

**МОДЕЛИРОВАНИЕ РЕСУРСОВ С СОСТОЯНИЕМ  
ПОСРЕДСТВОМ WEB-СЛУЖБ**

Версия 1.1

03/05/2004

<http://www-106.ibm.com/developerworks/library/ws-resource/ws-modelingresources.pdf>

\* \* \*

**MODELING STATEFUL RESOURCES  
WITH WEB SERVICES**

*Ian Foster (Globus Alliance/Argonne National Laboratory) (Редактор)*  
*Jeffrey Frey (IBM) (Редактор)*  
*Steve Graham (IBM) (Редактор)*  
*Steve Tuecke (Globus Alliance/Argonne National Laboratory) (Редактор)*  
*Karl Czajkowski (Globus Alliance / USI ISI)*  
*Don Ferguson (IBM)*  
*Frank Leymann(IBM )*  
*Martin Nally (IBM)*  
*Igor Sedukhin (Computer Associates International)*  
*David Snelling (Fujitsu Laboratories of Europe)*  
*Tony Storey (IBM)*  
*William Vambenepe (Hewlett-Packard)*  
*Sanjiva Weerawana (IBM)*

## Уведомление об авторском праве

© Копирайт Computer Associates, Inc., Fujitsu Limited, Hewlett-Packard Development, International Business Machines Corporation и The University Chicago 2003, 2004. Все права защищены.

Настоящим гарантируется права бесплатно и без авторских отчислений копировать и воспроизводить документ "Modeling Stateful Resources with Web Services" (далее «Документ») на любых носителях, при условии, что во ВСЕ созданные вами копии данного Документа или его части вы включаете:

1. ссылку или URL на данный Документ в этом месте.
2. это уведомление об авторском праве как показано в данном Документе.

ДАННЫЙ ДОКУМЕНТ ПРЕДОСТАВЛЯЕТСЯ «КАК ЕСТЬ», И COMPUTER ASSOCIATES INTERNATIONAL, FUJITSU LIMITED, IBM, HEWLETT-PACKARD DEVELOPMENT COMPANY И UNIVERSITY OF CHICAGO (ДАЛЕЕ ВСЕ ВМЕСТЕ НАЗЫВАЕМЫЕ «КОМПАНИИ») НЕ ДАЮТ НИКАКИХ ПОДТВЕРЖДЕНИЙ ИЛИ ГАРАНТИЙ, ВЫРАЖЕННЫХ ЯВНО ИЛИ ПОДРАЗУМЕВАЕМЫХ, В ТОМ ЧИСЛЕ, НО НЕ ТОЛЬКО, ГАРАНТИИ ТОВАРНОГО КАЧЕСТВА, ПРИГОДНОСТЬ ДЛЯ КОНКРЕТНОЙ ЦЕЛИ, НЕУЩЕМЛЕНИЯ ИНТЕРЕСОВ ИЛИ ПРАВ; НИ О ТОМ, ЧТО СОДЕРЖИМОЕ ДАННОГО ДОКУМЕНТА ПОДХОДИТ ДЛЯ ЛЮБОЙ ЦЕЛИ; НИ ТОГО, ЧТО ИСПОЛЬЗОВАНИЕ ЭТОЙ ИНФОРМАЦИИ НЕ НАРУШАЕТ ПАТЕНТЫ, КОПИРАЙТЫ, ТОРГОВЫЕ МАРКИ ИЛИ ДРУГИЕ ПРАВА ЛЮБОЙ ТРЕТЬЕЙ СТОРОНЫ.

КОМПАНИИ НЕ БУДУТ НЕСТИ ОТВЕТСТВЕННОСТЬ ЗА ЛЮБЫЕ ПРЯМЫЕ, КОСВЕННЫЕ, УМЫШЛЕННЫЕ, СЛУЧАЙНЫЕ ИЛИ ЛОГИЧЕСКИ ВЫТЕКАЮЩИЕ УБЫТКИ, ВОЗНИКШИЕ ВСЛЕДСТВИИ ИЛИ В СВЯЗИ С ЛЮБЫМ ИСПОЛЬЗОВАНИЕМ ИЛИ ВОСПРОИЗВЕДЕНИЕМ ДАННОГО ДОКУМЕНТА.

Названия и торговые марки Компаний НЕ могут использоваться никоим образом, в том числе для рекламы или популяризации, касающихся данного Документа или его содержимого, иначе как без особого предварительного письменного разрешения. Право на копият данного Документа всегда остается у Компаний.

Никакие другие права не гарантируются косвенно, явно или как-либо иначе.

ФРАГМЕНТЫ ЭТОГО МАТЕРИАЛА БЫЛИ ПОДГОТОВЛЕНЫ КАК ОТЧЕТ О РАБОТЕ, ФИНАНСИРУЕМОЙ IBM CORPORATION AT UNIVERSITY OF CHICAGO'S ARGONNE NATIONAL LABORATORY. НИ АВТОРЫ, НИ ПРАВИТЕЛЬСТВО СОЕДИНЕННЫХ ШТАТОВ, НИ КАКОЕ-ЛИБО ЕГО АГЕНТСТВО, НИ УНИВЕРСИТЕТ ЧИКАГО, НИ IBM, НИ ОДИН ИЗ ИХ СОТРУДНИКОВ ИЛИ РУКОВОДИТЕЛЕЙ, НИКАКИЕ ДРУГИЕ ВЛАДЕЛЬЦЫ КОПИРАЙТОВ ИЛИ СОАВТОРЫ НЕ ДАЮТ НИКАКИХ ГАРАНТИЙ, ПРЯМО ИЛИ КОСВЕННО, НЕ БЕРУТ НА СЕБЯ ЮРИДИЧЕСКУЮ ОТВЕТСТВЕННОСТЬ ИЛИ ОБЯЗАТЕЛЬСТВА ЗА ТОЧНОСТЬ, ПОЛНОТУ ИЛИ ПОЛЕЗНОСТЬ ЛЮБОЙ ИНФОРМАЦИИ, ПРИБОРА, ПРОДУКТА ИЛИ ПРОЦЕССА, КОТОРЫЕ БЫЛИ УПОМЯНУТЫ, И НЕ УТВЕРЖДАЕТ, ЧТО ИХ ИСПОЛЬЗОВАНИЕ НЕ НАРУШИТ ПРАВА ЧАСТНОЙ СОБСТВЕННОСТИ. ИМЕЮЩИЕСЯ ЗДЕСЬ ССЫЛКИ НА ЛЮБОЙ КОНКРЕТНЫЙ КОММЕРЧЕСКИЙ

ПРОДУКТ, ПРОЦЕСС ИЛИ СЛУЖБУ ПО КОММЕРЧЕСКОМУ НАЗВАНИЮ, ТОРГОВОЙ МАРКЕ, ПРОИЗВОДИТЕЛЮ ИЛИ КАК-ЛИБО ИНАЧЕ НЕ ОБЯЗАТЕЛЬНО ОЗНАЧАЕТ ИЛИ ПОДРАЗУМЕВАЕТ ОДОБРЕНИЕ, РЕКОМЕНДАЦИЮ ИЛИ ПООЩРЕНИЕ КОМПАНИЕЙ ИВМ, ПРАВИТЕЛЬСТВОМ СОЕДИНЕННЫХ ШТАТОВ ИЛИ ЛЮБЫМ ЕГО АГЕНТСТВОМ ЭТОГО ИЛИ ЛЮБОГО ДРУГОГО ВЛАДЕЛЬЦА КОПИРАЙТА ИЛИ СОАВТОРОВ. ТОЧКА ЗРЕНИЯ И МНЕНИЯ АВТОРОВ, ВЫРАЖЕННЫЕ ЗДЕСЬ, НЕ ОБЯЗАТЕЛЬНО СООТВЕТСТВУЮТ ИЛИ ОТРАЖАЮТ МНЕНИЕ ИВМ, ПРАВИТЕЛЬСТВА СОЕДИНЕННЫХ ШТАТОВ ИЛИ ЛЮБОГО ЕГО АГЕНТСТВА, ИЛИ ОРГАНИЗАЦИИ, В КОТОРОЙ МОГУТ РАБОТАТЬ АВТОРЫ.

Эта рукопись частично была создана в University of Chicago as Operator of Argonne National Laboratory (“Argonne”) в рамках Контракта No. W-31-109-ENG-38 Министерства энергетики США. Правительство США сохраняет за собой и другими, действующими от его имени, оплаченную, не эксклюзивную, не подлежащую отмене международную лицензию на указанную статью, предусматривающую воспроизведение, подготовку производных материалов, распространение копий для широкой публики, а также на ее публичное воспроизведение и демонстрацию самим Правительством или от его имени.

## Аннотация

Архитектура Web-служб широко используется как средство структурирования взаимодействий распределённых программных служб. В настоящее время необходимо провести дальнейшую стандартизацию для того, чтобы способствовать дополнительной интероперабельности между службами. Одна из важных областей, в которой такая стандартизация необходима, охватывает вопросы организации взаимодействий с *ресурсами, обладающими состоянием (stateful resources)*. В этой статье мы рассматриваем конструкции, которые дают возможность Web-службам обрабатывать состояние, используя при этом некую последовательную и интероперабельную технологию. Мы предлагаем так называемый *WS-Resource* (далее **WS-ресурс**) *подход* для описания и реализации связей между Web-службой и одной или более поименованными компонентами состояния. При этом подходе мы моделируем состояние как ресурс, обладающий состоянием, и устанавливаем правила для связи между Web-службами и такими ресурсами в терминах *шаблона неявного ресурса* – набора соглашений в рамках технологии Web-служб, в частности, спецификации WS-Addressing. Мы описываем WS-ресурс в терминах ресурса с состоянием и Web-службы, связанной с ним. Мы также описываем подход для обеспечения доступа к свойствам WS-ресурса посредством интерфейса его Web-службы и для управления временем жизни WS-ресурса.

## Статус

Эта публикация является первичным представлением проекта и имеет своей единственной целью инициировать его обсуждение и развитие. Компании надеются учесть ваши замечания и предложения в ближайшем будущем. Компании не дают каких-либо гарантий и подтверждений относительно данной публикации.

## Содержание

<b>Аннотация</b> .....	4
<b>Статус</b> .....	4
<b>Содержание</b> .....	5
<b>1 Введение</b> .....	6
<b>2 Истоки Web-служб</b> .....	7
2.1 Что такое Web-служба .....	7
2.2 Среда Web-служб .....	8
2.3 Краткая систематизация состояния и служб .....	10
2.4 Реализации без состояния, интерфейсы с состоянием .....	10
<b>3 Моделирование состояния в Web-службах</b> .....	11
3.1 Моделирование состояния: ресурсы с состоянием.....	12
3.2 Шаблон неявного ресурса .....	13
3.3 WS-ресурс и спецификация WS-Addressing.....	14
3.4 Мощность множества отношений WS-ресурса.....	16
3.5 Инкапсуляция WS-ресурса.....	17
<b>4 Время жизни WS-ресурса</b> .....	18
4.1 Создание WS-ресурса .....	18
4.2 Уникальное имя WS-ресурса .....	18
4.3 Уничтожение WS-ресурса .....	20
<b>5 Свойства WS-ресурса</b> .....	20
5.1 Документ свойств WS-ресурса .....	21
5.2 Композиция свойств WS-ресурса .....	22
5.3 Доступ к значениям свойств WS-ресурса .....	23
<b>6 WS-ресурс и ACID-свойства</b> .....	23
<b>7 Безопасность WS-ресурса</b> .....	24
<b>8 Заключение</b> .....	25
<b>9 Благодарности</b> .....	25
<b>10 Литература</b> .....	25

## 1 Введение

Архитектура Web-служб [WS-ARCH] определяет ориентированную на службы модель распределённого компьютеринга, в которой службы взаимодействуют, обмениваясь XML-документами. Базовые элементы архитектуры Web-служб определяют синтаксис для обмена информацией. В настоящее время ведутся различные исследования и разработки, направленные на развитие этой базовой архитектуры посредством дополнительных соглашений, позволяющих взаимодействующим службам, используя стандарты, демонстрировать более сложную функциональность, включая аутентификацию, транзакции и надёжную передачу сообщений [Web Services].

В этой статье мы предлагаем набор соглашений, направленных на формализацию взаимодействий с *состоянием*. Мотивация для этих новых предложений основана на понимании того факта, что есть много способов представления состояния в Web-службах, но не существует установленного соглашения, которое поддерживало бы интероперабельность между Web-службами и их взаимодействие с ресурсами, обладающими состоянием. Даже те *реализации* Web-служб, которые обычно рассматриваются как *не имеющие состояния (stateless)*, часто используются для оперирования с состоянием, то есть значениями данных, которые сохраняются и изменяются в результате взаимодействий с этой Web-службой. Например, интерактивная система резервирования авиабилетов должна поддерживать состояние, описывающее статус рейсов, резервирование билетов клиентами, и саму систему: её текущее размещение, загрузку и работоспособность. Интерфейсы Web-служб, которые позволяют запрашивать статус рейсов, резервировать авиабилеты, изменять статус сделанных заказов и управлять самой системой резервирования, обязательно должны обеспечивать доступ к такому состоянию.

Суть того, что мы называем *WS-ресурс подходом*, состоит в моделировании состояния в виде *ресурсов с состоянием* и кодификации взаимосвязей между Web-службами и ресурсами с состоянием в терминах *шаблона неявного ресурса* – набора соглашений в рамках технологий Web-служб, в частности, таких как XML, WSDL, и WS-Addressing [WS-Addressing]. Мы описываем WS-ресурс в терминах ресурса с состоянием и Web-службы, связанной с ним. Мы также описываем подход к обеспечению доступа к свойствам WS-ресурса средствами интерфейса его Web-службы, и управления временем жизни WS-ресурса.

Эта работа содействует идущим в сообществе Web-служб обсуждениям вопроса, следует ли использовать Web-службы для представления состояния и как это сделать. В этих дебатах одна точка зрения состоит в том, что “Web-службы не имеют понятия состояния” [Vogels], и что “Взаимодействия с Web-службами не имеют состояния; при этом в качестве одного из способов моделирования взаимодействий с состоянием предлагается учет контекста” [Parastatidis]. В то же время другие, включая авторов, утверждают, что критически важная роль, которую играет состояние при распределённом компьютеринге, требует, чтобы состояние учитывалось в рамках архитектуры Web-служб [Physiology]. Конструкция WS-ресурса может помочь примирить эти две позиции, показывая как можно прямо формализовать отношения между web-службами и состоянием, опираясь на другие спецификации Web-служб.

Мы рассматриваем в этой работе концепции и конструкции, которые лежат в основе WS-ресурса подхода, не отображая его в терминах конкретных обменов

сообщениями Web-служб. Конкретное отображение, в виде набора спецификаций, называемого WS-Resource Framework, предлагается в работе [WSRF].

Концепция WS-ресурса подхода сформировалась в контексте деятельности Global Grid Forum’s Open Grid Services Infrastructure (OGSI) Working Group [Physiology, GSI-Spec]. Сопоставительный анализ WS-ресурса подхода и его инфраструктуры с OGSI содержится в другой работе [OGSI-Refactor].

## 2 Истоки Web-служб

Прежде чем мы определим средства, с помощью которых Web-службы могут быть связаны с ресурсами, обладающими состоянием, нам необходимо пояснить несколько терминов и понятий.

### 2.1 Что такое Web-служба

Термин *Web-службы* появился в 2000 году в связи с введением таких технологий как SOAP, WSDL и UDDI. Одновременно с этим вошел в обращение термин *ориентированная на службы архитектура (service-oriented architecture - SOA)* [Tao], который использовался для описания общего подхода к построению слабо связанных распределённых систем, компоненты которых минимально согласованы между собой. Многочисленные публикации и практические разработки существенно продвинули понимание этих понятий участниками сообщества информационных технологий.

В то время как отдельные технологии, например, SOAP, WSDL и UDDI достаточно хорошо определены, пока не удаётся выработать повсеместно принятое определение термина *Web-служба*. Рабочая группа “W3C Web services Architecture working group” предлагает следующее определение [WS-Arch]:

*Web-служба - это система программного обеспечения, предназначенная для поддержки интероперабельного взаимодействия между компьютерами в сети. Её интерфейс определен в удобном для компьютерной обработки формате (а именно, на языке WSDL). Другие системы взаимодействуют с Web-службой так, как это предписано ее описанием, и используют для этого SOAP-сообщения, обычно транспортируемые по протоколу HTTP совместно с другими Web-стандартами.*

Ориентированная на службы архитектура определяет распределённую систему в виде совокупности агентов, известных как службы, действующих согласованно посредством обменов сообщениями. Прочитируем ещё одно определение из [WS-Arch]:

*SOA представляет собой распределённую систему особого типа, в которой агентами являются “службы”. Служба – это программный агент, который выполняет строго определённую операцию (то есть “обеспечивает некую услугу”) и может быть вызван вне контекста более крупной прикладной программы. Другими словами, хотя служба может быть реализована как экспонирование свойств более крупной прикладной программы, пользователям этой услуги необходимо всего лишь знать описание интерфейса данной службы. “Службы” имеют адресуемые в сети интерфейсы и общаются, используя стандартные протоколы и форматы данных.*

Соблазнительно интерпретировать предложение “пользователям этой услуги необходимо всего лишь знать описание интерфейса службы” как следствие того, что поведение службы определяется только обменами сообщениями, поддерживаемыми службой. Однако определения интерфейсов служб часто подразумевают существование ресурсов с состояниями, которые используются и обрабатываются в процессе реакции на запрос к Web-службе. Например, система резервирования авиабилетов может поддерживать следующие три типа сообщений:

- *GetReservation*, которое возвращает XML-документ, описывающий выполненный заказ.
- *AddFlightSegment*, которое добавляет новый рейс для резервирования.
- *RemoveFlightSegment*, которое удаляет рейс из списка резервируемых.

Этот интерфейс предполагает, что служба управляет набором документов, описывающих сделанные заказы. Программист может также извлечь идентификатор заказа из сообщений, объявленных в интерфейсе службы. Главное положение настоящей работы состоит в том, что желательно представлять такие отношения между Web-службами и состоянием явно и стандартным образом, вместо того, чтобы полагаться на интуитивные предположения. Мы утверждаем, что такое открытое и стандартизированное представление повысит интероперабельность служб, упростит определение интерфейсов новых служб и даст возможность создавать более совершенные средства обнаружения, управления службами и инструментарий для их разработки.

## 2.2 Среда Web-служб

Начнём с рассмотрения нескольких важных аспектов Web-службы, которые требуют дополнительного обсуждения. Они представлены в виде компонент среды Web-служб, показанной на Рис.1 и объясняются ниже.

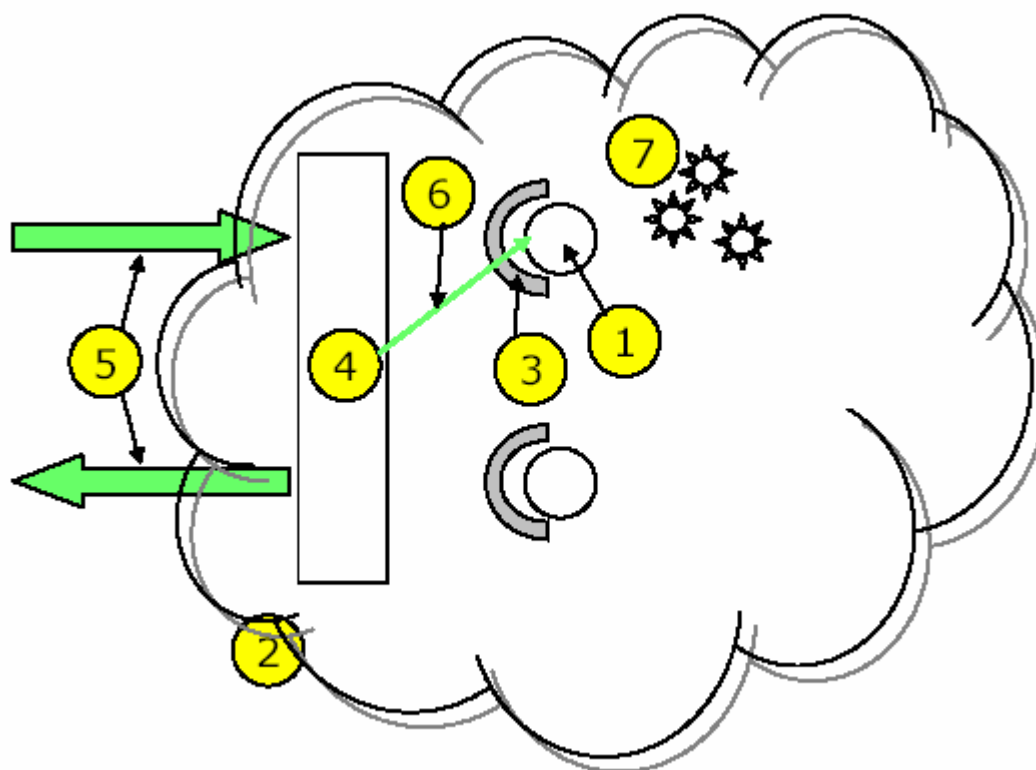


Рисунок 1 – Компоненты, связанные с Web-службами

Здесь Web-служба (помечена 1) представляет собой программную компоненту, выполняющую некоторую функцию, например, оформляющую заказ на покупку. Web-службы могут проводить операции, которые имеют доступ к состоянию, хранящемуся на других ресурсах системы. Web-служба – это компонента, которая устанавливается в некоторой исполнительной среде (2), например, Web-сервера приложений, такого как IBM WebSphere или JBoss. Эта среда обеспечивает размещение Web-службы и диспетчеризацию предназначенных ей сообщений. Исполнительная среда может также предоставлять Web-службе другие услуги, например, такие как обеспечение безопасности или обработка транзакций.

Интерфейс Web-службы (3), представленный на языке описания Web-служб, например, WSDL [WSDL 1.1], определяет функциональность Web-службы в терминах набора операций, которые могут быть вызваны другими сущностями в распределённой системе, называемыми *заказчиками службы (service requestors)*. Каждая операция описывается в терминах конструкций обмена сообщениями, которые определяют как формат сообщения, используемого для вызова операции, так и форматы возможных ответных сообщений, в том числе сообщений об ошибках.

В исполнительной среде имеется средство (4) обработки сообщения, которое может получать сообщения (5) от заказчиков. Эта компонента способна поддерживать один или более сетевых транспортных протоколов, таких как HTTP, SMTP или ИОП, и на практике её часто называют *крайняя точка (endpoint)*, поскольку именно эта компонента доступна распределённой компьютерной фабрике по конкретному сетевому адресу. Компонента крайней точки также осуществляет перевод конверта сообщения в формат, понимаемый службой (например, преобразует XML-посылку в набор Java-объектов), и (6) переправляет

сообщение реализации целевой службы, которая идентифицируется адресом (URL) и другими частями сообщения. Следует отметить, что хотя Web-службы создаются (или, как мы иногда говорим, устанавливаются) в исполнительной среде по адресу конкретной крайней точки (то есть уникального в сети имени сообщения, обрабатываемого исполнительной средой), тем не менее, каждая Web-служба сама уникально идентифицируется посредством адреса, представляющего собой объединение адреса её крайней точки, по которой установлена служба, и некоторого дополнительного идентификатора, специфического для данной Web-службы.

Реализация Web-службы принимает сообщение, обрабатывает его, при этом, возможно взаимодействуя с другими службами и ресурсами с состоянием (7) – и, если это предусмотрено при данном обмене, форматирует и отправляет ответное сообщение. Многие Web-службы сами играют роль заказчика услуги, иницируя обмен сообщениями с другими Web-службами.

### ***2.3 Краткая систематизация состояния и служб***

Строгая систематизация интерфейса, состояния и экземпляров выходит за рамки настоящей работы. Тем не менее, чтобы подготовить контекст для следующего ниже материала, мы дадим краткий обзор возможных связей состояния с интерфейсом.

1. *Не имеющая состояния служба (stateless service)* при обмене сообщениями, не имеет доступа и не использует никакую информацию, не содержащуюся во входном сообщении. Простым примером этого является служба, которая только упаковывает и распаковывает информацию в документах, содержащихся в сообщениях, поступающих к данной службе.
2. *Диалоговая служба (conversational service)* реализует последовательности операций, так что результат одной операции зависит от результата предыдущей и/или подготавливает последующую операцию. При этом каждое сообщение в логическом потоке сообщений используется службой для определения дисциплины работы этой службы. Дисциплина выполняемой операции определяется результатом обработки предшествующих сообщений в их логической последовательности. Многие интерактивные Web-сайты реализуют эту модель, используя HTTP-сеансы и «ключики» (**cookies**).
3. *Служба, которая оперирует с ресурсами, обладающими состоянием*, имеет доступ и обрабатывает набор логических ресурсов с состоянием (документов), опираясь на сообщения, которые она отправляет и получает.

В данной работе мы рассматриваем вопросы, касающиеся третьей модели. Мы считаем, что подходы, основанные на передаче контекста исполнения в заголовках сообщений, подобные тем, что введены спецификациями WS-Coordination, WS-Context и WS-Policy, обеспечивают средства, с помощью которых можно реализовать вторую модель.

### ***2.4 Реализации без состояния, интерфейсы с состоянием***

Необходимо подчеркнуть, что, говоря в третьей модели о *службе, которая работает с ресурсами, обладающими состоянием*, мы имеем в виду службу, чья реализация (**implementation**), оперирует с динамическим состоянием, то есть состоянием, за которое служба несёт ответственность между обменами сообщениями с её заказчиками.

Служба, которая оперирует с ресурсами, *обладающими состоянием*, может быть описана как не имеющая состояния (“stateless”), если она делегирует ответственность за управление состоянием другой компоненте, например, базе данных или файловой системе. Отсутствие состояния при реализации службы имеет своей целью повышение степени надёжности и масштабируемости: Web-службу без состояния можно повторно запустить после отказа, не заботясь при этом, какова история её предыдущих взаимодействий, а при изменении нагрузки можно создать (а впоследствии уничтожить) и новые копии такой Web-службы. Поэтому отсутствие состояния, вообще говоря, рассматривается как хорошее инженерное решение при реализации Web-служб.

Следствием отсутствия состояния является то, что любое динамическое состояние, необходимое для выполнения данной операции обмена сообщениями, должно быть:

- обеспечено *явно* внутри запросного сообщения либо прямо (значением), либо косвенно (ссылкой) и/или
- поддержано *неявно* в рамках другой системной компоненты, с которой может взаимодействовать Web-служба.

Конечно, Web-служба может быть реализована так, что она может работать со статическим состоянием (например, с заранее конфигурированными ссылками на другие системные компоненты).

Третья модель характеризует службу, которая *оперирует с ресурсами, обладающими состоянием*, и допускает, что реализация Web-службы без состояния будет часто использовать и обновлять динамическое состояние, поддерживаемое в других системных компонентах, например, в базе данных. В таких случаях уникальное имя элемента(ов) состояния может либо быть передано в сообщении запроса, либо поддерживаться как статические данные Web-службой. Интерфейс, предлагаемый такой Web-службой, конечно, обладает состоянием в том смысле, что его поведение определено с учётом нижележащего состояния.

### 3 Моделирование состояния в Web-службах

Мы уже отмечали, что даже тогда, когда сама *реализация* Web-службы может быть описана как не обладающий состоянием процессор сообщений, обмена сообщениями, которые он реализует (как это определено его *интерфейсом*), часто предназначены для обеспечения возможности доступа и/или обновления состояния, поддерживаемого другими системными компонентами, будь то база данных, файловые системы или другие объекты.

Учитывая ту существенно важную роль, которую играет доступ к состоянию во многих интерфейсах Web-служб, важно идентифицировать и стандартизировать шаблоны, используя которые можно представлять и обрабатывать состояние, так чтобы облегчить построение и использование интероперабельных служб.

С этой целью мы предлагаем подход к моделированию ресурсов с состоянием в инфраструктуре Web-служб, основанный на конструкции, которую мы называем *WS-ресурс (WS-Resource)*. Более конкретно, мы предлагаем средства, с помощью которых:

- WS-ресурс формируется из Web-службы и ресурса с состоянием (в этом разделе);
- ресурс с состоянием используется при обработке сообщений Web-службой (в этом разделе);
- WS-ресурсы могут создаваться и уничтожаться (в разделе 4); и
- определение ресурса с состоянием может быть ассоциировано с описанием интерфейса Web-службы так, чтобы дать возможность стандартным способом опрашивать состояние WS-ресурса, а также обращаться и обновлять состояние WS-ресурса посредством обменов сообщениями с данной Web-службой (в разделе 5).

### 3.1 Моделирование состояния: ресурсы с состоянием

Термин *состояние* воспринимается не вполне ясно и в принципе может охватывать многие аспекты вычислительной системы, от содержимого записи, хранящейся в базе данных, до времени поиска или даже температуры дисководов. Наше внимание сосредоточено на том, что называется *ресурсом с состоянием*, который по определению:

- имеет конкретный набор данных состояния, представленный в виде XML-документа;
- имеет определённое время жизни; и
- известен под каким-то именем, и доступен для обработки одной или несколькими Web-службами.

Примерами системных компонент, которые могут моделироваться как ресурсы с состоянием, являются файлы в файловой системе, строки в реляционной базе данных и инкапсулированные объекты, например такие, как сущности Entity Enterprise Java beans. Ресурс с состоянием также может быть совокупностью или группой других ресурсов с состоянием.

Следует подчеркнуть, что это определение касается того, как ресурс с состоянием моделируется, а не того, как он реализован или представлен. Конкретное состояние ресурса может быть реализовано в виде фактически существующего XML-документа, который хранится в памяти, в файловой системе, в базе данных или в некотором XML-репозитории. Альтернативно, тот же самый ресурс с состоянием может быть реализован в виде некой логической проекции данных, *динамически* сконструированной или сформированной из объектов языка программирования (например, такого как J2EE EJB Entity Bean) или из данных, полученных в результате обращения по частному коммуникационному каналу к обычной системе данных или приложению.

Для данного типа ресурса с состоянием может быть создано и уничтожено множество независимых экземпляров. Как мы покажем в Разделе 4, экземпляр ресурса с состоянием может быть создан с помощью Web-службы, называемой *фабрикой ресурсов с состоянием*.

В Разделе 5 мы покажем, что ресурс с состоянием определяется посредством глобальной декларации элемента (Global Element Declaration - GED) XML в заданном пространстве имён. Эта декларация определяет тип корневого элемента XML-документа ресурса и, следовательно, тип самого ресурса с состоянием.

Сущность, создающая экземпляр ресурса с состоянием, может присвоить ему уникальное имя. Приложения, использующие этот ресурс, также могут присваивать ему дополнительные уникальные имена (псевдонимы). Конкретная форма уникального имени ресурса с состоянием может конфиденциально использоваться одной или несколькими реализациями Web-служб, чтобы идентифицировать его при выполнении данного обмена сообщениями. Использование *идентификатора ресурса с состоянием* при обработке сообщения Web-службы обсуждается в следующем разделе.

### 3.2 Шаблон неявного ресурса

Определив способ моделирования элементов состояния ресурсов с состоянием, вернёмся к вопросу о том, как заказчики Web-служб могут ссылаться на ресурсы с состоянием. Для того чтобы описывать конкретный вид отношения между Web-службой и одним или более ресурсами с состоянием, мы предлагаем использовать конструкцию, называемую *шаблон неявного ресурса (implied resource pattern)*.

*Шаблон неявного ресурса* указывает на механизм, используемый для связывания ресурса с состоянием с выполнением обменов сообщениями, реализуемым Web-службой.

Термин *неявный* используется потому, что при выполнении запроса ресурс с состоянием, связанный с данным обменом, трактуется как неявный ввод. Используя слово *неявный*, мы хотим подчеркнуть, что заказчик услуги *не указывает* в теле запроса идентификатор этого ресурса *в качестве явного параметра*. Вместо этого, ресурс неявно ассоциируется с выполнением обмена. Это может быть осуществлено либо статически, либо динамически. Мы говорим, что ресурс с состоянием ассоциируется с Web-службой статически, если связывание происходит при установке Web-службы. Мы говорим, что ресурс с состоянием ассоциируется с Web-службой динамически, когда связывание происходит во время выполнения обмена сообщениями. При динамическом связывании идентификатор ресурса служит указанием того, что этот неявный ресурс, может быть инкапсулирован в предусматриваемой WS-Addressing-спецификацией ссылке на крайнюю точку, которая используется для адресации целевой Web-службы.

Мы используем термин *шаблон*, чтобы показать, что отношения между Web-службами и ресурсами с состоянием подчиняются набору соглашений, касающихся использования существующих технологий Web-служб, в частности XML, WSDL и WS-Addressing [WS-Addressing].

Определенная в спецификации WS-Addressing ссылка на крайнюю точку представляет собой XML-сериализацию глобального сетевого *указателя* Web-службы. Этот указатель может быть возвращен в качестве результата обращения к фабрике Web-служб с запросом о *создании* нового ресурса, или как результат запроса на поиск в реестре ресурсов, или же, как результат выполнения некоторого специфического Web-приложения.

Спецификация WS-Addressing стандартизирует конструкцию ссылки на крайнюю точку, используемую для представления адреса Web-службы, установленной в данной сетевой крайней точке. Ссылка на крайнюю точку кроме адреса крайней точки Web-службы может содержать другие метаданные, ассоциированные с Web-службой, например такие, как информацию описания службы и свойства ссылки, которые помогают

определить контекстуальное использование ссылки на крайнюю точку. Свойства ссылки на крайнюю точку играют важную роль в шаблоне неявного ресурса.

Отметим, что для обеспечения доступа к ресурсам с состоянием можно использовать и другие шаблоны. Например, Web-служба могла бы управлять уникальным именем ресурса, в виде статического состояния службы, тем самым, устраняя необходимость передачи этого имени в WS-Addressing-ссылке на крайнюю точку. Выбор такой конструкции подразумевает взаимно-однозначное отображение множества ссылок на крайние точки Web-служб на множество ресурсов с состоянием и, тем самым, необходимость в уникальной крайней точке Web-службы для каждого ресурса с состоянием.

### 3.3 WS-ресурс и спецификация WS-Addressing

Мы называем *WS-ресурсом* компоненту, которая представляет собой композицию Web-службы и связанного с ней ресурса с состоянием, участвующего в шаблоне неявного ресурса.

Рассмотрим соглашения, касающиеся использования спецификации WS-Addressing в шаблоне неявного ресурса. На Рисунке 2 WS-Addressing-ссылка на крайнюю точку удовлетворяет соглашениям о шаблоне неявного ресурса.

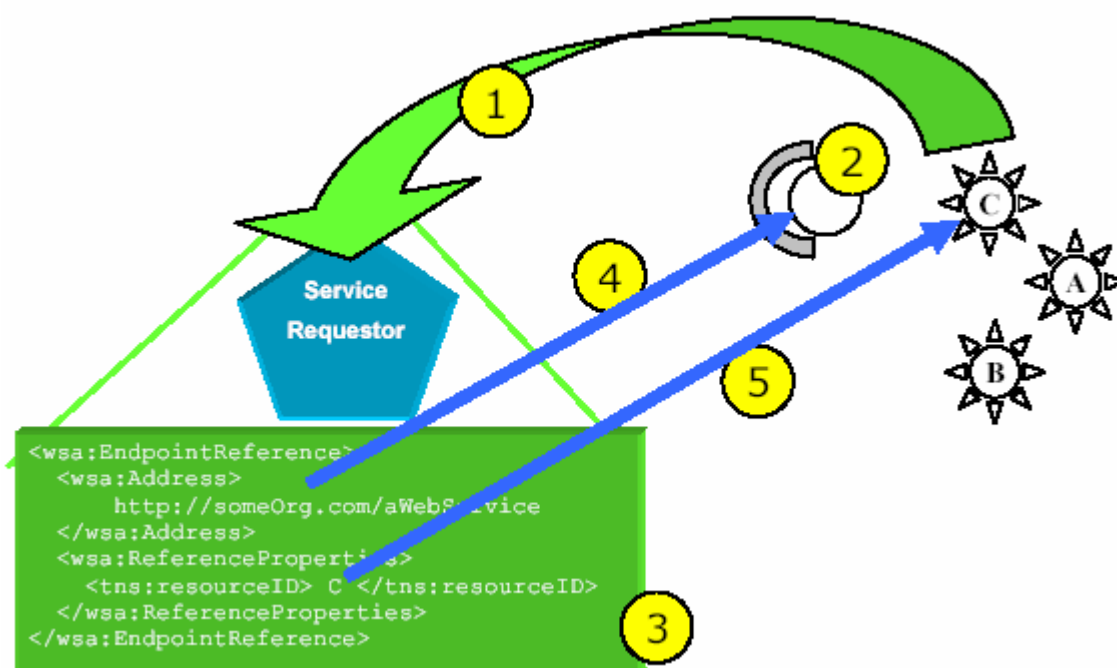


Рисунок 2 – Ссылка на крайнюю точку, содержащая идентификатор ресурса с состоянием.

Ссылка на крайнюю точку (на Рисунке 2 помечена 1) возвращается к заказчику в ответ на некоторый запрос, посланный Web-службе (2). Предположим, что обработка этого запроса привела к созданию ресурса с состоянием “С”. Мы говорим, что эта Web-служба представляет собой *фабрику явного WS-ресурса*. Она является таковой, поскольку ответное сообщение содержит ссылку на крайнюю точку WS-ресурса, который сформирован из заново созданного ресурса с состоянием и ассоциированной с ним Web-службы. В ссылке на крайнюю точку содержится информация, которая на основе шаблона

неявного ресурса выражает отношение между Web-службой и заново созданным ресурсом с состоянием.

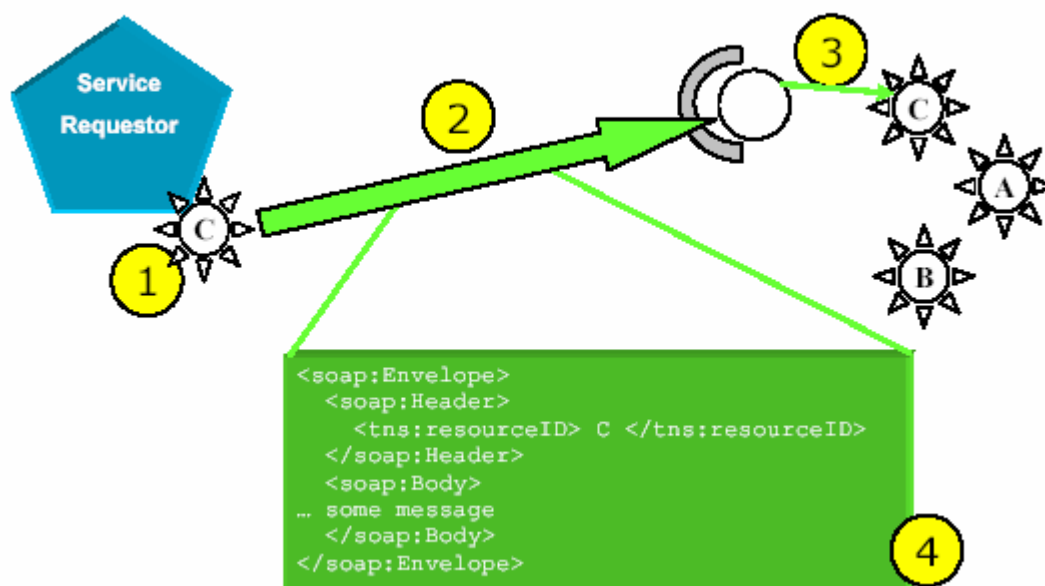
Ссылка на крайнюю точку (3) содержит две важных компоненты:

- <wsa: Address>-компонента (4), которая ссылается на сетевой адрес Web-службы для конкретного типа транспорта, (часто URL, в случае использования транспорта, базирующегося на протоколе HTTP). Это тот же самый адрес, который появился бы внутри элемента <port> WSDL-описания Web-службы.
- <wsa:ReferenceProperties>-компонента может содержать XML-сериализацию идентификатора ресурса с состоянием, понимаемого Web-службой, адресуемой ссылкой на крайнюю точку. Этот идентификатор ресурса с состоянием представляет собой ресурс с состоянием, который должен быть использован при выполнении запроса (5). Ссылка на крайнюю точку, содержащая идентификатор ресурса с состоянием, является *квалифицированной ссылкой на крайнюю точку WS-ресурса (WS-Resource qualified endpoint reference)*.

В XML-сериализации идентификатора ресурса с состоянием содержится специальный XML-элемент для представления информации об идентификаторе ресурса, которая должна игнорироваться заказчиком службы. Приложения заказчика службы не должны проверять или пытаться интерпретировать содержимое идентификатора ресурса с состоянием. Идентификатор ресурса с состоянием понятен только Web-службе и используется ею для идентификации WS-ресурса, связанного с ресурсом с состоянием, который нужен для выполнения поступившего запроса, так, как это принято в реализации данной Web-службы.

Идентификатор ресурса с состоянием должен идентифицировать уникальный ресурс с состоянием, который будет использоваться при выполнении поступившего запроса. Не требуется, чтобы значение этого идентификатора было уникальным вообще, но оно должно быть пригодным для Web-службы настолько, чтобы она могла использовать его для однозначной идентификации имеющегося в виду WS-ресурса, связанного с ресурсом с состоянием. Другими словами область действия идентификатора ресурса с состоянием должна быть уникальной внутри области действия Web-службы и может быть уникальной вне этой области. Кроме того, различные идентификаторы внутри области действия Web-службы могут ссылаться на один и тот же WS-ресурс.

С точки зрения заказчика ссылка на крайнюю точку представляет собой *указатель (pointer)* на WS-ресурс, включающий Web-службу, которая в дальнейшем, возможно, будет настроена на обмен сообщениями с конкретным ресурсом с состоянием. Заказчик услуги должен понимать, что ссылка на крайнюю точку указывает на WS-ресурс. Другими словами заказчик услуги должен распознать, что ссылка на крайнюю точку является квалифицированной ссылкой на крайнюю точку WS-ресурса. Использование квалифицированной ссылки на крайнюю точку WS-ресурса иллюстрируется на Рисунке 3.



**Рисунок 3 – Использование квалифицированной ссылки на крайнюю точку WS-ресурса.**

Приложения заказчика услуги должны использовать ссылку на крайнюю точку (помеченную 1 на Рисунке 3), чтобы посылать сообщения (2) идентифицированной Web-службе (3). Поскольку в ResourceProperties-компоненте ссылки на крайнюю точку WS-ресурса содержится идентификатор ресурса с состоянием, то всякий запрос, направленный службе, использующей эту ссылку на крайнюю точку, должен включать идентификатор ресурса с состоянием.

Заметим, что WS-ResourceProperties-компонента WS-Addressing сообщения обрабатываются в соответствии с конкретным способом связывания. WS-Addressing-спецификация предусматривает, что ResourceProperties-компонента ссылки на крайнюю точку должна являться частью любого сообщения, посланного Web-службе, идентифицированной ссылкой на крайнюю точку. Каждый тип WSDL-связывания должен декларировать как должны представляться дочерние элементы ResourceProperties-элемента в сообщениях, использующих такое связывание. Например, спецификация WS-Addressing устанавливает, что ResourceProperties-элементы должны появиться в сообщении в виде элементов SOAP-заголовка. На Рисунке 3 компонента, помеченная (4) показывает использование SOAP-заголовка для передачи идентификатора ресурса с состоянием, в данном случае представляющего ресурс с именем “С”. Web-служба (3) затем извлекает идентификатор ресурса из SOAP-сообщения и использует его для определения местоположения ресурса, необходимого для выполнения поступившего запроса.

### **3.4 Мощность множества отношений WS-ресурса**

Необходимо расширить представление о том, каковы в WS-ресурсе отношения между Web-службами и ресурсами с состоянием. В частности, нам необходимо описать мощность множества отношений между ресурсом с состоянием и Web-службой как в аспекте уровней типов, так и уровней экземпляров.

Web-служба может вообще не выполнять или выполнять обмены с несколькими ресурсами, которые определены как экземпляры документа свойств ресурса. В обычной ситуации одна Web-служба, привязанная к конкретной крайней точке, связана с несколькими отдельными ресурсами с состоянием. В некоторых случаях число ресурсов, используемых через одну Web-службу, может оказаться чрезвычайно большим как, например, в случае взаимодействия с файловой системой через интерфейс Web-службы, которая моделирует каждый файл как отдельный WS-ресурс, связанный с ресурсом с состоянием.

На уровне типа, элемент `portType` на языке WSDL 1.1, определяющий интерфейс с Web-службой, может быть связан с более чем одним документом свойств ресурса с состоянием. Стандартные средства для формулирования такой ассоциации описаны ниже. Любая Web-служба, которая реализует этот `portType`, является по определению Web-службой, ассоциированной с ресурсом с состоянием, определённым документом свойств ресурса.

Один документ свойств ресурса с состоянием может быть связан с несколькими элементами `portType`. Это отношение “один-ко-многим” на уровне типов позволяет связывать отдельные ресурсы с состоянием с множеством Web-служб, каждая из которых реализует собственный интерфейс.

На уровне экземпляра, ресурс с состоянием может быть ассоциирован с одной или несколькими Web-службами. Отношение “один-ко-многим” между экземпляром ресурса с состоянием и Web-службой может быть использовано для того, чтобы разрешить множество сетевых протоколов или сетевых крайних точек для обработки запросов к WS-ресурсу, или определить различные интерфейсы Web-служб, позволяющие классифицировать и разбивать на подмножества сообщения, которые воздействуют на ресурс с состоянием.

### **3.5 Инкапсуляция WS-ресурса**

Полезность инкапсуляции данных хорошо известна. Строгая инкапсуляция гарантирует, что инкапсулируемые данные могут быть доступны только через четко определённые операции. Эти операции обеспечивают пункт управления, принудительно реализующий политику, приводящую к увеличению согласованности, управляемости и целостности данных. Инкапсуляция данных облегчает использование данных пользователю, скрывая от него детали реализации, и тем самым уменьшает зависимость от реализации, а также повышает гибкость разработки.

Мы предлагаем такую конструкцию шаблона неявного ресурса, которая облегчает варьирование степеней инкапсуляции Web-службой ресурсов с состояниями. Самая высокая степень инкапсуляции обеспечивается тогда, когда полный доступ к состоянию данного ресурса может быть выполнен путём обменов сообщениями, реализуемыми с использованием единственного типа WS-ресурса. С другой стороны ресурс с состоянием может быть частью определения множества типов WS-ресурсов.

Дополнительный вид инкапсуляции используется для выражения ассоциации ресурса с состоянием с Web-службой. В этом случае идентификатор ресурса задаётся самой Web-службой, а не заказчиком услуги. Использование идентификатора ресурса как

части ссылки на крайнюю точку исключает необходимость для заказчика знать уникальное имя и локализацию ресурса с состоянием, инкапсулированного Web-службой.

#### **4 Время жизни WS-ресурса**

Время жизни WS-ресурса определяется как период между его созданием и уничтожением. Фактически существующие механизмы, с помощью которых создаётся и уничтожается WS-ресурс, зависят от реализации. Тем не менее, мы выделяем следующие три аспекта времени жизни WS-ресурса, которые мы обсуждаем в следующих трех подразделах, а именно:

1. Создание WS-ресурса путем использования фабрики WS-ресурсов,
2. Присваивание и использование идентификатора ресурса с состоянием и
3. Уничтожение WS-ресурса.

##### **4.1 Создание WS-ресурса**

WS-ресурс может быть создан с помощью некоторого специального механизма или альтернативно (как мы обсуждаем здесь) путём запроса к фабрике WS-ресурсов. *Фабрикой WS-ресурсов* является любая Web-служба, способная породить WS-ресурс. Порождение WS-ресурса включает такие акты, как создание нового ресурса с состоянием, присваивания этому новому ресурсу уникального имени и оформление связи между новым ресурсом и ассоциированной с ним Web-службой. Ответное сообщение, поступающее от фабрики WS-ресурсов, включает квалифицированную ссылку на крайнюю точку WS-ресурса, содержащую идентификатор ресурса с состоянием, который адресует новый ресурс. Хотя фабрика может сообщать такую ссылку на новый WS-ресурс и другим способом, например, размещая квалифицированную ссылку на крайнюю точку WS-ресурса в реестре для последующего поиска.

Заметим, что может быть много типов Web-служб (например, реестры ресурсов), которые возвращают квалифицированные ссылки на крайние точки WS-ресурсов в их ответных сообщениях. Тем не менее, если только запрос к Web-службе не привёл к фактическому созданию WS-ресурса, адресуемого в возвращаемой квалифицированной ссылке на крайнюю точку WS-ресурса, переданный запрос не считается операцией фабрики WS-ресурсов.

Заметим также, что описанная выше фабрика WS-ресурсов является полезным образцом Web-службы, но не единственной стандартной операцией. Эта модель может быть запрограммирована в ряде различных операций Web-служб, которые могут, например, создавать один или много WS-ресурсов.

##### **4.2 Уникальное имя WS-ресурса**

Мы рассмотрим и сопоставим роль и использование уникального имени WS-ресурса с двух точек зрения:

1. С профессиональной позиции разработчика WS-ресурса, и
2. С широкой потребительской позиции заказчика, которому передаётся ссылка на крайнюю точку WS-ресурса.

Напомним, что, как утверждалось в Разделе 3.3, каждый ресурс с состоянием имеет, по крайней мере, одну форму уникального имени, которая однозначно идентифицирует его в составе WS-ресурса. Это уникальное имя МОЖЕТ использоваться как “*идентификатор ресурса с состоянием*”, который играет особую роль, как часть ссылки на WS-ресурс. Идентификатор ресурса с состоянием размещается в разделе свойств ссылок WS-Addressing-ссылки на крайнюю точку. В дальнейшем эта ссылка на крайнюю точку называется *квалифицированной для WS-ресурса*. Квалифицированная ссылка WS-ресурса на крайнюю точку затем может быть сделана доступной для других сущностей в распределённой системе, которые могут в последующем использовать эту ссылку для прямых обращений к WS-ресурсу.

Для Web-службы, с которой ассоциирован ресурс с состоянием, идентификатор ресурса, передаваемый в составе запросного сообщения, имеет существенное значение. Реализация Web-службы понимает содержимое идентификатора ресурса, которое зависит от реализации, и может использовать эту информацию для того, чтобы идентифицировать ресурс, который должен быть использован при выполнении запроса.

Заказчик услуги, который получает доступ к квалифицированной ссылке на крайнюю точку WS-ресурса, не должен проверять или пытаться интерпретировать содержимое идентификатора ресурса. Некорректной считается даже попытка заказчика услуги сравнить содержимое двух идентификаторов. С позиции заказчика услуги содержимое идентификатора ресурса внутри ссылки на крайнюю точку не имеет смысла.

Итак, если идентификатор ресурса с состоянием не может использоваться как общедоступная форма уникального имени ресурса, то что же тогда с точки зрения заказчика должно быть общедоступным именем ресурса с состоянием как компоненты WS-ресурса? Короткий ответ состоит в том, что семантическое значение уникального имени ресурса с состоянием и средства, с помощью которых он определён и экспонирован заказчику, зависят от реализации Web-службы. В настоящее время ещё не приняты спецификации Web-служб, которые поддерживали бы определение уникального имени ресурса с состоянием. Нет и каких либо определений средств, с помощью которых уникальное имя ресурса с состоянием предоставляется заказчику.

Экспонировать или нет уникальное имя ресурса с состоянием заказчику, это является свойством конкретной реализации Web-службы. Тем не менее, мы полагаем, что многие Web-службы будут предоставлять возможность получения уникального имени ресурса с состоянием как компоненты WS-ресурса. Уникальное имя должно быть переносимым и ограниченным в пространстве имен значением. Переносимость важна, поскольку она позволяет передавать уникальное имя от одного приложения другому. Ограничение пространством имён важно, поскольку оно даёт возможность исключить неоднозначность во множестве уникальных имен, которые могут происходить от различных источников.

Нам представляется, что общий подход к экспонированию уникального имени ресурса с состоянием как компоненты WS-ресурса будет состоять в трактовке уникального имени как одного или нескольких свойств состояния ресурса, выраженных в документе ресурсных свойств WS-ресурса. Этот подход позволит заказчику непосредственно запрашивать этот документ, имея в виду свойства, необходимые для представления уникального имени ресурса с состоянием как компоненты WS-ресурса. Если уникальное имя экспонируется в виде одного или более свойств WS-ресурса, то

Web-служба должна гарантировать защиту этих свойств от записи. Конечно, было бы некорректно позволить заказчику изменять уникальное имя ресурса с состоянием.

В качестве другой возможности, Web-служба может реализовать специфический для данного приложения обмен сообщениями, предназначенными для обеспечения доступа к уникальному имени ресурса с состоянием как компоненты WS-ресурса. Мы ожидаем, что во многих приложениях проявится потребность использовать сообщения, связанные с выявлением уникального имени ресурса с состоянием. Некоторые такие сообщения могут обеспечивать поиск уникального имени, а некоторые обеспечивать сравнение ресурсов с состоянием.

### 4.3 Уничтожение WS-ресурса

Заказчик, отправляющий запрос к фабрике WS-ресурсов, который влечёт создание нового WS-ресурса, обычно будет заинтересован в этом новом WS-ресурсе в течение некоторого конечного периода времени. После истечения такого периода времени, этот WS-ресурс, возможно, следует ликвидировать, для того чтобы связанные с ним системные ресурсы могли быть снова использованы.

Определение конкретных интерфейсов, используемых для поддержки ликвидации WS-ресурсов, выходит за пределы данной статьи. Тем не менее, мы можем описать некоторые общие требования.

Заказчик, который хочет уничтожить WS-ресурс, использует соответствующую квалифицированную ссылку на крайнюю точку WS-ресурса, чтобы послать запрос на ликвидацию Web-службе, идентифицируемой этой ссылкой на крайнюю точку. Идентификатор ресурса с состоянием, содержащийся в ссылке на крайнюю точку, используется для того, чтобы идентифицировать этот ресурс, а стало быть, и тот WS-ресурс, который должен быть ликвидирован. Поступление ответа на запрос о ликвидации представляет собой точку синхронизации между заказчиком и Web-службой, получающей запрос на ликвидацию. До поступления ответа любой последующий обмен сообщениями со службой, использующей идентификатор ликвидируемого ресурса, должен привести к сообщению об ошибке. Это сообщение указывает, что WS-ресурс неизвестен, причём без какого-либо комментирования обстоятельств отказа, которые могли иметь место ранее.

Мы можем также определить сообщения, используемые для установки и обновления планируемого времени уничтожения WS-ресурса с тем, чтобы выполнять это в определённый момент времени в тех случаях, когда заказчик не может или не хочет осуществлять ликвидацию WS-ресурса явно.

## 5 Свойства WS-ресурса

Теперь мы обсудим средства, которые дают заказчику возможность рассматривать и модифицировать тип и значения характеристик состояния WS-ресурса, через интерфейс его Web-службы. Основные принципы здесь следующие:

- WS-ресурс имеет *XML-документ свойств ресурса*, определённый XML-схемой.
- Заказчики служб могут выяснить тип WS-ресурса, просматривая с помощью стандартных средств WSDL его определение типа порта (portType).

- Заказчики служб могут, используя средства обмена сообщениями между Web-службами, читать, модифицировать и опрашивать XML-документ, характеризующий состояние WS-ресурса.

Мы используем термин *свойство ресурса (resource property)*, чтобы указывать на отдельную компоненту состояния WS-ресурса. Мы называем XML-документ описывающий тип ресурса с состоянием в составе WS-ресурса *документом свойств WS-ресурса (WS-Resource properties document)*. В документе свойств WS-ресурса каждое его свойство представлено в виде XML-элемента.

### 5.1 Документ свойств WS-ресурса

Документ свойств WS-ресурса рассматривается как проекция фактического состояния WS-ресурса. Этот документ служит для определения структуры, на которую могут быть направлены сообщения клиента. Любая операция, которая обрабатывает свойство ресурса через документ свойств WS-ресурса, должна отражаться в фактической реализации состояния WS-ресурса.

Документ свойств WS-ресурса описывается с использованием XML-схемы. А именно, документ свойств WS-ресурса представляется как XML-декларация глобального элемента (GED) в некотором XML-пространстве имён. Например, рассмотрим ресурс с состоянием “С”, упомянутый в предыдущих разделах. Если состояние “С” содержит три компоненты, именуемые p1, p2 и p3, то его документ свойств ресурса названный “ExampleResourceProperties” должен быть описан следующим образом:

```
<xs:schema
  targetNamespace="http://example.com/ResourcePropertiesExample"
  xmlns:tns="http://example.com/ResourcePropertiesExample"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  ...
  ...>

  <xs:element name="p1" type= .../>
  <xs:element name="p2" type= .../>
  <xs:element name="p3" type= .../>

  <xs:element name="ExampleResourceProperties"
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="tns:p1" />
        <xs:element ref="tns:p2" />
        <xs:element ref="tns:p3" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  ...
</xs:schema>
```

Заказчики могут получить и ознакомиться с определением XML-схемы документа свойств WS-ресурса, которая представляет тип ресурса с состоянием “С” с помощью различных средств, включая обмен сообщениями, предусмотренный спецификацией [WS-MetaDataExchange].

Однако, как заказчик службы может узнать, что GED, названный “ExampleResourceProperties”, определяет документ свойств WS-ресурса, связанного с этой Web-службой? Описание документа свойств WS-ресурса для Web-службы включено в WSDL-определение интерфейса Web-службы. Декларация документа свойств WS-ресурса связана с WSDL определением типа порта (portType) посредством использования стандартного атрибута, resourceProperties, как это показано в следующем примере:

```
<wsdl:definitions
  targetNamespace="http://example.com/ResourcePropertiesExample"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl"
  xmlns:xs="http://www.w3.org/2002/XMLSchema"
  xmlns:wsrp="http://www.ibm.com/xml/stdwip/web-services/
    ws-resourcesProperties"
  xmlns:tns="http://example.com/ResourcePropertiesExample"
...>
...
<wsdl:types>
  <xs:schema>
  <xs:import
    namespace="http://example.com/ResourcePropertiesExample">
    schemaLocation="..." />
  </xs:schema>
</wsdl:types>
<wsdl:portType name="SomePortTypeName"
  wsrp:resourceProperties="tns:ExampleResourceProperties">
  <operation name="...
...
</wsdl:portType>
...
</wsdl:definitions>
```

Это определение типа порта, вместе с ассоциированным документом свойств ресурса, полностью описывает тип WS-ресурса.

## 5.2 Композиция свойств WS-ресурса

Web-службы позволяют построить новый интерфейс как композицию нескольких уже существующих интерфейсов. Используя WSDL 1.1 такую композицию можно получить с помощью копирования-вставки операций, определенных как составные части типов портов, используемых при композиции. Например, операции, определенные в примере определения типа порта “foo”, могут быть скомбинированы с операциями, определенными в различных стандартах и спецификациях, для того чтобы получить окончательный, полный набор обмена сообщениями, которые должны быть реализованы Web-службой.

В дополнение к возможности создавать композиционные интерфейсы, конструктор может также агрегировать свойства WS-ресурса, определённые в документах свойств WS-ресурсов различных типов портов, для того чтобы получить окончательный, полный документ свойств WS-ресурса, объявленный с помощью окончательного типа порта. Используя атрибут xs:ref, такую композицию документа свойств WS-ресурса можно завершить, подключив дополнительные декларации XML-элементов так, как это иллюстрируется в следующем примере.

```
<xs:element name="ExampleResourceProperties">
```

```
<xs:complexType>
  <xs:sequence>
    <xs:element ref="tns:p1" />
    <xs:element ref="tns:p2" />
    <xs:element ref="tns:p3" />

    <xs:element ref="xxxx:SomeAdditionalResourceProperty"
      xmlns:xxxx= ... />

  </xs:sequence>
</xs:complexType>
</xs:element>
```

Этот документ свойств WS-ресурса был сконструирован путём объединения элементов свойств ресурсов исходного документа свойств WS-ресурса для ресурса с состоянием “C” и элемента свойств ресурса (SomeAdditionalResourceProperty), определённого в некотором другом пространстве имён.

### 5.3 Доступ к значениям свойств WS-ресурса

Состояние WS-ресурса, т.е. значений свойств ресурса экспонированных в документе свойств WS-ресурса, может быть прочитано, модифицировано и запрошено путём использования стандартных сообщений Web-служб. Здесь мы только в общих чертах рассмотрим сообщения, которые должны быть использованы для этой цели; детальное обсуждение этого вопроса лежит вне рамок в данной статьи.

Базовая функциональность состоит в том, чтобы отыскать значение одного свойства ресурса, используя простой обмен сообщениями с Web-службой типа вопрос/ответ. При этом запросное сообщение идентифицирует WS-ресурс, используя квалифицированную ссылку на крайнюю точку WS-ресурса, как это описано ранее, и идентифицирует свойство ресурса квалифицированным именем его GED-элемента. Усовершенствованный вариант такой поисковой функции может обеспечить поиск значений множества свойств ресурса с помощью обмена единственным сообщением. При посылке ответа Web-служба сообщает в нём значения требуемых свойств WS-ресурса.

Для того чтобы выполнять произвольные XPath-выражения поиска в документе свойств ресурса, можно также воспользоваться стандартным обменом сообщениями. Могут применяться различные типы запросных выражений, например таких, которые обеспечивают обнаружение ресурса на основе использования текущих значений состояния WS-ресурса.

Мы также предвидим возможность операции, которая позволяла бы вставлять, обновлять и исключать значения свойств ресурсов, на основе представления, обеспечиваемого документом свойств WS-ресурса.

## 6 WS-ресурс и ACID-свойства

Акроним **ACID** обозначает четыре важных свойства, которыми, как правило, должны обладать ресурсы с состоянием, используемые в контексте транзакции при традиционной двухфазной системе подтверждения транзакции.

- **Атомарность (Atomicity)** требует, чтобы обновления ресурса с состоянием, используемого в контексте транзакции, были выполнены в стиле *всё или ничего*.
- **Целостность (Consistency)** означает, что транзакция оставляет ресурсы в непротиворечивом состоянии, даже в случае отказа
- **Изоляция (Isolation)** гарантирует, что частичные обновления ресурсов с состоянием, используемых внутри транзакции, невидимы вне транзакции до момента ее завершения. Изоляция реализуется посредством параллельного контроля, или, как это иногда называется, блокирования транзакции.
- **Устойчивость (Durability)** обеспечивает неизменность обновлений ресурсов с состоянием, выполненных во время транзакции.

Возможность ассоциирования политики восстановления транзакций с обменом сообщениями с Web-службой описана в спецификации [WS-AtomicTransaction]. При проведении транзакции Web-служба, способная участвовать в транзакционном протоколе, должна строго следовать правилам двухфазного подтверждения транзакции. Однако в отсутствии политики управления транзакциями Web-служба не обязана, в случае отказа, восстанавливать состояние WS-ресурса.

Спецификации WS-Resource Framework не содержат предписаний, касающихся политики управления параллельными операциями доступа к WS-ресурсу. Определение конкретной политики управления параллельными обновлениями, независимо от того, могут ли перекрываться выполнения отдельных сообщений, нацеленных на один и тот же ресурс с состоянием, и могут ли частично выполненные обновления при обработке данного запроса быть видны другим одновременным запросам, выходит за рамки инфраструктуры WS-Resource Framework. Если необходима изоляция WS-ресурса, то мы предлагаем использовать транзакцию [WS-AtomicTransaction], чтобы поддержать, контекст внутри которого может быть обеспечена изоляция обновлений. При отсутствии транзакции уровень атомарности, обновления, восстановления, изоляции и устойчивости, обеспечиваемый Web-службой, зависит от реализации последней.

Мы полагаем, что возможность декларировать и привязывать политику уровня изоляции к определению обмена сообщениями Web-службой, независимо от транзакций, представляет собой требование общего характера, на которое нет ответа в текущей версии архитектуры Web-служб. В будущем, декларации политики уровня изоляции могут быть введены как формальная часть WS-Resource Framework.

## 7 Безопасность WS-ресурса

Возможность ассоциировать политику безопасности с Web-службой описана в спецификациях WS-Policy и WS-SecurityPolicy, которые являются частью Web Services Security Roadmap. В условиях корректного контекста безопасности, связанного с обменом сообщениями, Web-служба, способная участвовать в протоколах безопасности, должна поддерживать и обеспечивать исполнение политик безопасности. При отсутствии такой политики безопасности, Web-служба не обязана обеспечивать ни безопасность выполнения обмена сообщениями, ни безопасность состояния WS-ресурса.

Определение WS-ресурса не содержит указания о политике управления правами доступа к WS-ресурсу. Определение конкретной политики безопасного управления доступом к WS-ресурсу выходит за пределы WS-Resource Framework. Если требуется

обеспечить управление доступом к WS-ресурсу, то мы предлагаем использовать функции, определённые в спецификациях WS-Security, чтобы обеспечить контекст безопасности для WS-ресурса. В отсутствие правильного контекста безопасности и связанных с этим политик управления доступом, дополнительные возможности, посредством которых Web-служба обеспечивает безопасность WS-ресурса, полностью зависят от её реализации

## 8 Заключение

Мы предложили основанный на концепции WS-ресурса подход к стандартизации представления ресурсов с состоянием и доступа к ним в распределённой среде. Этот подход определяет шаблон, который позволяет так представить и обрабатывать состояние, что Web-служба может описать ресурсы с состоянием, к которым она обеспечивает доступ, а заказчик службы – узнать тип этого ресурса и использовать стандартизированные операции для чтения, обновления и опроса его состояния, а также для управления временем его жизни.

Предлагаемый подход облегчает конструирование и использование интероперабельных служб, предоставляя различным провайдерам и потребителям служб возможность с помощью стандартных средств описывать ресурсы с состоянием, получать к ним доступ и управлять ими. Столь же важно и то, что предложенный подход предусматривает поддержку для ресурсов с состоянием, не исключая при этом возможности реализовывать Web-службы в виде не имеющих состояния процессоров сообщений.

## 9 Благодарности

Эта публикация является результатом совместной работы многих специалистов и исследовательских команд. Авторы хотят поблагодарить многих людей за их вклад в эту работу, в том числе Nick Butler, Glen Daniels, Christine Draper, Sonny Fulkerson, Rob High, Jim Knutson, Tom Maguire, Susan Malaika, David Martin, Bryan Murray, Peter Nilet, Jeff Nick, Ian Robinson, Chris Sharp и Jay Unger. Мы также признательны тем, с кем мы обсуждали проблемы, рассмотренные в этой статье, в том числе Malcolm Atkinson, Carl Kesselman и Savas Parastatidis.

## 10 Литература

### [OGSI-Refactor]

Foster, I., Frey, J., Graham, S., Tuecke, S., From Open Grid Services Infrastructure to Web Services Resource Framework: Refactoring and Evolution  
[http://www-106.ibm.com/developerworks/library/ws-resource/ogsi\\_to\\_wsrf\\_1.0.pdf](http://www-106.ibm.com/developerworks/library/ws-resource/ogsi_to_wsrf_1.0.pdf)

### [OGSI-Spec]

Open Grid Services Infrastructure (OGSI) V1.0  
<http://www.forge.gridforum.org/projects/ggf-editor/document/draft-ogsi-service-1/en/1>

### [Parastatidis]

Parastatidis, S., Webber, J., Watson, P., Rischbeck, T., A Grid Application Framework based on Web Services Specifications and Practices, Technical Report North East Regional e-Science Centre, University of Newcastle

### [Physiology]

Foster, I., Kesselman, C., Nick, J., Tuecke, S., The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, Globus Project, 2002.

<http://www.globus.org/research/papers/ogsa.pdf>

**Русский перевод:**

[http://gridclub.ru/library/publication.2004-11-29.8307957187/publ\\_file/](http://gridclub.ru/library/publication.2004-11-29.8307957187/publ_file/)

**[SOAP]**

The fundamental message enveloping mechanism in Web services

<http://www.w3.org/TR/SOAP>

**[Tao]**

Burbeck, S. The Tao of e-business Services

<http://www.ibm.com/developerworks/webservices/library/ws-tao/>, 2000

**[Vogels]**

Vogels, W. web Services are not Distributed Objects: Common Misconceptions About the Fundamentals of Web Service Technology. IEEE Internet Computing, 7 (6), 2003

**[Web Services]**

Ferguson, D., Lovering, B., Shewscuk, J., Storey, T., Secure, Reliable Transacted Web Services

<http://www-106.ibm.com/developerworks/webservices/library/ws-Securtrans/>

**[WS-Addressing]**

WS-Addressing, an XML serialization and standard SOAP binding for representing Network wide “pointers” to services

<http://www.ibm.com/developerworks/webservices/library/ws-add/>

**[WS-Arch]**

The W3C Web Services Architecture working group, public draft, August 2003

<http://www.w3.org/TR/2003/WD-ws-arch-20030808/>

**[WS-AtomicTransaction]**

<http://www.ibm.com/developerworks/webservices/library/ws-atomtran/>

**[WS-MetaDataExchange]**

WS-MetadataExchange is a set of Web service mechanisms to exchange policies, WSDL, schema and other metadata between two or more parties. This specification is part of the web services roadmap for WS-Federation.

**[WS-Security]**

The roadmap to the various security related Web services standards. See:

[http://www.oasis-open.org/committees/tc\\_home.php?wg\\_abbrev=wss](http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss)

**[WSDL 1.1]**

The Web Services Description Language, version 1.1. W3C Note:

<http://www.w3.org/TR/wsdl>

**[WSDL 2.0]**

The Web Services Description Language, version 2.0. W3C Note:

<http://www.w3.org/TR/wsdl20/>

**[WSRF]**

The Web Services Resource Framework.

<http://www-106.ibm.com/developerworks/library/ws-resource/ws-wsrfpaper.html>