OSTIS-2015
(Open Semantic Technologies for Intelligent Systems)

УДК 004.822

ТЕХНОЛОГИЯ РАЗРАБОТКИ РЕШАТЕЛЕЙ ЗАДАЧ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ С ИСПОЛЬЗОВАНИЕМ ИНСТРУМЕНТАЛЬНЫХ СЕРВИСОВ ОБЛАЧНОЙ ПЛАТФОРМЫ IACPAAS

Крылов Д.А., Москаленко Ф.М., Тимченко В.А.

Федеральное государственное бюджетное учреждение науки Институт автоматики и процессов управления Дальневосточного отделения Российской академии наук, г. Владивосток, Россия

dmalkr@gmail.com
philipmm@iacp.dvo.ru
vadim@dvo.ru

В работе описана технология разработки агентов, входящих в состав мульти-агентных решателей задач прикладных интеллектуальных облачных сервисов, с использованием инструментальных (системных) сервисов платформы IACPaaS. Технология направлена на снижение за счет ее применения трудоемкости разработки и, прежде всего, сопровождения интеллектуальных систем как облачных мульти-агентных сервисов.
Ключевые слова: интеллектуальные системы, мульти-агентные системы, агентно-ориентированное программирование, облачные сервисы.

Введение

Разработка и сопровождение интеллектуальных систем (ИС) или систем, основанных на знаниях (состоящих, в общем случае, из решателя задач, пользовательского интерфейса и базы знаний), является чрезвычайно сложным и трудоемким процессом. Данный процесс имеет свои особенности, поэтому стандартная технология разработки программных систем не может быть напрямую спроецирована на разработку ИС. В первую очередь это связано с тем, что база знаний ИС должна разрабатываться и сопровождаться экспертами предметной области и быть им понятной. Описанные в литературе технологии разработки ИС предполагают, в частности, что база знаний формируется в результате взаимодействия эксперта предметной области с инженером знаний. Однако в работах [Грибова и др., 2010], [Kleshchev, 2011], [Грибова и др., 2013a], [Грибова и др., 2013b] указываются недостатки такого подхода, особенно проявляющиеся при сопровождении базы знаний. В результате на сегодняшний день можно говорить об отсутствии принятой в качестве стандарта технологии, позволяющей разрабатывать жизнеспособные адаптивные ИС. Наряду с этим по-прежнему ощущается острая потребность в средствах разработки ИС, а также в повторном использовании компонентов ИС. Важной задачей при этом также является обеспечение доступа пользователей к ИС и средствам их разработки на протяжении всего жизненного цикла ИС. Существующие же технологии основаны на традиционном подходе к сопровождению программных средств, при котором версия системы передается конечному пользователю и который не предусматривает оперативное устранение возможных ошибок, централизованного хранения и обновления онтологий и баз знаний, модификацию решателей задач, сводя эти процессы лишь к смене/обновлению версий системы.

Современный подход к разработке программных систем основан на использовании технологии облачных вычислений, которые, с одной стороны, обеспечивают доступность программных систем широкому кругу пользователей, с другой – позволяют на протяжении всего жизненного цикла программной системы осуществлять ее сопровождение, поскольку все компоненты системы остаются доступными ее разработчику. К настоящему времени при использовании данного подхода для разработки ИС созданы редакторы информационного наполнения ИС, компоненты которых можно использовать при создании новых ИС [Gennari et al., 2003], [Орлов и др., 2006a], [Protege, 2014]; разработаны и широко используются платформы [Орлов и др., 2006b], [Sanderson, 2009], [TopQuadrant, 2014], которые поддерживают отдельные этапы цикла разработки облачных программных систем.

Однако на сегодняшний день отсутствуют облачные платформы, полностью поддерживающие разработку, функционирование и сопровождение всех трех компонентов ИС (базы знаний, решателя задач, интеллектуального интерфейса), а также концепцию единообразного представления, хранения и повторного использования информационного наполнения и программных компонентов ИС.

Для решения вышеперечисленных проблем в области разработки и сопровождения жизнеспособных ИС в работах [Грибова и др., 2011], [Gribova et al., 2013] предложена концепция облачной платформы IACPaaS, поддерживающей следующие технологические принципы разработки, сопровождения и использования облачных ИС:

- все информационные ресурсы (онтологии, знания, данные) и декларативные компоненты решателей задач имеют единое унифицированное представление (в виде иерархической семантической сети) [Lehmann, 1992];
- формирование и сопровождение знаний осуществляется экспертами предметной области на основе моделей онтологий;
- пользовательский интерфейс редакторов для эксперта генерируется по модели онтологии;
- метод решения задачи разбивается на подзадачи, где каждой подзадаче соответствует агент;
- для доступа агентов к информационным ресурсам, имеющим унифицированное представление, предусмотрены программные интерфейсы;
- ИС предоставляется пользователю как облачный мульти-агентный сервис.

В данной работе описывается технология разработки решателей задач ИС на платформе IACPaaS, каждый из которых представляет собой множество взаимодействующих между собой посредством обмена сообщениями агентов.

1. Облачная платформа IACPaaS

Облачная платформа IACPaaS представляет собой программно-информационный интернет-комплекс, предоставляющий контролируемый доступ и единую систему администрирования для создания и использования интеллектуальных сервисов и их компонентов, поддержку функционирования агентов (через передачу и обработку сообщений, запуск блоков продукций – обработчиков сообщений). Комплекс основан на технологии облачных вычислений и обеспечивает удаленный доступ конечным пользователям к интеллектуальным системам, а разработчикам и управляющим – к средствам создания интеллектуальных систем и управления ими. Концептуальная четырехуровневая архитектура платформы IACPaaS представлена на рис. 1.

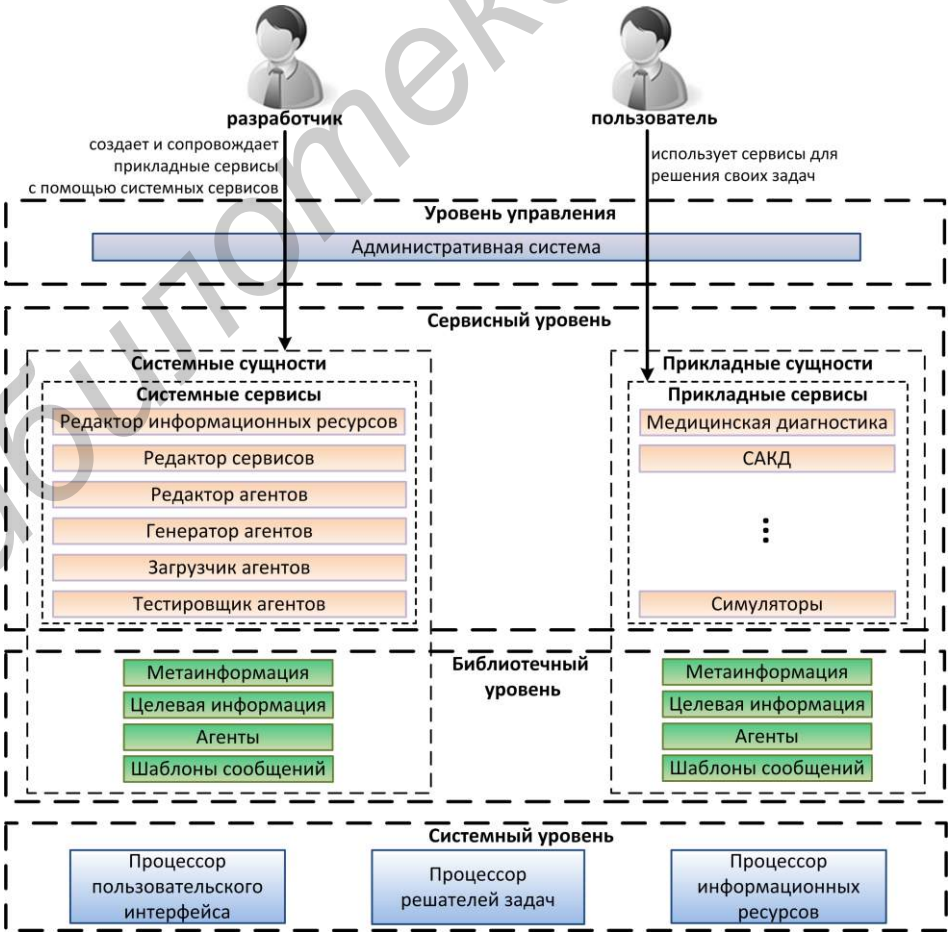

Рисунок 1 – Концептуальная четырехуровневая архитектура платформы IACPaaS

1. *Системный уровень* (уровень *виртуальной машины*). Виртуальная машина платформы IACPaaS состоит из процессора информационных ресурсов (ПИР), процессора решателей задач (ПРЗ) и процессора пользовательского интерфейса (ППИ), каждый из которых предназначен для поддержки соответствующих компонентов интеллектуальных систем. На данном уровне обеспечиваются доступ к Фонду информационных ресурсов платформы, запуск и работа сервисов, а также взаимодействие сервисов с пользователями.

2. *Библиотечный уровень*. Это уровень отдельных (повторно используемых при разработке и сопровождении) компонентов сервисов, к которым относятся агенты, шаблоны сообщений, используемые для взаимодействия между агентами, а также информационные ресурсы двух типов: представляющих метаинформацию и целевую информацию.

3. *Сервисный уровень*. Данный уровень представляет собой совокупность сервисов, каждый из которых представлен множеством взаимодействующих посредством обмена сообщениями агентов, обрабатывающих информационные ресурсы Фонда. Выделяются прикладные сервисы, создаваемые прикладными разработчиками для решения задач пользователей, и системные сервисы, необходимые для функционирования платформы и развития ее Фонда информационных ресурсов.

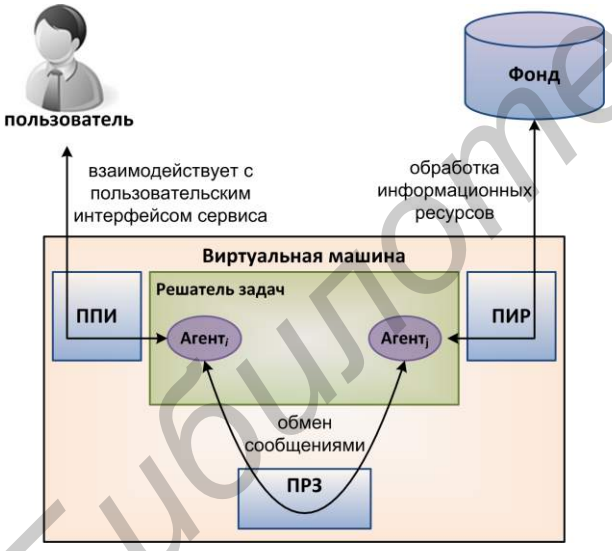

Рисунок 2 – Архитектурно-контекстная диаграмма сервисов

4. Уровень управления. Данный уровень представлен системным сервисом платформы *Административная система*, которая предназначена для обеспечения контролируемого доступа к функциональным возможностям платформы IACPaaS и управления правами доступа на использование прикладных и сервисов, разработанных на базе данной платформы.

Системные сервисы обеспечивают инструментальную поддержку технологии разработки прикладных облачных мульти-агентных сервисов и их компонентов (в т.ч. входящих в состав сервисов агентов). Технология разработки агентов и шаблонов сообщений, которые они могут принимать и/или отправлять, состоит из следующих традиционных для создания программных средств этапов [Фатрелл и др., 2004]: *разработка требований*, *проектирование*, *реализация*, *тестирование и отладка*, *ввод в эксплуатацию*.

2. Разработка агентов и шаблонов сообщений

2.1. Разработка требований к агенту

На этом этапе формулируются требования к функциональности агента; структуре его входных и выходных данных; структуре сообщений, посредством которых он взаимодействует с другими агентами; структуре информации, доступной в любом блоке продукций агента (при каждом обращении к агенту во время работы сервиса) и используемой для хранения собственных данных, настроек агента, управляющих логикой его работы и т.п. – локальной структуре данных (может отсутствовать, если все нужные данные доступны через сообщения и/или информационные ресурсы). Платформа IACPaaS не предоставляет средств для разработки и управления требованиями к агентам.

2.2. Проектирование агента

На данном этапе, также выполняемом без использования средств платформы, проектируется устройство и поведение агента как совокупности блоков продукций, обрабатывающих принимаемые сообщения. Для каждого блока продукций определен свой шаблон входных сообщений. Ввиду использования платформы IACPaaS на следующих этапах, при проектировании необходимо соблюсти следующие условия:

- проанализировать содержимое фонда платформы IACPaaS на наличие шаблонов сообщений, которые можно использовать при проектировании работы некоторых блоков продукций агента, и в случае обнаружения таких сущностей, их необходимо задействовать на этапе реализации агента;
- *Корневой агент* должен содержать блок продукций, выполнение которого инициируется сообщением по шаблону *Инициализирующее сообщение*;
- если агент должен завершать работу сервиса, то у него должен быть блок продукций, в процессе выполнения которого посылается сообщение по шаблону *Финализирующее сообщение*;
- агент *Интерфейсный контроллер* должен иметь блок продукций, выполнение которого инициируется сообщением по шаблону *Запрос от агента Вид*;
- для ответа агенту платформы *Вид* (с целью отображения интерфейса) агент сервиса должен

содержать блок продукций, в процессе выполнения которого посылается сообщение по одному из шаблонов: *Отобразить окно*, *Вернуть инфоресурс в окно*, *Вернуть строку в окно*.

2.3. Реализация агента

Реализация агента состоит в формировании в фонде платформы IACPaaS информационных ресурсов, представляющих декларативные спецификации разрабатываемого агента и используемых в нем новых шаблонов входных и выходных сообщений; генерации заготовок исходного кода агента и шаблонов сообщений по их декларативному описанию; написании кода агента и шаблонов сообщений (в частности, кода блоков продукций агента); получении байт-кода агента и шаблонов сообщений и загрузки его в фонд.

2.3.1. *Формирование информационных ресурсов*

Формирование в фонде информационного ресурса, представляющего декларативную спецификацию шаблона сообщений, выполняется по следующей схеме.

1. Создание с использованием Административной системы в фонде платформы IACPaaS нового информационного ресурса (с названием *Шаблон <имя>*, где *<имя>* – название шаблона, присвоенное ему на этапе проектирования) по метаинформации *Структура шаблона сообщений*, описывающей онтологию декларативных представлений шаблонов сообщений платформы IACPaaS.

2. Формирование содержимого созданного информационного ресурса с использованием системного сервиса *Редактор информационных ресурсов*, в котором процесс редактирования управляется метаинформацией *Структура шаблона сообщений*. Специфицирование шаблона сообщений состоит в задании:

- описания шаблона (на естественном языке);
- внутреннего имени шаблона (используемом при формировании имени класса шаблона сообщений на этапе генерации заготовки его исходного кода);
- требуемой структуры содержимого сообщений (если необходимо).

Формирование в фонде платформы информационного ресурса, представляющего декларативную спецификацию агента, выполняется по следующей схеме.

1. Создание с использованием *Административной системы* в фонде платформы IACPaaS нового информационного ресурса, представляющего декларативную спецификацию разрабатываемого агента, по метаинформации *Структура агента*, описывающей онтологию декларативных представлений агентов платформы.

2. Формирование содержимого созданного информационного ресурса с использованием системного сервиса *Редактор агентов*, в котором процесс редактирования управляется метаинформацией *Структура агента*. Специфицирование агента состоит в задании:

- описания агента (на естественном языке);
- внутреннего имени агента (используемом при формировании имени класса агента на этапе генерации заготовки его исходного кода);
- локальной структуры данных требуемого вида (если необходимо);
- для каждого блока продукций агента:
  - описания;
  - шаблона входных сообщений – сообщений, инициирующих выполнение данного блока продукций агента (путем создания ссылки на соответствующий информационный ресурс в фонде платформы IACPaaS);
  - шаблонов выходных сообщений – сообщений, создаваемых в процессе выполнения данного блока продукций агента и рассылаемых адресатам после завершения выполнения данного блока (если они есть).

2.3.2. *Генерация заготовок исходного кода*

На данном шаге с помощью системного сервиса *Генератор агентов* выполняется генерация заготовок исходного кода разрабатываемого агента и используемых в нем новых шаблонов входных и выходных сообщений по их декларативному описанию, а также получение байт-кода повторно используемых (пользовательских и встроенных в платформу IACPaaS) шаблонов сообщений (если таковые имеются) в виде jar-архивов сгруппированных в пакеты файлов, содержащих набор необходимых классов на языке программирования (в текущей реализации это Java), а также файлов, содержащих байт-код. Анализируя декларативную спецификацию агента, сервис формирует абстрактный класс на языке программирования с именем, совпадающим с внутренним именем агента, указанном в спецификации, который является подклассом системного класса облачной платформы *Agent* и содержит:

- конструктор класса, используемый платформой для создания и инициализации данного агента в виртуальной машине платформы IACPaaS;
- статический инициализирующий блок, содержащий описание блоков продукций агента (по количеству блоков продукций агента, указанных в его декларативном описании), используемый диспетчером сообщений платформы;
- набор внутренних статических классов (“создателей результатов”, количество которых совпадает с количеством блоков продукций агента, указанных в его декларативном описании), описывающих шаблоны тех (и только тех) сообщений, которые могут быть созданы и отправлены после окончания работы соответствующего блока продукций (т.е. выходных

для блока продукций сообщений) агента. Имя каждого такого класса есть внутреннее имя шаблона входного для блока продукции сообщения, к которому добавлен суффикс *MessageResultCreator*;

- набор методов-заглушек (количество которых совпадает с количеством блоков продукций агента, указанных в его декларативной спецификации) *void runProduction(…)*, соответствующих блокам продукций агента и имеющих два параметра (эти методы необходимо переопределить в классе-реализации):
  - типом первого параметра является класс, соответствующий шаблону входного для блока продукции сообщения;
  - типом второго параметра является класс "создатель результатов", описывающий шаблоны выходных для блока продукции сообщений.

На основе данного класса-заготовки агента разработчику необходимо создать класс-реализацию агента (наследующий от класса-заготовки) с именем *<внутреннее имя агента>Impl*. В теле каждого переопределяемого метода *void runProduction(…)* необходимо реализовать логику (алгоритм) работы соответствующего блока продукций. В классе-реализации можно, также, описать и реализовать множество вспомогательных методов, внутренних классов и т.п. для использования последних внутри методов *void runProduction(…)*.

Помимо абстрактного класса-заготовки агента сервис генерации заготовок исходного кода, анализируя декларативные спецификации всех шаблонов сообщений (входных и выходных), на которые присутствуют ссылки в спецификации агента, на основе каждого из них формирует класс на языке программирования с именем, *<внутреннее имя шаблона сообщений>Message*. Каждый такой класс является подклассом системного класса облачной платформы IACPaaS *Message* и содержит:

- конструктор класса, используемый платформой для создания и инициализации данного шаблона сообщений в виртуальной машине платформы IACPaaS;
- внутренний статический класс, содержащий два метода создания сообщений по данному шаблону для агентов и агентов-экземпляров, которым эти сообщения должны быть отправлены:
  - параметром первого метода является строковое имя агента, которому должно быть послано созданное сообщение;
  - параметром первого метода является указатель на экземпляр агента, которому должно быть послано созданное сообщение.

Сгенерированный класс кода шаблона сообщений можно оставить без изменений, либо дополнить вспомогательными методами (обычно, это методы чтения/модификации информационного ресурса сообщения), внутренними классами и т.п.

2.3.3. *Подготовка и загрузка байт-кода в фонд*

На данном шаге выполняется компиляция кода классов агента и шаблонов сообщений, объединение полученного в результате компиляции байт-кода в jar-архив и загрузка (или обновление) *class*-файлов с байт-кодом агента и шаблонов сообщений в фонд платформы IACPaaS – в информационные ресурсы, представляющие сформированные на первом шаге этапа реализации декларативные описания агента и шаблонов сообщений соответственно. Последнее выполняется с использованием системного сервиса *Загрузчик агентов* и необходимо для запуска и функционирования разработанных агентов на виртуальной машине платформы IACPaaS.

Необходимо отметить, что *Генератор агентов* и *Загрузчик агентов* проверяют полноту декларативного описания агента и используемых в нем шаблонов сообщений. Если декларативное описание не полно, то разработчику отображается сообщение, локализующее соответствующее место в описании. *Загрузчик агентов* проверяет, также, корректность байт-кода загружаемого агента и шаблонов сообщений, а также байт-кода используемых в них внутренних классов (если байт-код не корректен, то его загрузка или обновление не выполняется, а разработчику отображается сообщение о найденной ошибке).

2.4. Тестирование и отладка агента

На данном этапе для функционального тестирования разработанного агента используется системный сервис *Тестировщик агентов*, который обеспечивает многократный запуск и выполнение агентов на заданном множестве тестов, а также формирование и сохранение отчетов о результатах испытаний. Множество тестов для агента формируется с помощью *Редактора информационных ресурсов* по метаинформации *Структура тестов агента*. Тест для агента, в общем случае, есть четверка *<входное сообщение, множество ожидаемых выходных сообщений, множество изменяемых агентом информационных ресурсов, ожидаемые состояния изменяемых информационных ресурсов>*. Обязательным является только первый компонент. Тест считается успешно пройденным, если содержимое ожидаемых сообщений совпало с содержимым соответствующих сообщений, которые агент послал в процессе работы блока продукций, а содержимое изменяемых в процессе работы блока продукций информационных ресурсов совпало с содержимым соответствующих информационных ресурсов, перечисленных в последнем компоненте теста.

Для просмотра отчетов о результатах испытаний и журналов используется *Редактор информационных ресурсов* (в режиме просмотра). Журнал работы агента на конкретном тесте присоединяется к отчету и также доступен для просмотра. Журналирование используется для получения информации о том, какие события и в какой последовательности происходят во время

работы блоков продукций агента, а также для того, чтобы локализовать место возникновения ошибки и дать достаточно информации для ее воспроизведения. Сформированные наборы тестов могут использоваться для регрессионного тестирования в процессе сопровождения агента.

2.5. Ввод агента в эксплуатацию

Для включения агента в состав сервисов он с разрешения администратора предметной области (для которой разработан агент) переводится в режим публичного использования.

Заключение

Использование облачной платформы IACPaaS для разработки компонентов ИС соответствует современным требованиям к проектированию и реализации ИС и обеспечивает их жизнеспособность за счет того, что все компоненты доступны разработчикам и экспертам для поддержания их в актуальном состоянии. Разработка каждого агента, входящего в состав решателей задач ИС, состоит в описании компонентов его декларативной спецификации и написанию кода множества блоков продукций. На основе декларативных спецификаций выполняется генерация заготовок исходного кода агентов (и шаблонов сообщений), в них же помещается и хранится процедурная часть (байт-код) соответствующих агентов и шаблонов сообщений. Сопровождение агента состоит в изменении исходного кода агента и, возможно, шаблонов сообщений, используемых им, с последующей загрузкой обновленного байт-кода в фонд платформы. Также перед этим может быть изменена декларативная спецификация агента и используемых им шаблонов сообщений и выполнена повторная генерация заготовок исходного кода.

С использованием предложенной технологии и системных сервисов платформы созданы следующие облачные прикладные сервисы: сервис разработки профессиональных виртуальных облачных сред, компьютерные обучающие тренажеры по классическим методам исследования в офтальмологии, виртуальная химическая лаборатория, виртуальная модель городского района. В процессе разработки находятся сервисы по автоматизированному конструированию доказательств математических теорем и медицинской диагностике.

Работа выполнена при финансовой поддержке РФФИ (проект 13-07-00024а, проект 14-07-00299а).

Библиографический список

[Грибова и др., 2010] Грибова В.В., Клещев А.С., Шалфеева Е.А. Управление интеллектуальными системами // Известия РАН. Теории и системы управления. – 2010. – №6. – С.122-137.

[Kleshchev, 2011] Kleshchev, A.S. How can ontologies contribute to software development? // Lecture Notes in Artificial Intelligence. Springer-Verlag Berlin Heidelberg. – 2011. – Vol. 6581/2011. – Pp. 121-135.

[Грибова и др., 2013a] Грибова В.В., Клещев А.С. Технология разработки интеллектуальных сервисов, ориентированных на декларативные предметные базы знаний. Часть 1. Информационные ресурсы // Информационные технологии. 2013. №9. С. 7-11.

[Грибова и др., 2013b] Грибова В.В., Клещев А.С. Технология разработки интеллектуальных сервисов, ориентированных на декларативные предметные базы знаний. Часть 2. Решатель задач. Пользовательский интерфейс // Информационные технологии. 2013. №10. С. 10-14.

[Gennari et al., 2003] Gennari, J.H., Musen, M.A., Fergerson, R.W, etc. The evolution of Protege: An environment for knowledge-based systems development/International Journal of Human-Computer Studies. 2003. 58(1):89-123.

[Орлов и др., 2006a] Орлов В.А., Клещев А.С. Компьютерные банки знаний. Универсальный подход к решению проблемы редактирования информации. – Информационные технологии. – 2006. – №5 – С. 25-31.

[Protege, 2014] The Protégé Ontology Editor and Knowledge Acquisition System. [Electronic resource]. URL: http://protege.stanford.edu/ (дата обращения 25.11.2014).

[Орлов и др., 2006b] Орлов В.А., Клещев А.С. Компьютерные банки знаний. Многоцелевой банк знаний // Информационные технологии. – 2006. – №2. – С. 2-8.

[Sanderson, 2009] Sanderson D. Programming Google App Engine: Build and Run Scalable Web Apps on Google's Infrastructure. – Sebastopol, California : O'Reilly Media, 2009. – 394 p.

[TopQuadrant, 2014] TopQuadrant. [Electronic resource]. URL: http://www.topquadrant.com/ (дата обращения : 25.11.2014).

[Грибова и др., 2011] Грибова В.В., Клещев А.С., Крылов Д.А., Москаленко Ф.М., Смагин С.В., Тимченко В.А., Тютюнник М.Б., Шалфеева Е.А. Проект IACPaaS. Комплекс для интеллектуальных систем на основе облачных вычислений // Искусственный интеллект и принятие решений. – 2011. – №1. – С.27-35.

[Gribova et al., 2013] Gribova V.V., Kleschev A.S., Krylov D.A., Moskalenko Ph.M., Timchenko V.A., Shalfeyeva E.A., Goldstein M.L. A software platform for the development of intelligent multi-agent internet-services // Proceedings of the Distributed Intelligent Systems and Technologies Workshop (DIST'2013). – 1-4 July 2013. – St. Petersburg, Russia. – Pp. 29-36.

[Lehmann, 1992] Lehmann F.W. Semantic Networks // Computers & Mathematics with Applications. – 1992. – Vol.23. – №2-5.

[Фатрелл и др., 2004] Фатрелл Р.Т., Шафер Д.Ф., Шафер Л.И. Управление программными проектами. Достижение оптимального качества при минимуме затрат: Пер. с англ. – М.: Вильямс, 2004. – 1136 с.

A TECHNOLOGY FOR DEVELOPMENT OF PROBLEM SOLVERS OF INTELLIGENT SYSTEMS WITH THE USE OF IACPAAS CLOUD PLATFORM TOOLS

Krylov D.A., Moskalenko Ph.M.,
Timchenko V.A.

Federal State Budget Institution of Science
Institute for Automation and Control Processes
Far Eastern Branch of the Russian Academy of
Sciences, Vladiviostok, Russia

dmalkr@gmail.com

philipmm@iacp.dvo.ru

vadim@dvo.ru

The paper presents a technology for development of multi-agent problem solvers of applied intelligent cloud services with the use of system tools of IACPaaS platform. The technology is put to reduce the labour-intensiveness of development and primarily of support for intelligent cloud services.