Министерство сельского хозяйства Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
Иркутский государственный аграрный университет имени А.А. Ежевского
Институт экономики, управления и прикладной информатики
Кафедра информатики и математического моделирования

Бендик Н.В., Петрова С.А.

Системная архитектура информационных систем

Учебно-методическое пособие

Иркутск - 2016

УДК 004.273:681.518

Печатается по решению научно-методического совета Иркутского государственного аграрного университета имени А.А. Ежевского (протокол № от _____2016 г.).

Рецензенты: д.т.н., профессор кафедры информатики и математического моделирования ИрГАУ им. А.А. Ежевского Иваньо Я.М.

Бендик Н.В. *Системная архитектура информационных систем*: Учебно-методическое пособие / Н.В. Бендик, С.А. Петрова - Иркутск: Изд-во ИрГАУ, 2016, - 92 с. – ил.

В пособии изложены основы системного и архитектурного подходов к анализу и планированию создания информационных систем. Рассматриваются вопросы управления информационными системами в процессе выбора и реализации различных архитектур информационных систем, а также, состав и взаимосвязь процессов по разработке ИТ-стратегии и ИТ-архитектуры организации.

Рассмотрены вопросы применения на практике в проектировании информационных систем подходов и методов, позволяющих получать их архитектуры. Пособие построено по принципу выдачи заданий на практические работы и приведения комментариев и примеров к их выполнению. Практические работы взаимосвязаны между собой и предполагают последовательное выполнение. Приведенные задания на практические работы могут использоваться в рамках дисциплины «Системная архитектура информационных систем» для студентов, обучающихся по направлению 38.03.05 «Бизнес-информатика».

Содержание

Введение

В современном информационном обществе деятельность любого предприятия все больше становится электронной деятельностью. Таким образом компьютерные информационные системы становятся одним из основных факторов эффективности деятельности предприятия.

Разработка информационной системы состоит из трех этапов: анализа, проектирования и реализации, в результате итеративного выполнения которых происходит пошаговое «наращивание» системы. На этапах анализа и проектирования происходит построение архитектуры будущей информационной системы.

Архитектурный подход к разработке информационных и других автоматизированных систем обеспечивает жизнедеятельность организации и уделяет первоочередное внимание созданию и постоянному развитию комплексной архитектуры предприятия как основы, определяющей остальные работы по реализации и развитию систем. Этот подход предусматривает совместное взаимосвязанное и согласованное рассмотрение функций организации, среды ее деятельности, информационно-коммуникационной инфраструктуры, в которой она осуществляется, а также различных аспектов создаваемой системы, характеризующих ее представление как совокупности приложений и информационных ресурсов, воплощенных технологическими решениями.

Архитектура программного обеспечения системы или набора систем состоит из всех важных проектных решений по поводу структур программы и взаимодействий между этими структурами, которые составляют системы. Проектные решения обеспечивают желаемый набор свойств, которые должна поддерживать система, чтобы быть успешной. Проектные решения предоставляют концептуальную основу для разработки системы, ее поддержки и обслуживания.

Учебно-методическое пособие предназначено для направления «Бизнес-информатика» имеющее в учебной программе дисциплину «Системная архитектура информационных систем».

Цель данного пособия – изучение на практике применения в проектировании подходов и методов, позволяющих получать успешные архитектуры информационных систем.

Пособие построено по принципу выдачи заданий на практические работы и приведения комментариев и примеров к их выполнению. Практические работы взаимосвязаны между собой и предполагают последовательное выполнение. Студент по каждой практической работе обязан представить отчет и ответить на контрольные вопросы.

I Теоретические основы архитектуры информационных систем

1 Архитектура информационных систем

1.1 Основные понятия об информационной системе

В условиях рыночной экономики предприятие нуждается в решении задач управления на качественно более высоком уровне. Необходимость оперативного реагирования на конъюнктуру рынка и быстро меняющуюся экономическую ситуацию требует перестройки внутренней микроэкономики предприятия, постановки управленческого учета и оптимизации процессов управления.

Постоянно изменяющиеся требования рынка, огромные потоки информации научно-технического, технологического и маркетингового характера требуют от персонала предприятия, отвечающего за стратегию и тактику развития предприятия быстроты и точности принимаемых решений, направленных на получение максимальной прибыли при минимальных издержках.

В современных условиях производство не может существовать и развиваться без высокоэффективной системы управления, базирующейся на автоматизированной информационной технологии. Автоматизированная информационная технология тесно связана с информационной системой, которая является для нее основной средой.

Определим термин «информационная система». Существует множество определений информационной системы, рассмотрим наиболее емкие.

Система (греч. systema – целое, составленное из частей; соединение) – множество элементов, находящихся в отношениях и связях друг с другом, которое образует определённую целостность, единство в интересах достижения поставленных целей. Системы значительно отличаются между собой как по составу, так и по главным целям (табл.1).

Таблица 1 - Примеры систем направленных на реализацию разных целей

Система	Элементы системы	Главная цель системы
Организация	Люди, оборудование, материалы, здания и др.	Производство товаров
Электронно-вычислительные машины	Электронные и электромеханические элементы, линии связи и др.	Обработка данных
Коммуникационные линии связи	Модемы, кабели, сетевое программное обеспечение и др.	Передача информации
Информационная система	Компьютеры, компьютерные сети, люди, информационное и программное обеспечение	Производство профессиональной информации

Информационная система – это совокупность, состоящая из одного либо нескольких компьютеров, соответствующих средств программирования, операторов, физических процессов, средств телекоммуникаций и других, образующих автономное целое, способное осуществлять обработку или передачу данных. Другими словами, информационная система – это взаимосвязанная совокупность средств, методов и персонала, используемых для хранения, обработки и выдачи информации в интересах достижения поставленной цели.

Информационная технология является процессом, состоящим из четко регламентированных правил выполнения операций, действий, этапов разной степени сложности над данными, хранящимися в компьютерах. Основная цель информационной технологии: в результате целенаправленных действий по переработке первичной информации получить необходимую для пользователя информацию.

Информационная система является средой, составляющими элементами которой являются: аппаратные средства вычислительной техники, аппаратные средства телекоммуникаций (связи), программные средства, информационные базы данных и обслуживающий персонал. Основная цель информационной системы: организация обработки, хранения и передачи информации. Информационные системы, в которых представление, хранение и обработка информации осуществляется при помощи вычислительной техники, называются автоматизированными информационными системами или АИС.

Информационные системы являются основным средством, инструментарием решения задач и информационного обеспечения (рис. 1). Информационное обеспечение – это совокупность процессов сбора, обработки, хранения, анализа и выдачи информации, необходимой для обеспечения управленческой деятельности и технологических процессов. Под информацией понимают изменения объема и структуры знания о некоторой предметной области воспринимающей системой независимо от формы и способа представления знания.

В контексте обработки информации важное значение имеет понятие данных. Данные отличаются от информации конкретной формой представления и являются некоторым ее подмножеством, определяемым целями и задачами сбора и обработки информации. Данные характеризуются определенной формой представления и структурой, которая определяется структурой предметной области, информацию о которых содержат данные. Данные могут быть представлены в структурированной форме (анкеты, таблицы, графические данные в виде диаграмм) и неструктурированной форме (связный текст – документы на естественном языке, графические данные в идее фотографий и картинок).

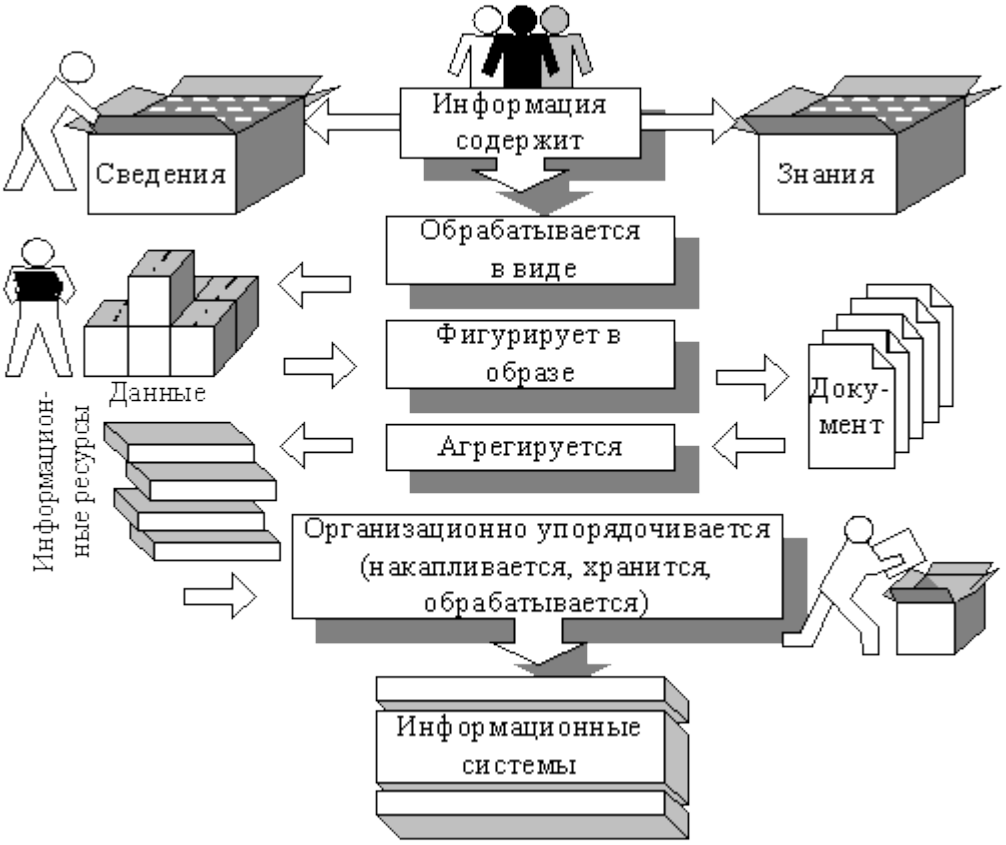

Рисунок 1 - Схема понятий информационного обеспечения

На предприятии в большинстве случаев информация фигурирует в виде документа или документированной информации. Документы подразделяются на служебные и организационно-распорядительные и представляют собой форму и способ выражения организационно-управленческих решений и воздействий. Документ – это зафиксированная на материальном носителе информация с реквизитами, позволяющими ее идентифицировать. Реквизиты определяются соответствующими руко-водящими документами по делопроизводству или отраслям технологической документации.

Документирование информации – запись информации на различных носителях по установленным правилам. Документирование представляет

собой выделение единичной смысловой части информации (данных) по некоторой предметной области, обособление и придание ей самостоятельной роли (имя, статус, реквизиты и пр.).

Процесс документирования превращает информацию в информационные ресурсы (Ressources d'information) – совокупность данных, организованных для эффективного получения достоверной информации. По законодательству Российской Федерации – это отдельные документы и отдельные массивы документов, документы и массивы документов в информационных системах: библиотеках, архивах, фондах, банках данных, других видах информационных систем.

В соответствии с вышесказанным, информационная система – это организационно-упорядоченная совокупность документов (массивов документов) и информационных технологий, в том числе и с использованием средств вычислительной техники и связи, реализующих информационные процессы.

Классификация информационных систем управления

Методологическую основу проектирования информационных систем составляет системный подход, в соответствии с которым любая система представляет собой совокупность взаимосвязанных объектов (элементов), функционирующих совместно для достижения общей цели. Для системы характерно изменение состояний объектов во времени, которое происходит в результате взаимодействия объектов в различных процессах и с внешней средой. В связи с этим для системы необходимо со-блюдение следующих принципов:

- целостности системы (эмерджентности – внутренней динамичности) на основе общей структуры, когда поведение отдельных объектов рассматривается с позиции функционирования всей системы;
- гомеостазиса (homeostasis) – способности системы сохранять равно-весие, т. е. обеспечивать устойчивое функционирование, благодаря саморегулируемому приспособлению к окружающей среде;
- адаптивности – способности системы адаптироваться к меняющимся условиям внешней и внутренней среды с помощью различных приспособительных механизмов, посредством воздействия на ее элементы;
- обучаемости путем изменения структуры системы в соответствии с изменением целей системы.

Процесс управления предприятием с позиции кибернетики представляет собой информационный процесс, который связывает внешнюю среду, объект управления и систему управления (рис. 2). Внешняя среда и объект управления информируют систему управления о своем состоянии. Система управления анализирует информацию и

вырабатывает управляющее воздействие на объект управления, в случае необходимости модифицируя цель и структуру всей системы.

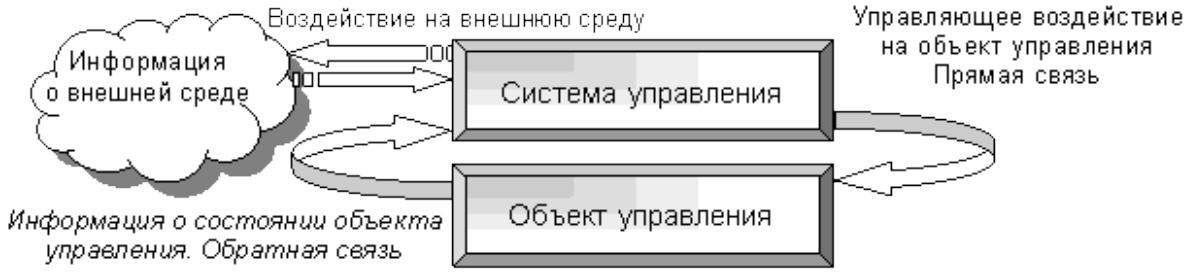

Рисунок 2 – Структура системы управления

В контексте обработки информации важное значение имеет понятие данных. Данные отличаются от информации конкретной формой представления и являются некоторым ее подмножеством, определяемым целями и задачами сбора и обработки информации. Данные характеризуются определенной формой представления и структурой, которая определяется структурой предметной области, информацию о которых содержат данные. Данные могут быть представлены в структурированной форме (анкеты, таблицы, графические данные в виде диаграмм) и неструктурированной форме (связный текст – документы на естественном языке, графические данные в идее фотографий и картинок).

На предприятии в большинстве случаев информация фигурирует в виде документа или документированной информации. Документы подразделяются на служебные и организационно-распорядительные и представляют собой форму и способ выражения организационно-управленческих решений и воздействий. Документ – это зафиксированная на материальном носителе информация с реквизитами, позволяющими ее идентифицировать. Реквизиты определяются соответствующими руководящими документами по делопроизводству или отраслям технологической документации.

Документирование информации – запись информации на различных носителях по установленным правилам. Документирование представляет собой выделение единичной смысловой части информации (данных) по некоторой предметной области, обособление и придание ей самостоятельной роли (имя, статус, реквизиты и пр.).

Процесс документирования превращает информацию в информационные ресурсы (Ressources d'information) – совокупность данных, организованных для эффективного получения достоверной информации. По законодательству Российской Федерации – это отдельные документы и отдельные массивы документов, документы и массивы документов в информационных системах: библиотеках, архивах, фондах, банках данных, других видах информационных систем.

В соответствии с вышесказанным, информационная система – это организационно-упорядоченная совокупность документов (массивов документов) и информационных технологий, в том числе и с использованием средств вычислительной техники и связи, реализующих информационные процессы.

Опыт реализации проектов создания информационных систем, особенно больших, охватывающих компании или организации целиком, показывает, что цели их создания достигаются довольно редко и сопровождаются неоправданно большими затратами времени и других ресурсов, что, обнаруживается, как правило, задним числом.

Причиной возникновения таких проблем является, как нечеткая постановка целей создания информационных систем, так и отсутствия комплексного, системного подхода ко всем процессам планирования, создания, внедрения и дальнейшей эксплуатации и развития систем такого масштаба.

Применение системного подхода, описанного выше, оказывается недостаточным для сложных проектов и требуется применение еще более комплексных подходов, позволяющих учитывать, описывать и управлять еще большим числом аспектов создаваемых информационных систем с точки зрения различных организационных ролей.

Такой комплексный подход, применение которого позволяет повысить результативность, эффективность, масштабируемость, управляемость, безопасность проектов создания больших информационных систем и предсказуемость их результатов, получил название «архитектурного подхода», а совокупность используемых при его применении описаний всех аспектов и точек зрения информационных систем – «архитектуры информационной системы».

Информационная система управления представляет собой совокупность организационных, технических, программных и информационных средств, объединенных в единую систему с целью сбора, хранения, обработки и выдачи информации, предназначенной для выполнения функций управления. Информационная система накапливает и перерабатывает поступающую нормативную, плановую и учетную информацию в аналитическую информацию, которая служит основой для прогнозирования развития системы управления, корректировки целей и планирования нового цикла воспроизводства. К обработке информации в информационной системе предъявляются следующие требования:

- полнота и достаточность информации;
- своевременность представления информации;
- достоверность информации;
- экономичность обработки информации;
- адаптивность к изменяющимся информационным потребностям пользователей.

Классификация информационных систем управления способствует выявлению наиболее характерных черт, присущих информационным системам. Классификация проводится по определенным признакам.

1. По характеру представления и логической организации хранимой информации:

– фактографические информационные системы;

– документальные информационные системы;

– геоинформационные информационные системы.

Фактографические информационные системы накапливают и хранят данные в виде множества экземпляров одного или нескольких типов структурных элементов (информационных объектов), которые отражают сведения по какому-либо факту, событию и пр., отделенному от других сведений. Структура каждого типа информационного объекта состоит из конечного набора реквизитов, отражающих основные аспекты и характеристики сведений для объектов данной предметной области. При комплектовании информационной базы обязательно используется структуризация, которая осуществляется через определение экземпляров информационных объектов определенного типа, информация о которых имеется в документе, и заполнение их реквизитов.

В документальных информационных системах единичным элементом информации является документ и информация на вводе (входной документ). При создании информационной базы процесс структуризации не производится или производится в ограниченном виде.

В геоинформационных системах данные организованы в виде отдельных информационных объектов, привязанных к общей электронной топографической основе (электронной карте). Такие системы применяются для информационного обеспечения предметных областей, структур информационных объектов и процессов, в которых имеется пространственно-географический компонент (маршруты транспорта, коммунальное хозяйство и пр.).

2. По выполняемым функциям и решаемым задачам:

- справочные информационные системы, которые предоставляют пользователям получать определенные классы объектов (телефоны, адреса, литературу и пр.) – электронные справочники, картотеки, программные или аппаратные электронные записные книжки и т. д.;

- информационно-поисковые информационные системы, которые дают пользователям возможность поиска и получения сведений по различным поисковым образам в информационном пространстве;

- расчетные информационные системы, которые производят обработку информации по определенным расчетным алгоритмам, например, вычисление определенных статистических характеристик;

- технологические информационные системы, функции таких систем заключаются в автоматизации всего технологического цикла или

отдельных его компонент производственной или организационной структуры, например, автоматизированные системы управления, системы автоматизации документооборота и пр.

3. По масштабу и интеграции компонент:

- локальный АРМ (автоматизированное рабочее место) – программно-технический комплекс, предназначен для реализации управленческих функций на отдельном рабочем месте; информационно и функционально не связан с другими информационными системами;
- комплекс информационно и функционально связанных АРМ, реализующих в полном объеме функции управления;
- компьютерная сеть АРМ на единой информационной базе, обеспечивающая интеграцию функций управления в масштабе предприятия или группы бизнес-единиц;
- корпоративная информационная система (КИС), обеспечивающая полнофункциональное распределенное управление крупномасштабным предприятием.

4. По характеру обработки информации на различных уровнях управления предприятием:

- системы обработки данных (EDP – Electronic data processing);
- информационные системы управления (MIS – Management Information System);
- системы поддержки принятия решений (DSS – Decision Support System).

Системы обработки данных предназначены для учета и оперативного регулирования хозяйственных операций, подготовки стандартных документов для внешней среды (отчетов, накладных, платежных поручений). Оперативное управление хозяйственными процессами составляет от одного до нескольких дней и реализует регистрацию и обработку событий, например, оформление и мониторинг выполнения заказов, приход и регистрацию материальных ценностей на складе, ведение табеля учета рабочего времени и т. д. Эти задачи имеют итеративный регулярный характер, выполняются непосредственно исполнителями хозяйственных процессов и связаны с оформлением и пересылкой документов в соответствии с четко определенными алгоритмами. Результаты выполнения хозяйственных операций через экранные формы вводятся в базу данных. Формы входных и выходных документов, схемы документооборота жестко регламентированы.

К системам оперативной обработки данных относятся информационные системы учета и регистрации первичной информации (бухгалтерские, складские, системы учета готовой продукции и т. д.), в которых выполняется сбор и регистрация больших объемов первичной информации, и используются простые алгоритмы расчетов и запросов к базе данных, структура которой стабильна в течение длительного времени. В таких системах большое значение имеет защита баз данных от несанк-

ционированного доступа, аппаратных и программных сбоев в работе. Для повышения эффективности функционирования используются компьютерные сети с архитектурой «клиент-сервер».

Информационные системы управления ориентированы на тактический уровень управления: среднесрочное планирование, анализ и организацию работ в течение нескольких месяцев (недель), например, анализ и планирование поставок, сбыта, составление производственных программ. Решение подобных задач предназначено для руководителей верхнего звена различных служб (отдел снабжения и сбыта, плановый отдел и пр.). Для данного класса задач характерны периодическая повторяемость формирования результатных документов и четко определенный алгоритм решения. Задачи решаются на основе накопленной базы оперативных данных.

Системы поддержки принятия решений используются на верхнем уровне управления и предназначены для решения задач по формированию стратегических целей, задач планирования, задач привлечения ресурсов и источников финансирования и пр. Задачи ориентированы на реализацию сложных бизнес-процессов, требующих аналитической обработки информации и имеют, как правило, нерегулярный характер. Анализ информации имеет определенную целевую ориентацию, например финансовый анализ предприятия. Для задач высшего менеджмента свойственно: недостаточность информации, ее противоречивость и нечеткость, преобладание качественных оценок целей и ограничений, слабая формализованность алгоритма решения.

5. По признаку структурированности задач:

- структурированные (формализуемые) задачи, где известны все ее элементы и взаимосвязи между ними;
- неструктурированные (неформализуемые) задачи – задачи, в которых невозможно выделить элементы и установить между ними связи;
- частично структурированные задачи.

При создании информационных систем возникают проблемы, связанные с формальным математическим и алгоритмическим описанием решаемых задач. От степени формализации зависит эффективность работы системы и уровень автоматизации, определяемый степенью участия человека при принятии решения на основе получаемой информации. Чем точнее математическое описание задачи, тем выше возможности компьютерной обработки данных и тем меньше степень участия человека в процессе ее решения. Это и определяет степень автоматизации задачи.

В структурированной задаче удается выразить ее содержание в форме математической модели, имеющей точный алгоритм решения. Подобные задачи обычно приходится решать многократно, и они носят рутинный характер. Целью использования информационной системы для решения структурированных задач является полная автоматизация их решения, т. е. сведение роли человека к нулю.

Решение неструктурированных задач из-за невозможности создания математического описания и разработки алгоритма связано с большими трудностями. Возможности использования здесь информационной системы невелики. Решение в таких случаях принимается человеком из эвристических соображений на основе своего опыта и, возможно, косвенной информации из разных источников.

Задачи, в которых известна часть элементов и связей между ними, называются частично структурированными. Информация, получаемая в информационной системе, анализируется человеком, который играет определяющую роль в принятии решения. Информационные системы, используемые для решения частично структурированных задач, подразделяются на два вида:

- информационные системы, создающие управленческие отчеты и ориентированные главным образом на обработку данных (поиск, сортировку, агрегирование, фильтрацию);
- информационные системы, разрабатывающие альтернативы решений (модельные или экспертные).

Информационные системы, создающие управленческие отчеты, обеспечивают информационную поддержку пользователя, т. е. предоставляют доступ к информации в базе данных и ее частичную обработку. Процедуры манипулирования данными в информационной системе должны обеспечивать следующие возможности:

- определенные комбинации данных, получаемых из различных источников;
- быстрое добавление или исключение того или иного источника данных и автоматическое переключение источников при поиске данных;
- управление данными с использованием возможностей систем управления базами данных;
- логическую зависимость данных одного типа от других баз данных, входящих в подсистему информационного обеспечения;
- автоматическое отслеживание потока информации для наполнения баз данных.

Модельные информационные системы предоставляют пользователю математические, статистические, финансовые и другие модели, использование которых облегчает выработку и оценку альтернатив решения. Пользователь может получить недостающую ему для принятия решения информацию путем установления диалога с моделью в процессе ее исследования. Основные функции модельной информационной системы:

- возможность работы в среде типовых математических моделей,
- достаточно быстрая и адекватная интерпретация результатов;
- оперативная подготовка и корректировка входных параметров и ограничений модели;
- возможность графического отображения динамики модели;

- возможность объяснения пользователю необходимых шагов формирования и работы модели.

Экспертные информационные системы обеспечивают выработку и оценку возможных альтернатив пользователем и связанны с обработкой знаний. Экспертная поддержка принимаемых пользователем решений реализуется на двух уровнях. Работа первого уровня экспертной поддержки исходит из концепции типовых управленческих решений, в соответствии с которой часто возникающие в процессе управления проблемные ситуации можно свести к некоторому типовому набору альтернатив. Для реализации экспертной поддержки на этом уровне создается информационный фонд хранения и анализа типовых альтернатив. Если возникшая проблемная ситуация не согласуется с имеющимися классами типовых альтернатив, в работу вступает второй уровень, который генерирует альтернативы на базе имеющихся данных, правил преобразования и процедур оценки альтернатив.

6. По функциональному признаку, который определяет назначение подсистемы, ее основные цели, задачи и функции:

- производственные системы, связанные с выпуском продукции и направленные на создание и внедрение в производство научно-технических новшеств;
- системы маркетинга, направленные на анализ рынка производителей и потребителей выпускаемой продукции, анализ продаж, организацию рекламной кампании по продвижению продукции и рациональную организацию материально-технического снабжения;
- финансовые и учетные системы, направленные на организацию контроля и анализа финансовых ресурсов на основе бухгалтерской, статистической и оперативной информации;
- системы кадров по подбору и расстановке специалистов и ведению служебной документации по различным аспектам предназначены для реализации функций оперативного планирования и учета личного состава;
- системы управления вспомогательным производством предназначены для автоматизации оперативного управления инструментальным производством, ремонтным и транспортным хозяйством и энергетическим обеспечением.

7. По уровням управления.

- информационные системы оперативного (операционного) уровня;
- информационные системы специалистов;
- информационные системы для менеджеров среднего звена;
- стратегические информационные системы.

Информационные системы оперативного уровня (бухгалтерские, банковские, обработки заказов и пр.) поддерживают специалистов, об-рабатывая данные о сделках и событиях (счета, накладные, зарплата,

кредиты, поток сырья и материалов). Задачи, цели и источники информации на операционном уровне заранее определены и структурированы. Система является связующим звеном между организацией и внешней средой, и основным поставщиком информации для остальных информационных систем.

Информационные системы специалистов помогают пользователям повысить продуктивность и производительность. Их задача – интеграция новых сведений и помощь в обработке бумажных документов.

Информационные системы менеджмента используются работниками среднего управленческого звена для мониторинга, контроля, принятия решений и администрирования. Основные функции систем: сравнение показателей, составление периодических отчетов за определенное время, обеспечение доступа к архивной информации и пр. Выделяют два типа систем:

- управленческие системы, обслуживающие менеджеров информацией о состоянии дел, ориентированы на контроль, отчетность и приня-тие решений по оперативной обстановке;
- системы поддержки принятия решений используются для решения частично структурированных задач, результаты которых трудно спрогно-зировать заранее, оснащены сложными инструментальными средствами моделирования и анализа.

Стратегические информационные системы обеспечивают поддержку принятия решений по реализации стратегических перспективных целей развития организации и помогают высшему звену управленцев осуществлять долгосрочное планирование. Основная задача – сравнение происходящих во внешнем окружении изменений с существующим потенциалом организации.

8. По характеру использования информации:

- информационно-поисковые системы производят ввод, систематизацию, хранение, выдачу информации по запросу пользователя без сложных преобразований данных (информационно-поисковая система в библиотеке, в железнодорожных кассах);
- информационно-решающие системы осуществляют все операции переработки информации по определенному алгоритму, выделяют управляющие и советующие системы.

Управляющие информационные системы вырабатывают информацию, на основании которой человек принимает решение. Этим системам свойственны задачи расчетного характера и обработка больших объемов данных, например, система оперативного планирования выпуска продукции, система бухгалтерского учета.

Советующие информационные системы вырабатывают информацию, которая принимается человеком к сведению. Они обладают более высокой степенью интеллекта и для них характерна обработка знаний. Например, медицинские информационные системы для постановки диагноза и

определения процедуры лечения, стратегические информационные системы.

9. По сфере применения:

- информационные системы организационного управления предназначены для автоматизации функций управленческого и оперативного контроля и регулирования, оперативного учета и анализа, перспективного и оперативного планирования, бухгалтерского учета, управления сбытом и снабжением и пр.;
- информационные системы управления технологическими процессами предназначены для автоматизации функций производственного персонала: организации поточных линий, изготовления микросхем, под-держания технологического процесса и пр.;
- информационные системы автоматизированного проектирования предназначены для автоматизации функций инженеров-проектировщиков, конструкторов, архитекторов дизайнеров для проведения инженерных расчетов, создания графической документации (чертежей, схем, планов), создания проектной документации, моделирования проектируемых объектов;
- корпоративные информационные системы используются для автоматизации всех функций организации и охватывают весь цикл работ от проектирования до сбыта продукции.

10. Укрупненная классификация систем, предназначенных для автома-тизации различных видов хозяйственного учета:

- локальные системы;
- средние интегрированные системы;
- крупные интегрированные системы.

Локальные системы достаточно успешно справляются с решением отдельных задач учета на предприятии, но, как правило, не предоставляют целостной информации для автоматизации управления. Преимуществом таких систем является низкая цена и простота внедрения. Например: «ИнфоБухгалтер» фирмы «Информатик», «Турбо-Бухгалтер» фирмы «Диц», «1С: Бухгалтерия» фирмы «1С». Программы обладают возможностями адаптации к особенностям предприятия, а некоторые из них представляют собой программные конструкторы, обладающие расширенными адаптационными возможностями, например, «Турбо-Бухгалтер».

Средние интегрированные системы представляют собой системы с ограниченными функциональными возможностями. Примеры: корпоративная информационная система «Галактика» фирмы «Галактика», комплексные информационные системы «Инфософт» фирмы «Инфо-софт», «NS2000» фирмы «Никос-Софт», «Abacus Financial» фирмы «Омега», система управления предприятием «Парус» фирмы «Парус», интегрированная система управления предприятием «БЭСТ ПРО» фирмы «Интеллект-сервис», система комплексной автоматизации финансово-

хозяйственной деятельности предприятия «Avacco» фирмы «Avacco Soft», «1С: Предприятие» фирмы «1С».

Крупные интегрированные системы представляют собой наиболее функционально развитые и соответственно наиболее сложные и дорогие системы, в которых реализуются стандарты MRP, ERP, SCRP. Примеры: «SAP» фирмы «R3 (Accelerated Solutions)», «BAAN» фирмы «Baan Midmarcet Solutions», «PeopleSoft» фирмы «PeopleSoft Select».

Российским лидером по производству и сопровождению информационных систем управления является корпорация «Парус». Технологии корпорации «Парус» используют в своей работе крупнейшие государст-венные структуры, отечественные и зарубежные коммерческие органи-зации. «Парус» предлагает своим клиентам весь спектр самых совре-менных информационных систем, предназначенных для управления производственными и торговыми предприятиями, бюджетными и стра-ховыми компаниями. Корпорация разработала эффективные инстру-менты, как для поддержки управленческих решений, так и для оказания услуг по выявлению внутренних резервов, внедрению систем бухгалтер-ского учета и перехода на международные стандарты отчетности.

Специалисты АО «Новый Атлант» и НТО «Топ Софт» разработали информационную систему «Галактика», которая предназначена для полной автоматизации управления всех служб предприятий различных форм собственности и позволяет повысить управляемость предприятия и его прибыльность.

Зарубежные корпоративные информационные системы, такие как R/3 фирмы SAP, Oracle Applications фирмы Oracle, Concorde XAL фирмы Columbus включают в себя больше подсистем, позволяющих оптимизировать управление корпорацией или фирмой на основе общепризнанных мировых стандартов. Эти системы не получили широкого распространения в России и странах СНГ за счет своей большой стоимости и некоторых отличий в методике ведения бухгалтерского учета.

1.2 Архитектура информационной системы

Следует отметить, что в настоящее время в российской и зарубежной практике проектирования информационных систем (ИС) и автоматизированных систем (АС) термин «архитектура системы» используется очень широко, но при этом имеет столь же широкое множество различных трактовок.

Одно из популярных в среде разработчиков ИС формальных определений архитектуры приведено в стандарте ANSI \ IEEE Std 1471 – 2000 Института инженеров-электриков и электронщиков, который предоставляет метамодель для определения архитектуры. Этот стандарт определяет такие абстрактные элементы архитектуры, как представления,

системы, среды, обоснования, заинтересованные стороны и т. д. в соответствии со схемой, показанной на рисунке 3.

В соответствии с этим представлением система обладает архитектурой, которая может быть описана с различных точек зрения заинтересованных лиц, рассматривающих архитектуру системы. Каждой точке зрения на архитектуру системы соответствует определенное представление, основу которого составляет набор моделей. Однако этот стандарт не определяет структуру собственно архитектуры предприятия. Например, говорится о том, что необходимо иметь различные представления архитектуры, но при этом не указывается, какие это должны быть представления.

Рисунок 3 – Рамочная модель разработки архитектуры по IEEE 1471

Можно рассмотреть различные аспекты понятия архитектуры ИС. В частности, можно выделять такие подмножества, как системная архитектура (архитектура систем – System Architecture) и программная архитектура (архитектура программного обеспечения – Software Architecture). На практике, в зависимости от контекста, термин "системная архитектура" может относиться либо к архитектуре ИС предприятия (в дополнение к бизнес- архитектуре) или, в более узком смысле, к технологической инфраструктуре информационной системы, либо – к архитектуре сложного продукта или семейства продуктов, выпускаемых предприятием. В последнем случае понятие разработки системной архитектуры близко по смыслу понятию «системное проектирование».

Методика описания и проектирования архитектуры отдельных прикладных систем имеет много общего с подходами к описанию архитектуры предприятия в целом, тем не менее, архитектура программных систем является отдельной областью знаний, которой посвящено большое количество соответствующей литературы. Под «программной архитектурой», в зависимости от контекста, может пониматься как архитектура взаимодействия приложений в рамках информационной системы предприятия (архитектура приложений), так и архитектура программных модулей, или архитектура взаимодействия различных классов в рамках одного приложения.

Каждая из отмеченных архитектур, в свою очередь, может рассматриваться с тем или иным уровнем детализации и под определенным углом зрения. Так, для программной архитектуры традиционными являются следующие перспективы или уровни описания архитектуры:

− концептуальная архитектура определяет компоненты системы и их назначения, обычно в неформальном виде. Это представление часто используется для обсуждения с нетехническими специалистами, такими как руководство, бизнес-менеджеры и конечные пользователи функциональных характеристик системы (что система должна уметь делать, в основном, с точки зрения конечного пользователя);

− логическая архитектура выделяет, прежде всего, вопросы взаимодействия компонент системы, интерфейсы и используемые протоколы. Это представление позволяет эффективно организовать параллельную разработку;

− физическая реализация, которая описывает привязку к конкретным узлам размещения, типам оборудования, характеристикам окружения, таким как, например, используемые операционные системы. Рассмотренные выше положения ANSI \ IEEE Std 1471 – 2000 задают лишь рамочную модель разработки архитектуры, но являются полезными для понимания основ архитектурного подхода.

В ГОСТ 34.320-96 дано описание архитектуры информационной системы, которая состоит из трех уровней: внешняя схема, внутренняя схема и уровень концептуальной схемы, информационной базы и информационного процессора. Ниже приведены (согласно стандарту) определения этих понятий.

Внешняя схема: Определение форм внешнего представления для возможных совокупностей предложений в пределах представления конкретного пользователя, а также аспектов манипулирования этими формами.

Внутренняя схема: Определение форм внутреннего представления в
компьютере совокупностей предложений концептуальной схемы и

информационной базы, а также аспектов манипулирования этими формами.

Концептуальная схема: непротиворечивая совокупность предложений, выражающих необходимые высказывания, относящиеся к проблемной области.

Информационная база: совокупность предложений, выражающих высказывания, отличные от необходимых высказываний, согласующиеся друг с другом и с концептуальной схемой, а также истинные в некотором пространстве сущностей.

Информационный процессор: процессор, который в ответ на команду выполняет действие над концептуальной схемой и/или информационной базой.

Российские стандарты не используют термин и не определяют понятие архитектуры автоматизированной системы (АС). Однако в структуре стадий и этапов процесса создания АС, которые определяются ГОСТ, необходимость в синтезе структуры системы появляется уже на стадии разработки концепции АС, а затем присутствует на стадии формирования технического задания на создание системы. На этих стадиях структуру АС можно отождествлять с архитектурой АС, что обычно и делается в реальной практике.

В зарубежных стандартах и методологиях присутствует большое количество различных вариантов определений термина (понятия) «архитектура системы».

Варианты определений понятия «архитектура системы» в явном или в
неявном видах присутствуют во многих источниках, к которым относятся отечественные и зарубежные стандарты, методологии, публикации ведущих специалистов.

Е.З. Зиндер (Фонд «Фостас») связывает развитие архитектуры ИС с необходимостью определения требований к архитектурам предприятия и внедрением архитектурного подхода в управлении развитием организации. По его мнению, наиболее фундаментальные положения современного подхода к созданию архитектуры предприятия (АП) оказались закреплены в стандарте ISO 15704.

Стандарт предназначен для определения требований к архитектурам и методологиям предприятия (Enterprise-reference Architectures and Methodologies). С учетом его положений в 2006 году был выпущен стандарт ISO 19439 «Enterprise integration — Framework for enterprise modelling»; сам ISO 15704 в 2005 году получил «Дополнительные представления с точки зрения пользователя». Стандарт ISO 15704 нацелен на решение задач трех типов: создание предприятия, его реструктуризация и инкрементальные изменения. Он ориентирован как на людей, так и на технологии (базовые и вспомогательные) и фиксирует необходимость комплексного подхода: «Было бы ошибкой ограничить обсуждение

интеграции только вопросами информации и систем управления…» — и далее: «…полное решение должно включать информацию, культуру и миссию». Характерно, что ISO 15704 создан комитетом не по ИТ, а по автоматизации предприятий (рабочей группой по архитектуре, связям на предприятии и его интеграции). Наверное, по этой причине в его основе лежит подход, отличающийся от «обычных» стандартов и методик ИТ-специалистов: в центре внимания постоянно находится именно предприятие как комплексный объект. Причем в это понятие включены и так называемые расширенные и виртуальные предприятия.

Принципиально важным в стандарте ISO 15704 является определение
архитектур двух типов. «Архитектура — это описание (модель) основной компоновки и взаимодействия частей системы (будь то физический либо абстрактный объект или сущность). Имеется два типа архитектур, относящихся к интеграции предприятий:

− архитектуры систем (иногда называемые архитектурами типа 1), которые имеют дело с конструкцией некоторой системы, например, компьютерной системы управления как части всеобъемлющей системы интеграции предприятия;

− архитектуры (планы/проекты) предприятия (иногда называемые архитектурами типа 2), которые имеют дело с таким проектом, как интеграция всего предприятия, или с иной программой его развития».

В стандарте рассматриваются требования в первую очередь к архитектурам типа 2, которым соответствуют «референсные архитектуры и методологии».

1.3 Архитектурный подход

Отечественные стандарты и руководящие документы не определяют и не используют термин «архитектура системы». Но в них определяются:

− виды структур ИС – функциональная, техническая, организационная, программная, информационная);

− основные структурные компоненты ИС – пользователи и комплекс средств автоматизации;

− виды обеспечения ИС – программное, информационное, техническое, организационное, методическое, математическое, лингвистическое, правовое и др.;

− необходимость выделения структуры функциональных систем и подсистем ИС, описания состава и характеристика автоматизируемых функций и задач ИС.

Control Objectives for Information and related Technology *(COBIT, 4-е издание):*

В COBIT не определяется и не используется термин «архитектура системы», но определяется и используется термин «ИТ-архитектура».

ИТ-архитектура – интегрированная структура для развития и поддержки существующих и приобретаемых новых информационных технологий, обеспечивающих выполнение стратегии и достижение бизнес-целей предприятия.

Кроме того, в СОВИТ определяется и используется термин «ИТ-ресурсы» для обозначения компонентов, из которых строится информационная система.

ИТ-ресурсы – это:

− *Приложения* – пользовательские программные системы, автоматизирующие обработку информации;

− *Информация* – бизнес-данные в формах ввода, обработки и вывода их информационными системами;

− *Инфраструктура* – технологии и оборудование (аппаратные вычислительные и коммуникационные средства, операционные системы, системы управления базами данных, средства мультимедиа, сооружения в которых установлены эти средства, инженерное оборудование мест инсталляции этих средств), делающие возможным функционирование приложений;

− *Персонал* – люди (специалисты), требующиеся для планирования, организации, приобретения, установки, эксплуатации и развития информационных систем и сервисов, нанимаемые по контрактам или используемые как внешний ресурс (аутсорсинг).

IEEE Recommended Practice for Architectural Description*, Draft 3.0 of IEEE P1471, May 1998:*

Архитектура – высокоуровневая концепция системы, учитывающая ее окружение.

ISO-15704, Industrial automation systems *– Requirements for enterprise-reference architectures and methodologies. August 20, 1999:*

Архитектура системы – описание (модель) основного расположения и взаимосвязей частей системы (физического либо концептуального объекта или сущности).

ANSI/IEEE Std 1471-2000, Recommended Practice for Architectural Description of Software-Intensive Systems:

Архитектура – фундаментальная организация системы, заключенная в своих компонентах, в их взаимоотношениях, в окружении, а также принципы, определяющие проектирование, создание и развитие системы.

OMG Unified Modeling Language (UML) Specification*, March 2003 Version 1.5:*

Архитектура – организационная структура и связанное с этой структурой поведение системы. Архитектура рекурсивно декомпозируется:

− на части системы, взаимодействующие через интерфейсы;

− на отношения между частями системы;

− на условия компоновки структур системы из ее частей.

Части системы, взаимодействующие через интерфейсы, включают классы, компоненты и подсистемы.

***Rational Unified Process** (www.ibm.com):*

Архитектура программной системы – организация или структура взаимодействия основных компонентов системы через интерфейсы, в том числе взаимодействия с компонентами, составленными из более мелких частей и интерфейсов.

Архитектура программной системы представляется в RUP множеством из следующих пяти архитектурных точек зрения, которые соответствуют основным элементам в соответствующих моделях:

− **The Use-Cases View** – структура вариантов использования системы;

− **The Logical View** – декомпозиция системы на классы и функциональные подсистемы;

− **The Implementation View** – структура программной реализации системы;

− **The Process View** – структура объединения подсистем и процессов;

− **The Deployment View** – структура физического распределения компонентов программной системы по аппаратным средствам.

***The Zachman Institute for Framework Advancement** (www.zifa.com)*

В методологии Дж. Захмана архитектура предприятия (информационной системы) представляется в виде структурированного набора моделей («The Zachman Framework», или «Framework for information systems architecture»), которые отражают различные содержательные точки зрения на структуру предприятия (системы) того круга лиц, которые вовлечены в его(ее) создание и развитие – собственника, менеджеров, проектировщика, конструкторов, субподрядчиков, пользователей. При этом различные точки зрения обращаются на различные структурные аспекты предприятия (системы) – структура данных, функции, сетевая инфраструктура, организация, время, мотивация.

Энциклопедические словари и другие источники

Архитектура – концепция, определяющая структуру и взаимосвязь компонентов сложного объекта (Международный центр научной и технической информации» - http://www.icsti.su).

Архитектура информационной системы – концепция, определяющая
модель, структуру, выполняемые функции и взаимосвязь компонентов информационной системы (http://www.glossary.ru).

Можно отметить, что на сайте американской организации «Software Engineering Institute» (http://www.sei.cmu.edu/archИТecture/definitions.html) собрано более 150 определений архитектуры программных систем.

1.4 Методология архитектуры предприятия

В западных странах (США, Канаде, Великобритании и др.) для управления развитием и функционированием государственных организаций и предприятий с использованием информационных технологий признана и практически используется методология «архитектуры предприятия» (www.enterprise-architecture.info).

Кроме этого, в США используется правительственная методология и практика по управлению информатизацией органов государственной власти и государственных предприятий – «Federal Enterprise Architecture» (FEA).

Методология FEA базируется на множестве взаимоувязанных референсных моделях, которые охватывают все структурные аспекты предприятия (учреждения), в том числе его информационной системы: структуру и функции деятельности, структуру программных и технических
средств, структуру обрабатываемой информации.

Мощный толчок развитию архитектур сверхкрупных распределенных человеко-машинных систем дали программы создания «электронного правительства» (ЭП). Концепции, методы и модели архитектур ЭП в них планомерно разрабатывались в течение многих лет — вплоть до обобщенных «архитектур электронного правительства».

В результате процесс заимствования идей на определённое время поменял направление: многие методы и модели архитектур, созданные для ЭП, стали заимствоваться коммерческими предприятиями. Существенный вклад в развитие архитектурного подхода сделала разработка FEA — «Федеральной архитектуры предприятия», развиваемой в США и, получившая наибольшую известность.

В FEA реализуются федеративные принципы архитектуры предприятий — оригинальное сочетание общего централизованного руководства и децентрализованного планирования при реализации архитектур отдельных организаций и отдельных информационных систем. Из методических особенностей FEA следует указать на ее полноценный состав, на принцип сегментного подхода и на развитие не только обобщенной схемы, но и важных референсных моделей. Состав FEA предусматривает следующие категории компонентов:

− стимулы (деловые и технические) и стратегическое направление развития архитектуры (включая его видение, цели и объекты);

− принципы, другие руководящие материалы, стандарты и глоссарий, а также примеры (образцы, сравнительные измерения)

передового
опыта;

− архитектурные референсные модели;

− текущую и целевую архитектуры;

− переходные процессы, включая планирование инвестиций, проектов
перехода к целевой архитектуре, управление проектами и их контроль;

− архитектурные сегменты для специфических областей деятельности;

− репозиторий для централизованного накопления и распределенного
использования архитектурных компонентов всех видов.

Одно из полезных положений FEA — принцип сегментного подхода — дает возможность ускорять практическое внедрение архитектуры, особенно в больших многоотраслевых образованиях, позволяя относительно независимо работать в рамках одного сегмента, обеспечивая минимизацию затрат, поддержку общих ресурсов и стандартов взаимодействия систем разных сегментов.

Несмотря на наличие большого числа методологий в области создания
архитектуры организации на практике большинство организаций ограничиваются при описании деятельности следующими предметными областями: цели, организационная структура, ключевые показатели результативности, бизнес-процессы, документы, информационные системы, знания и полномочия персонала.

Все эти предметные области можно найти в существующих методологиях описания архитектуры организации. Однако основная сложность для многих организации заключается в построении «мостика» от существующих бизнес-процессов к средствам их автоматизации. Поэтому одной из задач, которую сейчас решают многие российские организации, занимающиеся описанием бизнес-процессов, является переход от моделей и регламентов бизнес-процессов к вопросам построения ИТ-архитектуры.

В этой области существует определенный информационный разрыв при передаче информации от бизнес-аналитиков к ИТ-специалистам. И если в случае внедрения «тяжелых» систем с определенной бизнес-функциональностью (транзакции, структуры данных, отчетность) вопрос взаимодействия решается передачей построенных моделей бизнес-процессов специалистам по внедрению систем, то в случае использования систем-конструкторов или систем собственной разработки просто передачи моделей бизнес-процессов разработчикам недостаточно. Необходимо также на основании описанных бизнес-процессов определить перечень операций, структуру данных, дизайн экранных форм,

модульность системы и т.д. На этом этапе наиболее правильно использовать рекомендации, содержащиеся в вышеозначенных методологиях описания архитектуры организации, например, в TOGAF.

Это поможет решить и существующие организационные проблемы, связанные с тем, что в большинстве случаев работами по описанию и совершенствованию внутренних бизнес-процессов занимаются либо функциональные подразделения, либо специально созданные отделы совершенствования бизнес- процессов, а реализуются эти требования, сформированные на основании созданных описаний бизнес-процесса, ИТ-специалистами. Поэтому требования к информационной поддержке бизнес-процессов должны быть понятны как бизнес- специалистам, так и специалистам из ИТ-подразделений и компаний, что не всегда реализуемо.

Многие разработчики программного обеспечения используют для определения требований к информационной системе методологию UML, которая не совсем совместима с процессным подходом, поэтому фактически возникает два различных языка: процессные модели у бизнес-аналитиков и UML-модели у разработчиков.

Для устранения этого информационного разрыва между организацией
и ИТ необходимо расширять описание существующей архитектуры предприятия и, в частности, архитектуру процессов с учетом единства используемой методологии описания как для бизнес- аналитиков, так и для ИТ-специалистов.

Для перехода от описания архитектуры бизнес-процессов к описанию
ИТ-архитектуры необходимо формализовать несколько дополнительных предметных областей. В первую очередь следует описать архитектуру данных, которая строится на основании той информации и документов, которые используются в бизнес-процессах, а затем необходимо сформировать архитектуру приложений и архитектуру технологий (ИТ-инфраструктура) (рис. 4).

Для построения архитектуры данных необходимо выделить основные
сущности и агрегировать на них все «кванты» информации, собранные из описания бизнес- процессов. В результате использования стандартной методологии описания данных – модель «сущность-связь» (Entity Relationship Model – ERM) – можно четко структурировать всю информацию, тем самым определив структуру таблиц базы данных, что однозначным образом формализует структуру данных в компании в привязке к существующим бизнес-процессами будет понятно для ИТ-специалистов.

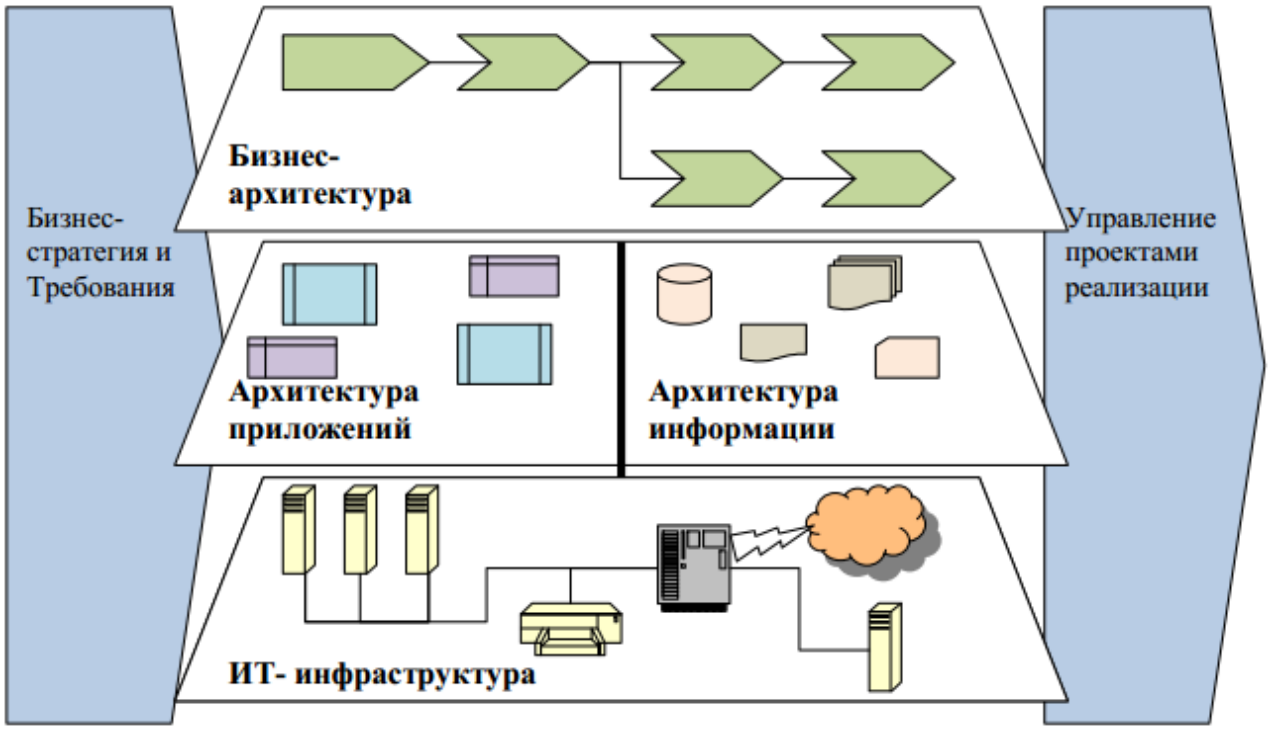

Рисунок 4 – Переход от бизнес-архитектуры к ИТ-архитектуре (приложения, информация, инфраструктура)

Следующим этапом является переход от архитектуры бизнес-процессов и архитектуры данных к созданию архитектуры приложений. На этом этапе необходимо определить классы информационных систем, требуемых для автоматизации, а затем определить необходимые модули для каждой информационной системы. Здесь основой для проектирования архитектуры приложений является модель процессов верхнего уровня (обобщенное представление всех бизнес-процессов предприятия). На этой модели располагаются основные типы информационных систем, которые далее детализируются в виде моделей модулей информационных систем и далее до уровня отдельных экранных форм. Задача построения архитектуры приложений является ИТ-задачей, но нельзя забывать, что решать ее можно только во взаимосвязи с бизнес-процессами.

Дополнительно предметной областью, влияющей на создаваемую архитектуру приложений, являются модели требований к информационной системе. Фактически модели требований – это целевой функционал ИТ-решения, который определяется ключевыми пользователями и структурируется либо по бизнес-процессам, либо по подразделениям. С учетом этих требований и существующих моделей бизнес- процессов, а также построенных моделей данных проектируется новая архитектура приложений. В дополнение к этому на основании взаимосвязи архитектуры бизнес-процессов и архитектуры приложений может быть построена такая прикладная модель, как «карта поддержки процессов информационными системами».

После того, как архитектура приложений сформирована, дальнейшим
этапом является создание архитектуры технологий, представляющей собой элементы ИТ-инфраструктуры, такие как сервера, сетевые элементы и другое оборудование, необходимое для поддержки функционирования приложений.

1.5 Системная архитектура и ее место в архитектуре предприятия

В современных условиях особенно важно постоянно и правильно согласовывать ИТ-аспекты устройства современного автоматизированного предприятия с актуальными бизнес-аспектами.

Известно, что бизнес любой современной компании все больше и больше становится зависим от информационных технологий. Развитие отдельных направлений бизнеса, например, развитие «карточного бизнеса» в банках, стало возможным исключительно благодаря появлению современных ИТ. Конечно, это справедливо и для предприятий других отраслей. Потому можно надеяться, что излагаемый опыт без значительных усилий по адаптации может быть использован в бизнесе любой другой, небанковской компании.

Следует отметить, что архитектура предприятия существует независимо как от нашего сознания, так и от размера этого предприятия — будь то глобальная корпорация, небольшой завод, малое торговое предприятие и т. п. У малого предприятия архитектура есть так же, как и у крупного, при этом они не слишком сильно различаются по составу компонентов. Однако одни руководители это понимают и могут себе позволить уделять внимание всем аспектам устройства своего же предприятия (это, как правило, руководители крупных компаний), а другие — нет. Иное дело, что у малого предприятия могут быть всего два-три продукта, миссия и стратегия в явной форме не зафиксированы, количество сотрудников составляет 30 человек и в производстве используется два компьютера с MS Word 2003. Но и в таком случае это все и составляет архитектуру данного предприятия.

Прозрачность бизнеса начинается с прозрачности понимания архитектуры предприятия, с четкого разделения ее на три взаимозависимых уровня: стратегический уровень, уровень бизнес-архитектуры, уровень системной архитектуры. Системная архитектура определяется бизнес-архитектурой, и ее проектирование может осуществляться только на основании бизнес-архитектуры, которая в свою очередь зависит от стратегии предприятия. Этот подход позволяет, не только правильно организовать сами работы и произвести корректное разделение функций и ответственности бизнес-архитектора («бизнес-девелопера»), отвечающего за развитие бизнеса, то есть бизнес-

архитектуры предприятия, и системного архитектора, но и, самое главное, выстраивать сбалансированную архитектуру предприятия, адекватно соответствующую его миссии и стратегии.

Для целей системного анализа архитектура предприятия может рассматриваться в двух аспектах:

- статическом - по состоянию организации в некоторый фиксированный момент времени;
- динамическом - как процесс перехода (миграции) предприятия от текущего состояния к некоторому желаемому состоянию в будущем.

Рассматриваемая в статике архитектура предприятия состоит из следующих элементов:

- миссия и стратегия, стратегические цели и задачи;
- бизнес-архитектура;
- системная архитектура.

Рассматриваемая в динамике архитектура предприятия — это логически связанный цельный план действий и скоординированных проектов, необходимых для преобразования сложившейся архитектуры организации к состоянию, определенному как долгосрочная цель, базирующийся на текущих и планируемых бизнес-целях и бизнес-процессах организации.

Таким образом, архитектура предприятия в общем случае описывается следующими последовательно зависимыми разделами (см. рис. 5 и рис. 6):

- сформулированные миссия и стратегия предприятия, стратегические цели и задачи;
- бизнес-архитектура в текущем (as is) и планируемом (to be) состоянии,
- системная архитектура в текущем (as is) и планируемом (to be) состоянии;
- планы мероприятий и проектов по переходу из текущего состояния в планируемое.

Рисунок 5 – Архитектура предприятия

На рис. 5 показано, что выполнение плана миграции не означает замораживания развития бизнес- и системной архитектуры. Таким

образом, планируемая системная архитектура является архитектурой «to be» только на определенном витке развития предприятия. Одновременно возврат к стратегическому уровню миссии и стратегических целей и задач не означает необходимость пересмотра миссии и стратегии. Но в конце каждого цикла обязательно проводится анализ эффективности разработанных и осуществленных мероприятий, при необходимости при второй итерации корректируются бизнес-архитектура, системная архитектура, реализуются новые планы миграции. В каждый момент времени может быть несколько циклов, каждый такой цикл не обязательно затрагивает все предприятие в целом, цикл может затрагивать отдельные направления, отдельные вопросы бизнеса и может быть зафиксирован в виде отдельного проекта.

При поэтапном плане миграции для фиксации достигнутых результатов возможно построение промежуточных (миграционных) одной или нескольких архитектур.

Миссия, стратегия и бизнес-цели определяют направления развития предприятия и ставят долгосрочные цели и задачи.

Бизнес-архитектура на основании миссии, стратегии развития и долгосрочных бизнес-целей определяет необходимые организационную структуру, структуру каналов продаж и функциональную модель предприятия, документы, используемые в процессе разработки и реализации продуктов. Функциональная модель описывает бизнес-процессы, направленные на реализацию текущих задач и перспективных целей.

Бизнес-архитектура включает в себя:

- предлагаемые и планируемые к реализации предприятием продукты и услуги (включая индивидуальные схемы их производства), формализованные в виде единого реестра продуктов и услуг, содержащего также клиентскую сегментацию, тарифы и владельцев продуктов;
- каналы продажи продуктов и услуг, построенные как на базе структурных и территориальных подразделений предприятия, так и на базе современных информационных технологий;
- функции и процессы по реализации внешних и внутренних продуктов и услуг, образующие деревья функций и процессов (далее - бизнес-функции и бизнес-процессы);
- финансовые и распорядительные документы (как в бумажном, так и в электронном виде), формализованные в виде единого реестра (альбома форм) документов предприятия;
- документопотоки, определяемые нормативными актами по внутреннему и внешнему документообороту;
- организационную структуру, включая штатное расписание предприятия и его территориальных подразделений, являющихся самостоятельными хозяйствующими единицами (юридическими лицами), комитеты, рабочие группы и ролевые функции отдельных сотрудников,

должностные инструкции, положения о подразделениях и рабочих органах и другие документы, регламентирующие взаимоотношения и распределение ответственности между сотрудниками банка, а также между структурными подразделениями предприятия.

Системная архитектура (ИТ-архитектура, архитектура ИС предприятия) — определяет совокупность технологических и технических решений для обеспечения информационной поддержки работы предприятия в соответствии с правилами и концепциями, определенными бизнес-архитектурой.

Далее под «решениями» будем понимать в зависимости от контекста не только конкретное оборудование и программные и информационные системы, но также принципы, стандарты и методологии, используемые при разработке или выборе информационных и программных систем, при выборе и конфигурации оборудования.

Планы миграции определяют сценарий перехода предприятия от текущего состояния к перспективному, определяемому стратегическими целями и задачами. Планы миграции определяют преобразования как бизнес-, так и системной архитектуры. При поэтапной миграции для целей формализации промежуточных результатов разрабатываются промежуточные (миграционные) одна или несколько бизнес- и системных архитектур. Планы миграции в соответствии с принятой на предприятии методологией управления проектами формализуются в виде отдельных проектов, включающих, в частности:

- определение проекта как совокупности задач и работ;
- фазы и сроки реализации проекта в целом и составляющих проект задач и работ;
- анализ конкурентной среды и рисков, связанных с реализацией проекта;
- состав статей расхода бюджета проекта;
- критерии успешности реализации проекта и ожидаемый экономический эффект.

Системная архитектура состоит из трех взаимосвязанных компонентов — прикладной архитектуры, архитектуры данных и технической архитектуры (см. рис. 6). Системная архитектура в системе стандартов данного предприятия определяет правила формирования своих компонентов и обеспечения взаимодействия между ними.

Прикладная архитектура включает в себя:

- прикладные системы (приложения), обеспечивающие исполнение бизнес- функций и бизнес-процессов;
- интерфейсы взаимодействия прикладных систем между собой и с внешними системами и источниками или потребителями данных;
- средства и методы разработки и сопровождения приложений.

Архитектура данных включает в себя:

- автоматизированные базы данных, обеспечивающие накопление, хранение и обработку данных, определяемых бизнес-архитектурой;
- применяемые для этого системы управления базами данных или хранилищами данных;
- правила и средства санкционирования доступа к данным.

Техническая архитектура состоит из сетевой архитектуры и архитектуры платформ.

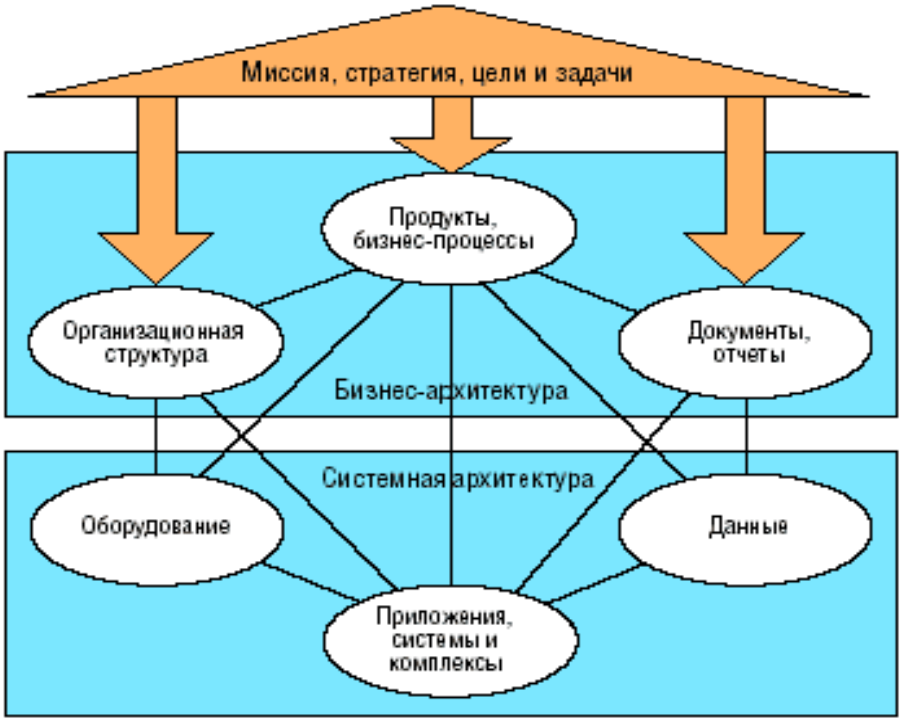

Рисунок 6 – Архитектура предприятия

Сетевая архитектура включает в себя:

- локальные и территориальные вычислительные сети, включая физические собственные и арендованные каналы связи и каналообразующую аппаратуру;
- используемые в сетях коммуникационные протоколы, сервисы и системы адресации;
- аварийные планы по обеспечению бесперебойной работы сетей в условиях чрезвычайных обстоятельств.

Архитектура платформ включает в себя:

- аппаратные средства вычислительной техники - серверы, рабочие станции, накопители и другое компьютерное оборудование;
- операционные и управляющие системы, утилиты и офисные программные системы;

- аварийные планы по обеспечению бесперебойной работы аппаратуры (главным образом - серверов) и баз данных в условиях чрезвычайных обстоятельств.

Для решения задач системной архитектуры в штате компании, как правило, создается выделенная Служба системного архитектора. Эта служба отвечает за решение следующих задач:

- координация работ ИТ-подразделений по документированию текущей системной архитектуры на начальном этапе и последующее поддержание базы знаний о системной архитектуре в актуальном состоянии;
- определение перспективных направлений развития системной архитектуры в соответствии со стратегическими целями и задачами банка, детализированными в форме перспективной бизнес-архитектуры;
- проектирование (совместно с другими профильными подразделениями по информационным технологиям) перспективной системной архитектуры и планов миграции от текущего состояния к перспективному;
- формулирование требований и ограничений к создаваемым или внедряемым средствам автоматизации, обеспечивающим качество и целостность системной архитектуры;
- контроль непротиворечивости системных архитектур, разработанных в рамках различных проектов;
- контроль соблюдения требований по обеспечению качества и целостности системной архитектуры подразделениями банка, осуществляющими разработку, обслуживание и эксплуатацию информационных систем.

Отдельно следует остановиться на одном принципиальном вопросе: кто осуществляет разработку системной архитектуры — системный архитектор или разработчик программного обеспечения, технолог. Правильным является решение, когда ответственность за разработку системной архитектуры закрепляется за ИТ-подразделениями, осуществляющими проектирование, разработку, тестирование, сопровождение (включая вывод из эксплуатации) программно-технических систем и комплексов. Документация по системной архитектуре является частью обязательной проектной и эксплуатационной документации. Этот подход позволяет создать службу системного архитектора небольшой численности. В противном случае разработка системной архитектуры выделенной службой требует значительного увеличения численности системных архитекторов, и процессы разработки либо сильно замедляются, либо разрабатываемая системная архитектура становится неадекватной уже в процессе ее разработки.

Взаимосвязи системной архитектуры и бизнес-архитектуры

Архитектура предприятия полностью описывается следующими сущностями (см. рис. 6).

- Миссия и стратегия, стратегические цели и задачи.
- Продукты и бизнес-процессы.
- Документы.
- Организационная структура.
- Приложения.
- Данные.
- Оборудование.
- Планы мероприятий и проектов по переходу из текущего состояния в планируемое.

На рис. 6 приведены только сущности верхнего уровня. Каждая из сущностей распадается на совокупность более детальных сущностей. Так, только сущность «Продукты» распадается в конечном счете на более чем десять таблиц, включая продуктовые группы, тарифные планы, целевые сегменты клиентов и т. д.

Очевидно, что между всеми этими сущностями имеются сильные взаимосвязи. К примеру, реализация какого-либо продукта сопровождается определенными документами, поддерживается со стороны информационного обеспечения определенными приложениями и модулями, которые используют определенные базы данных. В процессе реализации этого продукта задействованы различные сотрудники и подразделения. Продукт имеет владельца.

На рис. 7 дано укрупненное графическое отображение архитектуры предприятия и определяющих ее компонентов.

Жизненный цикл системной архитектуры

Для регламентации жизненных циклов (ЖЦ) систем в целом (в том числе предприятий), а также их информационных систем и программных средств (ПС), в частности, существует ряд стандартов. Они предусматривают возможности приспособления моделей ЖЦ, в том числе фаз (стадий) к особенностям конкретного предприятия и проекта. Таким образом, описанные в данном и следующем разделах фазы ЖЦ не противоречат нормативным и не являются догмой. Их преимуществом в смысле использования в данной статье является простота и опыт практического применения.

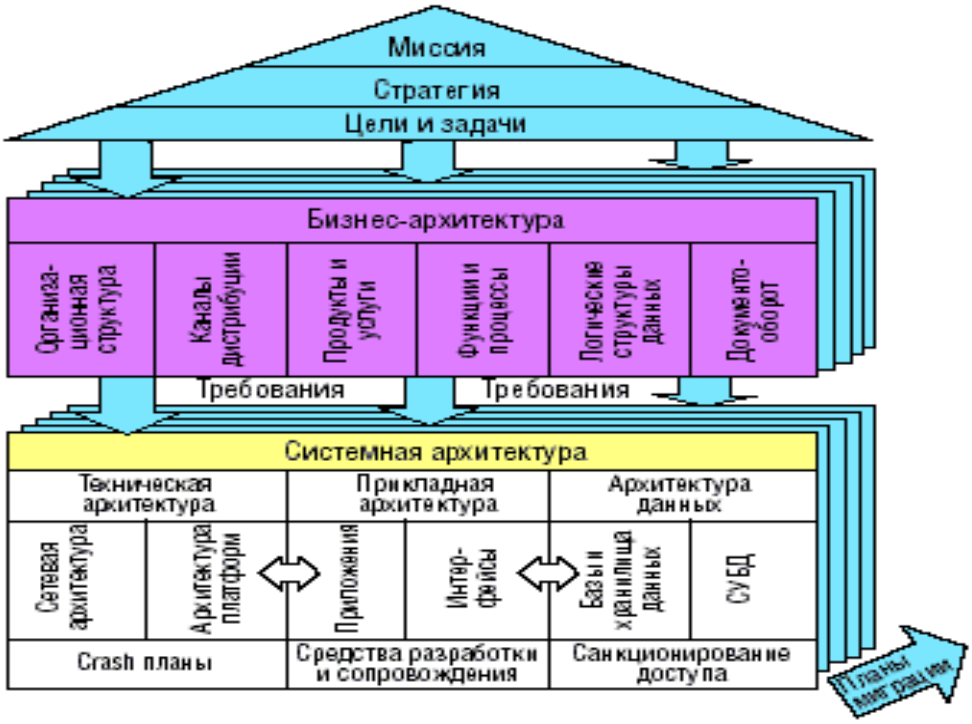

Рисунок 7 - Компоненты архитектуры предприятия

Жизненный цикл системной архитектуры состоит из следующих фаз:

- начальное документирование;
- использование;
- проектирование;
- миграция.

После завершения фазы миграции процесс повторяется, очередная итерация начинается с фазы использования. Фаза начального документирования при разработке новых ИС может отсутствовать. Разработка системной архитектуры начинается с фазы проектирования.

На фазе использования осуществляется эволюционное развитие системной архитектуры в соответствии с ранее сформулированными принципами и без изменения основных технических и технологических решений.

ПРИМЕР. Пусть на фазе проектирования была разработана системная архитектура программы ведения бухгалтерского учета в центральном офисе и филиалах и осуществлено ее внедрение (фаза миграции). Знания о системной архитектуре этого решения переходят в стадию использования до момента возникновения новых бизнес-требований на доработку/модернизацию построенной системы. Знания системной архитектуры созданного решения используются компанией для построения хранилища данных с целью консолидации информации и последующего получения управленческой отчетности. Но основе этих знаний проектируется системная архитектура хранилища данных и затем системы управленческой отчетности, которые в последующем, пройдя свои стадии миграций, входят в фазы использования. Таким образом,

можно говорить о многослойной модели системной архитектуры, в которой системная архитектура в различных слоях может находиться на различных стадиях жизненного цикла.

На фазе проектирования проводится разработка перспективной (to be) системной архитектуры, формулируются новые принципы построения системной архитектуры, вырабатываются в соответствии с этими принципами новые основные технические и технологические решения. Обычно причиной исполнения этой фазы являются существенные изменения в бизнес-архитектуре, появление новых бизнес-требований, существенным образом влияющих на системную архитектуру.

На фазе миграции осуществляется комплекс организационных, технических и технологических мероприятий, обеспечивающих переход системной архитектуры от текущего состояния к перспективному или к очередному промежуточному состоянию при поэтапной миграции в соответствии с подготовленными на предыдущей фазе миграционными планами.

Жизненный цикл системной архитектуры связан с жизненным циклом программных средств. Жизненный цикл программных средств (ПС) состоит из следующих основных фаз:

- анализ осуществимости;
- разработка технического задания;
- разработка технического проекта;
- разработка и документирование ПС;
- тестирование ПС;
- внедрение ПС;
- эксплуатация ПС;
- вывод ПС из эксплуатации.

Системный архитектор выполняет контроль проектных решений на всем жизненном цикле программных средств. Контроль осуществляется в форме согласования проектных документов, подготавливаемых и направляемых системному архитектору подразделениями, ответственными за реализацию той или иной фазы жизненного цикла ПС.

Описания фаз ЖЦ системной архитектуры, состава работ по системной архитектуре, проводимых в каждой фазе, исполнителей этих работ, а также соответствия фазам жизненного цикла ПС представлены ниже.

Начальное документирование

Фазе жизненного цикла системной архитектуры «Начальное документирование» нет прямого соответствия в фазах ЖЦ программных средств. Содержательно эта фаза представлена функциями ее активных участников.

Использование

Фазе жизненного цикла системной архитектуры «Использование» соответствуют следующие фазы ЖЦ программных средств.

- Разработка технического задания на ПС.
- Разработка технического проекта ПС.
- Тестирование ПС.
- Внедрение программных средств.

Проектирование

Здесь может возникнуть вопрос: а куда делась разработка постановки задачи? И нужна ли она вообще? Фазе жизненного цикла системной архитектуры «Проектирование» соответствуют следующие фазы ЖЦ программных средств:

- подготовка технического задания на ПС,
- подготовка технического проекта ПС.

Миграция

Фазе жизненного цикла системной архитектуры «Миграция» соответствуют следующие фазы ЖЦ программных средств:

- тестирование программных средств;
- Внедрение программных средств.

Таким образом, системная архитектура реально затрагивается на следующих фазах ЖЦ программных средств.

На фазе "Разработка технического задания на ПС" производится анализ существующей системной архитектуры с целью возможности и целесообразности использования существующих ресурсов для решения вновь поставленных бизнес-задач. Кроме того, при подготовке технического задания по возможности учитываются требования и ограничения, накладываемые существующей системной архитектурой.

На фазе "Разработка технического проекта ПС" производится собственно формирование или изменение системной архитектуры, необходимое для реализации вновь поставленных бизнес-задач. Требования и ограничения, накладываемые существующей системной архитектурой, учитываются в целях обеспечения преемственности и минимизации расходов на модернизацию.

На фазах "Тестирование" и "Внедрение" разработанных программных средств требования системной архитектуры используются для формирования необходимой технологической среды для проведения испытаний и эксплуатации этих ПС.

Таким образом в разделе рассмотрены основные определения и термины информационных систем. Детально описана архитектура информационной системы. Проанализирован архитектурный подход к реализации информационных систем. Помимо этого, дано определение методологии «архитектуры предприятия». Определено место системной архитектуры в архитектуре предприятия.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Перечислите основные определения информационных систем.

2. Что понимается под архитектурой информационной системы?

3. Опишите архитектурный подход к реализации информационных систем.

4. Основные принципы методологии «архитектуры предприятия».

5. Какое место занимает системная архитектура в архитектуре предприятия?

6. Связь стратегии и архитектуры ИС предприятия.

2. Причины применения архитектурного подхода.

3. Определения архитектуры.

4. Особенности и преимущества архитектурного подхода.

5. Перспективы или уровни описания архитектуры.

6. Эволюция представлений об архитектуре.

7. Контекст архитектуры.

8. Из каких фаз состоит жизненный цикл системной архитектуры?

9. Перечислите компоненты структуры системы управления.

10. Классификация информационных систем

11. Дайте определение ИТ-архитектуре.

12. Дайте определение ИТ-ресурсам.

13. Что включает в себя прикладная архитектура.

14. Перечислите компоненты архитектуры предприятия.

15. Что такое референсная модель?

2 Стратегия развития организации и проектирование архитектуры информационных систем

Стратегия развития информационных систем основывается на общей стратегии развития организации (учреждения, компании) и конкретизирует положения общей стратегии с точки зрения ИТ. Общая стратегия развития организации определяет настоящие и будущие виды деятельности, типы и виды услуг, рынки на которых работает компания и её доли на этих рынках, организационную и территориальную структуру организации. В свою очередь, ИТ-стратегия содержит основные положения использования ИТ в деятельности организации и определяет, как стратегия развития организации будет поддержана средствами ИТ.

2.1 Связь архитектуры информационных систем с ИТ-стратегией организации

Учет стратегии организации при планировании развития информационных систем. Согласно появившейся в 1980-м году и развивающейся по настоящее время концепции стратегической роли информационных систем, ИС являются не просто инструментом, обеспечивающим обработку информации для конечных пользователей внутри предприятия (фирмы). Они становятся «генератором» новых продуктов и услуг, которые должны обеспечить предприятию конкурентную позицию на рынке, а также играют решающую роль в реализации стратегических планов организации. Эффективное использование информации не означает, что необходимо механически повышать скорость и объёмы ее переработки. На первый план выходят такие проблемы, как:

- адекватность собираемой, хранимой, обрабатываемой и предоставляемой пользователям информации их реальным потребностям;
- чёткое представление об информационных потребностях участников бизнес-процессов;
- понимание соответствия бизнес-процессов стратегическим целям организации;
- видение перспектив развития организации и возможное влияние развития ИТ на них.

Для решения всех этих проблем необходим комплексный подход, позволяющий учесть все перечисленные моменты, связи между ними и их согласование со стратегией организации (рис.8).

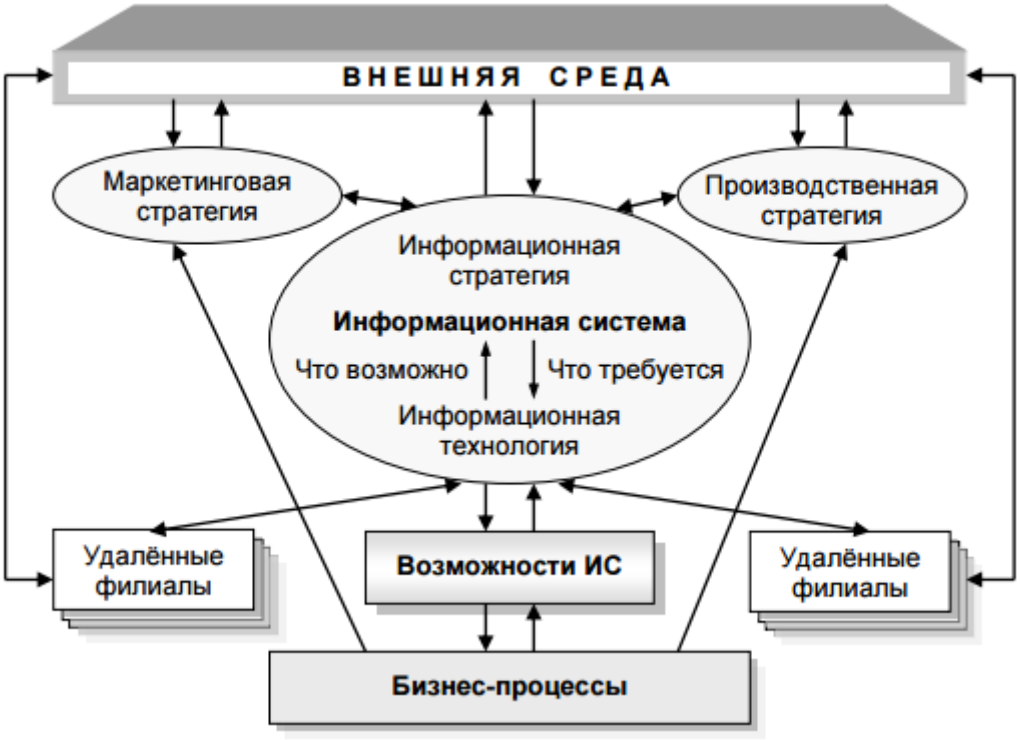

Рисунок 8 – Информационная система и ее окружение

В последнее время в области планирования и создания информационных систем ведутся разработки по созданию подходов разработки гибких корпоративных систем, способных перестраиваться за относительно короткое время согласно требованиям к реструктуризации бизнес-процессов в процессе реализации стратегических планов развития организации.

Анализ состояния развития ИТ в организации. Перед тем, как приступать к разработке (актуализации) ИТ-стратегии организации, необходимо проанализировать существующее состояние ИТ, изучить отраслевые и корпоративные стандарты, а также тенденции развития информационных технологий в конкретной области деятельности. Для того чтобы ИТ-стратегия соответствовала общей стратегии, в ней должны быть определены:

− философия развития ИТ в организации, в том числе место ИТ-подразделений в общей структуре;

− требования к ИТ с позиций стратегии развития организации;

− базовые принципы и направления развития ИТ;

− основные направления совершенствования процессов управления ИТ;

− интегральные характеристики ИТ-бюджета и списка проектов, необходимых для реализации ИТ-стратегии;

− показатели оценки качества и целевые показатели работы ИТ-системы;

− возможные риски и альтернативные варианты развития ИТ.

Важно подчеркнуть, что базовые принципы и направления развития ИТ должны быть детализированы до начала внедрения программных продуктов и информационных систем, например:

− внедрение комплексного продукта (например, системы класса ERP II) и автоматизация на его основе всех бизнес- процессов;

− внедрение нескольких специализированных продуктов, каждый из которых решает отдельный класс задач, и создание единой системы посредством интеграции этих продуктов;

− автоматизация отдельных участков (или бизнес-процессов) посредством внедрения отдельных модулей, входящих в один или в разные продукты;

− проведение заказной разработки одной из подсистем и интеграция её
с другими продуктами в единую информационную систему;

− осуществление комплексной разработки информационной системы организации (предприятия).

Перечисленные примеры позволяют сделать вывод, что стратегия должна включать в себя ответы на ключевые вопросы не только о целях и задачах, но и о процессе внедрения и использования информационных технологий. В частности, следует отразить следующие позиции:

− комплексность автоматизации;

− если не предполагается комплексная автоматизация, то определение
направлений деятельности, бизнес-процессов или подразделений, которые будут информатизироваться;

− порядок процесса информатизации, сроки отдельных этапов;

− выбор используемых продуктов, систем, платформ;

− применение заказных разработок;

− используемые методики интеграции информационных систем;

− способы реализации проектов (использование услуг сторонних компаний, аутсорсинг, выполнение работ силами собственного подразделения и пр.);

− способы поддержки основных ИТ-сервисов (традиционный, SLA).

Следует отметить, что ИТ-стратегия конкретизирует общую стратегию организации (предприятия) с точки зрения ИТ, а ИТ-архитектура рассматривает ИТ-аспекты общей архитектуры организации (предприятия). Определение архитектуры предприятия дано в стандарте ANSI/IEEE Std 1471-2000: «фундаментальная организация системы, реализованная в её компонентах, их взаимоотношениях друг с другом и средой и принципах, определяющих её конструкцию и развитие».

Архитектура предприятия – это концептуальное средство, которое помогает организации понять свою структуру и способы работы. Обычно архитектура предприятия имеет форму большого набора взаимосвязанных моделей, описывающих структуру и функции предприятия.

*Категории моделей архитектуры организации***.** Весь набор моделей архитектуры организации (предприятия) можно условно разделить на четыре категории (ракурса).

*Бизнес-ракурс***.** Бизнес-ракурс описывает бизнес-процессы предприятия или организационные процессы (процедуры) организации. Сюда включаются бизнес-стратегии и планы по переводу предприятия (организации) из текущего состояния в планируемое состояние в будущем. В типовом случае этот ракурс включает:

− цели и задачи верхнего уровня;

− бизнес- процессы, охватывающие всё предприятие или значительную
его часть;

− выполняемые бизнес-функции или организационные процедуры;

− основные организационные структуры организации (предприятия);

− взаимосвязи между всеми перечисленными элементами.

Бизнес-ракурс распространяется на все аспекты деятельности предприятия. Сюда входит технология производства, используемые финансовые и логистические схемы, структура основных средств, классификация норм запасов сырья и комплектующих, структура контрактов с персоналом и многое другое, что характеризует конкретный бизнес.

*Ракурс приложений***.** Ракурс приложений определяет набор приложений предприятия. Обычно этот ракурс включает:

− описание приложений или автоматизированных сервисов, поддерживающих бизнес-процессы;

− описание взаимодействия и взаимозависимостей (интерфейсов) прикладных систем предприятия (организации);

− планы разработки новых и переработки существующих приложений,
основывающиеся на целях и задачах предприятиях, а также на эволюции технологических платформ.

В ракурсе приложений должны быть представлены службы, информация и функциональность, необходимые в масштабах всего предприятия, используемые пользователями различной квалификации, выполняющими разные функции, для достижения общих бизнес- целей.

Ракурс информации. Ракурс информации описывает, какая информация необходима организации для функционирования (выполнения бизнес-процессов). Этот ракурс включает:

− стандартные модели данных;

− политики управления данными;

− описание шаблонов создания и использования информации в организации.

Ракурс информации также содержит описание того, как данные связаны с потоками работ, включая структурированные хранилища данных, такие как базы данных, и неструктурированные хранилища данных, такие как базы документов, таблиц и презентаций, которые используются всей организацией.

*Технологический ракурс***.** Технологический ракурс рассматривает аппаратное и программное обеспечение, используемое в организации. Ракурс включает:

− аппаратные средства серверов и рабочих станций;

− операционные системы;

− средства сетевого доступа;

− принтеры и МФУ;

− другие устройства.

Технологический ракурс обеспечивает логическое описание инфраструктуры и системных компонентов, которые необходимы для поддержания ракурсов приложений и информации. С этого ракурса определяется набор технологических стандартов и сервисов, необходимых для выполнения бизнес-миссии.

Хотя архитектура предприятия может содержать и большее число ракурсов, у каждого предприятия имеется только одна архитектура, которая описывает перспективу его развития. Значение архитектуры предприятия не определяется одним частным ракурсом, а состоит в определении взаимоотношений, взаимодействий и взаимозависимостей между различными ракурсами.

ИТ-архитектура предприятия (организации), являющаяся частью общей архитектуры, включает в себя ракурс приложений и технологический ракурс. Поэтому, рассматривая соответственно ИТ-архитектуру, мы можем говорить об архитектуре приложений и технологической архитектуре предприятия (организации).

Представления архитектуры приложений. Как архитектура приложений, так и технологическая архитектура состоят из представлений:

− концептуального;

− логического;

− физического.

Концептуальное представление является наиболее абстрактным и тяготеет к описанию в терминах, которые более понятны пользователям системы, не являющимся ИТ-профессионалами. Концептуальное представление используется для определения функциональных требований и для построения бизнес-модели на основе представления бизнес-пользователей приложений. Для построения описания ключевых бизнес-

процессов и используемых ими данных применяются такие техники концептуального моделирования, как анализ диаграммы состояний, диаграммы деятельности, моделирование бизнес-сущностей и т. д. Всё это направлено на удовлетворение бизнес-целей и бизнес-требований и не зависит от технологий реализации.

Логическое представление показывает основные функциональные компоненты и их взаимосвязи внутри системы, без определения технических деталей реализации необходимой функциональности. Архитекторы создают модели приложений, которые являются логическим представлением бизнес- моделей, поскольку они определяют, как удовлетворить бизнес- цели и бизнес- требования. Модели приложений представляют собой логические представления архитектуры приложений.

Архитекторы в данном случае работают с общей структурой приложений. Они решают, как будет отображаться управление данными и шаги бизнес-процессов, проектируют взаимодействие между компонентами модели в терминах логических сообщений и последовательностей, и определяют какие данные и состояния может содержать модель.

Физическое представление является наименее абстрактным и иллюстрирует специфику реализации компонентов и взаимосвязей между ними. Каждый элемент физического представления реализуется в процессе проектирования и разработки как программный или аппаратный компонент. Каждый элемент модели приложения должен быть поставлен в соответствие элементам реально существующих технологий. Этим способом модели приложений преобразовываются в модели реализации. Часть этих задач выполняется во время традиционной разработки, когда программисты прописывают детальную бизнес-логику в виде программного кода, но множество действий по реализации хорошо классифицируются как действия по применению специализированной среды. Это такая техника разработки, в которой множество элементов инфраструктуры распределённых приложений и управления данными поддерживается специализированной средой, которая расширяет пользовательскую логику приложений и декларативные структуры управления.

Применение специализированной среды скрывает от разработчика множество сложностей, например, поддержку асинхронного режима приёма-передачи сообщений, а также позволяет разработчику с невысоким профессиональным уровнем реализовывать сложные проекты.

2.2 Состав работ по разработке ИТ-стратегии и ИТ-архитектуры

Рассмотрев специфику применения архитектурного подхода в организациях следует проанализировать взаимосвязь между ИТ-стратегией и ИТ-архитектурой.

Эта взаимосвязь адекватна взаимосвязи между общей стратегией развития предприятия и архитектурой предприятия. Стратегия имеет более общий характер, не так детально рассматривает отдельные аспекты, как архитектура. Но, в отличие от архитектуры, стратегия продолжительна во времени. На оси времени архитектура отражает конкретный момент, а стратегия – период. Можно сказать, что стратегия описывает последовательность преобразования архитектуры во времени. При этом каждая конкретная архитектура в этой последовательности рассматривается не детально, а в общих чертах.

При этом ИТ-стратегия не сводится к описанию последовательности преобразований ИТ-архитектуры. Описание в ИТ-стратегии процесса развития ИТ-архитектуры во времени требует, чтобы в составе стратегии было дано общее направление этого развития, разработаны общие принципы развития, определены критерии достижения заданной цели и требуемые ресурсы.

Разработка ИТ-стратегии. Разработка ИТ-стратегии может осуществляться как в сочетании с последующей детальной разработкой ИТ-архитектуры на ближайшую перспективу, таки без этого этапа. Состав работ по разработке собственно ИТ-стратегии:

− разработка философии развития ИТ в компании и определение места
ИТ-подразделений в структуре предприятия;

− разработка требований к ИТ с позиций бизнес-стратегии;

− разработка оценок качества и целевых показателей работы ИТ-системы;

− определение альтернативных вариантов развития ИТ и анализ возможных рисков;

− определение базовых принципов и направлений развития ИТ;

− определение основных направлений совершенствования процессов управления ИТ;

− определение интегральных характеристик ИТ-бюджета;

− определение списка проектов, необходимых для реализации ИТ-стратегии, их последовательности и сроков;

− определение типовых способов реализации проектов (использование
услуг сторонних компаний, аутсорсинг, выполнение работ силами собственного подразделения и пр.);

− определение способов поддержки основных ИТ-сервисов (традиционный, SLA);

− эскизная разработка ИТ-архитектуры на ближайшую перспективу, включая архитектуру приложений и технологическую архитектуру;

− эскизная разработка ИТ-архитектуры на долгосрочную перспективу, включая архитектуру приложений и технологическую архитектуру.

Разработка архитектуры приложений. В настоящее время для разработки архитектуры приложений используются два подхода:

− разработка архитектуры на основе интеграции приложений (концепция Enterprise Application Integration – EAI);

− разработка сервис-ориентированной архитектуры (Service Oriented Architecture – SOA).

SOA – это новая парадигма проектирования распределенных интегрированных систем. Согласно SOA любые части информационных систем, имеющие функциональность, рассматриваются как службы (Service Providers – провайдеры служб), которые предоставляют свою функциональность другим частям системы по средством обмена сообщениями. Сервисы обеспечивают бизнес-логику и средства управления состояниями, относящиеся к проблеме, для решения которой они предназначены.

В связи с тем, что поставщики корпоративных приложений ещё только ведут работы по переводу своих продуктов на SOA, а пока все большие продукты поставляются в виде монолитных корпоративных приложений, возможны различные варианты рассматриваемой услуги:

− разработка архитектуры на основе концепции EAI, что в настоящее время больше применимо при построении системы на основе готовых существующих приложений;

− разработка сервис-ориентированной архитектуры (SOA), что в настоящее время больше применимо при построении системы на основе заказных разработок или при внедрении продуктов, уже построенных на основе принципов SOA;

− разработка сервис-ориентированной архитектуры (SOA) с преобразованием используемых унаследованных приложений к SOA (в этом случае процесс разработки самой архитектуры аналогичен предыдущему варианту, поэтому мы рассмотрим только этап преобразования используемых унаследованных приложений к SOA).

Разработка архитектуры приложений на основе концепции EAI. Методику разработки архитектуры приложений на основе концепции EAI, в случае, когда осуществляется полное перепроектирование, можно укрупнённо представить следующим образом:

1) обследование предприятия, определение основных функциональных требований к приложениям;
2) выбор базового полнофункционального пакета, удовлетворяющего
сформулированным требованиям;

3) проектирование методов интеграции, выбранной на этапе 2 базовой системы, с уже используемыми унаследованными системами, оценка затрат на интеграцию;
4) определение типов дополнительных систем, которые необходимо
будет внедрить, чтобы полностью удовлетворить потребности, выявленные на первом шаге, и выбор этих систем;
5) проектирование методов интеграции выбранной на этапе 2 базовой системы с дополнительными системами, определёнными на этапе 4, оценка затрат на интеграцию;
6) если затраты (сроки, деньги) на интеграцию сопоставимы с затратами на внедрение более «тяжёлого» пакета, необходимо вернуться на этап 2, повторив процесс выбора с анализом более «тяжелых» (комплексных) систем;
7) определение последовательности внедрения модулей выбранной
комплексной системы, внедрения дополнительных систем и интеграции с уже используемыми системами;
8) разработка требований к технологической архитектуре на основе разработанной архитектуры приложений;
9) в тех случаях, когда базовый пакет заранее предопределён, или частично внедрён и не подлежит замене, может проводиться неполный комплекс работ по уточнению или развитию имеющейся архитектуры приложений (этапы 3, 4, 5, 7 или некоторые из них).

Разработка сервис-ориентированной архитектуры приложений (SOA). Одним из подходов к созданию современных корпоративных информационных систем (ИС) является проектирование сервис-ориентированных архитектур на основе методологии SOA (Service Oriented Architecture). При этом сама SOA представляет собой набор бизнес-методов, методов процесса, организационных методов, методов управления и технических методов для создания гибкой среды. Сервис-ориентированная архитектура предлагает возможность гибкой работы с элементами бизнес- процессов и лежащей в их основе ИТ-инфраструктурой
как с компонентами, которые можно использовать многократно и комбинировать при изменении приоритетов организации.

Технически, реализация архитектур на основе SOA стала возможной в результате развития технологии Web-служб. Современные открытые стандарты Web-служб играют важную роль в организации процессов взаимодействия компонентов ИС различных производителей.

Архитектурные решения, реализованные на основе SOAP, WSDL и UDDI, несмотря на свою кажущуюся избыточность, показывают свою

жизнеспособность и полезность. Механизм сервисов SOAP является каркасом для интеграции бизнес-процессов и поддерживающей их ИТ-инфраструктуры в форме безопасных, стандартизированных компонентов (служб), предназначенных для многократного использования.

Использование подходов SOA в большинстве случаев позволяет реорганизовать процесс развития корпоративной ИС. С точки зрения сервис-ориентированной архитектуры жизненный цикл корпоративной системы целиком «распадается» на жизненные циклы составляющих ее компонентов. Такая декомпозиция позволяет не только оперативно реагировать на реструктуризацию бизнес-процессов, но и делает процесс развития ИС более предсказуемым и устойчивым.

В процессе проектирования сервис-ориентированной архитектуры приложений в первую очередь должно быть разработано концептуальное представление. В ходе его разработки должны быть определены следующие компоненты.

Сервисы. При проектировании сервисов основная задача состоит в том, чтобы эффективно инкапсулировать логику и данные, связанные с процессами в реальном мире. Значительные интеллектуальные усилия требуются для принятия решений, что можно объединить, а что должно быть реализовано отдельными сервисами.

Сообщения. Сервисы взаимодействуют между собой, обмениваясь сообщениями. Должны быть полностью определены сообщения, которые порождают и принимают сервисы, включая требования к последовательности этих сообщений.

Контракты. Каждый контракт описывает метод взаимодействия двух
сервисов. В это описание входит: перечень посылаемых каждым сервисом сообщений, их форматы, методы отправки, последовательность обмена сообщениями, перечень принимаемых каждым сервисом сообщений и способы приёма.

Политики. Политики должны давать возможность влиять на работу приложений, т. е. устанавливать и изменять правила, действующие во время выполнения, которые определяют методы работы сервисов и их взаимодействие. Разработка политик в ходе процесса проектирования ведёт к увеличению гибкости и управляемости приложений.

Состояния. Сервисы управляют состояниями и состояния, часто, являются главной причиной их существования. Состояние – это то, что хранится в некоторой долгосрочной среде, такой как файловая система или база данных. Сервисы гарантируют по средством своей бизнес-логики, содержательность, непротиворечивость и точность сохраняемых состояний. В процессе работы сервисы будут получать запросы от других сервисов, извлекать некоторые состояния из этой среды длительного хранения и строить ответы или корректировать эти состояния.

Процессы. Каждый процесс управляет последовательностью действий при выполнении некоторой работы, постепенно переводя систему из одного состояния в другое. В сервис-ориентированной архитектуре должны быть спроектированы бизнес-сервисы, построенные по традиционным принципам, и процессные сервисы, которые будут координировать выполнение бизнес-сервисов.

Приложения. Приложения объединяют процессные сервисы, бизнес-сервисы и сервисы пользовательских интерфейсов. Бизнес-сервисы обычно проектируются в четыре слоя: сервисы фасада, сервисы бизнес-процессов, сервисы бизнес-сущностей и сервисы представления данных. Такая модель работоспособна как для традиционных типов приложений, которые имеют интерфейс для взаимодействия пользователей с бизнес-сервисами, так и для сервисов, взаимодействующих с другими сервисами.

Помимо концептуального представления при проектировании сервис-ориентированной архитектуры должны быть спроектированы логическое
представление и физическое представление. Мы не будем на них подробно останавливаться, поскольку они в существенно меньшей степени отличаются от соответствующих представлений при проектировании традиционной архитектуры.

Преобразование приложений к сервис-ориентированной архитектуре (SOA). Процесс преобразования существующей архитектуры информационных систем в сервис-ориентированную архитектуру состоит из семи шагов и представлен схемой на рисунке 9.

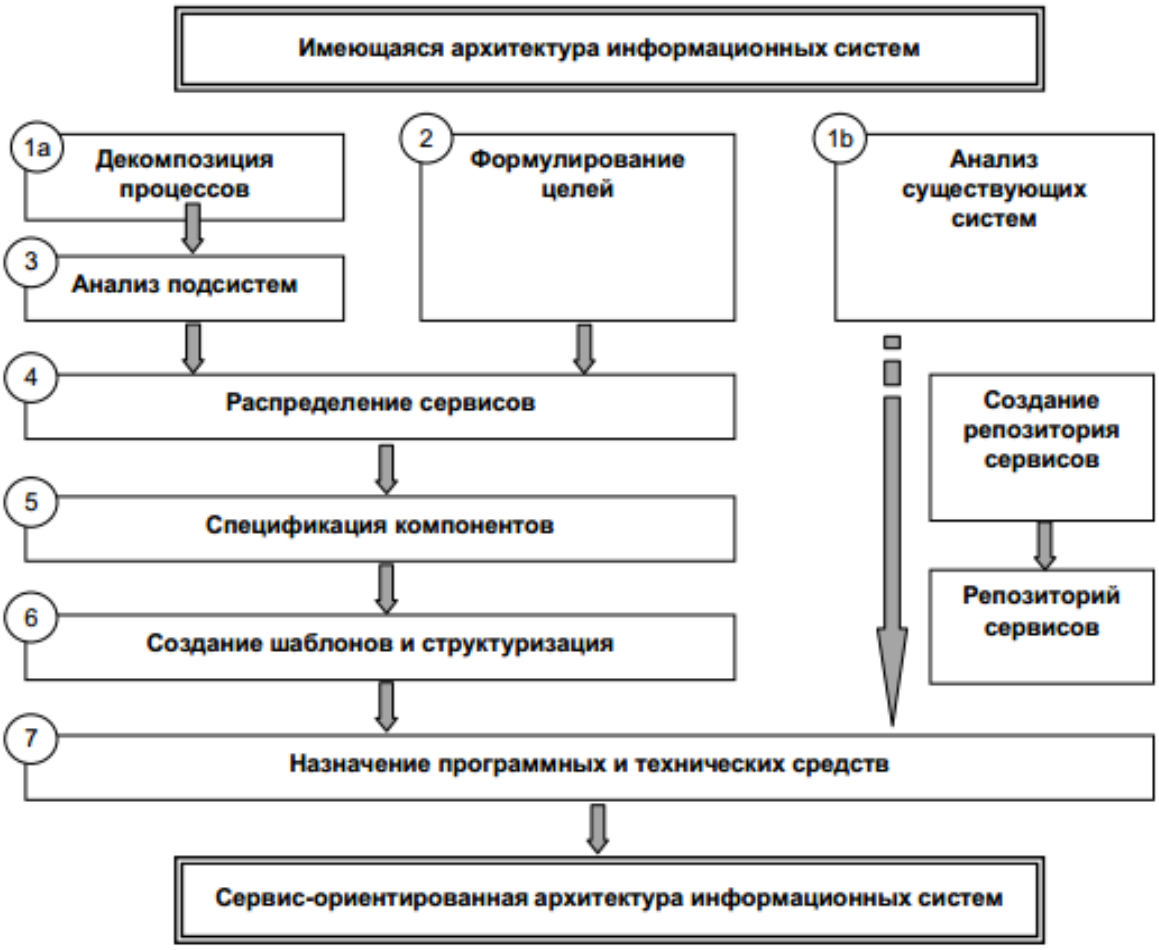

Рисунок 9 – Схема процесса преобразования имеющейся архитектуры информационной системы в сервис-ориентированную архитектуру

Схема описывает последовательность этапов (шагов) процесса преобразования архитектуры информационной системы в сервис-ориентированную:

− шаг 1a – декомпозиция предметной области (определение бизнес-процессов, подпроцессов, вариантов использования);

− шаг 1b – анализ существующих не объектно- ориентированных систем и преобразование их к компонентной архитектуре;

− шаг 2 – создание дерева целей сервисной модели для тестирования полноты сервисной модели (каждой подцели в дальнейшем будет соответствовать определённый сервис);

− шаг 3 – анализ подсистем, определение того, какие варианты использования реализуются какими компонентами системы, анализ взаимодействия компонентов и влияния нефункциональных требований на архитектуру системы;

− шаг 4 –определение, какие компоненты отвечают за какие сервисы, определение сервис- провайдеров и сервис-потребителей;

− шаг 5 – определение интерфейсов каждого компонента;

− шаг 6 – структуризация компонентов и сервисов на основе применяемых шаблонов архитектуры;

− шаг 7 – определение программных и технических средств с помощью которых будет реализован каждый сервис.

Разработка технологической архитектуры. Технологическая архитектура включает в себя следующие компоненты:

− сетевую архитектуру;

− архитектуру хранения;

− архитектуру инфраструктуры приложений;

− архитектуру управления;

− архитектуру безопасности.

Работы по разработке технологической архитектуры должны начинаться с обследования имеющейся ИТ-инфраструктуры предприятия (учреждения) и определения её соответствия требованиям архитектуры приложений. Далее для каждого из перечисленных компонентов технологической архитектуры должны быть выполнены:

− разработка концептуального представления;

− разработка логического представления;

− разработка физического представления.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Рамочная модель разработки архитектуры.
2. Домены (предметные области) архитектуры.
3. Архитектура информации.
4. Архитектура приложений.
5. Технологическая архитектура.
6. Сервис- ориентированная архитектура.
7. Модель Захмана.
8. Методика описания архитектуры TOGAF.
9. Основные элементы архитектурного процесса.
10. Творческий характер архитектурного процесса.
11. Учет стратегии организации при планировании развития информационных систем.
12. Анализ состояния развития ИТ в организации.
13. На какие категории можно условно разделить весь набор моделей архитектуры организации?
14. Из каких представлений состоит архитектура приложений?
15. Разработка ИТ- стратегии.
16. Разработка архитектуры приложений.

17. Разработка архитектуры приложений на основе концепции EAI.
18. Этапы процесса преобразования архитектуры информационной системы в сервис-ориентированную.
19. Какие компоненты включает в себя технологическая архитектура?

II Практикум по системной архитектуре информационных систем

Практическая работа 1 «Установление требований к разрабатываемой информационной системе»

Задание

Предложить для разработки информационную систему (ИС). ИС должна представлять собой программный комплекс, наделенный функциональностью, автоматизирующей конкретную деятельность в

рамках предметной области, для которой разрабатывается система. Примером таких систем могут служить:

- автоматизированные системы управления (АСУ);
- электронные магазины, аукционы;
- веб-порталы;
- сервисы и т.п.

Необходимо составить документ описания требований к разрабатываемой ИС согласно шаблону (рис.7).

Варианты заданий представлены в приложении.

Установление требований

Документ описания требований. Документ, описывающий требования, является осязаемым результатом этапа установления требований. Большинство организаций вырабатывает документ описания требований в соответствии с заранее определенным шаблоном. Шаблон определяет структуру (содержание) и стиль документа.

Ядро документа описания требований состоит из формулировок (изложения) требований. Требования могут быть сгруппированы в виде *формулировок сервисов* (зачастую называемых функциональными требованиями) и *формулировок ограничений*. Формулировки сути сервисов могут быть затем разделены на *требования к функциям* (*function requirements*) и *требования к данным* (*data requirements*). (В литературе термин «*функциональные требования*» (*functional requirements*) в широком и в узком смысле используется как взаимозаменяемый. При использовании в узком смысле он соответствует тому, что мы называем требованиями к функциям).

Не говоря уже о самих требованиях, документ описания требований должен обращаться к проектным вопросам. Обычно проектные вопросы рассматриваются в начале документа, а затем в конце документа.

Во вводной части документа рассматривается бизнес-контекст проекта, включая цель проекта, участников проекта и основные ограничения. Ближе к заключительной части документа поднимаются другие проектные вопросы, включая план-график выполнения проектных работ, бюджет, риски, документацию и т. д.

Шаблоны документа. Шаблоны для документов описания требований широко доступны. Их можно найти в учебниках, стандартах, выпускаемых такими организациями как ISO, IEEE и т. д., на Web-страницах консалтинговых фирм, программных средствах разработки и т. д. Со временем каждая организация разрабатывает свои собственные стандарты, которые соответствуют принятой в организации практике, корпоративной культуре, кругу читателей, типам разрабатываемых систем и т. д.

Шаблон документа описания требований определяет структуру документа и содержит подробные указания о содержании каждого из разделов документа. Указания могут включать содержание вопросов, мотивацию, примеры и дополнительные соображения.

На рис. 10 показано типичное оглавление документа описания требований. Последующие разделы включают объяснение к приведенному оглавлению.

Предварительные замечания к проекту. Часть документа описания требований, содержащая предварительные замечания к проекту, преимущественно дает ориентиры тем руководителям и участникам проекта, ответственным за принятие решений, которые, вероятно, не станут подробно изучать документ целиком. В начале документа необходимо ясно обозначить цели и рамки проекта, а затем описать деловой контекст системы.

Документ описания требований должен создать прецедент для системы. В частности, необходимо упомянуть обо всех усилиях, приложенных для обоснования необходимости системы на этапе планирования системы. Документ описания требований должен прояснить вопрос о том, каким образом предлагаемая система может способствовать достижению деловых целей и решению задач организацией.

Необходимо обозначить участников проекта системы. Важно, чтобы заказчик выступал не в виде безликого подразделения или офиса — необходимо привести конкретные имена. К концу дня человек должен быть в состоянии решить, приемлемо ли поставляемое программное обеспечение (ПО) для организации.

Хотя документ описания требований может быть как угодно далек от технических решений, все же важно обсудить идеи, касающиеся решения на самых ранних этапах жизненного цикла (ЖЦ) разработки.

Особый интерес представляют готовые решения. Всегда неплохо рассмотреть вариант приобретения готового продукта вместо его разработки «с нуля».

Документ описания требований

Содержание документа

Рисунок 10 - Содержание документа описаний требований

Документ описания требований должен предоставлять перечень существующих программных пакетов и компонент, которые должны быть в дальнейшем изучены в качестве вариантов возможных решений.

Обратите внимание, что приобретение готового решения изменяет процесс разработки, однако это не избавляет от необходимости проведения анализа требований и проектирования системы! Наконец,

неплохо в заключение раздела предварительных замечаний к проекту документа описания требований привести обзор оставшейся части документа. Это может подтолкнуть к тому, чтобы изучить остальные части документа, а также способствует лучшему пониманию содержания документа. Обзор также может содержать пояснения в отношении методологии анализа проектирования, выбранной разработчиками.

Системные сервисы. Основная часть документа описания требований посвящена определению системных сервисов. Эта часть может занимать до половины всего объема документа. Это также, пожалуй, единственная часть документа, которая может содержать обобщенные модели — модели бизнес-требований.

Рамки системы можно моделировать с помощью диаграммы контекста. В пояснениях к диаграмме контекста должны быть четко определены рамки системы. Без подобного определения проект не может быть застрахован от попыток «растянуть» его рамки.

Функциональные требования можно моделировать с помощью диаграммы бизнес-прецедентов. Однако диаграмма охватывает перечень функциональных требований только в самом общем виде. Все требования необходимо обозначить, классифицировать и определить.

Требования к данным можно моделировать с помощью диаграммы бизнес-классов. Так же, как и в случае функциональных требований, диаграмма бизнес-классов не дает полного определения структур данных для бизнес-процессов.

Каждый бизнес-класс требует дальнейших пояснений. Необходимо описать атрибутное наполнение классов и определить идентифицирующие атрибуты классов. В противном случае невозможно правильно представить ассоциации.

Системные ограничения. Системные сервисы определяют, что должна делать система. Системные ограничения определяют, насколько система ограничена при выполнении обслуживания. Системные ограничения связаны со следующими видами требований.

- Требования к интерфейсу.
- Требования к производительности.
- Требования к безопасности.
- Эксплуатационные требования.
- Политические и юридические требования.

Требования к интерфейсу определяют, как система взаимодействует с пользователями. В документе описания требований определяется только «впечатление и ощущение» от интерфейса. Начальное проектирование (закрашивание экрана) интерфейса проводится во время спецификации требований и позже во время системного проектирования.

В зависимости от области приложения требования к производительности могут играть довольно значительную роль в успехе проекта. В узком смысле они задают скорость (время отклика системы), с которой должны выполняться различные задания. В широком смысле, требования к производительности включают другие ограничения — в отношении надежности, готовности, пропускной способности и т. д.

Требования к безопасности описывают пользовательские права доступа к информации, контролируемые системой. Пользователям может быть предоставлен ограниченный доступ к данным или ограниченные права на выполнение определенных операций с данными.

Эксплуатационные требования определяют программно-техническую среду, если она известна на этапе проектирования, в которой должна функционировать система. Эти требования могут оказывать влияние на другие стороны проекта, такие как подготовка пользователей и сопровождение системы.

Политические требования и юридические требования скорее подразумеваются, чем явно формулируются в документе описания требований. Подобная ошибка может обойтись очень дорого. Пока эти требования не выведены явно, программный продукт может быть трудно развернуть по политическим или юридическим причинам. Возможны и другие виды ограничений. Например, в отношении некоторых систем могут предъявляться повышенные требования к легкости их использования (требования в отношении пригодности к использованию) или легкости их сопровождения (требования в отношении пригодности к сопровождению).

Значение выработки недвусмысленных определений для системных ограничений трудно переоценить. Существует немало примеров проектов, которые провалились из-за упущенных или неверно понятых ограничений. Эта проблема в равной мере относится как к заказчикам, так и к разработчикам. Недобросовестные или нерассудительные разработчики могут разыграть «карту системных ограничений», чтобы получить преимущество в своем стремлении уклониться от ответственности.

Проектные вопросы. Заключительная часть документа описания требований обращается к другим проектным вопросам. Один из важных разделов этой части называется «Открытые вопросы».

Здесь поднимаются все вопросы, которые могут сказаться на успехе проекта и которые не рассматривались в других разделах документа. Сюда относится ожидаемое возрастание значения некоторых требований, которые в текущий момент выходят за рамки проекта. Сюда можно отнести также любые потенциальные проблемы и отклонения в поведении системы, которые могут начаться в связи с развертыванием системы.

В этой же части необходимо представить предварительный план-график выполнения основных проектных заданий. Сюда же относится

предварительное распределение людских и других ресурсов. Для выработки стандартных плановых графиков можно использовать программные средства управления проектами, например, такие как система PERT (program evaluation_and_review technique — метод оценки и пересмотра планов) или карты Ганта.

Прямым результатом составления план-графика может быть разработка предварительного бюджета. Стоимость проекта может быть выражена скорее в виде диапазона значений затрат, а не конкретного значения. При наличии надлежащим образом документированных требований для оценки затрат можно использовать один из подходящих методов.

Приложения. Приложения содержат остальную, полезную для понимания требований, информацию. Основным добавлением здесь служит
глоссарий. Глоссарий определяет термины, сокращения и аббревиатуры, используемые в документе описания требований. Значение толкового глоссария трудно переоценить. Неверное истолкование терминологии таит в себе большую опасность для проекта.

Одна из особенностей, которую часто упускают из виду при составлении документа описания требований, состоит в том, что в проблемной области, определяемой документом, можно довольно неплохо разобраться с помощью изучения документов и форм, используемых в процессах делопроизводства. При возможности следует включать в документ заполненные формы — «пустые» формы не дают такого же уровня понимания бизнес-процессов.

Раздел ссылок содержит перечень документов, которые упоминаются или используются при подготовке документа описания требований. К ним могут относиться книги и другие опубликованные источники информации, но — что, пожалуй, даже более важно — необходимо также упомянуть протоколы совещаний, служебные записки и внутренние документы.

Пример документа описания требований

Документ описания требований ИС «Домашняя бухгалтерия»
1 Предварительные замечания к проекту

1.1 Цели и рамки проекта

Целью данного проекта является разработка информационной системы для ведения и оптимизации семейного бюджета. ИС «Домашняя бухгалтерия» должна быть проста в использовании и не требовать от пользователя знаний бухгалтерского учета.

1.2 Деловой контекст

Многие семьи в наше время планируют семейный бюджет. Ведение семейного бюджета при помощи подручных средств – карандаш, бумага – не всегда удобно и всегда трудоемко. Использование для этих целей компьютерных программ для ведения бухгалтерии не оправдано с точки зрения сложности их освоения и избыточного функционала для ведения домашней бухгалтерии. В связи с этим возникает необходимость создания специализированной программы ведения домашней бухгалтерии.

1.3 Участники проекта

Заказчик – Васильева Марья Федоровна (m.vasileva@mypochta.ru)

Разработчик – Петров Степан Николаевич (petrov@coolsoft.com)

1.4 Идеи в отношении решений

Программа должна быть реализована в виде настольного приложения для операционных систем семейств MS Windows.

1.5 Обзор документа

В разделе «Системные сервисы» описывается, что должна делать система. В разделе «Системные ограничения» определяется, насколько система ограничена при выполнении обслуживания. В разделе «Проектные вопросы» освещаются прочие проектные вопросы.

2 Системные сервисы

2.1 Рамки системы

Рамки системы можно моделировать с помощью диаграммы контекста (рис.11).

ИС «Домашняя Бухгалтерия» получает данные о доходах и расходах от внешней сущности «Домохозяин». Для передачи этих данных сущности «Домохозяин» должен авторизоваться. В своей работе сущность «Домашняя Бухгалтерия» использует информацию о ценах на товары и тарифах и курсах валют, получаемую от внешних сущностей «База тарифов и цен на товары» и «База курса валют». Результаты своей работы ИС «Домашняя Бухгалтерия» может отображать как внешней сущности «Домохозяин», так и генерировать в виде отчетов формата MS Excel для внешней сущности «MS Excel».

Рисунок 11 - Контекстная диаграмма ИС «Домашняя Бухгалтерия»

2.2 Функциональные требования

ИС должна обеспечивать следующие функциональные возможности:

- учет расходов;
- учет доходов;
- учет денег, отданных и взятых в долг;
- погашение долгов частями;
- проценты по долгам;
- контроль возврата долгов;
- система напоминания по долгам;
- составление бюджета расходов и доходов;
- планирование расходов;
- планирование доходов;
- система счетов;
- возможность использовать до пяти валют включительно;
- получение курсов валют из интернет;
- обмен валют;
- импорт данных из файлов Microsoft Excel;
- поиск по базе данных;
- фильтры и быстрый поиск по базе данных;
- экспорт данных в Excel, XML, текстовый файл;
- перенос данных;
- резервное копирование;
- печать данных;
- построение отчетов и диаграмм;
- настройка пользовательского интерфейса.

2.3 Требования к данным

ИС должна хранить свои данные в специализированных XML-файлах.

3 Системные ограничения

3.1 Требования к интерфейсу

ИС должна иметь стандартный интерфейс приложений, разработанных для ОС MS Windows.

3.2 Требования к производительности

Особых требований к производительности ИС нет.

3.3 Требования к безопасности

С программой могут работать несколько человек, входя в программу под своими именами. Для обеспечения конфиденциальности каждое имя можно защитить паролем. Добавление, изменение и удаление пользователей осуществляется в администраторе пользователей.

3.4 Эксплуатационные требования

ИС должна функционировать на ОС Windows XP, ОС Windows Vista, ОС Windows 7 и выше. Минимальные аппаратные требования определяются минимальными аппаратными требованиями к выше-перечисленным ОС.

3.5 Политические и юридические требования

Нет.

3.6 Другие ограничения

Нет.

4 Проектные вопросы

4.1 Открытые вопросы

Нет.

4.2 Предварительный план-график

1.09.2016 – 1.10.2016 – Анализ и установление требований к ИС
1.10.2016 – 1.11.2016 – Спецификация требований к ИС
1.11.2016 – 1.12.2016 – Кодирование ИС
1.12.2016 – 31.12.2016 – Тестовая эксплуатация ИС
11.01.2017 – 13.12.2017 – Ввод в эксплуатацию

4.3 Предварительный бюджет

Пятьдесят тысяч рублей.

5 Приложения

Глоссарий

ИС – информационная система
ОС – операционная система

Деловые документы и формы

Нет.

Ссылки

Нет.

Практическая работа 2 «Спецификация состояний информационной системы»

Задание

Составить спецификацию установленных в практической работе 1 требований для проектируемой ИС. Для составления спецификации использовать язык UML.

Необходимо построить UML-диаграммы: вариантов использования системы, классов и развертывания.

Примечание

Построение UML-диаграмм желательно осуществлять с помощью CASE-средств (StarUML, Rational Rose, MS Visio 2003 и др.).

Спецификация требований. Требования необходимо специфицировать (т.е. задать) графически или каким-либо иным формальным способом. Всесторонняя спецификация системы может потребовать использования многих типов моделей. Язык UML изобилует интегрированными методами моделирования, способными помочь бизнес-аналитику справиться с этой задачей. Спецификация — подобно процессу разработки ПО в целом — итеративный процесс с пошаговым наращиванием уровня детализации моделей. Немаловажную роль в успешном моделировании играет использование CASE-средств.

В результате спецификации требований вырабатываются три категории моделей: модели состояний, модели поведения и модели изменения состояния. Для каждой из категорий существует несколько методов работы с ними. Далее объясняются и иллюстрируются на примерах все основные методы моделирования языка UML.

Несмотря на то, что мы начнем изучение с моделей состояния, затем перейдем к моделям поведения, а затем — к моделям изменения состояний, это не отражает реальной последовательности, в которой проводится моделирование. Многие модели разрабатываются параллельно и служат источником взаимного развития. Это особенно справедливо в отношении двух основополагающих типов моделей — моделей классов и моделей прецедентов.

Принципы спецификации требований. Спецификация требований связана с доскональным моделированием требований заказчиков, определенных в процессе установления требований. При этом рассматриваются только услуги, которые стремятся получить от системы заказчики (формулировки сервисов). На этапе спецификации требований формулировки ограничений не подлежат дальнейшей проработке, хотя и могут претерпеть изменения как результат обычного цикла итерации.

В качестве входной информации процесса спецификации требований выступают неформальные требования заказчиков, а результатом этого процесса являются модели спецификации проектных конструкций. Эти

модели дают более формальное определение различных сторон (представлений) системы. Обычно требования пользователей в процессе спецификации подразделяются на две основные категории: функциональные требования и требования к данным. В качестве результата этапа спецификации выступает расширенный («детально проработанный») документ описания требований. Новый документ часто называют *документом спецификации требований* (или просто «спецификацией» на жаргоне разработчиков). Структура исходного документа не изменяется, однако содержание значительно расширяется за счет глав, которые определяют требования заказчиков. Постепенно для целей проектирования и реализации документ спецификации требований заменяет документ описания требований (на практике расширенный документ может по-прежнему называться документом описания требований).

Модели спецификаций можно разделить на три группы.

1. Модели состояний.
2. Модели поведения.
3. Модели изменения состояний.

Модели состояний «детализируют» требования к данным. Модели поведения обеспечивают детализированные спецификации для функциональных требований. Модели изменения состояний охватывают два вида требований. Они призваны объяснить, каким образом действие функций приводит к изменению данных.

Модели представляются в виде диаграмм на языке визуального моделирования (Visual Modeling Language) — в нашем случае это язык UML. Обычно диаграмма служит целям моделирования одной из сторон системы — состояний, поведения или изменения состояний. Заметное исключение составляет диаграмма классов, которая определяет все три аспекта — состояние и поведение объектов, и, косвенно, изменения состояний объектов.

Каждая диаграмма дает представление об определенной стороне системы. Взятые вместе диаграммы дают возможность разработчикам и пользователям взглянуть на предлагаемое решение с разных точек зрения, выделяя одни его стороны и игнорируя другие. Ни одна из диаграмм в отдельности не дает полного определения системы. Систему можно понять только через взаимосвязанный набор диаграмм. Аналогично случаю интерпретации завершенных моделей конструирование диаграмм — это не последовательный процесс построения одной диаграммы за другой.

Диаграммы разрабатываются параллельно, и в результате каждой последующей итерации к ним добавляются новые детали. В то время, как разработчики должны следовать строго определенному процессу разработки, решение о том, какая из моделей должна играть роль «движущей силы» разработки, в значительной мере зависит от личных предпочтений аналитика. Обычно диаграммы прецедентов и модели

классов — как наиболее важные типы моделей — конструируются параллельно, взаимно «обогащая» друг друга идеями.

С каждой новой итерацией разработки глубина и степень детализации спецификации возрастает. Многие более глубокие свойства объектов модели выражаются скорее в текстовом, нежели графическом виде. Некоторые свойства определяют замысел объекта модели, а не результат анализа. Некоторые другие свойства могут отражать особенности CASE-средств.

Спецификации состояний. Состояние объекта определяется значениями его атрибутов и ассоциаций. Например, объект BankAccount (Банковский счет) может находиться в состоянии «превышение кредитного лимита», если значение атрибута balance (баланс) отрицательно. Поскольку состояния объекта определяются структурам данных, модели структур данных называются *спецификациями состояний*. Спецификация состояний дает статический взгляд на систему (поэтому моделирование состояний часто называют *статическим моделированием*). Здесь основной задачей является определение классов проблемной области, их атрибутов и отношений с другими классами. Вначале операции классов обычно не рассматриваются. Они выводятся из моделей спецификации поведения.

В типичной ситуации сначала определяются классы-сущности, т. е. классы, которые определяют проблемную область и характеризуются постоянным присутствием в базе данных системы. Подобные классы иногда называются «бизнес-классами». Классы, которые обслуживают системные события (управляющие классы) и классы, которые представляют интерфейс (классы представления или пограничные классы), не устанавливаются до тех пор, пока не станут известны поведенческие характеристики системы.

Информационная система «Запись на университетские курсы». В качестве примера для спецификации требований выбрана информационная система «Запись на университетские курсы».

Постановка задачи.

Средний по размерам университет проводит набор студентов и аспирантов дневной и вечерней форм обучения для подготовки по ряду специальностей. Учебная структура университета состоит из факультетов. Факультет имеет в своем составе несколько кафедр. Хотя обучение и присвоение степени по определенной специальности является прерогативой факультета, обучение по специальности может включать дисциплины, преподаваемые на других факультетах.

В действительности, университет гордится свободой, предоставляемой им студентам в выборе дисциплин, изучаемых для получения специальности.

Гибкость выбора учебных курсов оказывает дополнительную нагрузку на университетскую систему набора. Индивидуально

подобранным программам обучения должны быть противопоставлены правила, регулирующие получение степени по выбранной специальности. Такие правила можно сформулировать, например, в виде структуры дисциплин, изучение которых является обязательной предпосылкой получения диплома, так, чтобы студент мог прослушать обязательные для данной специальности курсы. Выбор студентами дисциплин может быть ограничен несоответствием расписаний, максимальной вместимостью аудиторий и т. д.

Гибкость в получении образования, предлагаемого университетом, стала одной из причин роста количества студентов. Однако для сохранения своих традиционно сильных сторон система набора – по-прежнему, частично ручная – должна быть заменена новым программным решением. Предварительно поиск готовых программных пакетов не дал результатов. Университетская система набора достаточно уникальна, чтобы оправдать разработку ПО собственными силами.

От системы требуется оказывать помощь в предварительной деятельности по записи на курсы и управляться с процедурами набора. Предварительная деятельность по записи должна включать рассылку оценок последнего семестра студентам наряду с инструкциями по записи на лекции. В период записи на курсы система должна принимать заявленные студентами программы обучения и проверять их на предмет обязательности, конфликтов расписания, вместимости аудиторий, специальных санкций и т. д. Решения по некоторым проблемам могут требовать консультаций с научными руководителями или профессорами, ответственными за преподавание дисциплин.

Выявление классов. Два разных аналитика, как правило, не могут прийти к идентичным моделям классов для одной и той же проблемной области, и точно так же два разных аналитика не пользуются одним и тем же мыслительным процессом при выделении классов. Литература изобилует подходами, предлагаемыми для выявления классов. Аналитики могут поначалу даже следовать одному из этих подходов, однако последующие итерации, как правило, обязательно приводят к использованию нешаблонных и в чем-то даже случайных механизмов. Ниже перечислены эти подходы.

1. Подход на основе использования именных групп.
2. Подход на основе использования общих шаблонов для классов.
3. Подход на основе использования прецедентов.
4. Подход CRC (class–responsibility–collaborators – класс–обязанности–«сотрудники»).

Некоторые правила выявления классов. Ниже приведен далеко не полный перечень руководящих принципов или правил, которым должен следовать аналитик при выборе потенциальных классов. Вновь

напоминаем о том, что здесь мы имеем дело только с классами-сущностями.

1. Для каждого класса должно быть ясно сформулировано его назначение в системе.

2. Каждый класс — это шаблон описания множества объектов.

Единичные классы, для которых можно представить существование только одного объекта, весьма маловероятны среди «бизнес-объектов». Подобные классы обычно составляют в приложении «общее знание» и, как правило, жестко запрограммированы в программах приложения. Например, если система спроектирована для единственной организации, существование класса Organization (Организация) может быть не оправдано.

3. Каждый класс (т. е. класс-сущность) должен содержать набор атрибутов. Хорошим приемом является установление идентифицирующих атрибутов (ключей), чтобы помочь нам судить о мощности (cardinality) класса (т. е. ожидаемом количестве объектов данного класса в базе данных). Следует, однако, помнить о том, что класс не обязательно должен обладать пользовательским ключом.

4. Каждый класс должен отличаться от атрибута. Представляется ли понятие классом или атрибутом зависит от области приложения. Цвет автомобиля обычно воспринимается как атрибут класса Car (Автомобиль). Однако на фабрике по производству красок Color (Цвет) — это определенно класс со своими собственными атрибутами (яркостью, насыщенностью, прозрачностью и т. д.).

5. Каждый класс содержит набор операций. Однако на данном этапе мы не касаемся вопросов идентификации операций. Операции, входящие в интерфейс класса (сервисы, предоставляемые классом системе), являются логическим следствием формулировки назначения класса (пункт 1).

Пример выявления классов.

Рассмотрим следующие требования к системе «Запись на университетские курсы» и выделим потенциальные классы.

1. *Для получения каждой университетской степени существует несколько обязательных и несколько выборочных курсов.*
2. *Каждому курсу соответствует заданный уровень и значение условных очков (CreditPoint – условное очко, начисляемое за прослушивание какого-либо курса (за один курс может быть начислено несколько очков), студент обязан на одном году набрать такое число курсов, чтобы число очков за них было не ниже определенного значения).*
3. *Курс может быть составной частью системы получения произвольного количества степеней.*
4. *Каждая степень определяет минимальное общее значение условных очков, требуемое для получения степени (например, для*

степени бакалавра (вычислительные и информационные системы) требуется 68 очков, включая обязательные курсы).

5. *Студенты могут составлять из дисциплин программы обучения, приспособленные к их индивидуальным нуждам и обеспечивающие им получение степени, на которую они претендуют.*

Давайте проанализируем требования с целью выделения потенциальных классов. В первом утверждении подходящими классами являются классы Degree (Степень) и Course (Курс). Эти два класса удовлетворяют пяти правилам, перечисленным ранее.

Мы пока не уверены, должен ли и каким образом класс Course быть сужен до классов CompulsoryCourse (Обязательный курс) и ElectiveCourse (Выборочный курс). Например, ясно, что курс является обязательным или выборочным в зависимости от степени. Возможно, что различие между обязательными и выборочными курсами может быть зафиксировано с помощью ассоциации или даже атрибута класса. Таким образом, CompulsoryCourse и ElectiveCourse рассматриваются как нечеткие классы.

Второе утверждение идентифицирует только атрибуты класса Course, а именно course_level (уровень курса) и credit_point_value (количество условных очков). Третье утверждение характеризует ассоциацию между классами Course и Degree. Четвертая формулировка вводит атрибут min_total_credit_points (минимальное общее количество условных очков) в качестве атрибута класса Degree.

Последнее утверждение позволяет нам выделить три новых класса: Student (Студент), CourseOffering (Предлагаемый курс) и StudyProgram (Программа обучения). Первые два, безусловно, являются релевантными классами, а вот StudyProgram можно превратить в ассоциацию между классами Student и CourseOffering. Поэтому StudyProgram классифицируется как нечеткий класс. Наши рассуждения отражены в табл. 2.

Таблица 2 – Релевантные и нечеткие классы

Релевантные классы	**Нечеткие классы**
Course (Курс)	ComplosoryCourse (Обязательный курс)
Degree (Степень)	ElectiveCourse (Выборочный курс)
Student (Студент)	StudyProgram (Программа обучения)
CourseOffering (Предлагаемый курс)	-

Спецификация классов. После того как перечень потенциальных классов сформирован, необходима их дальнейшая спецификация: классы

требуется включить в диаграмму классов и определить их свойства. Некоторые свойства можно ввести и отобразить внутри графических пиктограмм, представляющих классы на диаграмме классов. Многие другие свойства, включенные в спецификацию класса, имеют только текстовое представление. CASE-средства, как правило, обладают возможностями редактирования, позволяющими легко вводить или модифицировать подобную информацию посредством диалоговых окон, снабженных вкладками, или с помощью аналогичных способов.

Выявление и спецификация атрибутов классов. Графическая пиктограмма, представляющая класс, состоит из трех отделений (имя класса, атрибуты, операции). Спецификация атрибутов классов принадлежит к спецификации состояний и рассматривается в этом разделе. Спецификация операций рассматривается позже.

Выделение атрибутов осуществляется параллельно с выделением классов. Идентификация атрибутов – своего рода «побочный эффект» установления классов. Это не означает, что выявление атрибутов — простая задача. Напротив, это процесс, требующий значительных усилий и многократных итераций.

Исходные модели спецификации определяют только атрибуты, являющиеся существенными для понимания состояний, в которых могут находиться объекты класса. Остальные атрибуты можно до поры до времени игнорировать (однако аналитик должен быть уверен в том, что установленная, но проигнорированная на определенном этапе информация не будет по ошибке утеряна и будет зафиксирована впоследствии). Маловероятно, чтобы все атрибуты класса были приведены в документе описания требований, однако важно не включать в спецификацию те атрибуты, которые не вытекают из требований. В последующих итерациях можно добавить больше атрибутов. Для имен атрибутов мы рекомендуем придерживаться простого соглашения: в именах атрибутов использовать только строчные буквы, а слова в составных именах отделять подчеркиванием.

Пример спецификации классов. Снова обратимся к примеру системы «Запись на университетские курсы». Рассмотрим следующие дополнительные требования, изложенные в документе описания требований.

1. Выбор студентам учебных курсов может быть ограничен из-за конфликтов расписания, а также за счет ограничения на количество студентов, которое может быть набрано на текущий предлагаемый курс.

2. Предлагаемая студентом программа обучения вводится в интерактивную систему записи на курсы. Система проверяет программу на непротиворечивость и сообщает о любых проблемах. Проблемы

требуется решать при помощи научного руководителя. Окончательная программа является предметом научного согласования со стороны представителя заведующего кафедрой, а затем направляется секретарю учебного заведения.

В первом утверждении упоминаются конфликты расписания, однако мы не знаем достоверно, как следует моделировать эту проблему. Возможно, что речь здесь идет о прецеденте, который процедурно определяет конфликты расписания. Вторую часть этой же формулировки можно смоделировать за счет ведения атрибута enrolment_quota (квота набора) в класс CourseOffering. Теперь также ясно, что класс должен обладать атрибутами year (год) и semester (семестр).

Вторая формулировка укрепляет нас во мнении о необходимости введения класса StudyProgram. Можно видеть, что класс StudyProgram сочетает в себе ряд дисциплин, предлагаемых к изучению в текущий момент. Поэтому класс StudyProgram также должен обладать атрибутами year и semester.

Ближайшее рассмотрение нечетких классов CompulsoryCourse и ElectiveCourse приводит нас к выводу, что учебный курс является обязательным или выборочным относительно определенной ученой степени. Один и тот же курс может быть обязательным по отношению к одной степени, выборочным, что касается другой, и вообще не допустимым применительно к некоторым другим степеням. Раз так, то CompulsoryCourse и ElectiveCourse не являются классами в полном смысле слова.

На рис. 12 представлена модель классов, соответствующая проведенным нами рассуждениям. Кроме того, на рисунке используются символы (стереотипы (stereotype)) <<PK>> и <<CK>> для обозначения первичных ключей и потенциальных ключей, соответственно. Это уникальные идентификаторы объектов для рассмотренных классов. Здесь же заданы типы данных для атрибутов.

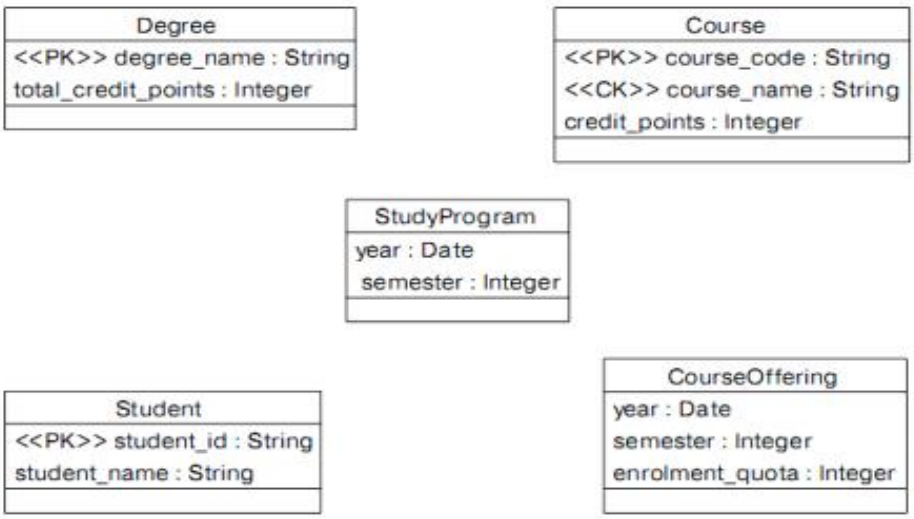

Рисунок 12 – Модель классов

Классы StudyProgram и CourseOffering не имеют пока идентифицирующих атрибутов. Они будут введены в эти классы после установления ассоциативных связей между классами.

Выявление ассоциаций. Нахождение основных ассоциаций представляет собой побочный эффект процесса выявления классов. При определении классов аналитик принимает решение об атрибутах классов, и некоторые из этих атрибутов являются ассоциациями с другими классами. Атрибуты могут относиться к элементарным типам данных либо могут вводиться в качестве других классов, устанавливая таким образом отношения с другими классами.

По существу, любой атрибут, относящийся к неэлементарным типам данных, должен моделироваться как ассоциация (или агрегация) по классу, представляющему этот тип данных.

Выполнение пробного прогона прецедентов позволяет выявить остающиеся ассоциации. Устанавливаются пути взаимодействия между классами, необходимые для прогона прецедентов. Обычно ассоциации должны поддерживать эти пути взаимодействия.

Каждая тернарная ассоциация должна быть заменена циклом или бинарной ассоциацией. Тернарные ассоциации привносят риск неверного семантического истолкования.

Иногда для того чтобы полностью выразить базовую семантику, циклы, образуемые ассоциациями, не должны коммутировать (быть замкнутыми). Это значит, что по меньшей мере одна из ассоциаций в цикле может быть производной (derived). Подобная ассоциация является избыточной в семантическом смысле и должна быть исключена (хорошая семантическая модель должна быть лишена избыточности). Вполне допустимо, что многие производные ассоциации все же войдут в проектную модель (например, из соображений эффективности).

Спецификация ассоциаций. Спецификация ассоциаций подразумевает выполнение следующих действий.

1. Присваивание имен ассоциациям.
2. Присваивание имен ассоциативным ролям.
3. Установление кратности ассоциации.

Правила именования ассоциаций должны соответствовать соглашениям по именованию атрибутов — имена ассоциаций состоят из строчных букв, отдельные слова в имени ассоциации разделяются подчеркиванием.

Если два класса связаны только одним ассоциативным отношением, задавать имя ассоциации и ассоциативные ролевые имена между этими классами необязательно. CASE-средства могут внутренне различать каждую ассоциацию через системные идентификационные имена.

Ролевые имена можно использовать для раскрытия более сложных ассоциаций, в частности самоассоциативных отношений (self associations) (рекурсивных ассоциаций, которые связывают объекты одного и того же класса). При задании ролевых имен их следует выбирать с учетом того, что в проектной модели они станут атрибутами классов, расположенных на противоположных концах ассоциативной связи.

Выявление агрегаций и композиций. Поиск агрегаций ведется параллельно с поиском ассоциаций. Если ассоциация проявляет одно или более из четырех семантических свойств, рассмотренных выше, то ее можно моделировать как агрегацию.

При объяснении отношения агрегации лакмусовой бумажкой выступают фразы «включает» («has») и «является частью» («is_part_of»). При истолковании отношения сверху-вниз по иерархии классов используется фраза «включает» (например, Книга «включает» Главу). При интерпретации снизу-вверх используется фраза «является частью» (например, Глава «является частью» Книги). Если предложение, описывающее отношение, прочитывается вслух с использованием этих фраз и оно лишено смысла на естественном языке, то это отношение не является агрегацией. Со структурной точки зрения агрегация часто связывает воедино большое количество классов, тогда как ассоциация степени выше двух бессмысленна. Когда требуется связать более двух классов воедино, отличным вариантом моделирования может быть агрегация типа Участник.

Спецификация агрегаций и композиций. Язык UML обеспечивает только ограниченную поддержку агрегации. Сильная форма агрегации называется в UML *композицией*. В композиции составной объект может физически содержать компонентные объекты (семантически это отношение берется «по значению»). Компонентный объект может принадлежать только одному составному объекту. Отношение композиции языка UML в большей или меньшей степени соответствует нашим агрегациям типа Безраздельно обладает и Обладает.

Слабая форма агрегации в UML называется просто *агрегацией*. Это отношение семантически берется «по ссылке» — составной объект физически не содержит компонентный объект. Один компонентный объект может обладать несколькими ассоциативными или агрегативными связями в модели. Попросту говоря, агрегация в языке UML соответствует нашим агрегациям типа Включает и Участник.

Сплошной ромб в языке UML представляет композицию. Пустой ромб используется для определения агрегации. В остальном спецификация агрегации совпадает с обозначениями для ассоциации.

Пример спецификации агрегации и композиции. Снова обратимся к системе Записи на университетские курсы. Рассмотрим следующие дополнительные требования.

1. *Академическая характеристика студента должна быть доступна по требованию. Характеристика должна включать информацию об оценках, полученных студентом по каждому из курсов, на которые*

записался студент (и которые он не покинул без взыскания, т. е. первые три недели с начала семестра).

2. *За каждый курс отвечает преподаватель, однако этот курс могут вести и другие преподаватели. В течение разных семестров за один курс могут отвечать разные преподаватели; кроме того, в течение разных семестров курс могут вести разные преподаватели.*

На рис. 13 показана модель классов, в которой выделено отношение агрегации. Понятие «студент» (объект Student) «включает» академическую характеристику (объект AcademicRecord) — это описание композиции в языке UML (с семантикой «по значению»). Каждый объект AcademicRecord физически помещен в один из объектов Student. Несмотря на существование ассоциации takes (брать [курс обучения]) объект AcademicRecord включает атрибут course_code (код курса).

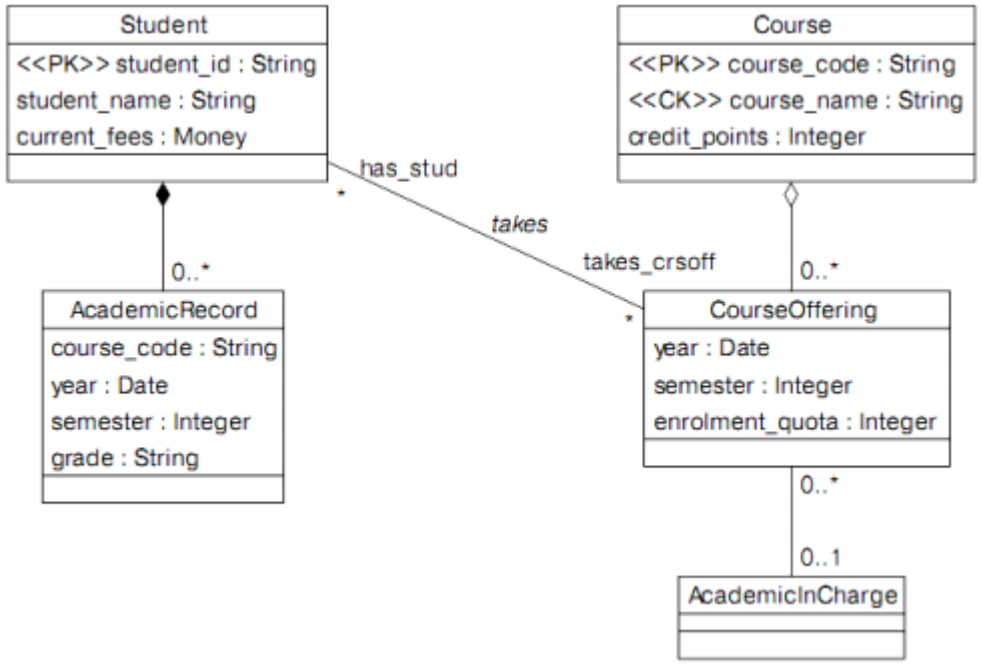

Рисунок 13 - Модель классов с отношениями агрегации и композиции

Это необходимо, поскольку ассоциация takes реализуется с помощью атрибута takes_crsoff («берет» дисциплину) объекта Student, введенного как коллекция (collection), например, Set[CourseOffering].

Атрибут takes_crsoff не зависит от информации во вложенном объекте AcademicRecord, хотя он безусловно связывает объект Student с объектом Course.

Понятие «курс» (объект Course) включает дисциплину (объект CourseOffering)» — это описание агрегации в языке UML (с семантикой «по ссылке»). Каждый объект CourseOffering только логически содержится в одном из объектов Course. Объект CourseOffering может также участвовать в других агрегациях и/или ассоциациях (например, с объектами Student и AcademicInCharge (ответственный преподаватель)).

Выявление обобщений. Многие суперклассы/подклассы аналитик отмечает еще в процессе формирования первоначального перечня классов. Многие другие обобщения можно обнаружить при определении ассоциаций.

Различные ассоциации (даже принадлежащие одному и тому же классу) может потребоваться связать с классом на различных уровнях обобщения/специализации. Например, класс Course может быть связан с классом Student (Student takes Course — студент берет курс), кроме того, этот класс может быть связан с классом TeachingAssistant (TeachingAssistant teaches Course — ассистент ведет курс). Дальнейший анализ может показать, что класс TeachingAssistant является подклассом Student. При поиске отношения обобщения лакмусовой бумажкой выступают фразы «может быть» («can_be») и «это нечто вроде» («is_a_kind_of»).

При истолковании отношения сверху-вниз по иерархии классов используется фраза «может быть» (например, Student «can_be» a TeachingAssistant — «студент «может быть» ассистентом»). При интерпретации отношения снизу-вверх используется фраза «это нечто вроде» (например, TeachingAssistant «is_a_kind_of» Student — «ассистент «это нечто вроде» студента»). Обратите внимание, что если также справедливо утверждение о том, что «ассистент — это «TeachingAssistant «is_a_kind_of» Teacher», то мы установили множественное наследование.

***Спецификация обобщений*.** Отношение обобщения между классами показывает, что один класс совместно использует структуру или поведение, определенные в одном или более классов. Обобщение представляется в языке UML сплошной линией со стреловидным наконечником, указывающим на суперкласс.

Полная спецификация обобщения включает несколько мощных возможностей.

Например, более подробное определение отношения может содержать квалификацию доступа к нему, указания на то, предоставляет ли класс права другому классу, решая, что необходимо делать в случае множественного наследования и т. д.

Практическая работа 3 «Спецификация поведения информационной системы»

Задание

Необходимо построить UML-диаграмму прецедентов.

Поведение системы — так как она выглядит для внешнего пользователя — изображается в вид прецедентов. Модели прецедентов можно разрабатывать на различных уровнях абстракции. Их можно применить к системе в целом для того, чтобы специфицировать основные функциональные блоки разрабатываемого приложения. Их также можно использовать для фиксации поведения пакетов UML, частей пакетов или даже класса внутри пакета.

На этапе анализа прецеденты вбирают в себя системные требования, концентрируясь на том, что делает или должна делать система.

На этапе проектирования представление проектных решений в виде прецедентов можно использовать для спецификации поведения системы в том виде, как оно должно быть реализовано.

Поведение системы, закрепленное с помощью прецедентов, требует осуществить соответствующие вычисления и обеспечить взаимодействие объектов для выполнения этих прецедентов. Вычисления можно смоделировать с помощью диаграмм видов деятельности. Взаимодействие объектов можно задать с помощью диаграмм последовательностей или диаграмм кооперации.

Спецификация поведения позволяет взглянуть на систему с точки зрения ее функционирования. Здесь основная задача состоит в том, чтобы определить прецеденты для области приложений и установить, какие классы участвуют в выполнении этих прецедентов. При этом необходимо идентифицировать операции классов и сообщения, передаваемые между объектами. Хотя взаимодействие объектов инициирует изменения состояния объектов, спецификации поведения дают функциональный взгляд на застывшее состояние системы. Изменения в состоянии объектов явным образом находят выражение в спецификациях изменения состояний.

Модели прецедентов должны нарабатываться итеративно и параллельно с моделями классов. Классы, введенные в спецификации состояний, подлежат дальнейшей детальной проработке, при этом обозначаются наиболее важные операции. Следует, однако, напомнить, что спецификация состояний определяет только классы сущностей («бизнес-объектов»).

По мере создания моделей поведения появляются еще два уровня классов.

1. Классы, которые обслуживают события, инициируемые пользователями, и представляют бизнес-процессы *(управляющие классы)*.
2. Классы, представляющие GUI-интерфейсы *(пограничные классы)*.

Выявление прецедентов. При выявлении прецедентов аналитик должен убедиться в том, что он твердо придерживается сущности концепции прецедентов. Прецеденты представляют следующие компоненты общей модели системы.

1. *Завершенный* фрагмент функциональных возможностей (включая *основной поток* логики управления, его любые вариации (*подпотоки*) и исключительные условия (*альтернативные потоки*)).
2. Фрагмент внешне *наблюдаемых функций* (отличных от внутренних функций).
3. *Ортогональный* фрагмент функциональных возможностей (прецеденты могут при выполнении совместно использовать объекты, но выполнение каждого прецедента независимо от других прецедентов).
4. Фрагмент функциональных возможностей, *инициируемый субъектом* (будучи инициирован, прецедент может взаимодействовать с другими субъектами).
5. При этом возможно, что субъект окажется только на принимающем конце прецедента (может быть, опосредовано), инициированного другим субъектом. Фрагмент функциональных возможностей, который предоставляет *субъекту* ощутимый *полезный результат* (и этот полезный результат достигается в пределах одного прецедента).

Выявление прецедентов основано на анализе следующих источников информации.

1. Требования, определенные в документе описания требований.
2. Субъекты и их цели применительно к системе.

Напомним, что требования представляются в отпечатанном виде. При поиске прецедентов нас интересуют только функциональные требования.

Прецеденты можно определить на основе анализа задач, выполняемых субъектами.

Ответы на эти вопросы могут позволить обозначить прецеденты.

- Каковы основные задачи, выполняемые каждым субъектом?
- Должен ли субъект получать доступ к информации в системе или модифицировать информацию?
- Должен ли субъект информировать систему о любых изменениях в других системах?
- Должен ли субъект быть проинформирован о непредвиденных изменениях в системе?

В ходе анализа прецеденты обращаются к личностным потребностям субъектов. В некотором роде это прецеденты субъектов. Поскольку прецеденты определяют основные строительные блоки для системы, то существует необходимость в выделении системных прецедентов. Системные прецеденты извлекают все то общее, что присутствует в прецедентах субъектов, и дают возможность разработать общее решение, применимое (через механизм наследования) к области прецедентов субъектов. «Субъектом» системного прецедента является разработчик/программист, а не пользователь. Системные прецеденты идентифицируются на этапе проектирования.

Спецификация прецедентов. Спецификация прецедентов включает графическое представление субъектов, прецедентов и четырех типов отношений, перечисленных ниже.

1. Ассоциация.
2. Включение.
3. Расширение.
4. Обобщение прецедента.

Отношение ассоциации устанавливает каналы связи между субъектом и прецедентом. В качестве стереотипов для отношений «включает» (include) и «расширяет» (extend) используются слова <<include>> и <<extend>>. Отношение обобщения позволяет специализировать прецедент посредством изменения любого из аспектов базового прецедента.

Отношение включения <<include>> позволяет вынести общее поведение за пределы включаемого прецедента. (Это отношение пришло на смену широко применяемой в более ранних версиях UML концепции отношения <<uses>> (использует)). Отношение <<extend>> обеспечивает контролируемую форму расширения поведения прецедента с помощью активизации другого прецедента в определенных точках расширения. Отношение <<include>> отличается от отношения <<extend>> в том, что «включаемый» прецедент является необходимым для завершения «активизирующего» прецедента.

На практике проект может легко столкнуться с проблемами, если прилагать слишком большие усилия для выявления отношений между прецедентами и установления, какие отношения применимы к определенной паре прецедентов. Кроме того, обобщенные прецеденты имеют тенденцию к таким тесным связям, что взаимосвязи станут превалировать и загромождать диаграмму, смещая акценты с надлежащей идентификации прецедентов в сторону отношений между прецедентами.

Пример спецификации прецедентов. Обращаясь к постановке задачи для системы «Запись на университетские курсы», а также к

требованиям, установим прецеденты на основе анализа функциональных требований.

На рис. 14 показана обобщенная диаграмма прецедентов для приложения «Запись на университетские курсы». Модель содержит четыре субъекта и четыре прецедента. Каждый прецедент инициируется субъектом и является завершенным, внешне видимым и ортогональным фрагментом функциональных возможностей. Все субъекты, за исключением субъекта Student, представляют собой инициирующих субъектов. Субъект Student получает результаты экзаменов и инструкции по записи на учебные курсы перед тем, как программа обучения в следующем семестре (учебном периоде) может быть введена и проверена.

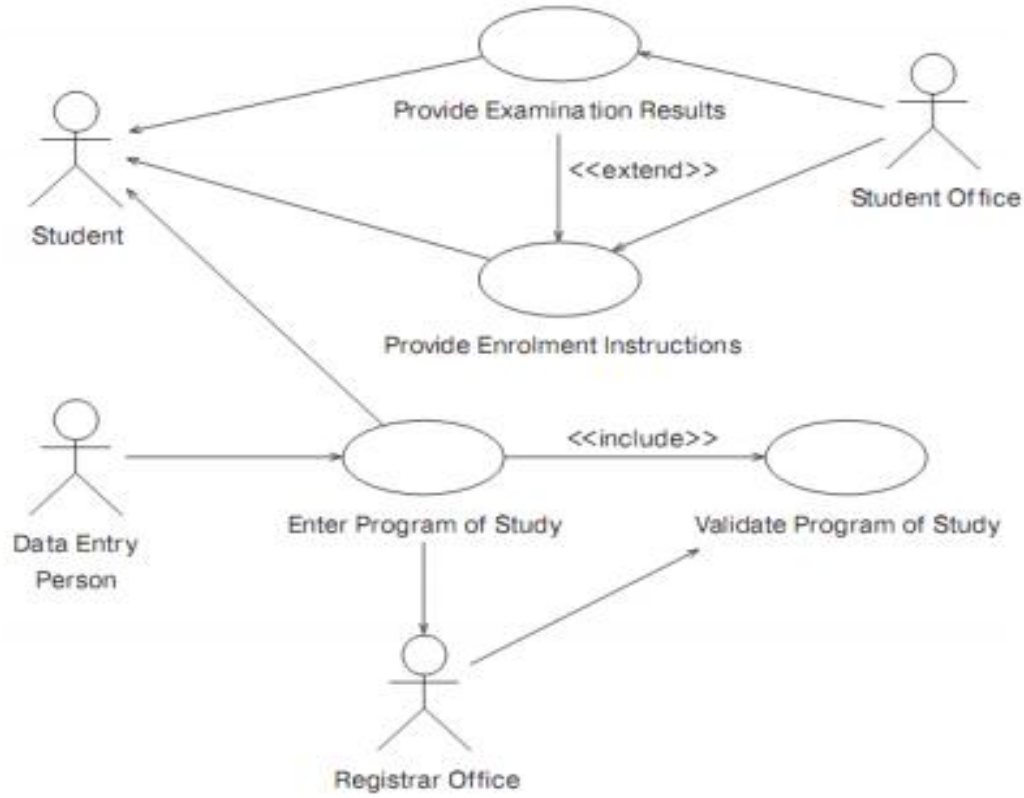

Рисунок 14 - Обобщенная диаграмма прецедентов для приложения «Запись на университетские курсы»

Прецедент Provide Examination Results (Предоставить результаты экзаменов) может «расширить» (<<extend>>) прецедент Provide Enrolment Instructions (Предоставить инструкции по записи). Первый прецедент не всегда расширяет последний прецедент. Например, для новых студентов результаты экзаменов неизвестны. Вот почему отношение моделируется с использованием стереотипа расширения (<<extend>>), а не включения (<<include>>).

Отношение <<include>> было установлено от прецедента Enter Program of Study (Ввести программу обучения) к прецеденту Validate Program of Study (Проверить программу обучения). Отношение <<include>> означает, что первый из прецедентов всегда включает последний. Как только программа изучения введена, она проверяется на предмет конфликтов расписания, специальных согласований и т.д.

Практическая работа 4 «Спецификация видов деятельности»

Задание
Необходимо построить UML-диаграмму последовательности.

Спецификация видов деятельности. Каждый прецедент можно моделировать с помощью одного или нескольких графов видов деятельности. Событие, источником которого служит субъект, инициирующий прецедент, это то же самое событие, которое запускает выполнение графа видов деятельности. Процесс выполнения последовательно переходит от одного состояния вида деятельности к другому. Состояние вида деятельности считается завершенным, когда завершается его вычисление. Внешние инициируемые событиями прерывания, которые могут вызвать завершение состояния вида деятельности, допускаются только в исключительных случаях. Если ожидается, что подобные события могут происходить часто, то следует вместо этого воспользоваться диаграммой состояний.

Виды деятельности лучше всего выявлять на основе анализа предложений неформальной спецификации прецедентов. Каждая фраза, содержащая глагол, может рассматриваться как потенциальный вид деятельности. Описание альтернативных потоков вводит в граф видов деятельности ветвление и разделение потоков.

Они приводят к исключительным (непредвиденным) состояниям деятельности. Возможны также параллельные потоки управления.

После выявления состояний видов деятельности спецификация видов деятельности выглядит как довольно простой процесс соединения этих состояний линиями переходов.

Параллельные потоки инициируются (разделяются) и сливаются, что отображается на диаграмме в виде жирной полосы, обозначающей синхронизацию потоков. Альтернативные потоки создаются (разветвляются) и объединяются, что отображается в виде ромбов ветвления.

Внешние события на графе видов деятельности обычно отсутствуют.

Однако существует графический метод включения внешних событий в граф. Аналогично существуют графические обозначения для состояний потоков объектов для представления объектов, которые являются входными или выходными для вида деятельности.

Выявление последовательностей сообщений. Выявление последовательностей сообщений является логическим продолжением моделирования видов деятельности. Если уровни абстракции, используемые для построения модели видов деятельности и модели последовательностей, совпадают, то осуществить отображение видов деятельности на сообщения довольно просто.

Спецификация последовательностей сообщений. При спецификации сообщений полезно проводить различие между сообщением, представляющим собой сигнал, и сообщением, которое являет собой вызов операции. Сигнал означает асинхронное взаимодействие объектов. Отправитель может продолжить выполнение потока сразу после отправки сигнального сообщения. Вызов означает синхронное обращение к операции с условием возврата управления отправителю. Возвращаемое сообщение может возвращать вызывающему объекту некоторые значения либо просто уведомлять его об успешном завершении операции. В последнем случае возвращаемое сообщение отображать на диаграмме последовательностей не обязательно.

Пример спецификации последовательностей. Приняв во внимание требования, а также обращаясь к примеру спецификации прецедентов, построим диаграмму последовательностей для прецедента «Enter Program of Study».

На рис. 15 показана диаграмма последовательностей для приведенного выше сценария. Субъект Data Entry Person (Лицо, вводящее данные) инициирует прецедент, отправляя сообщение с запросом объекту граничного класса ProgramEntryWindow (Окно программы ввода) для того, чтобы добавить студента (обозначенного аргументом std) к списку записавшихся на курс (аргумент crs) в данном семестре (аргумент sem).

Объект класса ProgramEntryWindow проверяет с помощью объекта aStudent, имеет ли право студент записаться (например, внес ли студент s_check соответствующую плату). Выходной аргумент возвращает значение «yes» или «no» объекту: ProgramEntryWindow. Если возвращается значение «no», запись не может продолжаться, и объект :ProgramEntryWindow отправляет автосообщение самому себе, чтобы уничтожить (destroy) себя (операция деструктора). Запись условия в квадратных скобках говорит о том, что операция деструктора является необязательной.

Использование автосообщения для указания на необходимость уничтожения объекта является просто сокращенной записью. В действительности объект_экземпляр не может уничтожить себя. Сообщение об уничтожении (destroy) должно быть отправлено объекту_классу. Кроме того, в языке UML имеется специальное обозначение для операции уничтожения объекта — большой знак X, помещенный на жизненной линии объекта в точке, в которой объект должен быть уничтожен.

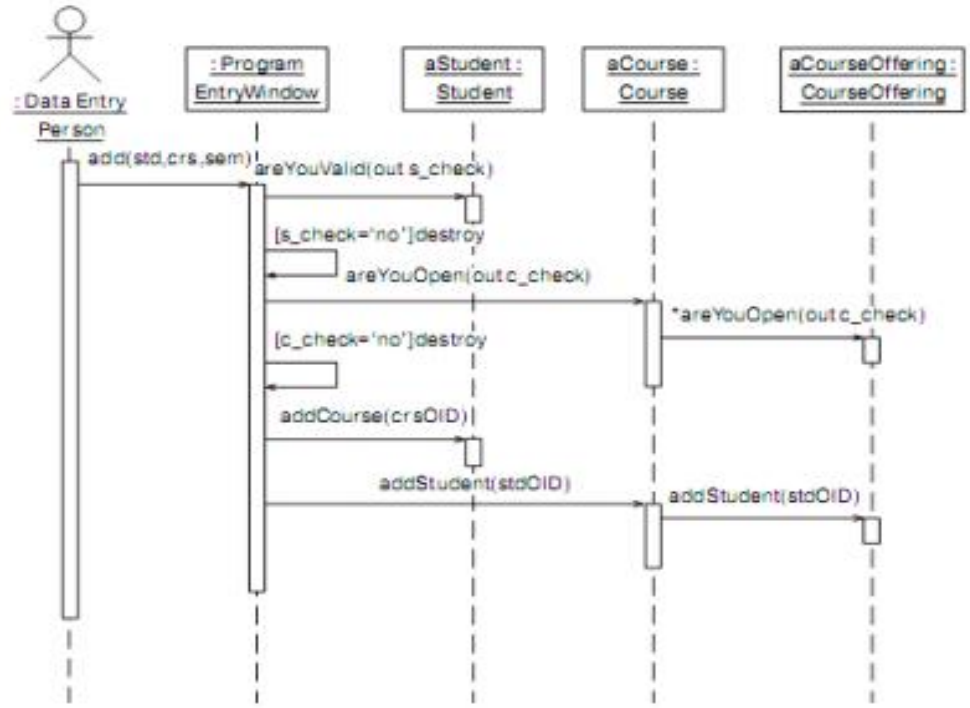

Рисунок 15 - Диаграмма последовательностей

Если студент aStudent может быть записан, нам требуется выяснить, открыта ли запись на курс aCourse. Для обслуживания этого требования объект aCourse должен запросить свой текущий объект aCourseOffering (на который указывает агрегативная связь от Course к CourseOffering (рис. 6)). Если в текущем объекте aCourseOffering свободных мест уже нет, запись не может продолжаться, и объект :ProgramEntryWindow снова отправляет себе автосообщение, чтобы разрушить себя.

Если процесс записи может продолжаться, объект :ProgramEntryWindow требует от объекта aStudent добавить себе объект aCourseOffering, а затем требует, чтобы объект aCourseOffering добавил объект aStudent к себе. Последовательность двух операций объекта :ProgramEntryWindow может быть изменена на обратную, если программа гарантирует ссылочную целостность между aCourseOffering и aStudent (т.е. если студент записался на предлагаемый курс, то в список студентов по данному курсу должен быть внесен данный студент).

Заметим, что если для хранения объектов Student и CourseOffering используется ПО СУБД, то объект :ProgramEntryWindow мог бы отправлять только одно из этих двух сообщений, т. е. отправить либо сообщение addCourse, либо сообщение addStudent.

После того как объект aStudent добавлен к объекту aCourseOffering, ПО СУБД берет на себя ответственность за поддержание ссылочной целостности и должно зафиксировать существование связи от aStudent к aCourseOffering, и наоборот.

Заметим также, что фактические аргументы, включенные в сообщения addCourse и addStudent, представляют собой идентификаторы объектов (OID), содержащие значения OID (дескрипторы), указывающие на объект aStudent и aCourseOfferin, соответственно. Эти значения OID были определены, когда объект: ProgramEntryWindow получил доступ к объектам aStudent и aCourse с помощью сообщений areYouValid и areYouOpen, представляющих собой запросы на возможность записи и наличие мест на курсе.

Заключение

Основной задачей при создании ИТ-архитектуры является отражение взаимосвязи бизнеса и ИТ, с одной стороны, через документирование, совершенствование и стандартизацию бизнес-процессов, а с другой – через описание элементов ИТ- архитектуры на логическом уровне, во взаимосвязи с бизнес- процессами. При достижении прозрачности и взаимосвязи архитектуры бизнес- процессов, данных, приложений и технологий можно говорить о создании базы для построения общекорпоративной системы управления изменениями и типизации требований к изменениям информационных систем.

При использовании системного подхода к документированию и управлению ИТ-архитектурой компания получает следующие преимущества:

− снижение общей стоимости владения ИТ (TCO) в стратегической перспективе;

− сокращение избыточности функционала существующих информационных систем;

− прозрачность существующего “зоопарка” систем;

− решение проблемы “лоскутной” автоматизации;

− возможность унификации информационных систем и элементов ИТ-архитектуры через стандартизацию в области ИТ и внедрение корпоративных стандартов;

− возможность идентификации критичных элементов ИТ-архитектуры на основе их взаимосвязи с критичными бизнес-процессами;

− возможность анализа взаимовлияния элементов ИТ- архитектуры между собой, а также с бизнес-процессами.

Имея картину существующего положения и разработав модель целевой ИТ-архитектуры, можно создать программу унификации и стандартизации, а также развития информационных технологий в компании.

В тоже время четкая формализация бизнес-требований, происходящая во время описания как бизнес-, так и ИТ-архитектуры, позволяет создать прозрачный ИТ-бюджет, поддержанный бизнес-партнерами и государственным заказчиком.

Выполнив практические работы, данные в учебно-методическом пособии, студент на практике получит опыт применения в проектировании подходов и методов, которые позволят получать успешные архитектуры информационных систем. Справедливости ради, стоит заметить, что изложенные подходы и методы не единственные. Самостоятельно осваивая другие методики и применяя их совместно с ранее изученными, студент сможет построить еще более гибкие и эффективные архитектуры.

Рекомендуемая литература

1. Данилин А.В. Электронные государственные услуги и административные регламенты. От политической задачи к архитектуре "электронного правительства". М.: Инфра-М, 2004.
2. Данилин А., Слюсаренко А. Архитектура и стратегия. «Инь» и «Янь» информационных технологий предприятия. М. Интернет-Университет Информационных технологий, 2005.
3. Забегалин Е.В. Архитектура информационных систем в теории и практике / IBS, Департамент управленческого консалтинга: http://www.evz.name/evzms-2.pdf
4. Забегалин Е.В. Технология моделирования архитектуры автоматизированных информационных систем: Сборник методических рекомендаций по определению и моделированию архитектуры автоматизированных информационных систем в консалтинговых проектах. Версия 1.0 /Декабрь 2006г. / IBS, Департамент управленческого консалтинга: http://www.evz.name/evzms-1.pdf
5. Зиндер Е.З. Архитектура предприятия в контексте бизнес-реинжиниринга. Часть 1 // Intelligent Enterprise. 2008.№ 4. - www.iemag.ru/articles/detail.php?ID=6612
6. Зиндер Е.З. Архитектура предприятия в контексте бизнес-реинжиниринга. Часть 2 // Intelligent Enterprise. 2008.№ 7. – http://www.iemag.ru/analИТics/detail.php?ID=18024
7. Дрожжинов В.И., Зиндер Е.З. Электронное правительство: рекомендации по внедрению в Российской Федерации. М.: Эко-Трендз, 2004.
8. Дрожжинов В.И., Штрик А. Стандартизация архитектуры государственных ведомств США // PC Week / RE. 2005. №28, 31.
9. Кристальный Б.В., Травкин Ю.В Электронное правительство. Опыт
США. М.:Эко-Трендз, 2003.
10. Лодон Дж., Лодон К. Управление информационными системами, /Перевод под ред. Д.Р. Трутнева. - СПб.: Питер, 2005.
11. Разработка стратегии развития и проектирование ИТ-архитектуры
больших и средних систем управления. Электронный ресурс: http://www.mrcb.ru/?mode=reply&pageId=3046&replyTo=28454
12. Сапегин С.В. Проектирование архитектуры информационных систем на основе SOA // Вестник Воронежского государственного университета. Серия: Системный анализ и информационные технологии. 2010. № 1.С. 80 - 84.

13. Трутнев Д.Р., Годин В.В. Управление информационными системами. -М.: ГУУНФПК, 1999.

14. Zachman A.A framework for Information Systems Architecture // IBM Systems Journal. 1987. Vol. 26.№ 3

Стандарты и другие документы

1. ГОСТ 34.601. Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания.

2. ГОСТ Р ИСО 15704-2008.Промышленные автоматизированные системы. Требования к стандартным архитектурам и методологиям предприятия. Электронный ресурс: http://protect.gost.ru/document.aspx?control=7&id=175381

3. ГОСТ 34.003-90 Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Термины и определения.

4. ГОСТ 34.602-89 Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы.

5. ANSI/IEEE Std 1471-2000, Recommended Practice for Architectural Description of Software-Intensive Systems; Control Objectives for Information and related Technology (COBИТ 4.0). ИТ Governance Institute (ITGI), www.ITgi.org, 2005.

6. ISO 15704:2000. Industrial automation systems — Requirements for enterprise-reference architectures and methodologies.

7. ISO 35.100. «Open systems interconnection».

8. ISO-15704, Industrial automation systems – Requirements for enterprise reference architectures and methodologies. August 20, 1999.

9. FEA Consolidated Reference Model Document, May 2005; IEEE Recommended Practice for Architectural Description, Draft 3.0 of IEEE P1471, May 1998.

10. OMG Unified Modeling Language (UML) Specification, March 2003 Version 1.5.

11. Глоссарий Фонда поддержки системного проектирования, стандартизации и управления проектами (ФОСТАС - http://www.fostas.ru).

Рекомендуемая литература для выполнения практических работ

1. Анализ требований и проектирование систем. Разработка информационных систем с использованием UML.: Пер. с англ. – М. :Издательский дом «Вильямс»,2002. – 432 с.

2. Инженерия программного обеспечения, –6-е изд. – : Пер. с англ. – М.: Издательский дом «Вильямс»,2002. – 624 с.

3. Принципы работы с требованиями к программному обеспечению. Унифицированный подход.: Пер. с англ. – М. : Издательский дом «Вильямс», 2002. – 448 с.

4. UML. Основы: Пер. с англ. – СПб.: Символ-Плюс, 2002. – 192 с.

5. ГОСТ 34.602-89 Техническое задание на создание автоматизированной системы. – М. , 1990. URL: http://www.nist.ru/hr/doc/gost/34-602-89.htm

6. IEEE Std 830-1998 IEEE Recommended Practice for Software

Глоссарий

Адаптация – процесс приспособления организации к существующим
или изменяющимся условиям.

Адаптивные изменения – спонтанные стратегические изменения, обусловленные рядом последовательных мер, принятых в течение длительного периода, оказывающие воздействие на традиционные критерии, структуру власти и компетентность менеджеров, которые возникают как реакция на постоянные воздействия из вне или на неудовлетворительные производственно-хозяйственные показатели деятельности организации.

Анализ SWOT (сила, слабость, возможности, угрозы) - анализ сильных и слабых сторон фирмы, оценка ее возможностей и потенциальных угроз.

Архитектура предприятия (Enterprise Architecture - EA, согласно стандарту ANSI/IEEE Std 1471-2000) – фундаментальная организация системы, реализованная в её компонентах, их взаимоотношениях друг с другом и средой и принципах, определяющих её конструкцию и развитие».

Архитектура приложений (Applications Architecture) - компонент технической архитектуры предприятия, который определяет основные приложения, необходимые для управления данными и для поддержки бизнес-функций.

Архитектурная политика предприятия (Enterprise Architecture Policy) – руководящие принципы разработки, реализации и поддержки архитектуры предприятия.

Архитектурный артефакт (Architectural Artifact) - релевантная документация, модели, диаграммы, описания и результаты анализа, а также базовый репозиторий, стандарты и профили защиты.

Базовые параметры – система критериев, способных адекватно отразить специфику конкретного объекта с учетом влияющих на него в тот или иной период времени факторов (система показателей, качественных характеристик, шкал и т.д.).

Бизнес-архитектура (Business Architecture, Деловая архитектура) – компонент текущей и целевой архитектуры, относящийся к основной федеральной задаче (миссии) данного предприятия и соответствующий его целям. Бизнес-архитектура включает содержание бизнес-моделей и концентрируется на федеральных областях бизнеса и бизнес-процессах, соответствующих бизнес-стимулам. Бизнес-архитектура определяет федеральные бизнес- процессы, федеральные информационные потоки, а также информацию, необходимую для осуществления бизнес-функций предприятия.

Видение целевой архитектуры (Target Architecture Vision) - краткое стратегическое описание конечного состояния целевой архитектуры через пять лет. Видение является основой для формирования стратегического направления и используется для того, чтобы принимать решения по ресурсам, уменьшать затраты, а также повышать эффективность деятельности.

Внешняя среда - факторы, условия, силы и субъекты, влияющие на ситуацию в организации, отрасли, стране из вне.

Внутренняя гибкость – обеспечение такой внутриорганизационной координации, при которой мощности, материальные, профессиональные и управленческие ресурсы организации могут быть быстро и легко переведены из одной бизнес-единицы в другую.

Глобальная стратегия – одинакова для всех стран, хотя и существуют небольшие отличия в стратегиях в каждой сфере, вызванные необходимостью приспособления к специфическим условиям, но основной конкурентный подход (например, низкие затраты, дифференциация или фокусировка на отдельных направлениях развития) остается неизменным.

Жизненный цикл предприятия (Enterprise Life Cycle) - динамический, итерационный процесс изменения предприятия в течение определенного времени, выражающийся во включении новых бизнес-процессов, новых технологий, новых возможностей, а также новых принципов технического обслуживания и передислокации существующих элементов предприятия. В большинстве случаев жизненный цикл предприятия охватывает три главных аспекта: жизненный цикл систем, человеческие ресурсы и управление безопасностью. В пределах этих трех внешних аспектов рассматривается динамическое развитие следующих элементов, определяющих деятельность предприятия:

− производственная деятельность предприятия и программы управления;

− процессы формирования архитектуры предприятия;

− процессы перспективного планирование капиталовложений и инвестиционного управления.

Жизненный цикл разработки системы (System Development Life Cycle - SDLC)

1) Руководство, стратегии и процедуры, предназначенные для разработки системы на протяжении всего ее жизненного цикла, включая определение требований, проектирование, реализацию, проведение испытаний, развертывание, функционирование и техническое обслуживание.

2) Сфера деятельности, связанная с некоторой системой, охватывающая инициацию проекта системы, ее разработку и комплектование, реализацию, функционирование и техническое обслуживание и, в конечном счете, избавление от нее, вызванное инициацией проекта другой системы.

Жизненный цикл системы (Life Cycle) [ISO-15704] - конечное множество фаз и шагов определенных видов, которые система может проходить на протяжении полной истории своей жизни.

Защита информации (Information Security) [ГОСТ-12207] - сохранение информации и данных так, чтобы не допущенные к ним лица или системы не могли их читать или изменять, а допущенные лица или системы не ограничивались в доступе к ним.

Защита информационной инфраструктуры (Information Infrastructure Protection) – функция организации, направленная на обеспечение: а) гарантий эффективного и безопасного функционирования систем, с адекватным уровнем конфиденциальности, целостности и доступности, б) информационной безопасности с адекватным уровнем риска и затрат. Защите подлежат такие элементы информационной инфраструктуры, как информация, конкретные системы (главные приложения, обеспечивающие системы, жизненно-важные системы), группы систем для поддержки конкретных операционных программ или сами такие программы (например, контроль авиационного трафика, медицинское страхование и др.).

Инновационный потенциал – возможности в достижении поставленных инновационных целей.

Инновация - создание, распространение и применение какого-либо новшества, ведущие к улучшению работы, повышению эффективности деятельности.

Информационно-коммуникационные технологии (ИКТ) – совокупность методов, производственных процессов и программно-технических средств, интегрированных с целью сбора, обработки, хранения, распространения, отображения и использования информации в интересах ее пользователей.

Информация – сведения об окружающем мире (объектах, явлениях, событиях, процессах, закономерностях…), которые уменьшают имеющуюся степень неопределенности, неполноты знаний, отчужденные от их создателя и ставшие сообщениями (выраженными на определенном языке в виде знаков, в том числе и записанными на материальном носителе), которые можно воспроизводить путем передачи устным, письменным или другим способом (с помощью условных сигналов, технических средств, и т. д.).

Информационная система должна рассматриваться как среда, обеспечивающая целенаправленную деятельность органов государственной власти. Т. е. она представляет собой совокупность таких компонентов как информация, регламенты, персонал, аппаратное и программное обеспечение, объединенных регулируемыми взаимоотношениями для формирования организации как единого целого и обеспечения её целенаправленной деятельности.

Миссия (стратегические установки, предназначение) - основная общая цель организации, четко выраженная причина ее существования, ее предназначение. Формулируется, прежде всего, с точки зрения повышения социальной роли организации. Концепция миссии – надежный элемент идеологической базы формирования организации.

Организационная культура организации – совокупность ценностей,
норм, правил, обычаев, традиций, ориентиров, разделяемых ее сотрудниками.

Прогнозирование – процесс научного предвидения, определение тенденций развития и образа будущего.

Риск – ситуативная характеристика деятельности, означающая неопределенность ее исхода, возможные неблагоприятные ее последствия, альтернативные варианты ошибки или успеха.

Риск управленческий – характеристика управленческой деятельности, осуществляемой в ситуации той или иной степени неопределенности, например, вследствие недостаточности или ненадежности информации, при выборе менеджером альтернативного решения, критерий эффективности которого связан с вероятностью проявления негативных условий реализации потерь или с вероятностью нейтрализации факторов неопределенности и увеличением прибыли.

Среда косвенного воздействия – факторы внешней среды, которые не оказывают непосредственного воздействия на поведение организации, но косвенно влияют на процесс формирования стратегии; к ней относятся экономические, политические, технологические и социальные факторы.

Среда прямого воздействия – совокупность факторов среды, непосредственно влияющих на поведение организации: поставщики, потребители, конкуренты, трудовые ресурсы, воздействие учреждений государственной и муниципальной власти.

Стратегический менеджмент (управление) - управленческая деятельность, связанная с постановкой долгосрочных целей и задач организации и с поддержанием ряда взаимоотношений между организацией и окружением, которые позволяют ей добиться своих целей, соответствуют ее внутренним возможностям и позволяют оставаться восприимчивой к внешним требованиям. С ростом уровня нестабильности условий деятельности организаций возрастает их потребность в ориентации на стратегическое управление. Способности к стратегическому менеджменту предполагают наличие пяти элементов: 1) умение смоделировать ситуацию; 2) умение выявить необходимость изменений; 3) умение разработать стратегию изменений; 4) умение использовать в ходе изменений надежные методы; 5) умение воплотить стратегию в жизнь.

Стратегия - (др. греч. στρατηγία — «искусство полководца») — общий, не детализированный план какой-либо деятельности,

охватывающий длительный период времени, способ достижения сложной цели. Обобщающая модель действий, необходимых для достижения поставленных долгосрочных целей путем координации и распределения ресурсов компании. По существу, стратегия есть набор правил для принятия решений, которыми организация руководствуется в своей деятельности. Процесс разработки стратегии включает: 1) определение миссии организации; 2) конкретизацию видения организации и постановку целей; 3) формулировку и реализацию стратегии, направленной на достижение целей.

Управление информационными системами:

− «Применение методов управления процессами планирования, анализа, дизайна, создания, внедрения и эксплуатации информационной системы организации для достижения ее целей» (ГОСТ РВ 51987-2002);

− «Структура взаимоотношений и процессов выбора вектора развития
предприятия и его управления, направленных на увеличение его стоимости при сбалансированном риске в сфере информационных и смежных технологий» (Cobit);

− «Определение прав на принятие решений и границ ответственности
для стимулирования желаемого поведения при использовании ИТ» (ПитерУэйл).

Электронное правительство — использование в практике органов государственной власти современных информационно-коммуникационных
технологий (ИКТ) для осуществления всего спектра правительственных функций, нацеленное на обеспечение доступа граждан к достоверной официальной информации, на создание новых возможностей для взаимодействия органов власти между собой, с населением, бизнесом и институтами гражданского общества, а также на повышение эффективности и прозрачности государственного управления.

Приложение

Варианты заданий

1. АСУ деятельностью отдела кадров предприятия
2. АСУ складского хранения
3. АСУ деятельностью библиотеки
4. Веб-магазин по продаже часов
5. Веб-магазин по продаже фотоаппаратов
6. АСУ деятельностью аптечной сети
7. Веб-сайт букмекерской конторы
8. ИС учета успеваемости студентов
9. Веб-магазин по продаже компьютерных комплектующих
10. Программный RSS-агрегатор
11. Веб RSS-агрегатор
12. ИС «Ежедневник»
13. АСУ деятельностью магазина видеопроката
14. АСУ деятельностью автосалона
15. Веб-магазин по продаже одежды
16. ИС «Почтовый коллектор»
17. АСУ деятельностью магазина бензозаправки
18. АСУ учетом пациентов в поликлинике
19. АСУ учетом коммунальных платежей
20. АСУ деятельностью службы такси
21. ИС сбора и обработки ошибок (багтрекер)
22. Веб-сайт кафедры
23. Веб-сайт факультета
24. ИС хранения и каталогизации фотографий
25. ИС «Каталог недвижимости»

Темы докладов

1. Общая характеристика и классификация информационных систем.
2. Формальные методы описания структуры системы.
3. Понятие архитектуры информационной системы.
4. Модели функционирования информационных систем.
5. Технологии разработки информационных систем.
6. Особенности реализации информационных систем в различных предметных областях
7. Модель распределенной обработки информации.
8. Программные и технические средства распределенных информационных систем.
9. Архитектура открытых систем.
10. Основные понятия архитектуры информационных сетей.
11. Модели и структуры информационных систем.
12. Компоненты информационных систем.
13. Архитектура информационных систем в научных исследованиях.
14. Эталонные аппаратные платформы.
15. Типовые архитектурно-структурные решения, используемые при создании информационных систем.
16. Программное обеспечение информационных систем.
17. Модели и проблемы человеко-машинного взаимодействия в информационных системах; правовые, экономические, социальные и психологические аспекты информационных систем
18. Методы оценки эффективности информационных систем.
19. Тенденции и перспективы развития информационных систем

Надежда Владимировна Бендик
Софья Андреевна Петрова

Системная архитектура информационных систем

Учебно-методическое пособие

Лицензия на издательскую деятельность
ЛР № 070444 от 11.03.98 г.
Подписано в печать 20.03.2016 г.
Тираж 30 экз.

Издательство Иркутского государственного аграрного
университета им. А.А. Ежевского
664038, Иркутская обл., Иркутский р-н,

пос. Молодежный