УДК 681.3

ПРОЕКТИРОВАНИЕ АРХИТЕКТУРЫ ИНФОРМАЦИОННЫХ СИСТЕМ НА ОСНОВЕ SOA

С. В. Сапегин

Воронежский государственный университет

Поступила в редакцию 05.04.2010 г.

Аннотация. Статья посвящена принципам построения информационных систем на основе методологии SOA. Исследуется схема абстракции концепции SOA. Рассматривается принцип оценки экономического эффекта от использования SOA.
Ключевые слова: Service-Oriented Architecture (SOA), корпоративные ИС, разработка архитектуры ИС

Abstract. In article the principles of CIS design, based on SOA methodology are considered. The abstract scheme of SOA concept is described. The principles of SOA using economic effect estimating are offered.
Key words. Service-Oriented Architecture (SOA), corporative IS, software architecture design.

ВВЕДЕНИЕ

Корпоративные системы, как средства управления информацией и достижения организационной динамичности, уже достаточно давно играют важную роль в функционировании современных предприятий. В условиях современного технологического прогресса и возрастающей конкуренции, а также экономически обусловленных тенденций к поиску наиболее продуктивных форм существования организаций, эффективные средства управления, основанные на широком использовании информационных технологий, являются жизненно необходимым фактором. Современный рынок предъявляет суровые требования к предприятиям любых размеров и отраслей, и наиболее успешными его игроками становятся те, кто вовремя осознал преимущества нового способа ведения бизнеса, основанного на повышенной скорости распространения и обработки информации.

Разработка и внедрение современных корпоративных ИС связана, как правило, с серьезным риском. Не секрет, что наиболее непредсказуемым фактором в области IT-решений является человеческий. Вместе с тем, влияние человеческого фактора сказывается и в процессе проектирования решений, и в процессе разработки, и в процессе дальнейшего использования и модификации разработанных ИС.

Статистические данные от многих аналитических компаний указывают, что по самым оптимистическим оценкам всего около 30% проектов в области IT можно считать успешно завершенными. Ни одна из существующих на данный момент методик разработки не в состоянии качественно переломить подобную ситуацию. При этом, в подавляющем большинстве случаев, для оценки успешности проекта берется только факт его завершения и сдачи в эксплуатацию, и совершенно не учитывается специфика использования разработанного ПО на предприятии. Вместе с тем, успешность IT-решений может быть измерена только экономическим эффектом от использования внедренного программного обеспечения в работе предприятия.

Одним из факторов, обусловливающих низкую предсказуемость результатов современных методик разработки ПО является постулат о том, что существует некий фиксированный «идеальный» набор требований к ИС, полностью определяющей разрабатываемую функциональность. Исходя из этого, можно выделить типичные этапы разработки программного средства (рис. 1).

При этом, этап выявления требований может включать в себя бизнес-моделирование, прототипирование и логическое проектирование

системы. К этапу разработки также может относится техническое проектирование ПС, собственно разработка, а также тестирование. Этап внедрения может включать доработку программного средства, локализацию и разработку специфичной функциональности для каждого заказчика.

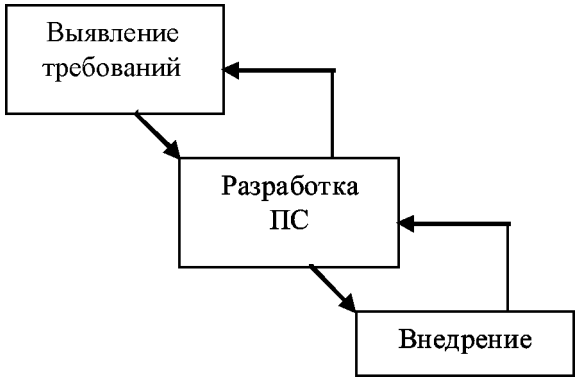

Рис. 1. Типичные этапы разработки ПС

Выявляемые несоответствия программных средств нуждам пользователей в ходе работ расцениваются, как ошибки проектирования или разработки, а концом проекта считается момент окончания доработок и начала промышленного использования ИС. Как показывает практика, практически любой подход, построенный с точки зрения этой модели дает достаточно низкую вероятность успешного завершения самого проекта (порядка 30%) и не позволяет сделать никаких предположений относительно длительности и эффективности использования разработанного программного обеспечения.

Исходя из этих реалий, в последнее время в области корпоративных информационных систем ведутся разработки по созданию подходов по созданию гибких корпоративных систем, способных перестраиваться за относительно короткое время согласно требованиям к реструктуризации бизнес-процессов. Движение современных предприятий в сторону деструктурированной, ориентированной на услуги индустрии, является одним из тенденций современной мировой экономики. Падение роли высокооптимальной производительности в повторяющихся процессах и повышение значимости времени выхода продукта на рынок приводит к возрастанию требований к гибкости современных корпоративных систем. Также, корпоративные системы должны предоставлять функциональность, обеспечивающую по возможности наиболее быстрое реагирование на требования клиентов и создание гибких решений.

КОНЦЕПЦИИ SERVICE-ORIENTED ARCHITECTURE (SOA)

Одним из подходов к созданию современных корпоративных ИС является проектирование сервис-ориентированных архитектур на основе методологии SOA (Service Oriented Architecture), разрабатываемой корпорацией IBM. При этом сама SOA представляет собой набор бизнес-методов, методов процесса, организационных методов, методов управления и технических методов для создания гибкой бизнес-среды с целью получения преимуществ в конкурентной борьбе. Сервис-ориентированная архитектура предлагает возможность гибкой работы с элементами бизнес-процессов и лежащей в их основе IT-инфраструктурой как с компонентами, которые можно использовать многократно и комбинировать при изменении приоритетов бизнеса.

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 на основе SOAP, WSDL и UDDI, несмотря на свою кажущуюся непроизводительность и избыточность, показывают свою жизнеспособность и полезность. Как правило, механизм сервисов SOAP является каркасом для интеграции бизнес-процессов и поддерживающей их IT-инфраструктуры в форме безопасных, стандартизированных компонентов (служб), предназначенных для многократного использования.

Использование подходов SOA в большинстве случаев позволяет реорганизовать процесс развития корпоративной информационной системы. 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.

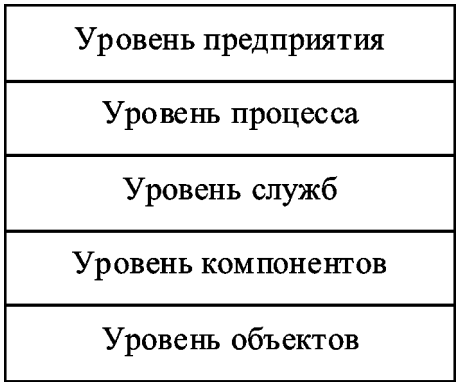

Рис. 2. Уровни абстракции SOA

С точки зрения SOA декомпозиция корпоративной ИС может осуществляться на нескольких уровнях абстракции, как показано на рис. 2. Каждый уровень абстракции SOA характеризуется несколькими важными категориями артефактов, каждая из которых обладает собственным набором свойств и связей. Уровень предприятия характеризуется бизнес-моделью деятельности предприятия. Бизнес-модель деятельности предприятия определяет критерии, согласно которым бизнес-процессы могут оцениваться либо как ключевые сферы деятельности, в которых можно достичь конкурентных преимуществ, либо как вспомогательные сферы деятельности, которые могут быть делегированы бизнес-партнерам.

Уровень процессов, включенных в бизнес-модель предприятия, характеризуется вниманием к структуре функциональных областей бизнеса. Каждая функциональная область может быть описана бизнес-процессов, который, в свою очередь, разделен на подпроцессы, необходимые для достижения заявленного набора целей. Декомпозиция подпроцессов может достигать нескольких уровней иерархии для выяснения зависимости от служб. Зачастую набор подпроцессов, необходимых для реализации бизнес-процессов является уникальным в контексте всего предприятия.

Абстракция уровня служб SOA характеризуется описанием отдельных бизнес-функций и служб, их реализующих. В SOA этот уровень часто является концептуальным связующим звеном между высшими уровнями предприятия и процесса и низшими уровнями компонентов и объектов. Некоторые бизнес-аналитики применяют этот уровень для выявления критических для работы бизнеса функций, а IT-специалисты используют этот уровень для выявления и демонстрации технических функций, соответствующих требованиям, сформулированным бизнес-аналитиками.

Уровень компонентов SOA реализует абстракции, направленные на выявление и описание отдельных компонентов ИС, которые могут быть объединены в реализациях служб. Анализ существующих прикладных систем зачастую позволяет выявить компоненты, которые являются кандидатами для многократного использования. Рассмотрение архитектуры SOA на уровне компонентов позволяет внести в структуру служб SOA изменения, вызванные техническими соображениями насчет оптимизации количества и функциональности компонентов в области организации их многократного использования.

Абстракция уровня объектов относится в SOA к широкому диапазону бизнес-объектов, их атрибутов и поведению, необходимому для выполнения каждой из бизнес-функций. С точки зрения построения SOA архитектур абстракции уровня объектов нацелены прежде всего на поиск типовых, «конвейерных» решений в области разработки служб для облегчения последующих задач интеграции и совместного использования служб.

Таким образом, использование архитектуры SOA для определения концепции развития корпоративных информационных систем позволяет сделать траекторию развития IT-инфраструктуры предприятия более предсказуемой. Использование архитектуры SOA сводит задачу оценки эффективности IT-решений к оценке экономических эффектов отдельных служб системы.

Задачу достижения максимального экономического эффекта от использования отдельной службы системы можно определить, как

$$\int_{T_0}^{T_0+T} f(S(t);F(t)) \to \max, \qquad (1)$$

где $S(t)$ — набор бизнес-требований к службе, $F(t)$ — реализованная функциональность, T — время использования службы, T_0 — момент начала использования службы, f — функция, оценивающая соответствие функциональности компонента $F(t)$ актуальным требованиям $S(t)$, $\in [0, 1]$. В качестве простейшего, самого грубого варианта функции f можно использовать выражение вида

$$f = \frac{D(S(t), F(t))}{|S(t)|} \qquad (2)$$

где D — мощность симметричной разности множеств $S(t)$ и $F(t)$ в момент времени t, $|S(t)|$ — мощность множества $S(t)$. Практика показывает, что зависимость расстояния между множествами $S(t)$ и $F(t)$ (т.е. степени соответствия функциональности компонента бизнес-требованиям) в случае процесса интенсивной (целенаправленной) разработки имеет S-образный характер. Это обусловлено тем, что в начале цикла разработки ресурсы затрачиваются не столько на реализацию бизнес-функций, сколько на выстраивание архитектуры программного средства. Примерно в середине цикла разработки достигается наиболее оптимальная производительность, спад которой в конце цикла разработки обусловлен усложнением как процесса реализации отдельных бизнес-функций (для реализации остаются наиболее сложные, комплексные бизнес-процессы), так и процесса интеграции разрабатываемых функций в существующую систему. Соответственно, варианты функции f, более соответствующие статистическим данным, следует искать среди семейств S-функций различной кривизны.

Рассмотрим модель функционирования службы, разработанной с использованием традиционного подхода программной инженерии. Схематичный график функции f, иллюстрирующий идеальный жизненный цикл такой службы, представлен на рис. 3.

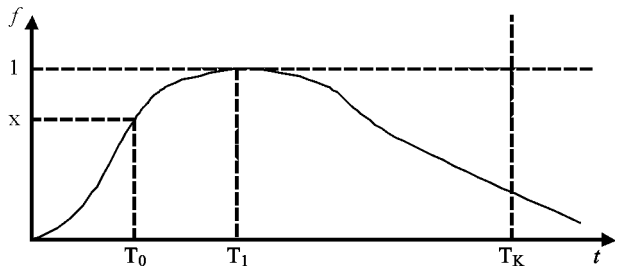

Рис. 3. Жизненный цикл службы при традиционном подходе к разработке

Фазы проектирования и разработки программного средства обычно планируются внутри отрезка времени $[0, T_0]$, до момента выпуска первой версии и начала пилотного внедрения. В случае использования каскадной модели для организации рабочего процесса к моменту времени T_0 полностью заканчиваются запланированные фазы разработки и проектирования службы. При этом, доля соответствия x функциональности разработанного программного средства $F(t)$ набору бизнес-требований $S(t)$ принадлежит диапазону $[0, 1]$. Иногда в случаях, если x меньше некоего порогового значения, принимается решение о закрытии проекта и исключении его результатов из процесса автоматизации предприятия. Как правило, после фактического закрытия неудачного проекта открывается повторный проект по решению той же самой автоматизации, где работа ведется уже с учетом опыта и наработок первого проекта. В рамках подхода SOA, однако, возможен также вариант, когда в рамках той же задачи новый проект не открывается, а наработки завершенного проекта используются при разработке смежных с точки зрения функциональности, служб. В этом случае область функциональности этих служб, соответственно, расширяется.

После принятия решения о внедрении начинается процесс доработки службы, тестирования и встраивания в работающие бизнес-процессы предприятия. Этому соответствуют фазы тестирования и внедрения. Иногда, в случае, если наблюдается значительное расхождение функциональности службы $F(t)$ со структурой бизнес-требований $S(t)$, процесс может вернуться на стадию разработки, а иногда — и на стадию проектирования системы (т.н. модель «выпрыгивающего лосося»). Одновременно с доработками начинается использование разработанного продукта в работе предприятия, поэтому на этапе приближения функциональности службы к желаемому результату $[T_0, T_1]$ уже можно говорить об эффекте от использования службы. Процесс доработки и внедрения продолжается до некоего порогового значения y (в идеале $y = 1$) соответствия функциональности службы $F(t)$ бизнес-требованиям $S(t)$, после которого происходит формальное завершение проекта с принятием решения о промышленной эксплуатации разработанного программного продукта. На этом проект с точки зрения традиционных методов может считаться завершенным.

Рассмотрим следующие стадии жизненного цикла разработанной службы. Процесс работы службы в режиме промышленной эксплуатации, как показывает практика, наиболее перспективен с точки зрения получения эффекта. При этом, за счет изменения среды существования службы, а также за счет неизбежной рест-

руктуризации бизнес-процессов полезность (степень удовлетворения бизнес-требований) программного средства снижается вплоть до момента T_k, когда принимается решение о выводе службы из эксплуатации. Основными причинами вывода программного средства из эксплуатации являются:

1. Снижение уровня используемой функциональности до некоего критического порога (служба становится фактически бесполезной в новых условиях).

2. Несоответствие службы реалиям архитектуры (при смене операционной системы, модели данных, среды взаимодействия служб, стандартов и т.д.).

3. Замена службы на новую версию, либо полное распределение функционала службы между другими, введенными в эксплуатацию, службами.

4. Изменение автоматизируемых бизнес-процессов, вызывающее серьезное несоответствие между функциональностью службы и новыми бизнес-требованиями.

ЗАКЛЮЧЕНИЕ

Таким образом, можно выделить следующие пути повышения эффективности автоматизации бизнес-процессов:

1. Увеличение площади фигуры на отрезке $[T_0, T_1]$. Это осуществляется, в основном, за счет увеличения длины отрезка $[T_0, T_1]$, т.е. за счет как можно более раннего начала использования функциональности службы. Это подтверждается эффективностью использования спиральных моделей разработки и практикой поэтапного внедрения.

2. Увеличение площади фигуры на отрезке $[T_1, T_k]$. Здесь эффект достигается как за счет увеличения длины отрезка (т.е. времени использования службы), так и за счет изменения характера графика (путем техподдержки разработанного средства, внесения своевременных изменений и постоянной адаптации уже работающей службы к новым требованиям и условиям).

3. Планирование поэтапного развития функциональности системы за счет использования SOA при проектировании архитектуры системы и декомпозиции программных модулей на службы, автоматизирующие отдельные бизнес-процессы.

Оценка реальной эффективности программных решений в области автоматизации бизнес-процессов невозможна без учета специфики изменяющихся в ходе развития предприятия требований, а также анализа полезности используемой службы в течение всего жизненного цикла разработанного ПО. Использование методологии SOA в процессе проектирования корпоративных ИС позволяет более гибко организовать развитие систем, адаптацию к изменяющимся требованиям бизнеса, а также максимизировать экономический эффект от использования системы за счет управления жизненными циклами разрабатываемых в рамках ИС служб.

СПИСОК ЛИТЕРАТУРЫ

1. *Брукс Ф.* Мифический человеко-месяц или как создаются программные системы. — Спб.: Символ-Плюс, 1999. — 304 с.

2. *Макконнелл С.* Сколько стоит программный проект. — М.: «Русская Редакция», Спб.: Питер, 2007. — 297 с.

3. *Биберштейн Н., Боуз С., Джонс К., Фиммант М., Ша Р.* Компас в мире сервис-ориентированной архитектуры (SOA): ценность для бизнеса, планирование и план развития предприятия / пер. с англ. — М.:КУДИЦ-ПРЕСС, 2007. — 256 с.

4. http://www.gartner.com

Сапегин Сергей Владимирович — к.т.н., доцент кафедры Программирования и информационных технологий, Воронежский государственный университет. Тел. (4732) 208-470. E-mail:sapegin@cs.vsu.ru

Sapegin Sergey V. — Candidat of Technical Sciences, Associate Professor, the dept. of the Programming and Information Technologies, Voronezh State University. Tel. (4732) 208-470. E-mail:sapegin@cs.vsu.ru