УДК 004.4

АРХИТЕКТУРА ИНТЕГРАЦИОННОЙ ОБЛАЧНОЙ ПЛАТФОРМЫ

Еловиков Андрей Евгеньевич
Компания NAUMEN, Екатеринбург, Россия

В статье рассмотрены элементы облачной интеграционной платформы, задачи, которые она решает, ее компоненты и взаимосвязи между ними. Определены типы приложений платформы и описано взаимодействие между ними

Ключевые слова: ОБЛАЧНЫЕ ВЫЧИСЛЕНИЯ, IPAAS, ИНТЕГРАЦИЯ В ОБЛАКЕ, МИГРАЦИЯ ДАННЫХ, ПЛАН МАРШРУТИЗАЦИИ, ПРОМЕЖУТОЧНАЯ ТРАНСФОРМАЦИЯ

UDC 004.4

ARCHITECTURE OF INTEGRATION CLOUD PLATFORM

Elovikov Andrey Evgenevich
NAUMEN Company, Yekaterinburg, Russia

This article presents the elements of the integration cloud platform, the tasks it solves, the components of the platform and the relationships between them. The types of applications of the platform are defined and the interactions between them are described

Keywords: CLOUD COMPUTING, IPAAS, CLOUD INTEGRATION, DATA MIGRATION, ROUTING PLAN, INTERIM TRANSFORMATION

Эволюция IT-систем приводит к тому, что в компаниях существует множество систем, отвечающих за различные аспекты внутренних процессов компании, что приводит к необходимости структурировать взаимодействия этих систем между собой [1]. Этой проблемой занимается область интеграции.

Облачные вычисления в настоящее время являются одной из самых быстро развивающихся IT-технологий. Поэтому задача интеграции IT-систем часто представляет собой задачу интеграции облачных или облачных и не облачных систем. Значимость решений по интеграции облачных вычислений растет по мере увеличения степени их распространенности.

Как отмечается в исследовании [2], более половины опрошенных участников IT бизнеса назвали проблемы в области интеграции как основную причину отказа от использования SaaS-приложений. Необходимость решения проблем интеграции облачных приложений привело к созданию технологии интеграционных облачных платформ Integration Platform as Service (iPaaS).

Ключевыми функциями платформы iPaaS являются [3]:

– средства поддержки потоков интеграции;

- средств разработки и сопровождения жизненного цикла интеграции;
- управление и мониторинг потоков приложений;
- обеспечение множественности владельцев приложений.

Обзор современного мирового состояния предложений по интеграции приведен в [4]. Там же показано, что основные современные решения реализуют не все ключевые функциями платформы iPaaS.

В работе [5] рассмотрены основные варианты построения интеграционной платформы и обосновано использование в качестве базового варианта построения интеграционной платформы решения по миграции данных.

Принципы построения интеграционной платформы

Платформа интеграции, реализующая решение по миграции данных между IT-системами, представленными облачными и не облачными системами, должна удовлетворять следующим принципам:

- платформа выполняется по модели SaaS;
- платформа имеет визуальный редактор настройки маршрутов и трансформаций. Этот редактор расположен в облаке;
- одним из режимов работы платформы должен быть такой, при котором поток пользовательских данных движется только в пределах пользовательской инфраструктуры (не уходит в облако);
- поток пользовательских данных должен быть защищен;
- поскольку платформа расположена в облаке, то протокол взаимодействия отдельных частей должен быть распространенным и поддерживаемым в условиях работы облака, а также межоблачного взаимодействия;
- отказ одного из приложений не должен влиять на работоспособность платформы в целом.

Платформа интеграции, реализованная как решение по миграции данных, должна обеспечивать:

- возможности горизонтального масштабирования;
- поддержанию необходимого уровня отказоустойчивости;
- реализацию многопользовательской среды.

Горизонтальное масштабирование должно позволять при необходимости увеличивать количество однотипных компонент для повышения общей производительности платформы.

Для поддержания необходимого уровня отказоустойчивости должна быть предусмотрена возможность поддерживать несколько реплик составляющих (и подсистем, и компонентов) для возможности продолжить работу в случае возникновения отказа.

Организация компонент платформы интеграции

Интеграционная платформа состоит из следующих компонент:

- компонент моделирования маршрутов интеграции;
- компонент применения планов маршрутизации и трансформации;
- компонент настройки промежуточных трансформаций;
- компонент управления развертыванием исполнительных модулей;
- исполнительных модулей;
- компонент оповещения о текущем ходе процесса.

Далее будут рассмотрены назначение и организация компонент платформы.

Компонент моделирования маршрутов интеграции предназначен для обеспечения решения задачи простой и удобной настройки интеграции. Для этой цели компонент должен включать визуальный редактор, с помощью которого пользователь может произвести настройку интеграции на интуитивно понятном уровне. Наиболее подходящим для решения

указанной задачи представляется реализация компонента в форме интерактивного редактора маршрутов.

Компонент применения планов маршрутизации и трансформации предназначен для приведения в действие интеграционных планов, заданных пользователем. Компонент должен обеспечивать:

- наличие необходимого числа коннекторов к системам;
- предоставление разнообразных форм трансформации сообщений;
- обеспечение необходимого для приложений уровня производительности.

Для обеспечения требований по производительности данный компонент должен быть горизонтально масштабируемым. Для упрощения масштабируемости компонент должен быть реализован как отдельное приложение.

Платформа интеграции должна обеспечивать трансформацию данных из формата одной системы в формат другой. Этот процесс также должен управляться пользователем. Для задания правил преобразования данных одной пользовательской системы в данные другой пользовательской системы при миграции, предназначен компонент настройки промежуточных трансформаций. Так же как к любому компоненту, реализующему настройку тех или иных процессов пользователем, основные требования к компоненту — наглядность описания процесса трансформаций, понятная визуализация процесса трансформаций. Для реализации этих требований наиболее подходящей представляется реализация компонента настройки промежуточных трансформаций в виде интерактивного редактора трансформаций.

Исполнительные модули, которые реализуют собственно процесс миграции данных, должны разворачиваться динамически. Это связано с тем, что исполнительные модули являются специфическими для каждого клиента. Поэтому при каждой регистрации нового клиента требуется

выделение исполнительных модулей, соответствующих специфике этого клиента. Управление развертыванием исполнительных модулей и их администрированием входит в задачи компонента управления развертыванием исполнительных модулей.

После того как процесс миграции данных начал исполняться, пользователь должен иметь возможность получать максимально возможную информацию о ходе выполнения процесса. На основании этой информации пользователь должен иметь возможность принятия оперативных решений по изменению настроек интеграционного процесса.

Основные требования к компоненту оповещения о текущем ходе процесса:

- возможность получения данных в режиме, приближенном к режиму реального времени;
- возможность визуализации получаемых данных (их предоставления в виде графиков, отчетов и т.д.).

Это обуславливает целесообразность построения данного компонента в виде графического редактора настройки получаемых и отображаемых данных о текущем ходе процесса.

Концептуальная модель системы управления маршрутизацией, коммутацией и трансформацией потоков данных

Решение интеграции в облаке лучше всего подпадает под архитектуру Master/Worker (мастер/работник). Архитектура Master/Worker представляет собой применение клиент-серверного подхода для реализации распределенных вычислений. Мастер (сервер) регистрирует подключения клиентов (работников) и, по их запросу, передает задания для вычислений. Работники рассматриваются как ненадежные и не требующие наличия постоянного соединения с мастером. Пользователь производит настройку на мастере, которой распределяет задания

(интеграционные планы) на исполнительные приложения. Проблема масштабирования в этом случае решается за счет динамического изменения количества работников, а так же за счет разделения интеграционных планов между работниками.

В Платформе каждое приложение идентифицируется с помощью определенного значения, которое генерируется при его запуске. Это значение уникальным образом идентифицирует экземпляр и сессию работы приложения. Все приложения в платформе разделяются на:

- Мастера;
- Работники;
- Очереди (Queue).

Мастер представляет собой центральное приложение, которое отвечает за регистрацию пользователей, настройку и презентацию настройки пользователю (визуализацией маршрутов и трансформаций), а так же занимается развертыванием рабочих процессов.

Пользователь взаимодействует с Мастером посредством веб-интерфейса (по HTTP протоколу), в то время как остальные приложения (Работники и Очереди) используют для взаимодействия с мастером REST-интерфейс.

Все остальные приложения при начале работы должны быть зарегистрированы в Мастере, а при ее завершении — также уведомить Мастера.

При начале работы приложение отправляет по REST-интерфейсу следующую команду:

/announce?id=<id>&type=<type>, где *id* — идентификатор сессии работы приложения, а *type* — тип приложения (Работник/Очередь).

При получении этой информации Мастером по команде POST происходит регистрация данного приложения во внутренних структурах

данных мастера, и он готов принимать запросы от зарегистрированного приложения.

При завершении работы приложение отправляет по REST-интерфейсу следующую команду:

/failure?id=<id>, где *id* — идентификатор сессии работы приложения.

При получении этой информации Мастером по команде POST происходит дерегистрация данного приложения во внутренних структурах, а зависимые элементы приложения мигрируют на существующие работающие приложения такого же типа.

Работник представляет собой приложение, содержащее исполнительный компонент и готовое выполнять интеграционные планы на мощностях, выделенных процессу операционной системой. В ходе своей деятельности Работник устанавливает несколько соединений. Одно из таких соединений является управляющим (соединение с мастером), с помощью него Работник получает команды и отдает статистику. Также устанавливается соединение, через которое перемещаются пользовательские данные.

При запуске приложения Работника генерируется уникальный идентификатор сессии этого приложения.

Работник соединяется с Мастером по «хорошо известному» доменному имени Мастера, заданному в конфигурационном файле Работника.

Для того чтобы зарегистрировать приложение у Мастера и получать от него информацию, позволяющую работнику выполнять действия по миграции, работник делает POST запрос на адрес Мастера. В ответ приходит сообщение с указанием текущей конфигурации маршрутов, и адресом приложения — очереди сообщений.

Работник начинает выполнять свой рабочий цикл — периодически опрашивать очередь на предмет новых сообщений. При поступлении сообщения работник пытается применить переданную конфигурацию. В случае невозможности передачи данных в соответствии с указанной конфигурацией, приложение-работник отправляет POST запрос на адрес Мастера для своей дерегистрации.

Приложение Очередь служит промежуточным звеном между Мастером и Работником. Оно необходимо для решения проблем масштабирования. При установлении соединения Мастера с Работником Мастер передает Работнику информацию об очереди, с которой должен быть ассоциирован рабочий процесс. Помимо доставки сообщений Очередь еще выполняет функцию наблюдения за работоспособностью работника. Это происходит при помощи механизма heartbeat'ов — работник получает сообщения от очереди через равные промежутки времени (polling). Если данный промежуток времени превышен, а ответ от работника не получен, то это означает, что работник находится в недопустимом состоянии. В этом случае происходит уведомление всех заинтересованных приложений (в частности, Мастера).

При запуске приложения типа Очередь генерируется уникальный идентификатор сессии этого приложения. Очередь соединяется с мастером по «хорошо известному» доменному имени мастера, заданному в конфигурационном файле очереди.

Для того чтобы зарегистрировать приложение у мастера, очередь делает POST запрос на адрес мастера. После этого приложение Очередь готова принимать запросы на создание очередей и производить запись сообщений в очереди.

После получения POST сообщения о регистрации очереди у мастера, приложение Очередь создает очередь и возвращает ее идентификатор Мастеру с помощью REST интерфейса.

Далее Очередь получает GET и POST сообщения. При получении GET сообщения очередь возвращает одно сообщение, которое было ранее поставлено в очередь. При получении POST сообщения приложение добавляет одно сообщение в очередь.

Таким образом, интеграционная платформа состоит из нескольких приложений, которые общаются друг с другом путем передачи сообщений по протоколу HTTP.

Взаимодействие приложений схематично можно представить так, как это показано на рисунке 1.

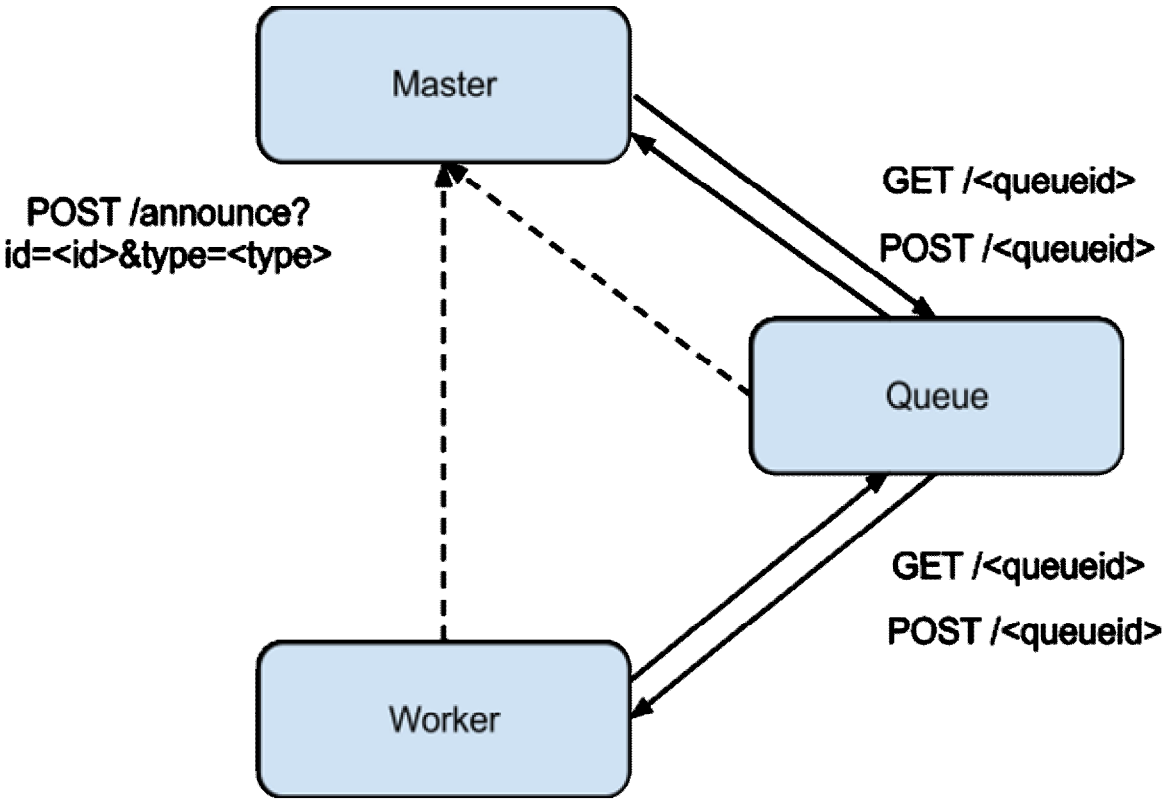

Рисунок 1 – Схема взаимодействия приложений платформы

Жизненный цикл каждого приложения состоит из двух фаз: фазы инициализации и исполнения.

Пунктирными линиями на рисунке 1 показаны действия компонентов на этапе инициализации.

При инициализации каждый компонент обращается с POST запросом по «хорошо известному» URL мастера. Таким образом, приложения дают сигнал о том, где они расположены и как к ним обращаться.

В общем случае фаза инициализации происходит следующим образом. После получения запроса Мастер регистрирует приложение в базе данных. Если это необходимо, то Мастер обращается к приложению Очереди с запросом на выделение очереди. После этого мастер отдает инициализируемому приложению ответ, в котором содержится необходимая для приложения информация и текущая полная конфигурация приложения, как это показано на рисунке 2.

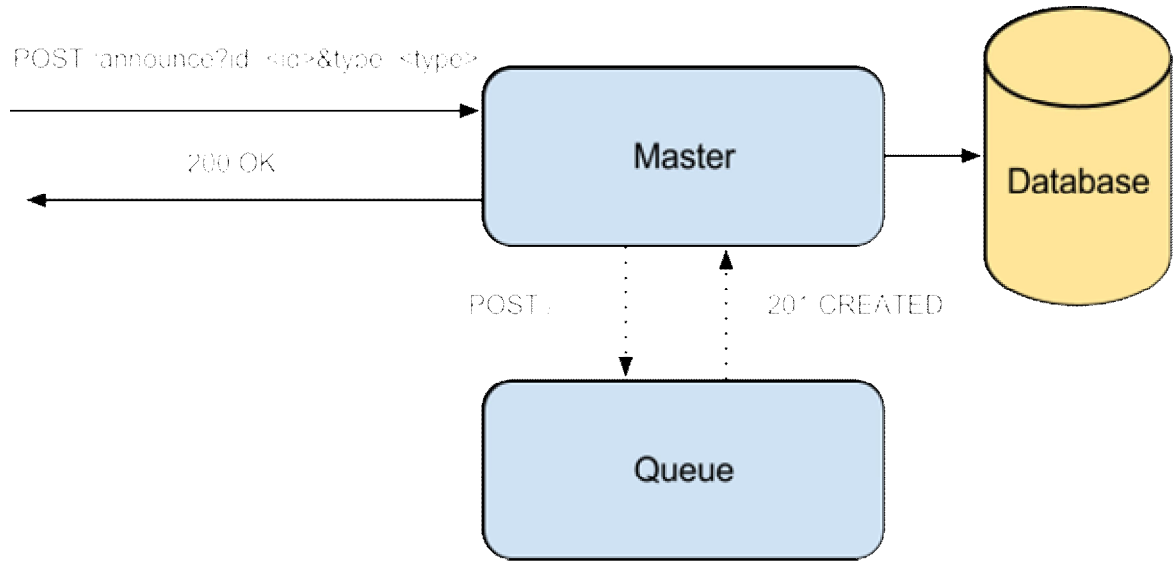

Рисунок 2 - Взаимодействие приложений по запросу на инициализацию приложения

Работа выполнена при финансовой поддержке Минобрнауки России по государственному контракту от 31.07.2012 г. № 14.514.11.4001 в рамках ФЦП «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2007-2013 годы».

Список литературы:

1. How to integrate with the cloud. David Linthicum. InfoWorld. [Электронный ресурс]. — http://www.infoworld.com/d/cloud-computing/how-integrate-the-cloud-714.
2. Интеграция: большой вызов облакам: [Электронный ресурс]. — http://www.mulesoft.com/integration-clouds-big-challenge.

1. Integration Platform as a Service: Moving Intergation to the Cloud. Gartner RAS Core Research Note G00210747, Massimo Pazzini, Benoit J. Lheureux, 7 march 2011.

2. Еловиков А.Е. Новые тенденции интеграции в облаке / А.Е. Еловиков // Политематический сетевой электронный научный журнал Кубанского государственного аграрного университета (Научный журнал КубГАУ) [Электронный ресурс]. – Краснодар: КубГАУ, 2012. – №09(083). С. 485 – 494. – IDA [article ID]: 0831209036. – Режим доступа: http://ej.kubagro.ru/2012/09/pdf/36.pdf, 0,625 у.п.л., импакт-фактор РИНЦ=0,577

3. Еловиков А.Е. Принципы построения облачной платформы / А.Е. Еловиков // Политематический сетевой электронный научный журнал Кубанского государственного аграрного университета (Научный журнал КубГАУ) [Электронный ресурс]. – Краснодар: КубГАУ, 2013. – №04(088). С. 498 – 508. – IDA [article ID]: 0881304033. – Режим доступа: http://ej.kubagro.ru/2013/04/pdf/33.pdf, 0,688 у.п.л., импакт-фактор РИНЦ=0,577

References

1. How to integrate with the cloud. David Linthicum. InfoWorld. [Jelektronnyj resurs]. — http://www.infoworld.com/d/cloud-computing/how-integrate-the-cloud-714.

2. Integracija: bol'shoj vyzov oblakam: [Jelektronnyj resurs]. — http://www.mulesoft.com/integration-clouds-big-challenge.

3. Integration Platform as a Service: Moving Intergation to the Cloud. Gartner RAS Core Research Note G00210747, Massimo Pazzini, Benoit J. Lheureux, 7 march 2011.

4. Elovikov A.E. Novye tendencii integracii v oblake / A.E. Elovikov // Politematicheskij setevoj jelektronnyj nauchnyj zhurnal Kubanskogo gosudarstvennogo agrarnogo universiteta (Nauchnyj zhurnal KubGAU) [Jelektronnyj resurs]. – Krasnodar: KubGAU, 2012. – №09(083). S. 485 – 494. – IDA [article ID]: 0831209036. – Rezhim dostupa: http://ej.kubagro.ru/2012/09/pdf/36.pdf, 0,625 u.p.l., impakt-faktor RINC=0,577

5. Elovikov A.E. Principy postroenija oblachnoj platformy / A.E. Elovikov // Politematicheskij setevoj jelektronnyj nauchnyj zhurnal Kubanskogo gosudarstvennogo agrarnogo universiteta (Nauchnyj zhurnal KubGAU) [Jelektronnyj resurs]. – Krasnodar: KubGAU, 2013. – №04(088). S. 498 – 508. – IDA [article ID]: 0881304033. – Rezhim dostupa: http://ej.kubagro.ru/2013/04/pdf/33.pdf, 0,688 u.p.l., impakt-faktor RINC=0,577