Аннотированный Манифест СОА

Томаса Ерл о Манифесте СОА, Комментарии и Взляд Внутрь

Сервис-ориентация - это парадигма, которая описывает то, что вы делаете. Сервис-ориентированная архитектура (СОА) - это такой тип архитектуры, который приводит к конечному результату за счет применения сервис-ориентации.

С самого начала было понятно, что это должен быть манифест о двух различных, но довольно тесно связанных между собой понятиях, модели сервис-ориентированной архитектуры и сервис-ориентации, той парадигмы, через которую эта архитектура определена. Формат этого манифеста был смоделирован подобно Манифесту Гибкой Методологии Разработки, Agile Manifesto, ограничивающему содержание до кратких утверждений, выражающих стремления и замыслы, а также руководящие принципы по реализации этих стремлений и замыслов. Такой магнифест не является ни спецификацией, ни эталонной моделью, и даже не статьей–публикацией; так что, не имея возможности обеспечить фактические определения, мы решили добавить эту преамбулу, чтобы объяснить, как и почему в других частях манифеста делаются ссылки на эти термины.

Мы применяем сервис-ориентацию...

Лучше всего рассматривать парадигму сервис-ориентации как метод или подход по реализации специфического конечного состояния [системы], которое далее определено набором стратегических целей и преимуществ. Применяя сервис-ориентацтию, мы формируем компьютерные программы и технологическую архитектуру [системы], таким образом реализовывая это конечное состояние. Это и есть то, что квалифицирует технологическую архитектуру, как сервис-ориентированную.

...чтобы помочь компаниям последовательно достигать устойчивой значимости бизнеса, повышенной гибкости и эффективной стоимости...

Это продолжение преамбулы освещает некоторые из самых очевидных и обычно ожидаемых стратегических преимуществ от сервис-ориентированной вычислительной среды. Понимание этих преимуществ позволяет пролить свет на вышеупомянутое конечное состояние [системы], которое мы намереваемся реализовать в результате применения сервис-ориентации. Гибкость на уровне бизнеса сравнима со способностью компании к быстрой ответной реакции. Чем легче и более эффективно компания может откликаться на изменения бизнеса, тем более эффективна и успешна будет адаптация к влияниям изменений, а

также дальнейшее использование всех тех преимуществ, которые эти изменения принесут.

Сервис-ориентация ставит сервисы в положение элементов ИТ, которые, как ожидают, со временем обеспечат ценность, во много раз превысящую начальные вложения, требуемые для изготовления этих сервисов. Рентабельность в основном имеет отношение к этой ожидаемой прибыли. Во многих отношениях, увеличение рентабельности идет рука-об-руку с увеличением гибкости; если есть больше возможности повторно использовать существующие сервисы, то следовательно потребуется меньше затрат на производство новых сервисов. "Устойчивая" ценность бизнеса определяется долгосрочными целями сервис-ориентации для того, чтобы формировать компьютерные программы как сервисы, с врожденной гибкостью, позволяющей постоянно составлять из них новые конфигурации и развивать их для непрерывно изменяющихся требований бизнеса.

...в соответствии с изменяющимися потребностями бизнеса.

Эти последние шесть слов преамбулы являются ключевыми к пониманию лежащей в основе философии сервис-ориентированной компьютеризации. Потребность удовлетворять постоянные изменения в бизнесе является основой сервис-ориентации и считается фундаментальной обобщающей стратегической целью.

В процессе работы мы установили следующие приоритеты:

Предстоящие заявления устанавливают основной набор ценностей, каждая из которых выражается как приоритет одной перед другой. Цель такой системы ценностей заключается в адресовании трудности выбора, производимого на регулярной основе с целью последовательной реализации стратегических целей и преимуществ сервис-ориентированной компьютеризации.

Значимость бизнеса перед технической стратегией

Как было сказано выше, потребность приспосабливаться к изменениям в бизнесе является всеобъемлющей стратегической целью. Поэтому, основополагающее качество сервис-ориентированной архитектуры и любых программ, решений и эко-систем, которые являются результатом сервисной ориентации, состоит в том, что они являются ориентированными на бизнес. Не технология определяет направление бизнеса, а видение бизнеса диктует использование технологии. Этот приоритет может иметь глубокий цепной эффект в подразделениях ИТ организации. Это вносит изменения практически во все части жизненного цикла разработки ИТ, от планирования и финансирования автоматизированных решений до построения и управления ими. Все остальные ценности и принципы в Минифесте, так или иначе, поддерживают реализацию этой ценности.

Стратегические цели перед выгодами, специфичными для данного проекта

Исторически, многие проекты ИТ фокусировались исключительно на

построении прикладного обеспечения, которое было специально спроектировано для автоматизации тех текущих требований бизнес-процессов, которые были действительными на тот момент. Это удовлетворяло непосредственные (тактические) потребности, однако, по мере разработки всё большего числа таких одноцелевых приложений, среда ИТ заполнялась островами логики и данных, называющихся "изолированными" приложениями. ¬¬¬По мере возникновения новых требований бизнеса либо создавались новые "изолированные" приложения, либо устанавливались каналы интеграции между "изолированными" приложениями. В то время, как количество изменений в бизнесе возрастало, создавались новые каналы интеграции, появлялось всё большее количество "изолированных" приложений, так что вскоре ландшафт среды ИТ стал слишком замысловатым и слишком запутанным, дорогим и медленно развивающимся. Во многих отношениях, сервис-ориентация появилась как ответ на эти проблемы. Эта парадигма является альтернативой к подходу, специфичному для данного проекта, основанному на разработке "изолированных", интегрированных приложений, за счёт непреклонного расположения по приоритетам достижения долгосрочных целей бизнеса. Конечное состояние, поддерживаемое сервис-ориентацией, не имеет "изолированных" приложений. И даже, когда унаследованные ресурсы и "изолированные" приложения существуют в среде с принятой сервис-ориентацией, даже в этом случае конечное состояние - это то состояние, в котором они предельно и разумно согласованы.

Свойственная способность к взаимодействию перед специально достигнутой интеграцией

Для того, чтобы программы могли совместно использовать данные, они должны быть взаимно совместимыми. Если программы не были спроектированы как взаимно совместимые, скорее всего, они не будут взаимодействовать. Чтобы достичь взаимодействия между несовместимыми программами, необходима интеграция. Поэтому интеграция - это усилие, требуемое для достижения взаимодействия несовместимых программ. Хотя часто необходимо это сделать, интеграция по заказу может быть дорогой, занять время и может привести к хрупкой, обременительной в развитии архитектуре. Одна из целей сервис-ориентации – это минимизировать необходимость интеграции по заказу путем формирования программ (внутри данной среды) таким образом, чтобы они были прирождённо совместимыми. Это качество называется свойственной способностью к взаимодействию. Парадигма сервис-ориентации охватывает ряд определенных дизайн принципов, которые направлены в сторону установления свойственной способности к взаимодействию на нескольких уровнях.

Свойственная способность к взаимодействию, как особенность программ внутри данной сферы, это ключ к реализации таких стратегических преимуществ, как рентабельность и гибкость.

Совместно используемые сервисы перед разработками с узкими, специальными целями

Как только-что было объяснено, сервис-ориентация устанавливает подход к проектированию, состоящему из ряда принципов. Будучи применеными в

разумной степени, эти принципы формируют программное обеспечение в единицу сервис-ориентированной логики, которую можно законно называть сервисом. Сервисы оснащены конкретных характеристиками (типа тех, которые позволяют свойственную способность к взаимодействию), которые непосредственно поддерживают ранее описанное целевое состояние. Одной из таких характеристик является инкапсуляция многоцелевой логики, которая может совместно и многократно использоваться в поддержку автоматизации различных бизнес-процессов. Совместно используемый сервис устанавливает себя в качестве такого элемента ИТ, который может обеспечить значимость бизнеса, при этом снижая затраты и усилия на производство новых решений в области автоматизации. Хотя есть ценность и в традиционных, одноцелевых приложениях, отвечающих тактическим требованиям бизнеса, совместно использыемые сервисы обеспечивают большую ценность в достижении стратегических целей сервис-ориентированных компьютерных технологий (что опять-таки включает в себя увеличение рентабельности и гибкости).

Гибкость перед оптимизацией

Это, возможно, является самым широким утверждением приоритетов ценностей и может быть лучше всего рассмотрено в качестве руководящей философии о том, как лучше приоритизировать различные соображения при разработке и развитии индивидуальных сервисов и сервис инвентариев. Оптимизация в первую очередь относится к обеспечению тактического выигрыша путем настройки данного дизайна приложения или ускорения его завершения с целью удовлетворения насущных потребностей. В этом нет ничего нежелательного кроме того, что это может привести к вышеупомянутой "изолированной"среде, если не установить приоритет в отношении гибкости. Например, характеристика гибкости выходит за пределы способностей сервисов к эффективному (и по сути свойственному) разделению использования данных. Чтобы по-настоящему реагировать на постоянно меняющиеся требования бизнеса, сервисы должны быть гибкими в том, как они могут быть объединены и соединены в композиционные решения. В отличие от традиционных распределенных приложений, которые зачастую были относительно статичными, несмотря на то, что они состояли из компонентов, композиции сервисов должны быть разработаны с таким уровнем врожденной гибкости, которая позволяет постоянное изменение. Это означает, что когда существующий бизнес-процесс изменяется или когда появляется новый бизнес-процесс, мы должны иметь возможность добавлять, удалять и расширять сервисы в составе архитектуры композиции с минимальным усилием по интеграции. Именно поэтому сервис композиционность является одним из ключевых принципов дизайна сервис-ориентации.

Эволюционные усовершенствования перед попыткой достичь изначального совершенства

Существует распространённое заблуждение о термине "гибкость" по отношению к сервис-ориентации. Некоторые подходы к проектированию пропагандируют быструю разработку программного обеспечения во имя немедленной выгоды. Такие подходы можно считать "тактической гибкостью", так как они фокусируются на тактических, краткосрочных выгодах. Сервис-ориентация пропагандирует достижение гибкости на организационном или

бизнес-уровне с целью расширения возможностей организации, как единого целого, реагировать на изменения. Эта форма организационной гибкости также может быть рассмотрена как "стратегическая гибкость", т.к. акцент делается на долгострочность, при которой с каждым разработанным программном продуктом мы хотим двигаться в направлении целевого состояния, которое способствует гибкости с долгосрочным стратегическим значением.

Для того, чтобы ИТ предприятия позволило организационную гибкость, она должна развиваться вместе с бизнесом. Как правило, мы не можем предсказать, как бизнес будет эволюционировать с течением времени, и поэтому мы не можем сначала построить идеальные сервисы. В то же время, как правило, большой объем знаний, который уже существует в рамках существующей бизнес-аналитики организации, может быть использован в ходе стадий анализа и моделирования проектов СОА.

Эта информация, наряду с принципами и проверенными методиками сервис-ориентации, может помочь нам выявить и определить набор сервисов, которые отражают то, как бизнес существует и функционирует сегодня, в тоже время будучи достаточно гибкими, чтобы приспосабливаться к тому, как бизнес меняется со временем.

Это значит, что мы ценим значение выше перечисленных понятий справа, однако ещё более ценим понятия слева

Изучая то, как эти ценности расставлены по приоритетам, мы обретаем понимание того, что отличает сервис-ориентацию от других парадигм. Этот тип понимания может помочь ИТ профессионалам в нескольких направлениях. Например, это может помочь установить основные критерии, которые мы можем использовать для определения того, насколько сервис-ориентация совместима с данной организацией или ИТ предприятия. Он может также помочь определить, в какой степени сервис-ориентация может быть или должны быть принята.

Оценка основных ценностей также может помочь нам понять проблемы, которые могут возникнуть для успешного выполнения проектов СОА в определенных условиях. Например, некоторые из ценностей могут вступить в противоречие с установленными убеждениями и предпочтениями. В таком случае, выгоды сервис-ориентации должны быть оценены относительно усилий и влияния, которые может иметь их принятие (не только на технологию, но и на культуру организации и ИТ).

Чтобы помочь решить многие из этих типов задач, были представлены последующие руководящие принципы.

Мы следуем этим принципам:

До этого момента, манифест определил общую концепцию, а также связанный с ней набор основных ценностей. Остальная часть декларации состоит из ряда принципов, которые приводятся в качестве руководства для следования ценностям и реализациям концепций.

Важно иметь в виду, что эти руководящие принципы являются специфичными

для этого манифеста. Существует отдельный набор установленных дизайн принципов, которые включают дизайн парадигму сервис-ориентации и многие другие документированные практики и шаблоны, специфичные для сервис-ориентации и сервис-ориентированной архитектуры.

Уважать социальную и руководящую структуру компании.

Одной из наиболее распространенных ошибок является отношение к СОА адоптации как к технология-ориентированной инициативе. Это почти всегда приводит к неудаче, потому что мы просто не готовы к неизбежным организационным последствиям.

Адоптация сервис-ориентации – это трансформация того, как мы автоматизируем бизнес. Однако, несмотря на планы, которые возможно мы имеем для осуществления такого преобразования, мы должны всегда начинать с понимания и признания организации, её структуры, цели и культуры.

Адоптация сервис-ориентации – это человеческий опыт. Она требует поддержки со стороны власть-имущих, а затем требует, чтобы ИТ адоптировало стратегический стиль мышления. Мы должны полностью признать и планировать этот уровень организационных изменений, с целью получения необходимых долгосрочных обязательств, необходимых для достижения целевого состояния сервис-ориентации. Такого рода соображения не только помогают нам определить, как лучше всего поступить с СОА инициативой, они также помогают нам в определении наиболее подходящего масштаба и подхода к адоптации.

Признать то, что, в конечном счете, СОА требует внесение изменений на многих уровнях.

Существует поговорка, которая гласит: "Успех готовит к возможности". Возможно, урок номер один, извлеченный из проведены до сих пор проектов СОА, является то, что мы должны в полной мере понять, а затем планировать и готовить количество и диапазон изменений, которые возникли в результате адоптации сервис-ориентации. Вот несколько примеров.

Сервис-ориентация изменяет то, как мы строим автоматизированные решения, позиционируя программное обеспечение как средства ИТ с долгосрочной, повторяемой бизнес ценностью. Необходимы первоначальные инвестиции для создания среды, состоящей из таких средств, так же как необходимо постоянное обязательство сохранять и усиливать их значение. Таким образом, с самого начала необходимы изменения того, как мы финансируем, оцениваем и поддерживаем системы в рамках ИТ предприятия.

Кроме того, поскольку сервис-ориентация представляет сервисы, которые позиционируются как ресурсы предприятия, будут внесены изменения в то, как мы владеем различными частями системы и регулируем их дизайн и эксплуатацию, не говоря уже об изменениях в инфраструктуре, необходимую для обеспечения непрерывной масштабируемости и надежности.

Масштабы внедрения СОА могут варьироваться. Держать усилия в этом направлении под контролем и в разумных рамках.

Общим мифом стало то, что в целях реализации стратегических целей сервис-ориентированных вычислений, сервис-ориентация должна быть принята в масштабах предприятия. Это означает введение и применение дизайн стандартов и отраслевых стандартов по всему ИТ предприятия для того, чтобы создать инвентарий внутренне-совместимых сервисов в масштабах всего предприятия. Хотя в этой идее нет ничего плохого, это нереальная цель для многих организаций, особенно больших ИТ предприятий.

Наиболее подходящий масштаб для любого данного усилия по адоптации СОА должен быть определен в результате планирования и анализа, связанного с прагматическими соображениями, таких как вышеупомянутое воздействие на организационные структуры, сферы полномочий и культурные изменения.

Такого рода факторы помогают нам определить контролируемый масштаб адоптации. Но для любого усилия по адоптации, которая приводит к среде, прогрессирующей ИТ предприятия в направлении достижения желаемого стратегического конечного состояния, этот масштаб должен иметь смысл. Другими словами, он должно иметь перекрестный смысл, чтобы коллекции сервисов могли быть разработаны в отношении друг друга в рамках заранее установленных границ. Иными словами, мы хотим создать "континенты сервисов" а не ужасные "острова сервисов".

Эта концепция создания независимо владеемых и управляемых сервис инвентариев в рамках одного и того же ИТ предприятия снижает риск, который обычно объясняется разработкой "колоссальных" проектов СОА, и кроме того снижает воздействие как организационных, так и технологических изменений (так как воздействие ограничено до сегментированного и управляемого масштаба). Кроме того, такой подход позволяет поэтапную адоптацию, когда домейны сервис инвентариев могут быть установлены по одному.

[Программные] продукты и стандарты сами по себе не дадут вам СОА и не применят сервис-ориентацию за вас.

Этот принцип касается двух отдельных, но очень тесно связаных мифов. Первый, что вы можете найти путь в СОА через приобретение современных технологий, а второй - это предположение о том, что адоптация отраслевых стандартов (например, XML, WSDL, SCA и т.д.) естественно приведет к сервис-ориентированной архитектуре.

Разработчикам и группам отраслевых стандартов отдаётся должное за разработку современных инновационных технологий по открытым структурам и платформам. Всё, от сервис виртуализации до облачного компьютинга и грид-компьютинга, помогло продвинуть потециал построения сложных и комплексных сервис-ориентированных решений. Тем не менее, ни одна из этих технологий не является исключительной для СОА. Вы можете так же легко построить изолированную системы в облаке, как вы можете это сделать на своих частных серверах.

Нет такого понятия, как "СОА в коробке", потому что для успешного внедрения сервис-ориентированные технологической архитектуры, сервис-ориентация должна быть с успехом внедрена, а это в свою очередь требует, чтобы все, что мы проектируем и строим, определялось единым направлением, видением и бизнес требованиями.

СОА может быть реализована путём применения различных технологий и стандартов.

Сервис-ориентация является технологически нейтральной и независимой от поставщика парадигмой. Сервис-ориентированная архитектура является технологически нейтральной и независимой от поставщика архитектурной моделью. Сервис-ориентированный компъютинг можно рассматривать как специализированную форму распределенного компъютинга. Поэтому сервис-ориентированные решения могут быть построены с использованием почти любых технологий и стандартов, подходящих для распределенного компъютинга.

В то время, как некоторые технологии (особенно те, которые основаны на отраслевых стандартах) могут существенно увеличить потенциал применения некоторых дизайн принципов сервис-ориентации, существует реальная возможность выполнить требования бизнеса, что в конечном итоге определяет выбор наиболее подходящих технологий и отраслевых стандартов.

Установить единый комплекс стандартов и положений предприятия, основанных на отраслевых стандартах, общественных стандартах и стандартах, принятых "по- факту".

Отраслевые стандарты представляют собой непроприетарные технические спецификации, которые, среди прочего, помогают установить последовательные основные характеристики технологической архитектуры (например, транспорт, интерфейс, формат сообщения и т.д.). Однако, использование отраслевых стандартов не гарантирует, что сервисы будут врождённо способны к взаимодействию.

Для того, чтобы две компьютерные программы были полностью совместимы, должны иметь место дополнительных конвенции (например, модели данных и положения). Именно поэтому организации ИТ должны устанавливать и следить за выполнением соблюдения дизайн стандартов. Неудача в стандартизации и регулировании стандартизации сервисов в данной области начнет рвать ткань взаимодействия, на котором основана реализация многих стратегических преимуществ.

Этот принцип не только выступает за использование дизайн стандартов предприятия, он также напоминает нам о том, что, когда это возможно и практически осуществимо, пользовательские дизайн стандарты должны быть основаны на и включать в себя стандарты, уже используемые в промышленности и обществе в целом.

Преследовать внешнее единообразие, одновременно позволяя внутреннее многообразие.

Федерации может быть определена как объединение множества разрозненных элементов. Допуская, что каждый элемент самостоятельно регулируется изнутри, все элементы согласны с тем, чтобы присоединиться к общему, единому фронту. Фундаментальной частью сервис-ориентированной архитектуры является введение федеративного слоя конечной точки, абстрагирующего детали реализации сервисов при публикации набора

конечных точек, которые представляют индивидуальные сервисы в данной области как единое целое. Выполнение этого правила в целом предполагает достижение единства, основанного на сочетании отраслевых стандартов и дизайн стандартов. Последовательность в этом единстве сервисов является ключом к реализации свойственная способности к взаимодействию.

Федеративный слой конечной точки также помогает расширить возможности для изучения вариантов разнообразия поставщиков. Например, один сервис может быть построен на совершенно иной платформе, чем другой сервис. До тех пор, пока эти сервисы поддерживают совместимые конечные точки, управление их реализацией может оставаться независимым. Это не только подчеркивает то, что сервисы могут быть построены с использованием различных форм реализации (например, EJB,. NET, SOAP, REST и т.д.), но это также подчеркивает то, что при необходимости.различные посреднические платформы и технологии могут быть использованы вместе.

Обратите внимание, что за этот тип разнообразия надо платить. Этот принцип сам по себе не выступает в роли диверсификации - он просто рекомендует, чтобы мы позволили диверсификацию, когда это оправдано, так что "лучшие в своем классе" технологии и платформы могут использоваться для максимального выполнения бизнес-требований.

Идентифицировать сервисы путём взаимодействия с заинтересованными представителями бизнеса и технологии.

Для того, чтобы технологические решения стали бизнес-ориентированными, технология и бизнес должны быть синхронизированы. Таким образом, еще одной целью сервис-ориентированных вычислений является выравнивание в одну линию технологии и бизнеса. Это согласование изначально выполняется в ходе анализа и моделирования процессов, которые обычно предшествуют фактической разработке и поставке сервисов. Критическим элементом для проведения сервис-ориентированного анализа является то, что специалисты бизнеса и технологий работают рука об руку, чтобы выявить и определить сервисы-кандидаты. Например, бизнес специалисты могут помочь точно определить функциональный контекст, относящийся к бизнес-ориентированным сервисам, а технические эксперты могут обеспечить практические данные для обеспечения того, что детализация и определение концептуальных сервисов остаются реальными в связи с их возможными средами реализации.

Максимизировать использование сервиса путём рассмотрения текущей и будущей сферы применения.

Масштабы данного проекта СОА может быть в масштабах всего предприятия или они могут быть ограничены частью предприятия. Независимо от масштаба, установливается граница, включающая инвентарий сервисов, которые должны быть концептуально смоделированы, прежде чем они могут быть разработаны. Моделируя многочисленные сервисы относительно друг друга, мы по существу созданием скетч сервисов, которые мы в конечном итоге будем строить. Такое моделирование важно при попытке выявить и определить сервисы, которые могут совместно использоваться в различных решениях. Существуют различные методики и подходы, которые могут быть использованы для

выполнения этапов сервис-ориентированного анализа. Тем не менее, общая нить между всеми ними заключается в том, что функциональные границы сервисов должны быть нормализованы для избежания дублирования. Даже в этом случае нормированные сервисы не обязательно станут высококлассными сервисами многократного использования. Тут вступают в игру другие факторы, такие как уровень детализации сервиса, автономия, управление состоянием, масштабируемость, компонуемость, и то, в какой степени сервис логика является достаточно общей, чтобы ее можно было эффективно использовать повторно.

Такого рода соображения, руководствуясь опытом бизнеса и технологии, дают возможность определить сервисы, которые отвечают современным требованиям использования, при этом будучи гибкими для адаптации к будущим изменениям.

Контролировать, чтобы сервисы удовлетворяли требованиям и целям бизнеса.

Как и со всем остальным, сервисы могут быть использованы неправильно. В процессе наращивания и управления портфелем сервисов, необходимо проверять и измерять их использование и эффективность выполнения требований бизнеса. Современные инструменты предоставляют различные средства контроля пользованием сервисов, но существуют нематериальные элементы, которые также должны быть приняты во внимание для обеспечения того, чтобы сервисы использовались не только потому, что они имеются в наличии, но чтобы убедиться, что они действительно выполняют требования бизнеса и соответствуют ожиданиемому.

Это особенно верно с совместно используемыми сервисами, которые поддерживают несколько зависимостей. Совместно используемые сервисы не только требуют адекватной инфраструктуры для обеспечения масштабируемости и надежности всех решений, которые повторно используют их, но они также должны быть разработаны с большой осторожностью, чтобы обеспечить, что их функциональный контекст никогда не был искажен.

Развивать сервисы и их организацию в ответ на их реальное использование.

Этот руководящий принцип непосредственно связан с утверждением "Эволюционные усовершенствования перед попыткой достичь изначального совершенства", а также с общей целью поддержки выравнивания в одну линию бизнеса и технологий.

Мы никогда не можем гадать, когда дело доходит до определения уровня детализации сервиса, набора функций, которые сервисы должны выполнить, или, как сервисы должны быть организованы в композиции. На основании любой степени анализа, который мы можем выполнить вначале, данному сервису будет присвоен определенный функциональный контекст и он будет содержать одну или несколько функциональных возможностей, которые вероятно включат его в одну или несколько композиций сервисов. По мере изменения бизнес требований и обстоятельств реального мира, может стать необходимым усилить, расширить переработать или даже заменить сервис.

Дизайн принципы сервис-ориентации строят врождённую гибкость в сервис архитектуру так что, будучи программным обеспечением, сервисы являются устойчивыми и адаптивными к изменениям и изменяются в ответ на использование в реальном мире.

Разделять различные аспекты системы, которые изменяются различными темпами.

Что делает монолитные и изолированные системы негибкими это то, что изменения могут оказать значительное влияние на их существующее использование. Именно поэтому часто проще создать новый изолированные приложения, чем увеличивать или расширять существующие.

Смысл разделения проблем (известной теории разработки программного обеспечения) заключается в том, что большая проблема может быть решена более эффективно, когда она раскладывается на множество меньших проблем или вопросов. При применении сервис-ориентации в разделении проблем, мы создаем соответствующие логические блоки для решения отдельных проблем, что позволяет нам агрегировать блоки для решения большей проблемы в дополнение к предоставлению нам возможности объединять их в различные конфигурации с тем, чтобы решать другие проблемы. Помимо содействия повторному использованию сервисов, этот подход представляет многочисленные слои абстракции, которые помогают защитить сервисные системы от воздействия изменений. Эта форма абстракции может существовать на различных уровнях. Например, если наследованные ресурсы, инкапсулированные одним сервисом, должны быть изменены, влияние этого изменения может быть смягчено, если сервис может сохранить свою первоначальную конечную точку и функциональное поведение. Другим примером является разделение агностик логики от не-агностик логики. Первый тип логики обладает высоким потенциалом повторного использования, если эта логика многоцелевая и вероятность её изменения невысока. Не-агностик логика, с другой стороны, как правило представляет собой одноцелевые части логики родительских бизнес-процессов, которые зачастую являются более нестабильны. Разделение этих соответствующих типов логики в различные сервис слои представляет дополнительную абстракцию, которая делает возможным повторное использование сервисов, в тоже время защищая сервисы, а также решения, которые используют их, от воздействия изменений.

Сокращать неявные зависимости и публиковать все явные зависимости с целью увеличения прочности и снижения последствий изменения.

Одним из самых известных дизайн принципов сервис-ориентации является слабая связь сервисов. Как сервис архитектура внутренне структурирована и как сервисы связанны с потребляющими их программами (которые могут включать другие сервисы), все это сводится к зависимостям, формирующимся на индивидуально движущихся частях, которые входят в состав сервис архитектуры.

Слои абстракции облегчают эволюционное изменение за счет локализации последствий изменения в контролируемых средах. Например, в рамках сервис архитектуры, могут быть использованы сервис фасады для абстрагиривания

частей реализации в целях сведения к минимуму уровня зависимостей реализации.

С другой стороны, опубликованные технические сервис контракты должны раскрывать зависимости, которые должны формировать сервис потребители для того, чтобы взаимодействовать с сервисами . За счет уменьшения внутренних зависимостей, которые могут повлиять на эти технические контракты, когда происходит изменение, мы избегаем распространения влияния этих изменений на зависимых сервис потребителей.

Организовать каждый сервис в соответствии с единой, управляемой единицей функциональности на каждом уровне абстракции.

Каждый сервис требует четко определенный функциональный контекст, определяющий логику, принадлежащую и непринадлежащую функциональной границе сервиса. Определение масштаба и степени детализации этих функциональных границ сервиса является одной из важнейших функций в процессе жизненного цикла сервиса. Сервисы с крупной степенью функциональной детализации могут быть излишне негибкими, чтобы стать эффективными, особенно, если они, как ожидается, будет многоразовыми в использовании. С другой стороны, сервисы со слишком мелкой степенью функциональной детализации могут повлиять на инфраструктуру так, что сервис композиции должны будут состоять из большего количествах членов композиции. Определение оптимального баланса между функциональным масштабом и степенью функциональной детализации требует сочетания знаний бизнеса и технологий, а также требует понимания того, как сервисы в рамках данных границ связаны друг с другом. Многие из руководящих принципов, изложенных в этом манифесте, помогут в принятии решения в поддержку позиционирования каждого сервиса, как элемента ИТ, способного содействовать направлению ИТ предприятий к этой конечной цели, при которой реализованы стратегические преимущества сервис-ориентированных вычислений. Однако, в конечном итоге, достижение бизнес стоимости реального мира, от концепции до реализации могократного использования, всегда будет диктовать эволюционный путь любого элемента сервис-ориентированной функциональности.

Благодарности: Я написал содержимое это документа в выходные дни 21-22 ноября 2009 года для следующего поколения книги СОА, которая в данный момент еще находится в разработке. Я хотел бы поблагодарить Prentice Hall за разрешение открыто опубликовать данный документ на этом сайте в качестве дополнения к первоначальному Манифесту СОА. Я также хотел бы поблагодарить членов рабочей группы, а также тех людей из сообщества SOAPatterns.org, которые предоставили комментарии и оказали помощь с этими аннотациями. Многие из членов рабочей группы уже опубликовали свои собственные статьи, блоги, и документы о манифесте СОА, и я призываю всех вас к этим публикациям за дальнейшими комментариями и идеями.

- Томас Ерл

Переводчик

Леонид Феликсон

Original SOA Manifesto PDF About the Translators