УДК 004.89
DOI: 10.15827/0236-235X.122.233-238
Дата подачи статьи: 15.09.17
2018. Т. 31. № 2. С. 233–238

ТЕХНОЛОГИЯ ВЗАИМОДЕЙСТВИЯ СЕРВИСОВ ОБЛАЧНОЙ ПЛАТФОРМЫ IACPaaS С ВНЕШНИМ ПРОГРАММНЫМ ОБЕСПЕЧЕНИЕМ

В.В. Грибова [1], *д.т.н., зам. директора, gribova@iacp.dvo.ru*
Ф.М. Москаленко [1], *к.т.н., старший научный сотрудник, philiptm@iacp.dvo.ru*
В.А. Тимченко [1], *к.т.н., старший научный сотрудник, vadim@iacp.dvo.ru*
Л.А. Федорищев [1], *к.т.н., научный сотрудник, fedorischev@iacp.dvo.ru*

[1] *Институт автоматики и процессов управления ДВО РАН, ул. Радио, 5, г. Владивосток, 690041, Россия*

Современные облачные технологии обладают рядом преимуществ перед другими видами ПО (упрощение установки, сопровождения, доступа, командной работы и т.д.). В то же время существуют задачи, которые невозможно выполнить на облачных платформах (или их выполнение неоптимально). Поэтому очевидна необходимость в создании стандартов, интерфейсов, систем, позволяющих объединять возможности ПО разных видов. В данной статье рассмотрены архитектура и технология взаимодействия облачной платформы IACPaaS и внешнего традиционного ПО.

Приведена архитектура взаимодействия облачной платформы и внешнего ПО, которая включает пять основных компонентов: два из них на платформе – сервис и агент-посредник и три на внешней системе – веб-сервер, программа-посредник, внешнее ПО. Описаны особенности каждого из этих компонентов.

Представлена технология разработки сервиса с учетом приведенной архитектуры, состоящая из четырех основных этапов: разработка сервиса на платформе, разработка внешнего ПО, установка веб-сервера, разработка коммуникационного ПО – программы-посредника. Описаны особенности каждого из этих этапов.

Названы необходимые требования при разработке агентов на платформе, спецификации параметров при отправке и получении сообщений. Представлены соответствующие примеры.

Приведены примеры успешного использования описываемой технологии при решении задач транспортного моделирования. Показано, что основная вычислительная нагрузка сосредоточена на внешнем ПО, тогда как на платформе IACPaaS находятся сервисы визуализации результатов этих вычислений. Продемонстрировано поэтапное выполнение рассмотренной технологии.

***Ключевые слова**: технология, сервис, облачная платформа, агенты, транспортное моделирование*.

Развитие облачных технологий обусловило активное использование сервисных платформ для разработки ПО [1–3]. Облачные платформы имеют ряд неоспоримых преимуществ, среди которых можно выделить простоту установки и сопровождения ПО, упрощение командной работы пользователей, расширение аудитории потенциальных пользователей и др.

Несмотря на все очевидные преимущества облачных технологий, другие типы ПО не потеряли своей актуальности: традиционное ПО (не SaaS-приложения) по-прежнему широко используется (особенно на высокопроизводительных суперкомпьютерных платформах) для обработки данных большого объема либо для расчетов, имеющих значительную вычислительную сложность. При этом анализ программных систем демонстрирует непрерывный рост гибридных вычислений на гетерогенных архитектурах [4]. В настоящее время активно ведутся работы по стандартизации таких вычислений. В частности, организацией OGF разработан стандарт Открытого интерфейса для облачных вычислений (OCCI), который описывает взаимодействие на уровне облака и пригоден для моделей IaaS, PaaS, SaaS [5, 6]. Однако названные известные стандарты и интерфейсы не подходят для объединения облачной платформы с традиционным ПО. В данной статье предлагается метод организации такого взаимодействия с традиционным ПО на примере облачной платформы IACPaaS.

Платформа IACPaaS [7] предназначена для разработки оболочек интеллектуальных сервисов (прежде всего специализированных) и их компонентов по нескольким технологиям, разработки с их помощью прикладных интеллектуальных облачных сервисов, а также использования этих сервисов для решения задач. Сами оболочки также предоставляются как облачные сервисы платформы. Для решения задачи использования внешнего традиционного ПО предложена специализированная технология удаленного взаимодействия платформы IACPaaS, аналогов которой в других облачных платформах не было найдено. Целью данной работы является описание указанной технологии удаленного взаимодействия, ее архитектуры и примеров применения, а также обсуждение полученных преимуществ.

Архитектура удаленного взаимодействия платформы

Архитектура удаленного взаимодействия платформы IACPaaS с внешним ПО основывается на агентном подходе, стандартном механизме

HTTP-запросов [8] и возможности запуска выполняемых exe-файлов внешнего ПО из других программ, расположенных на веб-сервере.

Для отправки HTTP-запросов из программных агентов платформы на внешние адреса на платформе IACPaaS разработан специализированный агент-посредник, который инкапсулирует все необходимые задачи по обработке HTTP-запросов, описанные выше (работа с адресом, операции GET, POST, параметры сообщения). Агент-посредник работает с агентами сервиса с помощью механизма шаблонов сообщений. На платформе IACPaaS для разработки сервисов используются решатели задач. Решатель задач – это набор агентов, обменивающихся между собой сообщениями, которые формируются по шаблонам. Таким образом, агенты сервиса взаимодействуют не напрямую с внешним ПО, а через сообщения с агентом-посредником.

Для коммуникации с внешним ПО необходимо дополнительное программное средство, работающее на веб-сервере и способное по мере необходимости вызывать это внешнее ПО. С этой целью на удаленной стороне необходима разработка специальной внешней системы.

На рисунке 1 представлены схема удаленного взаимодействия облачной платформы IACPaaS с внешним ПО через внешнюю систему и основные компоненты, участвующие в процессе, стрелками указаны направления передачи данных между ними.

Технология удаленного взаимодействия платформы

В соответствии с описанной архитектурой технология удаленного взаимодействия должна включать пять обязательных элементов: два из них (сервис и агент-посредник) на платформе и три (веб-сервер, программа-посредник, внешнее ПО) на внешней системе. Таким образом, технология разработки мультиагентных интеллектуальных сервисов с использованием удаленного взаимодействия с внешним ПО состоит из следующих этапов: разработка сервиса на платформе, разработка внешнего ПО, установка веб-сервера, разработка коммуникационного ПО – программы-посредника. (Агент-посредник – это часть платформы IACPaaS, его разрабатывать не требуется.)

Разработка сервиса на платформе основывается на базовой технологии [9] и состоит из этапов разработки:

– информационных единиц, используемых решателем задач;
– решателя задач сервиса;
– агентов решателя задач;
– новых шаблонов сообщений;
– пользовательского интерфейса сервиса.

Специальные требования при разработке агентов, непосредственно взаимодействующих с *агентом-посредником*, состоят в том, что в множестве их блоков продукций в обязательном порядке должны присутствовать блок продукций, множество шаблонов выходных сообщений которого содержат шаблон «*Шаблон Данные для внешнего запроса*», причем выполняется этот блок продукций после предыдущего описанного блока (возможно, это один и тот же блок продукций), и блок продукций, работа которого инициируется сообщениями, сформированными по шаблону «*Шаблон Ответ на внешний запрос*», причем выполняется этот блок продукций после работы предыдущего описанного блока (возможно, это один и тот же блок продукций).

Перечисленные шаблоны сообщений присутствуют в Фонде платформы IACPaaS в специальном разделе. В классах на языке программирования, представляющих процедурные части этих шаблонов сообщений, инкапсулированы доступные разработчикам агентов методы обработки (формирования, чтения и модификации) содержимого сообщений, созданных по этим шаблонам. Данные требования необходимо учесть на этапе формирования декларативной спецификации агента, а также при написании его исходного кода.

Рассмотрим этот процесс более детально.

Разработка агентов решателя задач. Для связи с внешним ПО агенту, входящему в состав решателя задач некоторого облачного сервиса платформы, необходимо отправить сообщение агенту-посреднику по шаблону «*Шаблон Данные*

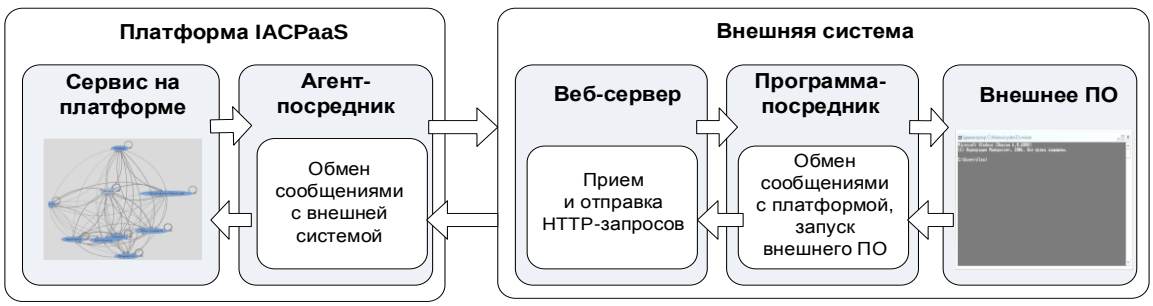

Рис. 1. Схема удаленного взаимодействия

Fig. 1. A remote interaction scheme

для внешнего запроса», который указывается на 4-м шаге в описании множества шаблонов выходных сообщений блока продукций агента. Шаблон «*Шаблон Данные для внешнего запроса*» описывает формат передачи данных из агента платформы внешней системе. Данный формат включает следующие возможности.

Передача параметров: используется метод *addParam*(<ключ>, <значение>).

Передача файлов: используется метод *addFile*(<имя файла (ключ), данные файла в формате BLOB (бинарные данные)>.

Указание URL-адреса внешней системы: добавление параметров и файлов необязательно; обязательным является только указание адреса внешней системы.

В API платформы IACPaaS для данного шаблона используется класс ExternalRequestDataMessage.

Пример отправки запроса. Создаем сообщение-запрос по шаблону:

ExternalRequestDataMessage mess = rc.externalRequestDataMessage.create();

Добавляем параметры:

mess.addParam("param1", "value1");

mess.addParam("param2", "value2");

Добавляем файлы:

Blob b1 = ...; Blob b2 = ...;

mess.addFile("file1", b1);

mess.addFile("file2", b2);

Указываем адрес внешней системы, на который будет отправлен запрос:

mess.setUrl("http://site.ru/test.php");

После завершения работы внешняя система отправляет результат обратно сервису платформы.

Следующая задача заключается в необходимости получения и обработки ответа от внешней системы. Для обработки агентом результата внешнего запроса в описание агента добавляется шаблон входного сообщения «*Шаблон Ответ на внешний запрос*». Формат данного шаблона сообщения хранит только один параметр – массив байтов, в который может быть помещена любая информация извне, без каких-либо правил со стороны платформы. На платформе IACPaaS для данного шаблона используется класс ExternalRequestReplyMessage.

В коде агента необходимо создать функцию runProduction с ожиданием сообщения класса ExternalRequestReplyMessage:

public void runProduction(ExternalRequestReplyMessage msg ...)

Ответ от внешней системы приходит в переменной типа BLOB (бинарные данные):

Blob b = msg.getAnswer();

Пример извлечения необходимого массива байтов с помощью функции getBytes():

<Массив полученных данных> = b.getBytes()

Внешняя система

Разработка внешнего ПО. Внешнее ПО может быть разработано как независимо от сервиса на платформе, так и с учетом некоторых его требований. Основное требование для внешнего ПО – возможность его запуска другим ПО, имеющим выход в Интернет.

Веб-сервер. В качестве веб-сервера выбирается одна из известных существующих программ, например Apache, IIS, nginx и другие. Установка веб-сервера производится на компьютер, либо с установленным требуемым внешним ПО, либо находящийся в локальной сети с ним. Веб-сервер выполняет роль интернет-коммуникатора для связи между платформой IACPaaS и внешним ПО. В отличие от другого ПО веб-сервер должен быть всегда подключен к Интернету, в его задачи входят ожидание запросов от сервисов платформы IACPaaS, передача их программе-посреднику, отправка результатов обратно сервисам платформы IACPaaS.

Программа-посредник. Программа-посредник может быть разработана на веб-сервере на любом языке программирования. Однако предполагается использовать для этой цели языки, применяющиеся в веб-программировании: PHP, Perl, Java и другие, так как именно они обеспечивают стандартные функции для работы с интернет-запросами.

Перечислим задачи программы-посредника.

1. Принять поступивший запрос от сервиса платформы IACPaaS через веб-сервер. Сервис платформы IACPaaS отправляет запросы по HTTP-протоколу с помощью метода POST. Таким образом, в программе-посреднике необходимо получить данные по указанному протоколу и методу. Пример получения параметра запроса на языке PHP: *$param = $_REQUEST['reqParam']*, где *$param* – произвольная переменная PHP для хранения данных; *'reqParam'* – название параметра, значение которого необходимо получить. Название параметра должно соответствовать переданному в запросе из сервиса платформы IACPaaS. Параметров может быть любое количество.

2. Вызвать внешнее ПО и передать ему параметры, полученные из поступившего запроса.

Пример вызова исполняемого файла программы на языке PHP: exec(<Путь к исполняемому файлу на сервере>).

3. Получить результат работы запущенной программы. Это стандартная задача о взаимодействии между разными программами на одном компьютере, и в данной статье она не описывается.

4. Отправить полученный результат через веб-сервер обратно сервису платформы IACPaaS. Отправляемые данные требуется привести к заранее определенному формату, согласованному с обработкой этого результата на сервисе платформы IACPaaS. Особых требований к отправляемому от-

вету технологией не накладывается: это могут быть произвольный текст, бинарные данные, отформатированные данные, файлы и т.д.

Примеры разработки прикладных сервисов

Технология удаленного взаимодействия между облачной платформой IACPaaS и внешним ПО была успешно протестирована в проектах транспортного моделирования города и транспортных коммуникаций между городами. В данных проектах основная вычислительная нагрузка сосредоточена на внешнем ПО, на платформе IACPaaS находятся сервисы визуализации результатов этих вычислений.

Сервис транспортных коммуникаций. Сервис предназначен для моделирования торговых потоков между территориями с определением наиболее вероятных значений для коммуникационных систем в условиях неполноты информации. Математическая модель задачи относится к классу выпуклых задач математического программирования, предполагает численное решение нелинейной оптимизационной задачи с линейными ограничениями и реализуется на высокопроизводительной серверной платформе.

В данном сервисе пользователь с помощью графического интерфейса может менять параметры коммуникаций и видеть результат изменения потоков товаров по городам. Интерактивное редактирование графа предполагает изменение параметров как вершин графа (рис. 2), так и его дуг. При нажатии на соответствующий элемент графа пользователь получает структурированную информацию об объекте, которую может изменить. Используя полученный граф, пользователь может добавить к нему реальную карту, чтобы более точно оценить результаты визуализации.

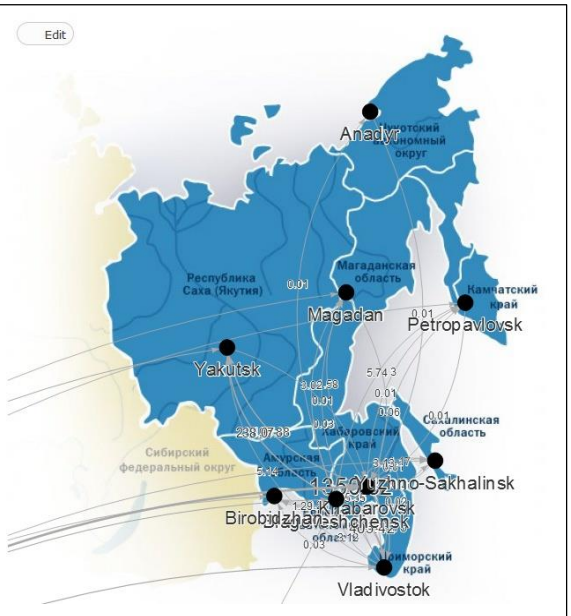

Рис. 2. Визуализация транспортных коммуникаций на облачном сервисе

Fig. 2. Visualization of transport communications in the cloud service

Технология реализации сервиса транспортных коммуникаций. Приведенный сервис транспортных коммуникаций разрабатывался в соответствии с описанной выше технологией удаленного взаимодействия облачной платформы IACPaaS с внешним ПО (математическое ПО, вычисляющее транспортную модель). В соответствии с представленной технологией были выполнены все четыре этапа.

Этап 1. Сервис на платформе IACPaaS.

1. *Разработка информационных единиц, используемых решателем задач*.

На данном этапе был создан информационный ресурс, представляющий декларативную спецификацию агента транспортной модели для решателя задачи, внутри которого были указаны стандартные и специальные шаблоны сообщений. Стандартные шаблоны сообщений: «*Шаблон Запрос от агента Вид*», «*Шаблон Отобразить окно*», «*Шаблон Вернуть инфоресурс в окно*», «*Шаблон Вернуть строку в окно*». Специальные шаблоны сообщений: «*Шаблон Данные для внешнего запроса*», «*Шаблон Ответ на внешний запрос*». Все шаблоны записаны в ресурс агента (см. http://www.swsys.ru/uploaded/image/2018_2/2018-2-dop/5.jpg).

2. *Разработка решателя задач сервиса на платформе.*

На данном этапе в решателе задач сервиса были указаны агент транспортной модели – «*Транспортная модель. Интерфейсный агент*» и стартовая страница – «*Транспортная модель*» (см. http://www.swsys.ru/uploaded/image/2018_2/2018-2-dop/6.jpg). Остальные элементы (входные, выходные, собственные и временные инфоресурсы) отсутствуют, корневой агент совпадает с интерфейсным агентом.

3. *Разработка агентов решателя задач*.

На данном этапе был разработан интерфейсный агент сервиса. В агенте был подключен клиентский интерфейсный модуль для визуализации графа транспортной модели. Также в данном агенте был реализован обмен данными с внешней системой в соответствии с описанной технологией.

4. *Разработка шаблонов сообщений*.

Новых шаблонов сообщений для данной задачи создано не было.

5. *Разработка пользовательского интерфейса сервиса на платформе.*

Пользовательский интерфейс сервиса реализован как клиентский модуль и подключен на шаге 3 к интерфейсному агенту. Его задача – визуализация графа транспортной модели на основе получаемых от внешнего ПО данных.

Этап 2. Внешнее ПО.

Внешнее ПО в данной задаче – это программа, вычисляющая математическую модель.

Этап 3. Веб-сервер.

На внешней системе был установлен веб-сервер Apache/2.2.29 (Linux/SUSE).

Этап 4. Программа-посредник.

В программе-посреднике реализованы функции приема двух видов запросов: на получение первоначальной модели и на изменение ее параметров. Взаимодействие между программой-посредником и внешним ПО осуществляется с помощью записи-чтения общего файла. Внешнее ПО записывает в общий файл результаты своих вычислений, программа-посредник считывает их и отправляет сервису на платформе IACPaaS.

Сервис модели города. Сервис предназначен для интерактивного моделирования транспортных потоков в развивающейся городской инфраструктуре. Внешнее ПО строит математическую модель прогнозирования загрузки транспортной сети в результате синтеза гравитационной модели описания транспортных корреспонденций и мультимодальной задачи потокового равновесия с эластичным спросом. Решение задачи построения такой модели является очень трудоемким и вычислительно сложным процессом, поэтому оно размещается на внешней системе. На облачной платформе размещается внутренний сервис, который отображает пользователям результаты этого сложного моделирования в виде трехмерной визуализации (см. http://www.swsys.ru/uploaded/image/2018_2/2018-2-dop/7.jpg).

Представленный сервис также реализован по описанной в статье технологии удаленного взаимодействия облачной платформы IACPaaS с внешним ПО. Подробно технология реализации данного сервиса не описывается, так как она во многом похожа на предыдущий пример. Основное отличие в реализации данного сервиса заключается в том, что для его разработки дополнительно используется инструментальное средство платформы IACPaas – «Редактор графических сцен».

Заключение

Описанная технология была успешно применена в двух реальных проектах, связанных с транспортными задачами. Рассмотрим критерии, которые имели значение в их реализации.

1. Преимущества платформы IACPaaS. Сервисы получили все преимущества, которые дает платформа IACPaaS, и были реализованы современными веб-средствами визуализации.

2. Преимущества внешнего серверного кластера. Внешнее ПО в данном случае – это сложные математические программы, выполняемые на высокопроизводительном аппаратно-программном обеспечении, которых нет на платформе IAPCaaS.

3. Независимость платформы и внешнего ПО. Математическая система располагается отдельно от облачной платформы IACPaaS. Взаимодействие между системами происходит только посредством обмена сообщениями.

Таким образом, представленные проекты реализованных облачных сервисов показали актуальность применения описанной в статье технологии удаленного взаимодействия, реализованного на платформе IACPaaS. Проекты продемонстрировали возможность удобного использования и визуализации сложных математических результатов через Интернет благодаря сетевому взаимодействию разнородных программных средств. Агент-посредник, реализованный на платформе IACPaaS, инкапсулировал функциональность для решения задач по взаимодействию с внешним ПО, предоставив агентам платформы только необходимые структуры данных и шаблоны для заполнения. Таким образом, на платформе IACPaaS была успешно решена задача удаленного взаимодействия с внешним ПО.

Работа выполнена при частичной финансовой поддержке РФФИ, проекты №№ 16-07-00340, 15-07-03193.

Литература

1. Батура Т.В., Мурзин Ф.А., Семич Д.Ф. Облачные технологии: основные модели, приложения, концепции и тенденции развития // Программные продукты, системы и алгоритмы. 2014. № 3. С. 64–72.

2. Медведев А. Облачные технологии: тенденции развития, примеры исполнения // Современные технологии автоматизации. 2013. № 2. С. 6–9.

3. Рыбина Г.В. Интеллектуальные системы: от А до Я. Кн. 3: Проблемно-специализированные интеллектуальные системы. Инструментальные средства построения интеллектуальных систем. М.: Научтехлитиздат, 2015. 180 с.

4. Богданова В.Г., Горский С.А., Пашинин А.А. Сервис-ориентированные инструментальные средства для решения задач булевой выполнимости // Фундаментальные исследования. 2015. № 2–6. С. 1151–1156.

5. Open Cloud Computing Interface (OCCI). URL: http://occi-wg.org/community/ (дата обращения: 23.05.2017).

6. Майерсон Д. Чего следует ожидать клиентам от стандартов облачных сервисов. URL: https://www.ibm.com/developerworks/ru/library/cl-srvcstandardsgap/ (дата обращения: 23.05.2017).

7. Орландо Д. Модели сервисов облачных вычислений: программное обеспечение как сервис. 2012. URL: http://www.ibm.com/developerworks/ru/library/cl-cloudservices3saas/ (дата обращения: 29.05.2017).

8. Gribova V., Kleschev A., Krylov D., Moskalenko P., Timchenko V., Shalfeeva E. A cloud computing platform for lifecycle support of intelligent multi-agent internet-services. Proc. Intern. Conf. PEEE. Hong Kong, 2015, vol. 20, pp. 231–235.

9. Типы HTTP-запросов и философия REST. URL: https://habrahabr.ru/post/50147/ (дата обращения: 17.06.2017).

10. Грибова В.В., Клещев А.С., Крылов Д.А., Москаленко Ф.М., Тимченко В.А., Шалфеева Е.А. Базовая технология разработки интеллектуальных сервисов на облачной платформе IACPaaS. Ч. 1. Разработка базы знаний и решателя задач // Программная инженерия. 2015. № 12. С. 3–11.

Software & Systems
DOI: 10.15827/0236-235X.122.233-238

Received 15.09.17
2018, vol. 31, no. 2, pp. 233–238

TECHNOLOGY OF INTERACTION OF IACPaaS CLOUD PLATFORM SERVICES WITH EXTERNAL SOFTWARE

V.V. Gribova [1], *Dr.Sc. (Engineering), Deputy Director, gribova@iacp.dvo.ru*
F.M. Moskalenko [1], *Ph.D. (Engineering), Senior Researcher, philipmm@iacp.dvo.ru*
V.A. Timchenko [1], *Ph.D. (Engineering), Senior Researcher, vadim@iacp.dvo.ru*
L.A. Fedorishchev [1], *Ph.D. (Engineering), Research Associate, fedorischev@iacp.dvo.ru*

[1] *Institute of Automation and Control Processes Far Eastern Branch of RAS,*
Radio St. 5, Vladivostok, 690041, Russian Federation

Abstract. Modern cloud technologies have a number of advantages over other types of software (simplification of installation, maintenance, access, teamwork, etc.). At the same time, some tasks are impossible to perform on cloud platforms (or non-optimal to perform). Therefore, there is an obvious need in creating standards, interfaces, systems that allow combining capabilities of different software types. This article discusses the architecture and technology of interaction between the cloud platform IACPaaS and external traditional software.

The article describes the interaction architecture of the cloud platform and external software, which consists of five main components. Two of the components are on the platform: a service and a proxy agent. The others are on the external system: a web server, a proxy program and external software. The paper describes the features of each component.

The authors also present a service development technology taking into account the given architecture. It consists of four main stages: 1) service development on the platform, 2) external software development, 3) web server installation, 4) communication software development – a proxy program. The features of each stage are described.

The paper considers the necessary requirements for the development of agents on the platform and the specification of parameters when sending and receiving messages. There are corresponding examples.

The paper gives the examples of successful use of the proposed technology when solving transport modeling problems. In the examples, the main computing load is focused on the external software, whereas services for visualizing the results of these calculations are provided on the IACPaaS platform. One example demonstrates the phased implementation of the technology.

At the end of the article, there is a discussion of the advantages gained from applying the presented technology.

Keywords: technology, service, cloud platform, agents, transport modeling.

Acknowledgements. *The work has been done with the partial financial support of the Russian Foundation for Basic Research, project no. 16-07-00340, no. 15-07-03193.*

References

1. Batura T.V., Murzin F.A., Semich D.F. Cloud technologies: basic models, applications, concepts and trends of development. *Programmnye produkty, sistemy i algoritmy* [Software Programs, Systems and Algorithms]. 2014, no. 3, pp. 64–72 (in Russ.).
2. Medvedev A. Cloud technologies: trends of development, examples of execution. *Sovremennye tekhnologii avtomatizatsii* [Modern Automation Technologies]. 2013, no. 2, pp. 6–9 (in Russ.).
3. Rybina G.V. *Intellektualnye sistemy: ot A do Ya. Kniga 3. Problemno-spetsializirovannye intellektualnye sistemy. Instrumentalnye sredstva postroeniya intellektualnykh sistem* [Intellectual Systems: from A to Z. Book 3: Problem-specific Intelligent Systems. Instrumental Tools for Building Intelligent Systems]. Moscow, Nauchtekhlitizdat Publ., 2015, 180 p.
4. Bogdanova V.G., Gorsky S.A., Pashinin A.A. Service-oriented tools for solving of boolean satisfiability problem. *Fundamentalnye issledovaniya* [Fundamental Research]. 2015, no. 2-6, pp. 1151–1156 (in Russ.).
5. *Open Cloud Computing Interface (OCCI)*. Available at: http://occi-wg.org/community/ (accessed May 23, 2017).
6. Mayerson D. *Chego sleduet ozhidat klientam ot standartov oblachnykh servisov* [What Should Clients Expect from the Standards of Cloud Services]. Available at: https://www.ibm.com/developerworks/ru/library/cl-srvcstandardsgap/ (accessed May 23, 2017).
7. Orlando D. *Modeli servisov oblachnykh vychisleny: programmnoe obespechenie kak servis* [Models of Cloud Computing Services: Software as a Service]. 2012. Available at: http://www.ibm.com/developerworks/ru/library/cl-cloudservices3saas/ (accessed May 29, 2017).
8. Gribova V., Kleschev A., Krylov D., Moskalenko P., Timchenko V., Shalfeeva E. A Cloud Computing Platform for Lifecycle Support of Intelligent Multi-agent Internet-services. *Int. Conf. on Power Electronics and Energy Engineering (PEEE)*. Hong Kong, USA, Lancaster, Destech Publ., 2015, pp. 231–235.
9. *Tipy HTTP-zaprosov i filosofiya REST* [Types of HTTP Requests and the REST Philosophy]. Available at: https://habrahabr.ru/post/50147/ (accessed June 17, 2017).
10. Gribova V.V., Kleshchev A.S., Krylov D.A., Moskalenko F.M., Timchenko V.A., Shalfeeva E.A. A base technology for development of intelligent services with the use of IACPaaS cloud platform. Part 1. A development of knowledge base and problem solver. *Programmnaya inzheneriya* [Software Engineering]. 2015, no. 12, pp. 3–11 (in Russ.).