***СОРОКИН Александр Викторович –*** *кандидат технических наук, доцент, менеджер университетских программ IBM*

# ПРОДУКТЫ И СТАНДАРТЫ СЕРВИС–ОРИЕНТИРОВАННОЙ АРХИТЕКТУРЫ, ОБЕСПЕЧИВАЮЩИЕ ГЕНЕРАЦИЮ И УПРАВЛЕНИЕ ПОТОКАМИ РАБОТ

## *1. Использование потоковых моделей в проектировании информационных систем*

До конца прошлого века при построении информационных систем доминировал подход, основанный на анализе и управлении информационными потоками. Однако со временем в качестве более универсального и информативного стал рассматриваться процесс-ориентированный подход, в рамках которого выбор и изменение структуры информационной системы, состав её компонент и основные характеристики осуществляются «от процесса», в зависимости от его построения и особенностей, определяемых конечными пользователями.

Такая смена позиции разработчиков вызвала к жизни мощное направление в теории, методах и технологии разработки систем – моделирование и управление бизнес-процессами, на основе которого были созданы все известные ERP-системы.

В настоящее время центр приложения анализа смещается ещё глубже - в сторону манипулирования работами, построения средств генерирования потоков работ и управления ими. Управление бизнес-процессами и управление потоками работ – не одно и то же.

Бизнес-процесс определяет ***что*** должно быть сделано, в том числе, устанавливая соотношение между входом и выходом. В составе бизнес–процесса могут оказаться неавтоматизируемые (и\или неавтоматизированные) составляющие, отрабатываемые вручную, и он может использовать любые виды ресурсов.

Поток работ описывает ***как*** определённый результат может быть достигнут.

Тем не менее, взаимосвязь между бизнес-процессом и потоком работ достаточно простая [1,3] и иллюстрируется схемой, приведённой на рис. 1.

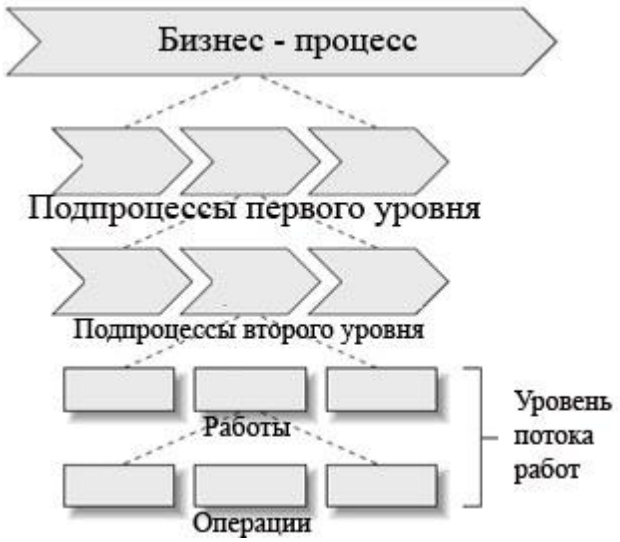


Рис. 1. Взаимосвязь бизнес-процессов и потоков работ

Как и в предшествующем случае, смена позиции разработчиков объясняется изменениями в концепции построения информационных систем. Происходящий в настоящее время переход к сервис-ориентированной архитектуре (СОА) означает, прежде всего, необходимость выделения функциональных сервисов–элементов СОА, из которых потом осуществляется синтез и модернизация бизнес-процессов.

По определению, даваемому организацией OASIS (Organization for the Advancement of Structured Information Standards, http://www.oasis-open.org/), занимающейся вопросами стандартизации компонентов СОА, СОА представляет собой парадигму распределенных организационных и утилитарных возможностей, работающих под управлением доменов, принадлежащих различным владельцам.

По определению IBM [7], СОА – архитектурный стиль для создания ИТ-архитектуры предприятия, основанный на сервисной ориентации для достижения более тесной взаимосвязи между бизнесом и поддерживающими бизнес-информационными системами. СОА вводит сервисную ориентацию в качестве подхода к интеграции бизнеса на основе связанных между собой сервисов.

В рамках концепции СОА крупные конгломераты программных и информационных средств должны быть разделены на небольшие унифицированные стандартизированные сервисы, каждому из которых, в свою очередь, соответствует базовая активность (работа).

Для выполнения этой функции используются механизмы моделирования бизнес-процессов.

В сочетании с возможностями быстрого внесения изменений в процессы наличие таких механизмов считается основой эффективности ИТ-систем, базирующихся на СОА. Для решения возникающих при этом задач используются бизнес–инструменты, в общем случае обеспечивающие:

- ✓ построение новых бизнес-функций;
- ✓ установление связей между бизнес-функциями, выделяемыми из существующих приложений;
- ✓ генерирование потоков работ для выполнения бизнес-процессов.

***2. Стандартизация средств и унификация процессов***

Реализация современных сложных информационных систем на изложенных выше принципах невозможна без стандартизации и унификации процессов и элементов, из которых системы строятся. На сегодняшний день в мире нет какого-либо единого центра, координирующего общую политику в данной области, и несколько организаций разрабатывают и поддерживают стандарты СОА.

- ✓ Во-первых, это уже упомянутый выше OASIS.
- ✓ Далее, W3C - World Wide Web Consortium (http://www.w3.org/Consortium) - международный консорциум, ведущий разработки в области совместимости в интернете.
- ✓ Web Services Interoperability Organization - WS-I. Открытая для членства организация, разрабатывающая рекомендации и стандарты для межплатформенного взаимодействия web-сервисов (http://www.ws-i.org).
- ✓ IETF (Internet Engineering Task Force) - Интернет-сообщество (www.ietf.org), включающее разработчиков, поставщиков, операторов и исследователей, интересующихся вопросами развития архитектуры интернета и решающее задачи, обеспечивающие плавный переход к web следующего поколения.
- ✓ OMG - Object Management Group (www.omg.org) - международный некоммерческий консорциум с открытым членством. Цель консорциума - разработка стандартов для объектного и графического проектирования на основе архитектуры Common Object Request Broker Architecture (CORBA), обеспечивающих интеграцию в интернете предприятий любого профиля.

Общими усилиями они разрабатывают стандарты, поддерживающие интеграцию различных компонент СОА.

На приведённом ниже рисунке 2 показана динамика разработки и состав действующих в этой области стандартов [11].

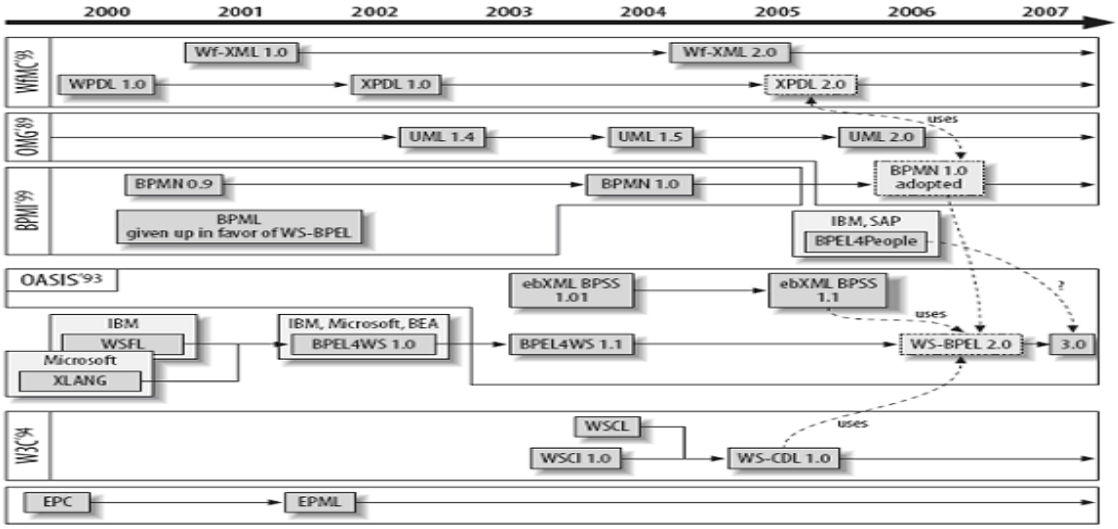


Рис. 2. Совокупность стандартов, используемых для моделирования Web-сервисов и потоков работ

***2.1 Языковые стандарты***

Для успешной интеграции систем на основе Web-сервисов используется целый ряд стандартов, основными из которых являются:

- ✓ BPEL - язык реализации бизнес-процессов;
- ✓ WSDL - язык описания сервисов;
- ✓ SOAP - протокол обмена сообщениями;
- ✓ UDDI - универсальный формат каталога для поиска и интеграции web - сервисов.

Композиция Web-сервисов на основе элементарных работ осуществляется с помощью языка реализации бизнес-процессов - Business Process Execution Language (BPEL). Он появился как результат объединения языка WSFL (Web Services Flow Language), разработанного корпорацией IBM, и языка XLANG, созданного в Microsoft. Язык построен на нотации XML.

Использование данного языка позволяет осуществить формирование и исполнение потока работ как последовательность логических действий, включающих:

- ✓ принятие запроса на включение работы в процесс;
- ✓ проверку описания, и в случае совпадения параметров подготовку положительного отклика на запрос;

- ✓ отклонение запроса в противном случае с выдачей обоснования.

Как правило, BPEL формирует поток работ, состоящий из последовательностей логических действий или активностей, каждой из которых соответствует свой квадратик на диаграмме потока работ или функция программного кода. При этом возможны две формы использования BPEL.

Первая - исполняемый BPEL - процесс, который также рассматривается в качестве сервиса, и может являться узлом оркестровки. Программные продукты, реализующие исполняемые BPEL–процессы, называются BPEL-engine (движок BPEL-процесса). Здесь один исполняемый процесс может включать другой, что дает эффект включения одной оркестровки (последовательности сервисов) в другую, как это показано на рис. 3.

Вторая форма – так называемый абстрактный процесс, который почти полностью идентичен исполняемому процессу за исключением наполнения данными. В этом качестве он представляет логику бизнес-процесса и используется в следующих целях:

- ✓ определяет поведение элементов организационной структуры, поддерживающей процесс;
- ✓ представляет собой руководство для программистов и разработчиков, автоматизирующих процесс;
- ✓ является входом для коммерческого программного обеспечения, образующего скелет системы, в качестве выхода может применяться Java, либо другой язык.

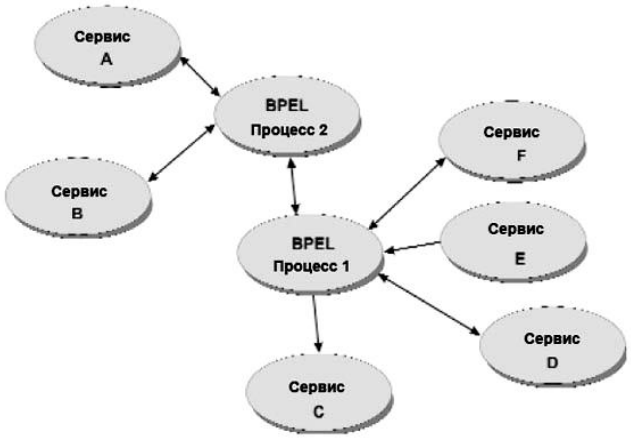


Рис. 3. Связанные оркестровки

WSDL (Web Services Description Language) — язык описания веб-сервисов, основанный на языке XML. Обеспечивает правильный выбор сервисов для передачи от провайдера к потребителю. Информация в формате WSDL используется разработчиками сервисов для включения сервисов в систему.

SOAP (Simple Object Access Protocol) - протокол обмена сообщениями, также написанный в формате XML, предназначен для передачи данных из/в web-сервисы. Файлы SOAP, создаваемые автоматически, включают данные из описания сервисов в формате WSDL.

Последние документы и версии SOAP находятся на странице: *http://www.w3.org/TR/2003/REC-soap12-part0-20030624*

UDDI представляет собой набор правил регистрации и извлечения данных об имеющихся сервисах. Разрабатывая программы, программисты могут осуществлять поиск в реестре UDDI необходимых сервисов для включения их в программы. Данный реестр может также быть востребован в процессе выполнения программы, которой необходимы сервисы, предоставляющие данные, например, о стоимости определенного продукта или услуги. Последний вариант использования UDDI менее употребим, чем первый.

***2.2 Унификация процессов***

Дальнейшие шаги в области унификации представления бизнес-процессов были сделаны компанией IBM, внедрившей в практику автоматизации унифицированное описание процесса - Rational Unified Process (RUP). Пример создания автоматизированной системы с использованием RUP и программных продуктов IBM, поддерживающих моделирование потока работ, показан на рис. 4 [5,6,7].

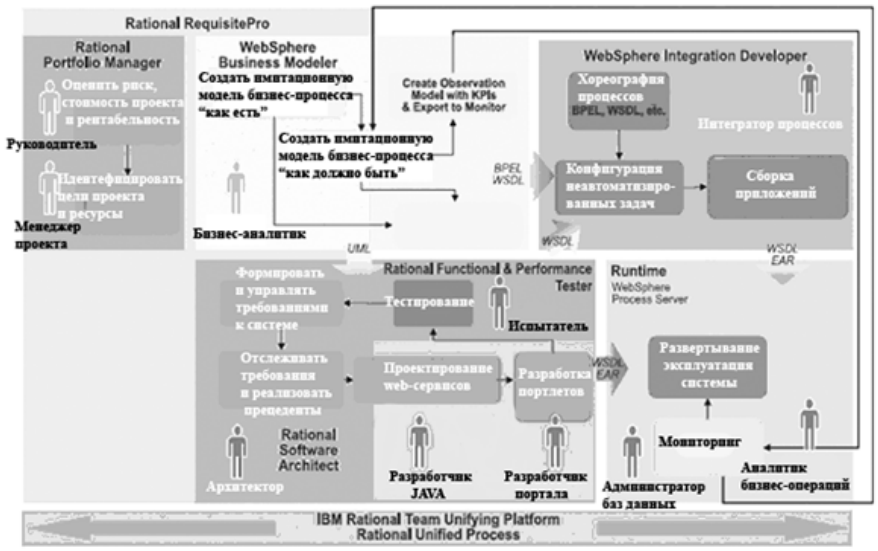


Рис.4. Создание автоматизированной системы с использованием унифицированного процесса Rational (RUP)

Ниже представлено пошаговое описание показанного на рисунке унифицированного процесса создания информационной системы.

**Шаг 1**. Приложение Rational Portfolio Manager помогает достичь согласования на верхнем уровне моделирования по таким показателям, как рентабельность (ROI), стоимость и другим условиям ведения бизнеса.

**Шаг 2.** Бизнес-аналитик разрабатывает модель бизнес-процесса, используя приложение WebSphere Business Modeler.

**Шаг 3.** Бизнес-процесс представляется в Rational Software Architect в стандартной нотации UML.

**Шаг 4.** Rational Software Architect преобразует модель в нотации UML в код Java.

**Шаг 5.** IBM Rational Application Developer автоматизирует процесс создания новых Web-сервисов на базе существующих ресурсов, таких как компоненты JavaBeans или EJB, посредством автоматического генерирования файлов WSDL, описывающих Web-сервисы, дескриптора SOAP и клиента для тестирования Web-сервисов. Например, в форме портлета. Портлет — настраиваемый стандартный портальный компонент пользовательского интерфейса (Wikipedia). Портлет выдаёт фрагменты разметки, которые встраиваются в страницу портала. Можно представить себе страницу портала как коллекцию неперекрывающихся портлетных окон, каждое из которых отображает портлет. Портлет как бы представляет собой веб-приложение, размещённое на портале. Примеры портлетов: e-mail, сообщения о погоде, последние новости.

Благодаря существующим стандартам портлетов разработчики могут создавать портлеты, встраиваемые в любой портал, следующий этим стандартам.

**Шаг 6.** WebSphere Integration Developer реализует бизнес-процесс в формате BPEL.

Разработчик, осуществляющий интеграцию Web-сервисов в бизнес-процесс, может извлечь готовый сервис из UDDI и включить его в бизнес-процесс, или включить в него функции, выполняемые людьми. Получаемый результат развертывается в виде приложения WebSphere Process Server.

При реализации проектов, в которых используется управление жизненным циклом, руководитель проекта и члены команды для поддержки разработки применяют Rational ClearQuest. Этот продукт автоматизирует назначение заданий и отслеживает ход их выполнения, а также используется для тестирования и выявления дефектов в спроектированном потоке работ. Он также применяется при редактировании, сертификации и верификации сервисов.

Для решения задач управления конфигурацией применяется IBM Rational ClearCase, интегрированный с Eclipse.

### *3. Эволюционный переход к СОА с использованием наследуемого программного обеспечения*

Обычным аргументом против перехода к СОА является утверждение, что он связан с полным сломом всей существующей системы, и поэтому требует единовременных очень высоких затрат. В действительности переход к СОА не требует единовременного и полного изменения существующей информационной системы. Отдельные шаги или приложения могут внедряться постепенно, и весь переход может быть осуществлен эволюционным путем. При этом используются технологии, обеспечивающие соблюдение ранее введенных стандартов, а также преемственность по отношению к ранее инсталированным работоспособным программным продуктам и техническому оборудованию. Обобщенная структура СОА показана на рис. 5.

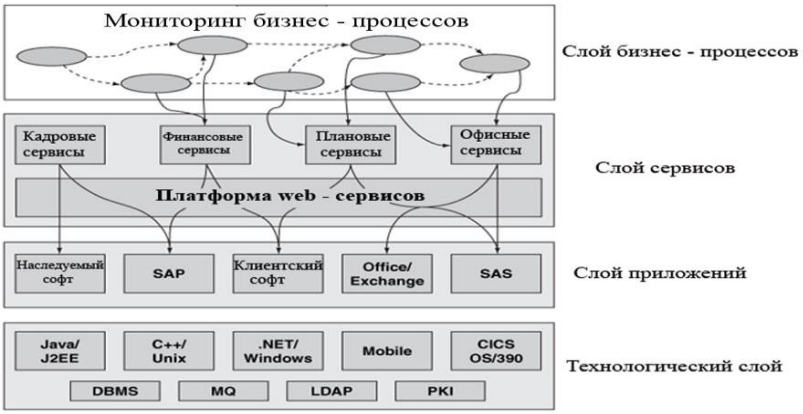

Рис. 5. Пример сервис–ориентированной архитектуры

На рисунке 5 изображена типичная ситуация, когда информационная система на базе СОА строится на базе существующего проприативного программного обеспечения. В качестве примера показаны такие популярные приложения, как SAP, SAS, офисные пакеты, а также клиентское программное обеспечение и программы, наследуемые от предшествующих этапов развития системы.

Вся структура состоит из четырех слоёв: бизнес–процессов, сервисов, приложений и технологического слоя. Многие аналитики выводят историю СОА, начиная с конца прошлого века, имея в виду развитие функционального подхода, а позднее концепцию COBRA и инструментарий Java [2]. Тем не менее, широкого распространения она не получила, поскольку в составе используемой в то время архитектуры отсутствовал сервисный слой. Множественные приложения имели различный интерфейс GUI, несовместимую логику бизнес–процессов, разные системы хранения данных (СУБД). Все это затрудняло работу системы. Реализация бизнес-процессов напрямую была связана с технологическим уровнем и зависела от скорости обмена информацией между одинаковыми по функциональному назначению, но разнородными по конструкции компонентами (например, между различными БД приложений).

Переход к современной СОА и включение в неё дополнительного сервисного слоя формирует линию воспроизводимых сервисов, каждый из которых привязан к соответствующему бизнес-домену и платформе Web–сервисов, что позволяет осуществлять их использование в значительной степени независимо от расположенных ниже слоя приложений и технологической платформы.

Введение слоя сервисов создаёт также благоприятные условия для построения расположенного выше слоя бизнес–процессов и их функционирования, а именно:

- ✓ Линия бизнес сервисов воспроизводит грубый набор функций, готовых для включения в бизнес-процесс.
- ✓ Описание (contract) каждого сервиса в линии сервисов однозначно его определяет, формируя интерфейс. Наличие последнего позволяет создавать бизнес–процессы независимо и без привлечения знаний относительно технологической платформы.
- ✓ Инструмент реестра и поиска сервисов поддерживает динамику и возможность изменений бизнес–процессов путем оперативного доступа к сервисам в случае такой необходимости.
- ✓ Модель данных, реализованная на уровне сервисов, опирается на структуру бизнес-домена и тоже не зависит от моделей данных отдельных приложений. Далее, поскольку XML применяется в качестве канонического формата обмена данными, включение сервиса в бизнес–процесс осуществляется также независимо от внутренней структуры данных в приложениях.
- ✓ Модель защиты сервисов, действующая в рамках этого слоя, обеспечивает универсальный контроль ролевого распределения и подключения сервиса, авторизируя использования сервиса некоторым процессом. Это также позволяет избежать трудностей непосредственного взаимодействия со средствами защиты приложений, с их различием в принципах организации, интерфейсах и т.п.
- ✓ Модель управления уровнем обслуживания генерирует статистику использования сервисов, что является частью мониторинга бизнес–процесса, выполняемого верхним слоем (бизнес-процессов).

**Отсутствие этого слоя в предыдущих поколениях информационных систем создавало множество уязвимостей, которые способствовали быстрой деградации бизнес–процессов, и именно этот фактор тормозил распространение СОА.**

Естественной платой за подобное совершенство является усложнение системы, что и выстраивает определённые барьеры на пути широкого перехода к СОА**.**

В примере, взятом из книги [2], показана возможность перехода к СОА путем «естественной» эволюции и усовершенствования системы, не требующим разовых больших затрат и полного слома привычного процесса функционирования системы. В этом примере переход к СОА осуществляется поэтапно (рис.6).

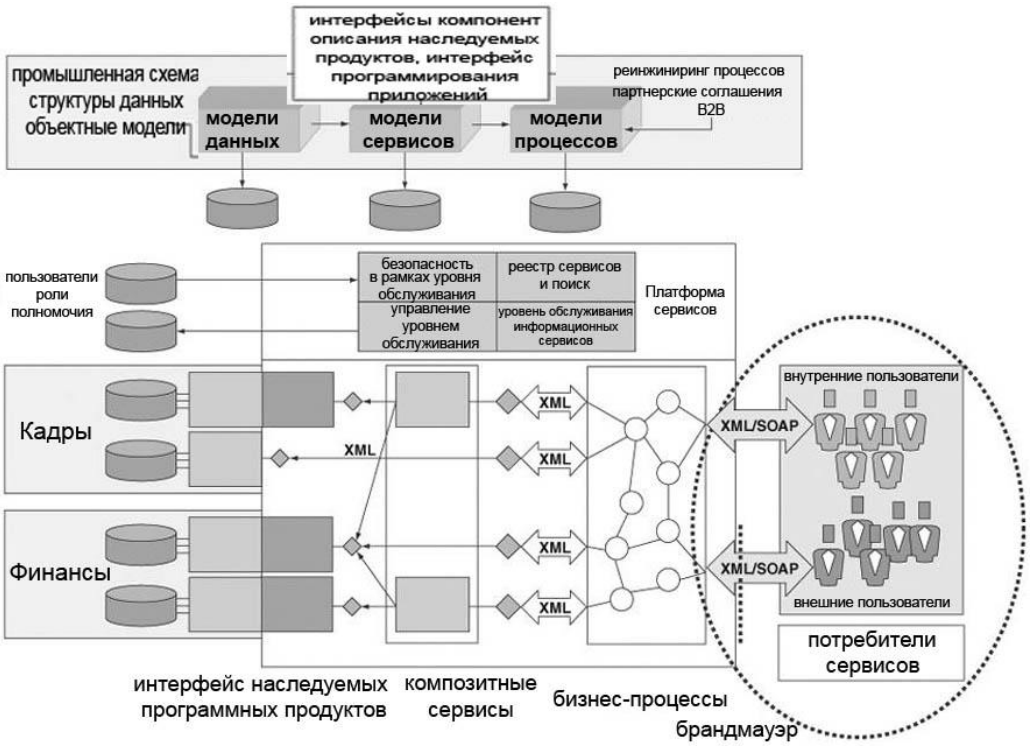


Рис. 6. Эволюция информационной системы на основе СОА с включением наследуемого программного обеспечения

На рисунке 6 в развиваемой системе выделены две подсистемы – кадров и финансов. На первом этапе эволюции разрабатываются новые сервисы, включаемые в эти подсистемы. На рисунке они крайние слева и обозначены более светлым серым цветом. Новые сервисы сразу же выполнены под стандарты и требования будущей СОА. Правее от них показаны сервисы, построенные на основе наследуемых приложений. Эти сервисы сохраняют конструктивные особенности старой системы. Для их использования в будущей системе они заключаются в «оболочку», в качестве которой выступает показанный на рисунке интерфейс, спроектированный уже под новую логику реализации процессов. Далее, еще правее следуют так называемые композитные сервисы. Они строятся как комбинация атомарных сервисов, в качестве которых могут быть задействованы как новые, так и наследуемые сервисы.

И, наконец, с помощью инструментария, включенного в слой сервисов (реестра, служб управления уровнем обслуживания, безопасности и построения бизнес–процессов), строится бизнес–процесс, объединяющий как новые сервисы, так и наследуемое программное обеспечение.

Разработка подобной схемы может служить основой организации и планирования процесса перехода к СОА посредством постепенного развития информационной системы.

***Литература***

*1. Mathias Weske. Business Process Management. - Springer-Verlag, Berlin-Heidelberg, 2007.*

*2. Eric Newcomer, Greg Lomow. Understanding SOA with Web Services. - Addison Wesley Professional, 2004.*

*3. IBM Redbook, IBM Enterprise Workload Manager. – V.2.1 (http://www.redbooks.ibm.com/redbooks/pdfs/sg246785.pdf).*

*4. Bobby Woolf. Exploring IBM SOA Technology&Practice. - Maximum press, 2007.*

*5. IBM Redbook. Patterns: Implementing an SOA Using an Enterprise Service Bus. - 2004.*

*6. SOA Development Using the IBM Rational Software Development Platform: A Practical Guide.- 2005.*

*7. IBM Redbook Patterns: Service-Oriented Architecture and Web Services. - April 2004.*

*8. Dan Woods. Enterprise Services Architecture. - O'Reily Field Guide. - 2007.*