МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

СОГЛАСОВАНО
Генеральный директор
ЗАО «АйТи»

_____ Бакиев О.Р.
"29" декабря 2011 г.

УТВЕРЖДАЮ
Ректор НИУ ИТМО

_____ Васильев В.Н.
"29" декабря 2011 г.

МНОГОПРОФИЛЬНАЯ ИНСТРУМЕНТАЛЬНО-ТЕХНОЛОГИЧЕСКАЯ ПЛАТФОРМА СОЗДАНИЯ И УПРАВЛЕНИЯ РАСПРЕДЕЛЕННОЙ СРЕДОЙ ОБЛАЧНЫХ ВЫЧИСЛЕНИЙ CLAVIRE

ПРОГРАММНЫЙ КОМПОНЕНТ ОБЕСПЕЧЕНИЯ ДОСТУПА К ИНФРАСТРУКТУРЕ CLAVIRE/INFRAACCESS

ОПИСАНИЕ ПРОГРАММЫ

ЛИСТ УТВЕРЖДЕНИЯ

RU.СНАБ.80066-06 13 27-ЛУ

Представители
Организации-разработчика

Руководитель разработки,
профессор НИУ ИТМО

_____ Бухановский А.В.
"29" декабря 2011 г.

Ответственный исполнитель,
с.н.с. НИУ ИТМО

_____ Луценко А.Е.
"29" декабря 2011 г.

Нормоконтролер
ведущий инженер НИУ ИТМО

_____ Позднякова Л.Г.
"29" декабря 2011 г.

2011

Инв.№ подл.	Подп. и дата	Взам.инв.№	Инв.№ дубл.	Подп. и дата

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

УТВЕРЖДЕН
RU.СНАБ.80066-06 13 Ошибка! Источник ссылки не найден.7-ЛУ

МНОГОПРОФИЛЬНАЯ ИНСТРУМЕНТАЛЬНО-ТЕХНОЛОГИЧЕСКАЯ ПЛАТФОРМА СОЗДАНИЯ И УПРАВЛЕНИЯ РАСПРЕДЕЛЕННОЙ СРЕДОЙ ОБЛАЧНЫХ ВЫЧИСЛЕНИЙ CLAVIRE

ПРОГРАММНЫЙ КОМПОНЕНТ ОБЕСПЕЧЕНИЯ ДОСТУПА К ИНФРАСТРУКТУРЕ CLAVIRE/INFRAACCESS

ОПИСАНИЕ ПРОГРАММЫ

RU.СНАБ.80066-06 13 ОШИБКА! ИСТОЧНИК ССЫЛКИ НЕ НАЙДЕН.**7**

ЛИСТОВ 16

2011

Инв.№ подл.	Подп. и дата	Взам.инв.№	Инв.№ дубл.	Подп. и дата

АННОТАЦИЯ

Программный компонент обеспечения доступа к инфраструктуре CLAVIRE/InfraAccess RU.СНАБ.80066-06 01 27 предназначен для предоставления удаленным интерфейсам единой точки доступа к платформе, а также для снижения числа жестких зависимостей компонентов системы между собой и контроля состава комплекса в процессе динамического управления выполнением композитных приложений.

Программный компонент обеспечения доступа к инфраструктуре разработан в ходе выполнения проекта «Создание распределенной вычислительной среды на базе облачной архитектуры для построения и эксплуатации высокопроизводительных композитных приложений» (Договор № 21057 от 15 июля 2010 г., шифр 2010-218-01-209) в рамках реализации постановления Правительства РФ № 218 «О мерах государственной поддержки развития кооперации российских высших учебных заведений и организаций, реализующих комплексные проекты по созданию высокотехнологичного производства».

СОДЕРЖАНИЕ

1. ОБЩИЕ СВЕДЕНИЯ

Программный компонент обеспечения доступа к инфраструктуре CLAVIRE/InfraAccess RU.СНАБ.80066-06 01 27 предназначен для предоставления удаленным интерфейсам единой точки доступа к платформе, а также для снижения числа жестких зависимостей компонентов системы между собой и контроля состава комплекса в процессе динамического управления выполнением композитных приложений.

Компонент состоит из двух модулей – сервисов WCF, реализованных на языке программирования C# версии 4.0. Оба модуля являются web-сервисами, т. е. приложениями для web-сервера с поддержкой технологии ASP .NET WebServices. Оба модуля для запуска требуют установленной среды .NET 4.0 или Mono 2.10. Компонент функционирует на аппаратных системах с архитектурой процессора x86, x86_64 и IA64.

2. ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ

2.1. Решаемые задачи

Компонент обеспечения доступа к инфраструктуре InfraAccess является интерфейсом взаимодействия различных компонентов МИТП и предоставляет унифицированный интерфейс для доступа к сервисам и ресурсам комплекса. Компонент решает следующие задачи.

1) Предотвращение несанкционированного доступа к сервисам и ресурсам МИТП и распределенной вычислительной среды.
2) Предоставление единого гибкого интерфейса доступа к сервисам распределенной вычислительной среды для других компонентов МИТП.
3) Предоставление пользователям и компонентам комплекса информации о текущем составе платформы.
4) Балансирование нагрузки между однородными сервисами.
5) Сокрытие от пользователя внутренней структуры МИТП.

2.2. Функциональные возможности

Для решения своих задач компонент InfraAccess тесно взаимодействует с компонентом контроля доступа CLAVIRE/GateKeeper, все остальные компоненты

комплекса являются потребителями сервисов, предоставляемых данным компонентом. На данный момент InfraAccess обладает следующими функциональными возможностями.

1) Обеспечивает безопасный обмен данными с пользователями через компонент пользовательского интерфейса.
2) Обеспечивает упрощенную коммуникацию между компонентами комплекса.
3) Ведет реестры установленных компонентов комплекса.
4) Проводит автоматическую регистрацию доступных сервисов и ведение реестра активных сервисов комплекса.
5) Предоставляет другим компонентам информацию о текущем наборе доступных сервисов комплекса.
6) Взаимодействует с компонентом контроля доступа для аутентификации и авторизации пользователей в системе.

3. ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ

3.1. Архитектура

Компонент обеспечения доступа к инфраструктуре состоит из двух модулей – модуля маршрутизации и модуля регистрации. Модуль регистрации служит для поддержки целостности системы и используется внутренними компонентами комплекса для коммуникации между собой (см. рис. 3.1). Модуль маршрутизации в основном предназначен для работы с компонентами пользовательского интерфейса – он является прослойкой между пользовательскими интерфейсами и системными сервисами. Модуль маршрутизации вместе с компонентом контроля доступа также ответствен за защиту сервисов комплекса от несанкционированного использования.

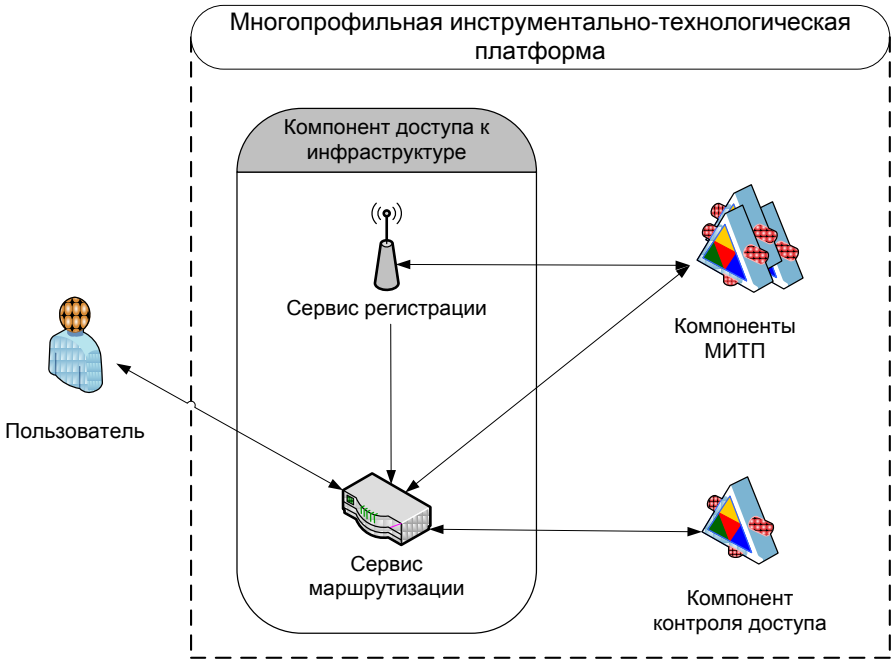

Рисунок 3.1 – Архитектура компонента обеспечения доступа к инфраструктуре

3.2. Модуль регистрации

Модуль регистрации хранит информацию об установленных компонентах системы и поддерживает информацию об активных сервисах комплекса в актуальном состоянии.

Модуль основан на спецификации WS-Discovery консорциума OASIS. Всякий раз, когда новый экземпляр любого сервиса запускается, он предоставляет модулю регистрации свои данные – контракт (имя сервиса в специальной нотации), адрес, протокол доступа. Первичное наполнение базы данных компонента выполняется при установке комплекса компонентом развертывания и конфигурирования.

Каждый компонент системы в любой момент времени имеет возможность запросить у модуля регистрации данные о конкретном сервисе и воспользоваться ими для выполнения операций, предоставляемых им. Происходит это следующим образом: когда компоненту необходим некоторый сервис другого компонента, он посылает запрос поиска с необходимыми параметрами (именем и версией сервиса) с помощью специального класса-агента DiscoveryAgent. Модуль маршрутизации по своей базе данных определяет, есть ли такой сервис в системе, и возвращает результат агенту. Если подходящих сервисов не нашлось – выполнение компонента завершается с ошибкой, которая детализируется в журнале работы для последующего анализа операторами комплекса. В случае успешного нахождения необходимых сервисов, агент получает его URL. С помощью этого адреса

агент определяет свойства соединения сервиса и инициирует сервис, после чего передает дальнейшее управление компоненту его запустившему.

Так как число сервисов с одинаковыми контрактами теоретически не ограничено, в модуле регистрации ресурсов есть инструменты получения списка сервисов по контракту и балансировке нагрузки между сервисами. Эти функции необходимы для различных специфичных модулей, например, с их помощью сервис маршрутизации формирует свою таблицу маршрутизации.

3.3. Классы модуля регистрации

3.3.1. Класс DiscoveryProxyService

Класс DiscoveryProxyService является web-сервисом, реализующим основную функциональность модуля регистрации компонента обеспечения доступа к инфраструктуре. Класс следует стандарту WS-Discovery и опирается на экспериментальные компоненты, разработанные командой разработчиков Windows Communitaion Foundation (WCF), в связи с этим он обладает необязательной функциональностью.

- Register – метод, позволяющий регистрировать сервисы в реестр платформы. Входные данные: строка – контракт регистрируемого сервиса и строка – адрес этого сервиса. Выходных параметров нет.
- Unregister – метод для удаления сервиса из реестра сервисов платформы. Входные данные: строка контракта сервиса, строка – адрес сервиса. Выходных параметров нет.
- OnlineAnnouncement – метод, позволяющий сервису объявить, что он становится активным (при разворачивании сервиса в структуре IIS 7.0 неактивные сервисы могут быть доступными для выполнения). Входной параметр: тип EndpointDiscoveryMetadata – описание сервиса, становящегося активным. Выходных параметров нет.
- Offline Announcement – метод, позволяющий сервису объявить, что он становится неактивным (неактивные незарегистрированные сервисы удаляются из базы платформы). Входной параметр: тип EndpointDiscoveryMetadata – описание сервиса, становящегося активным. Выходных параметров нет.
- Find – метод для поиска сервисов в реестре платформы. Входной параметр: FindRequestContext – описание искомого сервиса (по контракту и по версии).

Выходной параметр: список описаний EndpointDiscoveryMetadata сервисов, удовлетворяющих условиям поиска.

- Resolve – метод для определения экземпляра сервиса, сменившего адрес, по его старым данным. Входной параметр: ResolveContext – запрос поиска по старым данным. Выходной параметр: EndpointDiscoveryMetadata – новое описание искомого сервиса.
- SerializeToString – метод для конвертирования описания сервиса из типа EndpointDiscoveryMetadata в строку для записи в базу данных.
- DeserializeFromString – метод для конвертирования строкового описания сервиса в тип EndpointDiscoveryMetadata – для возвращения результатов методов, описанных выше.

Класс EndpointDiscoveryMetadata входит в состав сборок WCF, служит описанием web-сервисов для регистрирования в сервисе Discovery. В его состав входит полное квалифицированное имя контракта сервиса, его http-адреса и возможные протоколы доступа, а также версия сервиса.

3.3.2. Класс MetadataToSave

MetadataToSave – это класс описания сервиса, которое хранится в базе данных MongoDB из-за того, что аналогичный класс – EndpointDiscoveryMetadata – не может быть адаптирован для хранения в этой базе из-за закрытости своей реализации. Так как класс предназначен для хранения данных, приведем только его поля.

- Id (тип ObjectId) – уникальный идентификатор записи в базе.
- Uri (string) – http-адрес сервиса.
- Contract (string) – полное квалифицированное имя контракта сервиса.
- Registered (bool) – определяет, зарегистрирован ли сервис в реестре платформы.
- Online (bool) – определяет, находится ли сервис в активном состоянии.
- DiscoveryMetadata (string) – строка, полученная из описания сервиса EndpointDiscoveryMetadata применением метода SerializeToString класса DiscoveryProxyService, служит для хранения исходного описания сервиса.

3.3.3. Интерфейс IBindingProvider

Интерфейс IBindingProvider определяет поведение для классов, предоставляющих конфигурации протоколов доступа к сервисам. Реализации этого класса передаются в

конструктор класса DiscoveryAgent для определения правильных привязок к каждому сервису платформы.

- Метод GetDiscoveryBinding – возвращает параметры протокола доступа к развернутому сервису Discovery. Входных данных нет. Выходные данные – объект класса Binding (или его наследники), определенный в WCF для инкапсуляции параметров протокола доступа.
- Метод GetBinding<T> параметризован типом контракта – возвращенный объект класса Binding зависит как раз от этого типа-параметра.

Для этого интерфейса существует несколько простых реализаций, таких как BasicBindingProvider,WsBindingProvider и ConfigBindingProvider.

3.3.4. Класс DiscoveryAgent

Класс DiscoveryAgent – специально разработанный клиент сервиса DiscoveryProxyService модуля регистрации, служащий цели упрощения взаимодействия системных сервисов с модулем регистрации и между собой. DiscoveryAgent создается на клиентской стороне и формирует запросы к DiscoveryProxyService. Основные методы сервиса представлены ниже. Класс спроектирован по шаблону объектно-ориентированного программирования «одиночка».

- Instance – статический метод получения экземпляра класса. Входных параметров нет. Выходной параметр: DiscoveryAgent – действующий экземпляр класса.
- GetDiscoveryEndpoint – возвращает адрес сервиса Discovery, с которым взаимодействует на данный момент.
- GetEndpoints – возвращает описания сервисов, зарегистрированных в платформе в виде объектов EdpointDiscoveryMetadata.
- GetClient – генерация клиента для заданного контракта. Входной параметр: интрефейс-контракт, для которого требуется создать клиент. Выходной параметр: клиент заданного типа.
- GetClient<T> – генерация клиента для заданного контракта, метод, выполняющий то же что и предыдущий, но параметризированный искомым типом. Входной параметр: параметр-тип контракта. Выходной параметр: клиент заданного типа.
- Register<T> – метод регистрации сервиса в платформе по его интерфейсу. Проекция метода Register класса DiscoveryProxyService. Входной параметр: параметр-тип для регистрации, адрес сервиса. Выходных параметров нет.

- Unregister – метод удаления сервиса из реестра сервисов платформы CLAVIRE. Проекция метода Unregister класса DiscoveryProxyService. Входные параметры: строка – контракт сервиса, строка – адрес сервиса. Выходных параметров нет.

3.4. Модуль маршрутизации

Модуль маршрутизации является общей точкой доступа ко всем внутренним компонентам платформы. Пользователь может направлять запрос на использование любого компонента системы непосредственно к модулю маршрутизации, который разбирает запрос с помощью фильтров и перенаправляет соответствующему компоненту, при этом опрашивается компонент контроля доступа на возможность выполнения этой операции данным пользователем. Модуль маршрутизации формирует свою таблицу маршрутизации, получая текущий набор доступных сервисов от сервиса регистрации, и создает для нее фильтры, основанные на полученных контрактах. Обновление фильтров происходит с некоторой периодичностью, которая может быть настроена в файлах конфигурации.

Модуль маршрутизации обеспечивает и другую функциональность, например, балансировку нагрузки однородных компонентов внутри таблицы маршрутизации и повышение надежности комплекса с помощью задания таблицы перенаправления запросов на дублирующие компоненты.

Также модуль маршрутизации упрощает настройку политик безопасности при передаче информации, для этого настраивается безопасный канал между пользователем и точкой доступа к сервису, в то время как компоненты системы могут обмениваться между собой информацией по более простым и производительным, хотя и менее безопасным, протоколам. Также это избавляет клиента от необходимости разбираться с протоколами и версиями конкретных сервисов, равно как и администраторов платформы при настройке взаимодействия компонентов.

3.5. Классы модуля маршрутизации

Модуль маршрутизации практически полностью реализован на классах, предоставляемых фрейворком Windows Communication Foundation в сборке System.ServiceModel.Routing. Классы этой сборки разработаны для быстрого внедрения сервиса маршрутизации в составы распределенных программных комплексов. Сама по себе маршрутизация не требует хранения каких-либо данных в базах или в конфигурационных файлах, для получения данных используется модуль регистрации.

Именно обращение к модулю регистрации путем расширения встроенного сервиса маршрутизации было основной проблемой, также были реализованы элементы авторизации и аутентификации путем взаимодействия с компонентом контроля доступа.

3.5.1. Класс DynamicUpdateBehavior

Класс DynamicUpdateBehavior – расширение поведения для сервиса маршрутизации для периодического переформирования таблицы фильтрации, является реализацией интерфейса IBehaviorExtention.

- Метод AddBinding – обязательный для расширения сервиса (описан в IBehaviorExtention), позволяет добавлять новые протоколы доступа с помощью задания дополнительных объектов Binding (или его потомков) к сервису уже во время его работы. Входные параметры: ServiceDescription – описание сервиса и ServiceHostBase – описание объекта запуска сервиса. Выходных параметров нет.
- Метод Validate – тоже описан в интерфейсе IBehaviorExtention, позволяет проверить непротиворечивость настроек сервиса маршрутизации и его запускающего объекта. Входные параметры: ServiceDescription – описание сервиса и ServiceHostBase – описание объекта запуска сервиса. Выходных параметров нет.
- Метод ApplyBehavior – позволяет добавить к сервису дополнительное поведение – т. е. расширить его функциональность. Именно в этом методе добавляется периодическое обновление таблицы маршрутизации – класс RulesUpdateExtension. Входные параметры: ServiceDescription – описание сервиса и ServiceHostBase – описание объекта запуска сервиса. Выходных параметров нет.

3.5.2. Класс RulesUpdateExtension

Класс RulesUpdateExtension – расширение для объекта запуска сервиса маршрутизации, наследуется от интерфейса IExtention<T>.

- Метод Attach позволяет применить расширение в объекту запуска сервиса маршрутизации. Входным параметром является как объект запуска. У метода нет выходных данных.
- Метод Detach, наоборот, отменяет расширение объекта запуска сервиса. Входным параметром является объект запуска. У метода нет выходных данных.
- Метод UpadateRules является собственным методом объекта, не наследуется от класса или интерфейса. Этот метод запускается с заданной администратором

периодичностью, вызывает метод GetEndpoints модуля регистрации и на основе этих данных формирует свою таблицу маршрутизации.

3.5.3. Класс EasisValidator

Класс EasisValidator тоже является расширением сервиса маршрутизации, но иного плана – благодаря наследованию от класса UserNamePassordValidator. Каждый раз когда к сервису приходит какой-либо запрос, он еще до обработки фильтрами проверяется этим классом, а если точнее – его методом Validate. В этот метод передаются имя пользователя и пароль стороны, начавшей соединение. В случае успешной проверки пары логин/ пароль метод должен просто завершит свою работу, в случае неудачной попытки метод сгенерирует исключение, которое будет направлено графическому интерфейсу пользователя.

Метод Validate класса EasisValidator обращается к модулю аутентификации компонента контроля доступа для определения корректности пары пользователь/пароль.

3.5.4. Класс EasisAuthManager

Класс EasisAuthManager – наследуется от класса ServiceAuthorizationManager, предназначенного для проверки авторизации пользователя на сервис. У класса имеется один метод CheckAccess с входным параметром – OperationContext – контекстом выполнения операции. Результатом выполнения является логическое значение, определяющее разрешение на запуск данного сервиса.

Класс EasisAuthManager при реализации метода CheckAccess обращается к модулю авторизации компонента контроля доступа, который и определяет право запуска сервиса с помощью выдачи всех ролей вызывающего пользователя.

4. ИСПОЛЬЗУЕМЫЕ ТЕХНИЧЕСКИЕ СРЕДСТВА

Компонент предназначен для функционирования на аппаратных системах с характеристиками:

- архитектура процессора – x86, x86_64, IA64;
- минимальный объем оперативной памяти – 1 ГБ;
- минимальный объем свободного пространства на жестком диске – 1 ГБ;
- минимальная тактовая частота процессора – 1 ГГц.

Для работы компонента требуется наличие следующего системного ПО: ОС семейства Windows NT (версии старше Windows 2000), .NET 4.0, IIS (с версией старше 6.0). Для полноценной работы необходимо наличие развернутого и настроенного сервера баз данных MongoDB.

5. ВЫЗОВ И ЗАГРУЗКА

Программный компонент реализован в виде SOAP WCF-сервиса платформы .NET. Для запуска сервиса используется стандартный механизм используемого web-сервера. Загрузка сервиса в этом случае выполняется web-сервером автоматически по мере поступления запросов от клиентов сервиса.

Компилировать и размещать сервисы данного компонента можно независимо друг от друга, но при этом важно помнить об их тесном взаимодействии и, следовательно, обеспечивать их быстрым каналом передачи данных. Оба модуля компонента являются обычными SOA-сервисами, разворачивающимися стандартными средствами web-сервера (например, IIS). После запуска компонента доступ ко всем модулям можно получать по адресам, заданным при развертывании, например, адрес может выглядеть так: http://SampleServer.ru:8008/SampleService.

Модуль регистрации является образующим сервисом комплекса, поэтому он должен устанавливаться первым и не требует других предустановленных компонентов.

6. ВХОДНЫЕ ДАННЫЕ

Сервисы компонента являются классическими web-сервисами, реализованными на основе каркаса, предоставленного фреймворком Windows Communication Foundation. Разработка такого рода компонентов состоит из двух частей: написание ядра сервиса, удовлетворяющего контракту и обеспечивающего выполнение необходимой логики, и расширение поведения сервиса для предоставления дополнительной функциональности. Особенностью такой реализации является сокрытие внутренних потоков данных как от пользователя, так и от разработчиков. Входными данными для программного компонента обеспечения доступа к инфраструктуре могут являться конфигурационные файлы, но они ограничены стандартными конфигурациями платформы Windows Communication Foundation.

7. ВЫХОДНЫЕ ДАННЫЕ

Сервис маршрутизации не имеет собственного формата выходных данных, так как является посредником между пользователем и компонентами комплекса.

Сервис регистрации имеет свой формат выходных данных, являющийся расширением языка запросов SOAP. Информация в данном формате генерируется сервисом, хранится на диске и передается по запросу другим компонентам комплекса. Пример данных в этом формате представлен в листинге 7.1.

Листинг 7.1. Расширение языка SOAP для сервиса регистрации

```
      <ProbeMatchType>
        <EndpointReference xmlns="http://www.w3.org/2005/08/addressing">
          <Address>http://192.168.0.31:8008/SecurityService</Address>
        </EndpointReference>
        <d:Types xmlns:dp0="http://tempuri.org/" xmlns:d="http://docs.oasis-
open.org/ws-dd/ns/discovery/2009/01">dp0:ISecurityService</d:Types>
        <XAddrs xmlns="http://docs.oasis-open.org/ws-
dd/ns/discovery/2009/01">http://192.168.0.31:8008/SecurityService</XAddrs>
        <MetadataVersion xmlns="http://docs.oasis-open.org/ws-
dd/ns/discovery/2009/01">0</MetadataVersion>
        <root xmlns="">
          <BindingData>System.ServiceModel.WSHttpBinding</BindingData>
          <Version>1.0</Version>
          <ImplementationType>dev</ImplementationType>
        </root>
      </ProbeMatchType>
```

Значения полей для этого формата приведены в табл. 7.1.

Таблица 7.1

Поля формата данных сервиса регистрации

Название поля	Описание поля
Address	Адрес (URL) сервиса
Types	Контракт, которому удовлетворяет сервис
MetadataVersion	Версия метаданных сервиса
BindingData	Тип привязки, определяющей параметры соединения с сервисом
Version	Версия реализации сервиса
ImplementationType	Определяет завершенность разработки сервиса: dev – сервис еще разрабатывается, prod – уже готов к использованию

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

SOAP	Simple Object Access Protocol (простой протокол доступа к объектам)
БД	База данных
МИТП	Многопрофильная инструментально-технологическая платформа
ОС	Операционная система
ПО	Программное обеспечение

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

Лист регистрации изменений									
Изм.	Номера листов (страниц)				Всего листов (страниц) в докум.	№ документа	Входящий № сопроводительного докум. и дата	Подп.	Дата
	изменённых	заменённых	новых	Аннулированных					