Занятие 4.
Открытые системы.
Сервис-ориентированная архитектура.

Открытые информационные системы

Одной из главных тенденций современной индустрии информатики является создание *открытых систем.*

Свойство открытости означает:

1) переносимость (мобильность) ПО на различные аппаратные платформы,
2) приспособленность системы к ее модификациям (модифицируемость или собственно открытость)
3) приспособленность системы к комплексированию с другими системами в целях расширения ее функциональных возможностей и (или) придания системе новых качеств (интегрируемость).

Переход к открытым информационным системам позволяет существенно ускорить построение ИС в результате замены длительной и дорогостоящей разработки новых систем по полному циклу их компоновкой из ранее спроектированных подсистем или быстрой модернизацией уже существующих систем (реинжиниринг).

Открытость подразумевает выделение в системе интерфейсной части (входов и выходов), обеспечивающей сопряжение с другими системами или подсистемами, причем для комплексирования достаточно располагать сведениями только об интерфейсных частях сопрягаемых объектов. Если же интерфейсные части выполнены в соответствии с заранее оговоренными правилами и соглашениями, которых должны придерживаться все создатели открытых систем определенного приложения, то проблема создания новых сложных систем значительно упрощается. Из этого следует, что основой создания открытых систем являются стандартизация и унификация в области информационных технологий.

Значительное развитие концепция открытости получила в области построения вычислительных сетей, что нашло отражение в эталонной модели взаимосвязи открытых систем, поддерживаемой рядом международных стандартов. Идеи открытости широко используются при построении программного, информационного и лингвистического обеспечений ИС; в результате повышается степень универсальности программ и расширяются возможности их адаптации к конкретным условиям.

Аспекты открытости отражены в стандартизации:

- *API {Application Program Interface)* - интерфейсов прикладных программ с операционным окружением, в том числе системных вызовов и утилит операционной системы (ОС), т.е. связей с ОС;
- межпрограммного интерфейса, включая языки программирования;
- сетевого взаимодействия;
- пользовательского интерфейса, в том числе средств графического взаимодействия пользователя с ЭВМ;
- средств зашиты информации.

Стандарты, обеспечивающие открытость ПО, в настоящее время разрабатываются такими организациями, как ISO (International Standard Organization), IEEE (Institute of Electrical and Electronics Engineers), EIA (Electronics Industries Association) и др.

Среди других стандартов, способствующих открытости ПО ИС, следует отметить стандарты графического пользовательского интерфейса, хранения и передачи графических данных, построения баз данных и файловых систем, сопровождения и управления конфигурацией программных систем и др.

Важное значение для создания открытых систем имеют унификация и стандартизация средств межпрограммного интерфейса, или, другими словами, необходимо наличие профилей ИС для информационного взаимодействия программ, входящих в ИС. *Профилем* открытой системы называют совокупность стандартов и других нормативных документов, обеспечивающих выполнение системой заданных функций.

Так, в профилях ИС могут фигурировать унифицированный язык SQL обмена данными между различными СУБД, стандарты сетевого взаимодействия и т. п.

Введение в SOA

SOA - это архитектурный подход к определению, связыванию и интеграции повторно используемых бизнес-сервисов, имеющих четкие границы и самодостаточных по своей функциональности. В рамках такой архитектуры можно организовывать бизнес-сервисы в бизнес-процессы. Внедряя концепцию сервисов (более высокого уровня абстракции, не зависящего от приложений и платформы информационной инфраструктуры, а также от контекста или других сервисов), SOA переносит информационные технологии на следующий уровень, более подходящий для обеспечения функциональной совместимости и реализации в гетерогенных средах.

Поскольку SOA основывается на стандартах (таких, например, как Web-сервисы), утвержденных и поддерживаемых основными поставщиками информационных технологий, сервисы можно быстро создавать и объединять. Можно организовать взаимодействие предприятий независимо от их инфраструктуры, что обеспечивает делегирование, совместное применение, повторное использование и максимальную эффективность существующих активов.

При внедрении SOA вы переводите внутреннюю информационную инфраструктуру на более высокий, более открытый и управляемый уровень. С появлением повторно используемых сервисов и высокоуровневых процессов внесение изменений становится проще, чем когда бы то ни было, и начинает больше походить на разборку и сборку частей (сервисов) в новые процессы, направленные на осуществление бизнес-деятельности. Это не только способствует повышению эффективности и повторному использованию, но и дает возможность изменять и подстраивать информационные технологии под меняющиеся бизнес-требования.

Что лучше всего подходит для SOA?

Хотите узнать, для каких бизнес-функций и ситуаций архитектура SOA подходит лучше всего и где она может полностью проявить свой потенциал? Существуют определенные ситуации и бизнес-функции, когда следует немедленно обратиться к SOA, поскольку эта архитектура может существенно повысить конкурентоспособность и производительность и четко проявить свои преимущества. К таким ситуациям главным образом относятся:

- **Централизованные бизнес-функции, используемые несколькими субъектами**.

SOA помогает идентифицировать эти функции и собрать их в повторно используемые самодостаточные сервисы, не подверженные влиянию изменений в процессах, их использующих.

- **Интеграция с партнерами**. SOA способствует применению стандартов, создающих единые критерии для работы всех заинтересованных сторон. Кроме того, обеспечиваемая архитектурой SOA гибкость улучшает процесс интеграции благодаря возможности подключать, изменять и обновлять сервисы практически незаметно для ваших клиентов.

- **Наличие работающих старых технологий**. Некоторые организации не желают отказываться от проверенных и надежных старых технологий. Вопросы безопасности делают некоторых пользователей, особенно в сфере банковского обслуживания, недоверчивыми к новым программным системам и их неисследованным уязвимостям. В таких ситуациях SOA может помочь облачить старые технологии в новые стандарты, отобразить их в основанной на стандартах среде и сделать пригодными для интеграции и повторного использования.

Чем обусловлена быстрая настройка под меняющиеся бизнес-требования?

Из-за неизбежности изменений единственным гарантом обеспечения непрерывности бизнес-деятельности является способность адаптироваться к изменениям и быть готовым к ним (подвижность бизнеса - agility). SOA обеспечивает возможность адаптации к бизнес-требованиям (что имеет решающее значение для будущего любой деятельности), благодаря следующим факторам:

Слабое связывание

1. Устраняет жесткие связи, препятствующие изменениям.
2. Меньше вложений в реализацию и больше в повторное использование.
3. Улучшает возможности удаленного доступа к оригинальным источникам информации, уменьшая задержки и зависимости.
4. Проекты по интеграции управляются бизнес-требованиями (то есть бизнес-деятельность является основной движущей силой).
5. Благодаря отображению и совместному использованию информации, слабое связывание позволяет компаниям извлекать в режиме реального времени больше данных об эффективности бизнес-деятельности.
6. Облегчает партнерам взаимодействие с вашей компанией.
7. Способствует продвижению и публикации ваших сервисов, облегчая клиентам обнаружение их и вашей компании.
8. Облегчает поиск новых партнеров и сервисов, помогая найти более подходящий под ваши требования сервис.

Повторное использование

1. Делает процессы более согласованными, поскольку они базируются на одних и тех

же компонентах.

2. Способствует повышению качества благодаря конкуренции между провайдерами сервисов.
3. Позволяет изменять систему независимо от изменений бизнес-деятельности.
4. Уменьшает влияние изменений, поскольку они выполняются централизовано и охватывают все участвующие стороны.

Расширяемость

1. Делает SOA-решения доступными организациям любого размера.
2. Изменяет процесс разработки на более динамичный, более подходящий для ведения бизнес-деятельности.
3. Ускоряет слияния и поглощения.

Так что же теряет компания, отказываясь от внедрения SOA?

Если SOA является приемлемым решением для компании, отказ от ее реализации может привести к следующим трем главным негативным последствиям:

- Неспособность охватить более крупные рыночные ниши, обеспечивающие дальнейшее развитие бизнес-деятельности. Поскольку компания привязана к своим специализированным системам, она застревает в своей первоначальной рыночной нише и затрачивает много усилий, чтобы пробиться выше. Напротив, используя SOA, организация может менять бизнес-тактики и реализовать новые, оставаясь на гребне.

В каких случаях применение SOA не обосновано

Архитектура SOA приносит пользу бизнес-организациям практически во всех ситуациях. Однако в очень специализированных случаях она может помешать улучшению бизнес-деятельности. К таким ситуациям относятся:

- **Когда информационная среда гомогенна**. Если организация использует комплекс согласованных продуктов (принадлежащих, например, одному производителю), SOA может оказаться помехой, а не полезной стратегией.
- **Когда критична производительность в режиме реального времени**. В силу слабого связывания между различными потребителями и производителями архитектура SOA зависит от протоколов взаимодействия, которые по своей природе являются медленными. Она также склонна применять логику посредничества и асинхронные протоколы, которые не подходят для эффективной работы в режиме реального времени.
- **Когда ничего не меняется**. Если потребитель не видит изменений в бизнес-логике, представлении, потоке данных, процессе или любых других аспектах приложения, преобразование старых систем в SOA может не оправдать затраченных усилий.
- **Когда тесное связывание не является недостатком**. Слабое связывание приносит пользу, когда оно используется с компонентом, который вами не управляется и изменения которого вы, следовательно, не можете контролировать. С другой

стороны, когда компонент ваш и находится под вашим контролем, слабое связывание может потребовать дополнительных накладных расходов, особенно если компонент не является повторно используемым.

Концепции SOA

Давайте рассмотрим некоторые концепции архитектуры SOA, чтобы лучше понять, что она собой представляет.

Определение сервиса в SOA

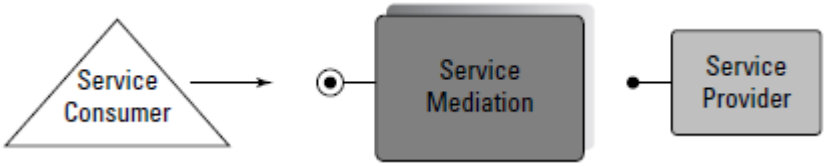
Существует множество различных определений сервисов, но, на мой взгляд, лучше всего объясняют сущность сервисов определения, приведенные ниже.

Сервис - это функция, являющаяся четко определенной, самодостаточной и не зависящей от контекста или состояния других сервисов.

Концепция слабого связывания в SOA

Виртуализация сервиса (Service Mediation).

Разделение сервиса происходит с помощью механизма виртуализации. Виртуальный сервис является прокси-объектом для реального сервиса. Прокси-сервис представляет собой желаемый потребителем сервиса интерфейс. Потребители обращаются к прокси-сервису, передающему сообщения к действительному сервису.



Виртуализация сервиса обеспечивает гибкость, необходимую при внедрении SOA. Эта гибкость основана на факте, что виртуальный сервис разделяет поставщика и потребителя в терминах местоположения, передачи данных и сообщений.

Независимость местоположения.

Виртуальный сервис позволяет скрыть действительное местоположение сервиса от потребителей. Это дает свободу перемещать реализацию сервиса без уведомления потребителей. Например, вы можете переместить сервис на сервера большей мощности для увеличения производительности.

Независимость передачи данных.

Виртуализация сервиса позволяет снабжать сервис несколькими средствами передачи данных. Предположим, вы создали сервис «CreateOrder», доступный через JMS (Java Message Service). Сервис стал популярен и некоторые пользователи желают расширить функциональность своих приложений данным сервисом. Сложность в том, что они могут использовать HTTP-протокол. Обычно требуется создать другую реализацию сервиса “CreateOrder” для поддержки HTTP, но возможности виртуальных сервисов позволяет создать виртуальный HTTP-сервис без изменения реализации. Это прозрачно решает проблему взаимодействия и позволяет расширять число пользователей сервиса.

Независимость сообщений.

Иногда потребители сервиса не синхронизированы с поставщиками в смысле

ожидаемых сервисом XML-сообщений. В таких ситуациях виртуализация сервиса предлагает трансформировать сообщения между форматами поставщика и потребителя. Подобный эффект может быть получен, например, при введении в эксплуатацию новой версии сервиса и изменении XML-схем, определяющих параметры сообщений. Предполагается, что потребители сервиса должны всегда соблюдать ожидаемый поставщиком формат. Но, при изменениях, сложно заставить всех потребителей мгновенно приспособиться.

Типовые функции виртуального сервиса

Виртуальный сервис – наилучшее место реализации некоторых технических условий или обеспечения качества сервиса (QualityOfService):

1. Проверка XML сообщений на корректность формата и соответствие интерфейсу сервиса.
2. Аутентификация и авторизация: идентификация потребителя сервиса и проверка наличия у него прав для вызова сервиса.
3. Расшифровка сообщений и проверка подписи.
4. Балансировка нагрузки и гарантии наличия ресурсов для работы сервиса.
5. Маршрутизация сообщений. Передача сообщений различным реализациям сервиса в зависимости от содержимого сообщений или внешних условий.
6. Мониторинг работы сервиса, производительности, а также проверка предоставления поставщикам требуемых услуг (SLA).

Данные требования изменяются много чаще, чем функциональная логика сервисов.

Механизм виртуализации позволяет реализовать разное качество сервиса (QoS) для разных клиентов, например, HTTP-аутентификацию внутри предприятия и передачу шифрованного XML для внешних клиентов.

Чтобы понять концепцию слабого связывания в SOA, следует прежде всего рассмотреть концепцию слабого связывания (loose coupling) в целом. Следующие утверждения демонстрируют, что такое слабое связывание и каково его значение:

- Сущность связывается, если ее изменение одной взаимодействующей стороной требует изменений другими сторонами (например, модели данных).

- Сущность объявляется, если ее поведение определяется в интерфейсе к сервису, а инициаторы запросов и провайдеры могут взаимодействовать только при совпадении объявленного поведения. К объявляемым аспектам относятся система защиты, транзакционное поведение и качество сервиса (например, время реакции и доставки).

- Сущность преобразовывается, если она объявляется как инициаторами запросов, так и провайдерами сервисов, но инициаторами и провайдерами заявлено различное поведение сущности и инфраструктура обеспечивает определенные возможности преобразования, позволяющие наладить взаимодействие.

- Сущность согласовывается, если как инициатор запросов, так и провайдер сервиса объявляют спектр поведений, которые они могут поддерживать, а посредническая инфраструктура может договориться о согласовании их поведения для каждого взаимодействия.

- Сущность развязывается, если изменения аспекта одной взаимодействующей стороной не требуют соответствующих изменений другими сторонами.

Слабое связывание проявляет себя в парадигме SOA следующим образом:

- Оно помогает организовать уровень абстракции между производителями и потребителями сервисов.
- Оно способствует реализации гибкости в изменении реализации сервисов без воздействия на потребителей сервисов.
- В архитектуре SOA функциональность организуется как набор модульных повторно используемых общих сервисов. Эти сервисы имеют четко определенные интерфейсы, инкапсулирующие ключевые правила доступа к ним. Они также строятся без каких-либо допущений о том, кто будет использовать или потреблять эти сервисы. Таким образом, они слабо связаны с потребителями сервисов.