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 СОБЫТИЙНОГО ВЗАИМОДЕЙСТВИЯ CLAVIRE/EVENTING

ОПИСАНИЕ ПРОГРАММЫ

ЛИСТ УТВЕРЖДЕНИЯ

RU.СНАБ.80066-06 13 23-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.**-ЛУ**

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

ПРОГРАММНЫЙ КОМПОНЕНТ СОБЫТИЙНОГО ВЗАИМОДЕЙСТВИЯ CLAVIRE/EVENTING

ОПИСАНИЕ ПРОГРАММЫ

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

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

ЛИСТОВ 17

2011

АННОТАЦИЯ

Документ содержит описание программного компонента событийного взаимодействия CLAVIRE/Eventing RU.СНАБ.80066-06 01 23, реализующего необходимый для работы других компонентов МИТП CLAVIRE слой взаимодействия, а также предоставляющего компонентам МИТП информацию о состоянии распределенной среды в форме отчетов о событиях. Программный компонент 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

Компонент событийного взаимодействия CLAVIRE/Eventing RU.СНАБ.80066-06 01 23 (далее Eventing) предназначен для обеспечения взаимодействия компонентов многопрофильной инструментально-технологической платформы (МИТП) CLAVIRE с использованием событийной модели обмена информацией. Данный компонент разработан на языке C# в виде web-сервиса с применением технологии WCF.

Компонент функционирует на аппаратных системах с архитектурой процессора x86, x86_64 и IA64. Для его работы необходимо следующее системное программное обеспечение: ОС семейства Windows NT (версии старше Windows 2000), .NET 4.0, Microsoft Internet Information Services (с версией старше 6.0); опционально – наличие БД MySql для дополнительного журналирования происходящих событий.

2. ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ

Программный компонент предназначен для выполнения следующих функций:

- обеспечение других компонентов МИТП CLAVIRE средством взаимодействия, основанным на шаблоне проектирования pub/sub (публикация/подписка). Eventing реализует уровень абстракции над стандартными web-сервисами и позволяет гибко связывать компоненты МИТП по схеме «многие ко многим» (вместо стандартного метода связи «один к одному»);
- предоставление информации о процессах, происходящих в распределенной среде под управлением МИТП, компоненту мониторинга CLAVIRE/Monitoring RU.СНАБ.80066-06 01 24, который, в свою очередь, выполняет ее обработку.

3. ОПИСАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ

3.1. Принципы функционирования Eventing

В терминологии программного компонента Eventing событие – структура, содержащая формализованные данные о произошедшем в МИТП явлении, которая включает в себя заголовочную часть и тело. Заголовок содержит следующие поля: время, в которое явление произошло; URI (Uniform Resource Identifier) источника события; тип события (или тема), по которому можно идентифицировать внутреннюю структуру тела

события; адрес, по которому опубликована схема структуры тела события. Тело события представляет собой документ в формате XML. Внутри тела события содержатся специфические для типа события данные. Для упрощения программной обработки событий (проверки на корректность тела события, операций объектной сериализации и десериализации) применяется подход, основанный на использовании технологий XML/XSD для описания события. Он позволяет:

- фиксировать контракт между сервисами комплекса;
- предоставлять гибкие механизмы структурирования информации о событии;
- расширять систему типов событий, не затрагивая при этом сам Eventing;
- использовать весь набор существующих инструментов и библиотек для обработки XML-документов.

Жизненный цикл события начинается с момента его генерации источником. Для этого источник события формирует информационную структуру (EventReport), соответствующую типу события, и выполняет вызов операции FireEvent web-сервиса EventingBroker. В свою очередь, менеджер оповещений производит запись события в журнал и генерирует оповещения. При генерации предварительно определяется список адресатов в менеджере подписок, для чего выполняется процесс фильтрации. Фильтрация подписок – это механизм, который используется для определения того, каким подписчикам посылать уведомления о случившемся событии. Процесс фильтрации заключается в том, что к событию последовательно применяются фильтры каждой из активных подписок, и в результате определяется, каким подписчикам будет сгенерировано оповещение. Каждая подписка содержит в себе набор фильтров, которые применяются согласно логическому «И». Фильтр характеризуется диалектом, телом фильтра и действием, которое производится с подпиской при срабатывании фильтра. Диалект фильтра – это способ фильтрации, по которому будет вычисляться предикат P(Body,Event), где Body – тело фильтра, а Event – событие. Программный компонент позволяет использовать механизмы фильтрации по заголовку события и по телу. К методам фильтрации по заголовку относятся: TopicFilter – фильтрация по типу (теме) события; SourceFilter – фильтрация по источнику события. Для фильтрации по телу события используется механизм XPath. В этом случае тело фильтра содержит XPath-выражение, которое применяется к телу события.

Результатом фильтрации подписок является список адресатов, подписанных на данное событие. При оповещении для каждого адресата формируется объект Notification (который содержит в себе EventReport и идентификатор подписки) и помещается в

очередь оповещения. Процедура оповещения выполняется в отношении web-сервиса, адрес которого указан в подписке. Схема оповещения предполагает, что целевой web-сервис реализует сервисный контракт INotifiable, в соответствии с которым сервис должен иметь операцию с именем Notify и параметром – объектом класса Notification. Если в процессе доставки уведомлений возникают проблемы, то производится отмена подписки (CancelSubscription): в случае, если подписка не системная, она удаляется из списка активных, а на адрес ответственного web-сервиса высылается уведомление типа SubscriptionEnd с указанием причины отмены подписки.

Механизм подписки совмещает в себе две схемы: первая – каждый компонент МИТП подписывается на события, которые ему интересны, и следит за тем, чтобы подписка оставалась активной. Вторая – выделенный компонент подписывает компоненты МИТП на интересующие их события и следит за функционированием существующих подписок. В состав EventingBroker входит системный подписчик (SystemSubscriber), который отвечает за подписку основных компонентов на служебные события. Он хранит конфигурацию системных подписок и периодически производит синхронизацию ее с сервисом подписки. Компоненты, которые по тем или иным причинам не обслуживаются системным подписчиком (например, динамически создаваемые сервисы, внешние сервисы), производят подписку самостоятельно.

3.2. Программная архитектура Eventing

Архитектура Eventing приведена на рис. 3.1, ядро компонента – EventingBroker, централизованный брокер событий, в функции которого входят регистрация событий, ведение журнала событий, подписка на события, оповещение подписанных на событие компонентов. Он состоит из менеджера подписок (SubscriptionManager), менеджера оповещений (NotificationManager) и системного подписчика (SystemSubscriber). Интерфейсом брокера является web-сервис EventingBrokerService.

Источниками событий в МИТП могут являться любые компоненты и модули, регистрирующие определенные события. Технически источником может являться любая программа, которая способна выполнить вызов операции FireEvent web-сервиса брокера.

Адресатом уведомления может быть компонент системы, имеющий web-сервис, который реализует интерфейс INotifiable. Такой web-сервис должен иметь операцию Notify с определенной сигнатурой.

Подписчик выполняет служебную функцию подписки определенного компонента системы (включая себя) на получение оповещений об определенных событиях,

происходящих в системе. Подписчиком на события может быть как сервис, который сам нуждается в получении уведомлений, так и сервис, подписывающий другие подчиненные сервисы.

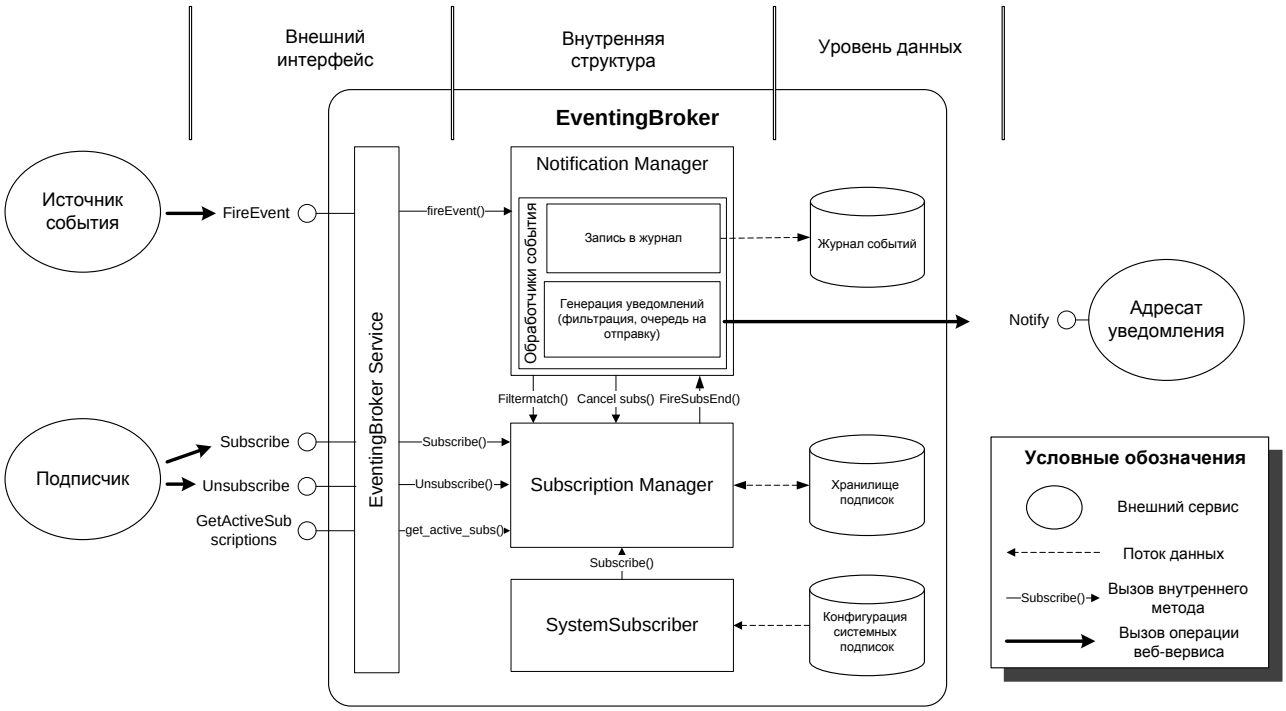

Рисунок 3.1 – Архитектура компонента событийного взаимодействия

Eventing реализован в виде SOAP web-сервиса платформы .NET 4.0 на языке C#. Web-сервис брокера разработан с использованием технологии WCF (Windows Communication Foundation) на основе стандарта SOAP. Все события заносятся в таблицу базы данных MySQL (опциональная возможность). Для реализации хранилища подписок SubscriptionManager использовано средство управления базами данных SQLite. Доступ к БД реализован с использованием технологии Entity Framework.

Программная реализация компонента событийного взаимодействия включает в себя два модуля: *библиотеку классов* и *брокер событий*. Программы, входящие в состав библиотеки классов событийного взаимодействия, написаны на языке C# и поставляются в форме стандартного проекта, созданного с использованием интегрированной среды разработки VisualStudio 2010. Основные классы, входящие в библиотеку Eventing, описаны в файлах с расширением *.cs. Брокер событийного взаимодействия поставляется в форме стандартного проекта WCF web-сервиса, созданного с использованием интегрированной среды разработки VisualStudio 2010. Основные классы, входящие в брокер, описаны в файлах с расширением *.cs.

3.3. Основные классы Eventing

Ниже приводятся сокращенные описания структуры и методов основных классов компонента событийного взаимодействия.

3.3.1. Класс EventingBrokerService

Основной класс WCF-сервиса брокера событий. Реализует интерфейс IEventingBrokerService, является интерфейсом всего компонента.

Открытые методы

- FireEvent – регистрация нового события
    a) входной параметр Evt (тип: EventReport) – информация о событии;
    b) возвращаемое значение отсутствует (void).
- GetActiveSubscriptions – получение списка активных подписок, зарегистрированных в менеджере подписок.
    a) возвращает список активных подписок (тип: List<Subscription>).
- Subscribe – зарегистрировать новую подписку.
    a) входной параметр subscriptionRequest (тип: SubscriptionRequest) – запрос на подписку возвращает идентификатор подписки (тип: SubscriptionId);
- Unsubscribe – прекращение подписки (отписка)
    a) входной параметр subscriptionId (тип: SubscriptionId) – идентификатор подписки;
    b) возвращаемое значение отсутствует (void).

3.3.2. Класс GlobalSubscriber

Системный подписчик: устанавливает системные подписки, периодически синхронизирует список системных подписок с файлом конфигурации. Реализован согласно шаблону проектирования «одиночка» (singleton). Поддерживает многопоточный доступ.

Свойства

- Instance (тип: GlobalSubscriber) – экземпляр класса.
- ConfigFilePath (тип: string) – путь до конфигурационного файла.
- ConfigSyncInterval (тип: double) – временной период синхронизации с файлом конфигурации.

Открытые методы

- SyncronizeConfig – синхронизация подписок с файлом конфигурации.

Закрытые методы

- SubscribeNew – подгрузка новых (отсутствующих в списке активных подписок) системных подписок из файла.
- LoadSubscriptions – загрузка подписок из конфигурационного файла.
- UnloadSubscriptions – выгрузка подписок в файл.

3.3.3. Класс NotificationManager

Менеджер оповещения. Помимо оповещения о событии осуществляет ряд других действий, таких как сохранение в БД информации о событии. Все действия, применяемые к событию, организованы в виде упорядоченной цепочки обработчиков. Класс реализован согласно шаблону проектирования – одиночка (singleton). Поддерживает многопоточный доступ.

Внутренние классы

- Класс AddressedEvent – адресованное событие.
- Интерфейс IChainOfEventHandlers – интерфейс для цепочки обработчиков.
- Интерфейс IEventHandler – интерфейс для обработчика события.
- Класс MongoHistoryEventHandler – обработчик событий, сохраняющий их в БД mongodb.
- Класс MySQLHistoryEventHandler – обработчик событий, сохраняющий их в БД MySQL.
- Класс NotificationGeneratorEventHandler – обработчик событий, рассылающий уведомления. Использует ThreadPool для отправки уведомлений.
- Класс SequentalChainOfEventHandlers – последовательная цепочка обработки событий. Все события обрабатываются во всех зарегистрированных в цепочке обработчиках по порядку.

Открытые методы

- FireEvent – сигнализация о событии
    a) входной параметр ev типа EventReport – информация о событии;
    b) возвращаемое значение отсутствует (void).
- FireSubscriptionEndNotification – сигнализация о прекращении подписки. Вспомогательный метод для вызова FireAddressedEvent; вызывается из SubscriptionManager

a) входной параметр subscription типа Subscription – целевая подписка;
b) входной параметр reason типа ReasonEnum – причина прекращения подписки;
c) возвращаемое значение отсутствует (void).

Свойства

- Instance (тип: NotificationManager) – экземпляр класса.

3.3.4. Класс Subscription

Информация об активной подписке. Внутренний класс для брокера. Внешне может быть использован в качестве контейнера, содержащего информацию о подписке.

Свойства

- Status (тип: SubscriptionStatusEnum) – статус подписки.
- isSystem (тип: bool) – определяет, является ли подписка системной.
- DeliveryMode (тип: DeliveryModeEnum) – метод доставки. Поддерживается только Push.
- NotifyTo (тип: EndPoint) – адрес оповещаемого сервиса.
- EndTo (тип: EndPoint) – адрес сервиса, которому будет доставлено оповещение об отмене подписки.
- Expires (тип: DateTime) – дата, когда подписка станет недействительной. По достижении данной даты подписка будет удалена.
- Filters (тип: List<SubscriptionFilter>) – активные фильтры, определяющие, подходит ли событие для данной подписки.
- Id (тип: SubscriptionId) – идентификатор подписки.
- UniqName (тип: string) – уникальное имя подписки.

Методы

- fromSubscriptionRequest – создать экземпляр класса на основе информации из запроса на подписку (SubscriptionRequest)
  a) входной параметр sreq (тип: SubscriptionRequest) – описание запроса на подписку;
  b) возвращает созданную подписку (тип: Subscription).

3.3.5. Класс SubscriptionManager

Менеджер подписок. Хранит список активных подписок. Позволяет регистрировать, удалять подписки клиентам, а также поддерживает набор системных подписок. Класс реализован согласно шаблону проектирования «одиночка» (singleton). Поддерживает многопоточный доступ.

Методы

- CancelSubscription – отмена подписки
    a) входной параметр subscriptionId (тип: SubscriptionId) – идентификатор подписки;
    b) входной параметр reason (тип: ReasonEnum) – причина отмены;
    c) возвращаемое значение отсутствует (void).
- FilterMatch – метод фильтрации поступившего события через все имеющиеся подписки
    a) входной параметр evt (тип: EventReport) – информация о событии;
    b) возвращает список подходящих подписок, фильтры которых сработали для события (тип: List<Subscription>).
- GetActiveSubscriptions – возвращает все активные подписки
    a) возвращает список (тип: List<Subscription>).
- Subscribe – установка новой подписки
    a) входной параметр subReq (тип: SubscriptionRequest) – запрос на подписку;
    b) возвращает идентификатор подписки (тип: SubscriptionId).
- Unsubscribe – прекращение подписки
    a) входной параметр subscriptionId (тип: SubscriptionId) – идентификатор подписки;
    b) возвращаемое значение отсутствует (void).

3.3.6. Класс SubscriptionRequest

Запрос на установку подписки. Содержит в себе все информационные поля Subscription, передаваемые клиентом. На основании объекта данного класса создается объект класса Subscription.

Открытые атрибуты

- Duration (тип: TimeSpan) – срок действия подписки. Начинается с момента поступления запроса в SubscriptionManager.

Свойства

- Status (тип: SubscriptionStatusEnum) – статус подписки.
- isPermanent (тип: bool) – определяет, является ли подписка системной.
- DeliveryMode (тип: DeliveryModeEnum) – метод доставки. Пока поддерживается только Push.
- NotifyTo (тип: EndPoint) – адрес оповещаемого сервиса.
- EndTo (тип: EndPoint) – адрес сервиса, которому будет доставлено оповещение об отмене подписки.
- Filters (тип: List< SubscriptionFilter >) – активные фильтры.
- Id (тип: SubscriptionId) – идентификатор подписки.
- UniqName (тип: string) – уникальное имя подписки.

4. ИСПОЛЬЗУЕМЫЕ ТЕХНИЧЕСКИЕ СРЕДСТВА

Компонент событийного взаимодействия 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.

Eventing требует для своей работы наличия следующего системного ПО: ОС семейства Windows NT (версии старше Windows 2000), .NET 4.0, IIS (с версией старше 6.0).

5. ВЫЗОВ И ЗАГРУЗКА

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

Eventing предоставляет интерфейс в виде WCF-сервиса, реализованного классом EventingBrokerService (см. спецификацию EventingBrokerService в разделе 3.3.1). Вызов сервиса производится стандартным для технологии WCF способом: по опубликованному описанию сервиса (WSDL) необходимо создать прокси-класс, через который

осуществляется взаимодействие с сервисом путем вызова необходимых операций (методов). Процедура создания прокси-класса зависит от того, на базе каких технологий строится клиентское приложение. Если выбраны язык программирования C# и платформа .NET, построение клиента производится за счет вызова служебного программного средства svcutil.exe, распространяемого в составе платформы .NET, либо за счет создания ссылки на сервис в среде Microsoft VisualStudio.

6. ВХОДНЫЕ ДАННЫЕ

Входными данными для Eventing являются конфигурационные файлы, а также входные параметры вызова интерфейсных методов. Модуль использует хранилище системных подписок, которое организовано в виде файла subscriptions.config в формате XML. При старте системы конфигурации подписок считываются и активируются. На рис. 6.1 представлен пример файла конфигурации с одной системной подпиской.

Интерфейсные операции web-сервиса принимают данные о подписке (SubscriptionRequest) и данные о событии (EventReport). На рис. 6.2 представлена структура используемых данных и приведены примеры значений для каждого поля.

```
    <?xml version="1.0" encoding="utf-8"?>
    <GlobalSubscriberConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
      <subscriptionConfigs>
        <SubscriptionConfig>
          <SubscriptionRequest>
            <DeliveryMode>PushMode</DeliveryMode>
            <isPermanent>true</isPermanent>
            <NotifyTo> <Address>http://srv08:81/FlowSystemService.svc</Address>
              <Tag>WFStateUpdatedEvent</Tag>
            </NotifyTo>
            <EndTo> <Address>http://localhost:8735/GlobalSubscriberService/</Address>
              <Tag>OnSubscriptionEnd</Tag>
            </EndTo>
            <Duration />
            <UniqName>PesToFlowSystemService</UniqName>
            <Filters>
              <SubscriptionFilter>
                <Dialect>TopicFilterDialect</Dialect>
                <Body>WFStateUpdatedEvent</Body>
              </SubscriptionFilter>
            </Filters>
          </SubscriptionRequest>
        </SubscriptionConfig>
      </subscriptionConfigs>
    </GlobalSubscriberConfig>
```

Рисунок 6.1 – Фрагмент конфигурационного файла системных подписок

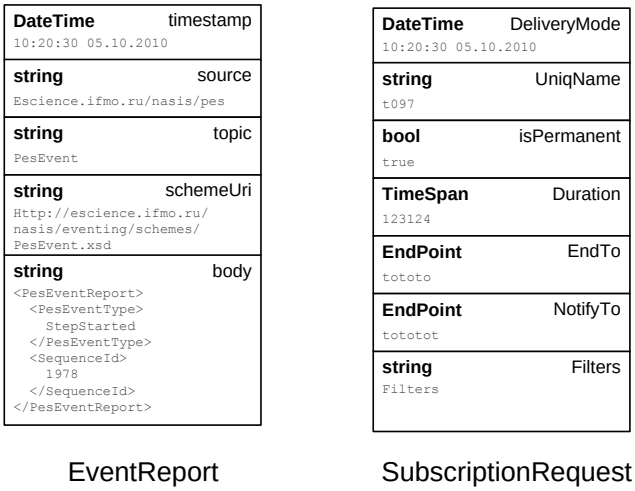

Рисунок 6.2 – Схема входных данных для сервиса событийного взаимодействия

7. ВЫХОДНЫЕ ДАННЫЕ

Выходными данными компонента событийного взаимодействия являются данные о событии, отсылаемые при оповещении (см. табл. 7.1), журнал событий и опубликованные схемы событий.

Таблица 7.1

Структура данных оповещения о событии (*Notification*)

Тип параметра	Имя параметра	Описание
SubscriptionId	SubscriptionId	Идентификатор сработавшей подписки
EventReport	Event	Информация о событии
String	Tag	Вспомогательное поле, соответствующее подписке, для удобства обработки событий из разных подписок

Для ведения журнала происходящих в системе событий используется база данных, описываемая схемой, представленной на рис. 7.1.

```
    CREATE TABLE  `ipse`.`event_log` (
      `Id` int(10) unsigned NOT NULL AUTO_INCREMENT,
      `Timestamp` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
      `Source` varchar(256) NOT NULL,
      `Topic` varchar(256) NOT NULL,
      `Body` text NOT NULL,
      `SchemeUri` varchar(256) NOT NULL,
      `LoggedTimestamp` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',
      PRIMARY KEY (`Id`)
    ) ENGINE=InnoDB AUTO_INCREMENT=144 DEFAULT CHARSET=utf8;
```

Рисунок 7.1 – Описание схемы БД в формате SQL конфигурационного файла системных подписок

Схемы типов событий (в формате XSD), необходимые для функционирования Eventing, располагаются в специальной директории «Schemes» web-сервиса, откуда они доступны другим компонентам платформы.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

SOAP	Simple Object Access Protocol (простой протокол доступа к объектам)
URI	Uniform Resource Identifier (унифицированный идентификатор ресурса)
WCF	Windows Communication Foundation
XML	eXtensible Markup Language (расширяемый язык разметки)
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
	изме-ненных	заме-ненных	но-вых	аннули-рован-ных					