.
4. Разработать ИТ-стратегии в виде ИТ-целей:
   - изменение функционала ИС;
   - архитектуры вычислительной системы ИС;
   - архитектуры программного обеспечения ИС;
   - архитектуры информационного обеспечения ИС.
5. Установить соответствие бизнес-целей и ИТ-целей.
6. Определить состав и структуру ИТ-решений в рамках ИТ-стратегий.
7. Трансформировать связь «бизнес-цель == ИТ-цель» в связи «бизнес-цель == ИТ-решение».
8. Установить окончательный порядок реализации ИТ-решений.

В условиях высокой динамики экономической системы и развития ИТ (появления новых средств и методов реализации ИТ-решений, возникновения рисков и т.п.) стратегические бизнес-цели и ИТ-цели должны периодически пересматриваться. Управление ИТ-стратегиями должно стать обязательной составляющей проектной деятельности по информатизации.

II. Архитектурный подход к разработке ИТ-решений

Согласно стандарту ANSI/IEEE Std 1471-2000: «Архитектура – это фундаментальная организация системы, реализованная в её компонентах, их взаимоотношениях друг с другом и средой и принципах, определяющих её конструкцию и развитие».

ИТ-решение использует архитектуру предприятия, которая представляется как набор взаимосвязанных самостоятельных моделей различных уровней, описывающих структуру и функции предприятия:

- **Бизнес-модель**: описывает цели и задачи верхнего уровня, бизнес-процессы, охватывающие всё предприятие или значительную его часть, бизнес-функции, организационная структура, внешние информационные и материальные связи (логистические цепочки, информационное взаимодействие).
- **Модель приложений**: состав приложений и автоматизированных сервисов для поддержки бизнес-процессов; состав и структура прикладных систем, стандартные интерфейсы.
- **Информационная модель**, описывающая структуру данных в интегрированном виде, политику управления данными в течение жизненного цикла, системы документации и документооборота, системы классификации и кодирования информации, политику информационной безопасности.
- **Модель ИТ-инфраструктуры** позволяет представить аппаратные средства вычислительной системы, базовое программное обеспечение, средства сопряжения с внешними информационными системами.

Согласно международному стандарту IEEE 1471 «Рекомендованная практика архитектурного описания систем с интенсивным применением программного обеспечения» (Recommended Practice Architectural Description of Software Intensive Systems):

- бизнес-система реализует определенную миссию (Mission), вступает во взаимодействие с внешним окружением (Environment);
- ключевые фигуры (Stakeholders) владеют бизнесом, определяют основные бизнес-стратегии и цели (Concern), ракурсы видения бизнеса (Viewpoint);
- архитектура компании (Architecture) использует нотацию для описания (Architecture Description), которое учитывает целесообразность включения в архитектуру (Rationale) различных представлений (View).

Проектирование ИТ-решения состоит в целенаправленном изменении архитектуры приложений, информации (данных) и ИТ-инфраструктуры в рамках заданной бизнес-архитектуры, хотя она может изменяться под влиянием ИТ-решений.

Для разработки архитектуры приложений наиболее часто используется:

- концепция интеграции корпоративных приложений (Enterprise Application Integration, EAI);
- концепция сервисно-ориентированной архитектуры (Service Oriented Architecture – SOA).

EAI предполагает различные варианты интеграции приложений, разработанных независимо друг от друга:

- использование методов объектно-ориентированного программирования;
- создание распределенной, кросс-платформенной программы для связи приложений;
- применение интегрированных систем корпоративного типа (стандарты MRP II, ERP);
- создание централизованно хранимых сетевых баз данных;
- использование стандартов для интеграции разнородных данных, формата обмена сообщениями XML (Extensible Markup Language;
- программное обеспечение класса Middleware и другие подходы.

EAI обеспечивает интеграцию бизнес-процессов (Business Process Integration, BPI) в режиме реального времени на основе стандартов CORBA, EDI, COM+/DCOM, JavaRMI и XML. Как правило, используется два базовых метода интеграции приложений: «точка-точка» – согласование взаимодействия приложений попарно (для большого количества приложений возникает проблема поддержания разнообразных интерфейсов); «звезда» – каждое отдельное приложение согласовывает свою работу с единственным центром, который обеспечивает централизацию правил взаимодействия и виды интерфейсов. Разработка архитектуры приложений на основе концепции EAI предполагает:

- Сбор и анализ требований к приложениям, оценка их «покрытия» программными средствами.
- Выбор метода интеграции («точка-точка», «звезда»).

- Разработка требований к ИТ-архитектуре и информационной архитектуре, выбор инструментов интеграции.
- Детализированная разработка интеграции для каждого приложения.

Для распределенных ИТ-решений применяется архитектура SOA , приложения строятся на основе сервисов или служб, обеспечиваемых провайдерами (Service Provider). Существует три класса сервисов: бизнес-функций, инфраструктуры и управления ИС/ИТ. Для SOA создана технология Enterprise Service Bus, которая обеспечивает унифицированное взаимодействие приложений на базе интеграционной платформы. В разработке приложений осуществлен переход от объектно-ориентированного программирования к сервисно-ориентированному с целью уменьшить степень связанности различных элементов системы. Приложения в SOA обмениваются через стандартизированные интерфейсы сообщениями, в которых содержится семантика и бизнес-логика приложений. Отдельное приложение представляет собой объединение процессных сервисов, бизнес-сервисов и сервисов пользовательских интерфейсов. Бизнес-сервисы обычно проектируются в четыре слоя: фасад, бизнес-процессы, бизнес-сущности и сервисы представления данных.

Развитие SOA идет в направлении создания *событийно-управляемой* архитектуры (Event Driven Architecture, EDA), обеспечивающей тесную интеграцию приложений в бизнес-процессах, их гибкость и адаптивность. Для EDA характерно наличие трех компонентов: генератор события, обработчик события и менеджер событий. Менеджер событий регистрирует все возникающие в системе события, идентифицирует и передает их обработчику событий. События могут попадать в очередь либо сразу же передаваться обработчику, инициализирующий соответствующий сервис согласно выбранной стратегии:

- простая обработка (Simple Event Processing) – инициализация сервисов по мере регистрации соответствующих событий, работа в режиме реального времени, но без учета приоритета событий;
- потоковая обработка (Event Stream Processing) – объединение зависимых сервисов в общий поток с целью снижения расхода ресурсов на поиск и извлечение информации из базы данных;
- комплексная обработка (Complex Event Processing) – управление «по отклонениям» от заданного состояния равновесия.

В SOA используются клиентские веб-приложения (Rich Internet Applications, RIA), средой исполнения которых становится веб-браузер. Технология AJAX используется для организации интерфейса между клиентской частью и веб-сервисом, что позволяет строить интерактивный пользовательский интерфейс для веб-приложений, поддерживать «фоновый» обмен данными между браузером и веб-сервером, что приводит к существенному ускорению работы приложений.

При изменении архитектуры приложений возможны изменения в технологической архитектуре, которая включает в себя: сетевую архитектуру;

архитектуры хранения; архитектуру инфраструктуры приложений; управления и безопасности ИТ, и архитектуре данных.

III. Управление требованиями для ИТ-решений

Разработка ИТ-стратегий осуществляется с учетом информационных потребностей бизнес-системы. Управление требованиями пользователей к ИС (Requirement Management) является непременным условием успешного проекта информатизации. Возможно расширение и пересмотр требований в процессе создания ИТ-решений, это обусловлены сложностью окончательной формулировки требований, динамикой развития системы управления, выбором новых средств реализации ИТ-решения и т.п.

Требования к ИТ-системе оформляются с помощью различных нотаций, например, в настоящее время все большое применение получает графический язык SysML – *язык системного моделирования*, являющийся расширением подмножества языка унифицированного моделирования (UML). Далее документированные требования подвергаются анализу, проводится классификация и унификация требований, устанавливается их приоритет с целью рационализации ИТ-решений, снижения затрат на их реализацию. Поддержание в актуальном состоянии заявленных требований, с одной стороны, и контроль за их адекватной и своевременной реализации в ИТ-решении, с другой стороны, создает основу для обеспечения качества проектных решений.

IV. Гибкость и итеративность ИТ-решений

ИТ-решение разрабатывается как проект – уникальное временное целенаправленное мероприятие, ограниченное по бюджету и трудовым ресурсам. Если требования к ИТ-решению недостаточно конкретны и меняются, но при этом в наличии квалифицированные разработчики и заказчик, который согласен участвовать в разработке, в управлении ИТ-проектом может применяться *гибкая методология разработки – Agile software development,* которая объединяет ряд методик[1]. Разработчиками программного обеспечения принят манифест Agile Manifesto, который провозгласил базовые идеи (но имеется и ряд альтернатив доработки этого документа!):

- Личности и их взаимодействия выше, чем процессы и инструменты – подчеркивает важность слаженной работы команды ИТ-проекта.
- Работоспособное программное обеспечение выше, чем обширная документация – необходимо стремиться к получению качественного результата – реализованного ИТ-решения.
- Сотрудничество с заказчиком выше, чем переговоры по контрактам.
- Умение реагировать на изменения выше, чем следование плану.

[1] Agile Modeling, Agile Unified Process (AUP), Agile Data Method, Essential Unified Process (EssUP), Экстремальное программирование (Extreme programming, XP), Open Unified Process (OpenUP), Scrum, Бережливая разработка программного обеспечения (Lean Software Development) и др.

В манифесте сформулированы принципы гибкой разработки программного обеспечения:

1. Главным приоритетом является удовлетворение заказчика посредством ранней и непрерывной поставки работоспособного программного обеспечения.
2. Приветствуйте меняющиеся требования даже на поздних стадиях разработки. Гибкие процессы используют изменения как средство получить конкурентные преимущества для заказчика.
3. Поставляйте работоспособное программное обеспечение часто: от раза в несколько недель, до раза в несколько месяцев, отдавая предпочтение коротким интервалам.
4. Представители бизнеса и разработчики должны работать вместе в течение всего проекта.
5. Стройте проекты вокруг мотивированных личностей. Предоставьте им среду и поддержку, в которой они нуждаются, и доверьте им самим сделать работу.
6. Наиболее эффективный способ передать информацию в команду проекта (а также передавать её внутри команды) – это непосредственное живое общение.
7. Основной мерой прогресса проекта является работоспособное программное обеспечение.
8. Гибкие процессы поощряют разработку с постоянной скоростью. Спонсоры проекта, разработчики и пользователи должны быть способны поддерживать постоянную скорость на неограниченной дистанции.
9. Непрерывное внимание техническому совершенству и хорошему дизайну увеличивает степень гибкости.
10. Простота – искусство максимизации работы, которую не надо делать, – является существенным фактором.
11. Наилучшие архитектуры, требования и дизайн создаются самоорганизующимися командами.
12. Через регулярные промежутки времени команда должна проводить анализ того, как стать более эффективной и улучшать свой процесс работы.

Подобная гибкая методология управления ИТ-проектом обеспечивает минимизацию проектных рисков за счет итерационных циклов, длительностью 2-3 недели, добавляющих функциональность гибкому продукту за счет выполнения планирования, анализа требований, проектирования, кодирования, тестирования и документирования. После каждой итерации проектная команда имеет возможность выполнить переоценку приоритетов разработки программного продукта. Одно из самых серьезных ограничений этих методологий – применение их в больших командах разработчиков, а также прозрачность и стабильность требований к ИТ-решению. В этих случаях лучше использовать предсказуемый процесс управления ИТ-проектом.

V. Соответствие ИТ-решений стандартам взаимодействия открытых систем

Взаимодействие открытых систем (ВОС) предполагает создание единого информационного пространства, в котором обеспечены:

- переносимость (portability) приложений с одной прикладной платформы на другую;
- интероперабельность (interoperability) приложений – возможность совместного использования информации и ресурсов компонентами распределенной системы;
- масштабируемость (scalability) приложений, их независимость от физических размеров системы (технических параметров).

ИТ-решения должны опираться на стандарты ВОС и рекомендации для открытых систем: *международные* стандарты, принятый ISO, *функциональные* стандарты, охватывающие несколько базовых стандартов или профилей и *эталонные модели* (Reference Model) предметных областей, в которых содержатся объекты, определяющие концептуальную основу для информационного взаимодействия с внешними информационными системами. Эталонная модель *среды открытой системы* (Open System Environment Reference Model – OSE/RM), предложенная комитетом IEEE POSIX 1003.0, используется в США, модель является многомерной. *Объектные* компоненты эталонной модели – это приложения; платформа; внешняя среда и интерфейсы (приложений и платформы, платформы и внешней среды). *Функциональные* компоненты эталонной модели представлены в виде различных служб: операционной системы, интерфейса «человек-машина», управления и обмена данными, машинной графики, компьютерной сети. *Обеспечивающие* компоненты эталонной модели – службы поддержки разработки программного обеспечения, служба защиты информации, служба интернационализации и служба распределенной обработки. Для различных предметных областей разработаны эталонные модели, определяющие концептуальную и методологическую основы ИТ-решения, структуру базовых спецификаций. Наиболее известными эталонными моделями являются: базовая эталонная модель взаимосвязи открытых систем (Basic Reference Model for Open Systems Interconnection – RM-OSI); эталонная модель для открытой распределенной обработки (Reference Model for Open Distributed Processing – RM-ODP); эталонная модель управления данными (Reference Model for Data Management – RM DF) и др. ИТ-решение формирует и поддерживает определенный профиль открытых систем общего или конкретного применения.

Перечисленные системные принципы процесса проектирования являются минимально необходимыми для получения значимого ИТ-решения. При использовании CASE-технологий проектирования ИТ-решений также следует обращать внимание на поддержку рассмотренных принципов.