Гвоздевский И. Н., аспирант
Белгородский государственный технологический университет им. В.Г. Шухова

ПРИМЕНЕНИЕ АГЕНТНО-ОРИЕНТИРОВАННОГО ПОДХОДА ДЛЯ РАСШИРЕНИЯ ВОЗМОЖНОСТЕЙ АВТОМАТИЗИРОВАННЫХ СИСТЕМ ДИСПЕТЧЕРСКОГО УПРАВЛЕНИЯ С ИСПОЛЬЗОВАНИЕМ ОНТОЛОГИЙ

igorek@intbel.ru

Информационные системы современных предприятий являются сложными иерархическими структурами, включающие в себя огромное количество разнородных элементов. Многообразие существующей элементной базы обусловлено спецификой подходов к построению такого рода систем. При анализе вопросов взаимодействия элементов среды в географически распределенных информационных системах, встает вопрос создания и адаптации к работе интерфейсов обмена данными между различными аппаратными и программными ресурсами. При этом необходимо учитывать возможности по унификации протоколов взаимодействия данных сред, специфику методов, алгоритмов получения, обработки и хранения информации. Модернизация современных систем диспетчеризации и постоянное обновление или дополнение элементной базы требует использовать подходы построения мультиагентных систем, которые позволяют использовать различные инновационные методы искусственного интеллекта, параллельного программирования.

Ключевые слова: *агент, агентные системы, онтология, распределенные вычислительные системы, автоматизированная система диспетчерского управления.*

Информационные системы современных предприятий являются сложными иерархическими структурами, включающие в себя огромное количество разнородных элементов. Многообразие существующей элементной базы обусловлено спецификой подходов к построению такого рода систем. При анализе вопросов взаимодействия элементов среды в географически распределенных информационных системах, встает вопрос создания и адаптации к работе интерфейсов обмена данными между различными аппаратными и программными ресурсами. При этом необходимо учитывать возможности по унификации протоколов взаимодействия данных сред, специфику методов, алгоритмов получения, обработки и хранения информации.

Для управления распределенными вычислительными системами используют комплекс различных подходов таких, как сервис-ориентированный и агентный.

Модернизация современных систем диспетчеризации и постоянное обновление или дополнение элементной базы требует использовать подходы построения мультиагентных систем, которые позволяют использовать различные инновационные методы искусственного интеллекта, параллельного программирования. Использование данной технологии приводит к необходимости применения усовершенствованных методов связанных с накоплением, переработкой и структурированием знаний в таких системах, что позволит обеспечить более высокий уровень интеграции с современными информационными подсистемами и веб-сервисами[1].

Мультиагентные системы к контексте использования распределенного искусственного интеллекта предполагает нахождение решения путем взаимодействия различных типов подсистем. Результат достигается из-за возможности применить универсальность взаимодействия цепочки агент – пользователь – система – интернет. Применение онтологической модели знаний (онтологии) позволяет использовать более приемлемую для машинной обработки информацию. Таким образом, модернизация существующих систем диспетчерского управления, где существующие элементы имели бы возможность использовать унифицированную модель взаимодействия, где предметная область была бы структурирована и описана в виде онтологии, открывало бы возможности для решения задач диагностики и восстановления работоспособности, а также позволило использовать преимущества онтологий, агентных платформ, искусственного интеллекта.

Ключевые слова: агент, агентные системы, онтология, распределенные вычислительные системы, автоматизированная система диспетчерского управления.

Введение. Одним из преимуществ автоматизированной системы диспетчерского управления БГТУ является использование различных технологий, применяемых при построении телекоммуникационной среды[2], для интеграции в единое целое различных модулей, элементов и подсистем. Неоднородность при выборе оборудования и программного обеспечения в подобных АСДУ системах обусловлена поэтапным вводом новых

подсистем, модернизацией программного обеспечения, а также добавлением новой усовершенствованной элементной базы к текущим подсистемам, что в свою очередь позволяет использовать агентно-ориентированный подход для решения задач оптимизации работы, диагностики и унификации подсистем.

Методология. Применение в существующей системе автоматизированного диспетчерского управления в рамках технологической площадки Белгородского государственного технологического университета агентных подходов, позволяющих оптимизировать выполнение диагностических задач. Унификация методов взаимодействия подсистем за счет применения онтологического подхода и формирования структурированной предметной области элементов системы.

Основная часть. Для обеспечения взаимодействия компонентов системы, весь программно-аппаратный комплекс можно разделить на три базовых уровня на которых можно осуществить стандартизацию подходов и методов[2].

- Технологический уровень – включает набор элементов, осуществляющих взаимодействие с технологическими системами (контроллеры, триггеры, агенты управления)
- Коммуникационный уровень – обеспечивает возможность интеграции элементов технологического уровня с помощью различных протоколов связи в единую телекоммуникационную среду (коммуникационное оборудования, агенты обеспечения взаимодействия)
- Уровень управления (SCADA - уровень) – программный слой управления, мониторинга и сбора информации (программные пакеты, агенты мониторинга, управления и принятия решений)

Каждый уровень АСДУ может быть унифицирован по определенным признакам, зачастую такими признаками могут быть типы оборудования, поддерживаемые протоколы, типы программного обеспечения, а также многие другие параметры компонентов. Для обеспечения высокого уровня взаимодействия между любыми элементами и возможности оперативного расширения функционала комплекса в целом был проведен анализ существующих методов управления разнородными системами.

Для реализации систем взаимодействия агентов используют готовые технологические платформы, включающие в себя набор функций обеспечивающий функционирование среды MAC [3]. Из свободно распространяемых или проприетарных программных решений, существующих в настоящее время можно выделить наиболее развитую в техническом плане модель агентной платформы JADE.

JADE является платформой и средой для функционирования агентов, контроля выполнения, целостности, утилизации ресурсов, а также включает классы Java, необходимые для полного спектра разработки. Основной единицей среды исполнения агентов в подходе JADE является контейнер, содержащий необходимый минимум для работы агента. Совокупность контейнеров образует общую платформу исполнения и реализовано в виде бесшовной структуры единого слоя без привязки к конечному программно-аппаратному комплексу (операционная система, технологический элемент, контроллер, сетевая инфраструктура)[3].

Данная платформа позволяет агенту использовать базовые принципы работы, каждый агент можно считать элементом пиринговой сети в общей оболочке, где фактически среда может быть географически распределена. JADE позволяет агенту выполнять функции по ориентированию в оболочке, находить элементы, настраивать интерфейсы взаимодействия групп агентов.

На программном уровне идентификация агента происходит с помощью уникального именования, тут же происходит формирования базовых сервисов, согласно специфики деятельности агента. Агент является интеллектуальным элементом в том числе может управлять своим жизненным циклом, изменять набор своего функционала путем взаимодействия с онтологической моделью, внешним источником информации или других агентов.

Данная пиринговая модель позволяет агентам обмениваться различными сообщениями, зачастую модель коммуникации элементов представляется в виде отсылки асинхронных пакетов данных, причем сама платформа позволяет создавать единую коммуникационную среду не упираясь в существующие ограничения нижнего уровня взаимодействия. Фактически для проведения обмена информацией между элементами агенту достаточно указать имя получателя, не имеет значения где он находится, и кем он является. Сообщение будет сконвертировано в определённый формат необходимый для получателя с сохранением семантики, без ограничений по временным параметрам. Основным преимуществом можно выделить доступность отправителя и получателя в рамках взаимодействия[2].

Однако, в данном контексте встает вопрос о продолжительности периода доставки данного сообщения.

Предположим, что в результате выполнения поставленной задачи агентной передачи сообщения S, каждое действие $k \in j$ будет исполнено за время $C_k(S)$.

На решение задачи j, потребуется времени не меньше, чем

$$C_j(S) = \max C_k(S)$$

где j – поставленная задача передачи сообщения; k – действия, направленные на решение задачи.

Найдем время действия $k \in j$ как p_k. Время решения p_j задачи j можно вычислить по следующей формуле

$$p_j = C_j(S) - \min\left(C_k(S) - p_k\right)$$

Значения, полученные в результате вычислений дают возможность оценить интегральные свойства распределенной многоагентной структуры. Показатель максимального опоздания задач можно использовать для представления уровня качества анализа службы, вычисляемое

$$L_{\max} = \max\left(C_j(S) - d_j\right)$$

где d_j – время, до которого пользователь желает получить сообщение.

Для получения наиболее оптимальных уровней работы информационной системы нужно обеспечивать минимальные значения данного показателя, а также обеспечить подсчет опоздавших событий[4].

$$j \in \tau \wedge C_j \geq d_j$$

При вычислении данных параметров в системе можно получить полную информацию о загрузке среды при прохождении сообщений в частности обращений агентов, пользователей и необработанных пакетах.

Модель передачи сообщения в мультиагентной системе (рис. 1) изображена в форме графа

$$G(S,T)$$

где S – множество состояний

T – множество переходов при передаче сообщения

$$T \subset S^2$$

Дуга $(v,t) \subset T$ существует, если существует правило разговора для перехода с текущего состояния v в следующее состояние t. В графе переходов все состояния должны быть достижимы из начального состояния [5].

Для решения задачи интеграции с внешними источниками данных, обеспечение поддержки онтологий и действий с ними используется модуль JADE content, при этом информация об онтологиях представляется не ввиде языка описаний, а структурирована в формате java-объекта. Платформа также имеет возможность подключения дополнительных классов и пакетов для увеличения возможностей разработчика. Библиотеа AgentOWL поддерживает модели агентов RDF/OWL и является более эффективным. Описание модели знаний агента при использовании онтологий основано на пяти основных элементах: Ресурсы, Акторы, Действия, Содержание и События[4]. Обмен сообщениями происходит в формате RDF/OWL, возможно включение элементов в созданные онтологии. При расширении возможностей для работы агентов с онтологиями необходимо добавить методы работы и передачи сообщений OWL и SPARQL, что обеспечивает в том числе при взаимдодействии с онтологическими моделями знаний. В основе работы AgentOWL лежит фреймворк Jena, поэтому данный подход легко расширять новыми возможностями работы с онтологической моделью знаний, используя Jena API.

Использование модели использования агентов и онтологий ставит сложную задачу создания интерфейса взаимодействия приложений SCADA-уровня и мультиагентной платформы[3]. Возможно использовать несколько решений по осуществлению интеграции. Решение JADEGateway реализует обмен сообщениями между платформой JADE и прикладным сервисом. Класс JADE.wrapper.gateway создает интерфейс взаимодействия веб-агентов, при непосредственном контроле всех запросов к сервису.

Мультиагентная система работает по принципу взаимодействия прикладной сервис – шлюз – мультиагентная система, при этом достигается наилучший результат интеграции[5].

Интеграция посредством компонента JADE4spring – обеспечивает взаимодействие фреймворка Spring и функционирующих контейнеров агентов JADE. Решение позволяет расширять возможности агента путем применения фукционала технологии Spring. Работа по визуализации онтологических моделей осуществляется с использованием модуля Flare Prefuse. Flare это ActiveX библиотека позволяющая использовать объекты Adobe Flash. Эта библиотека позволяет преобразовать онтологическую модель знаний агента в owl-граф с последующей публикацией в виде элемента Adobe или SWF-объекта[1].

Для преобразования OWL в формат, требуемый для Flare, используется OWL2Prefuse. Используя библиотеку Flare можно получить OWL-граф, информация о котором сохраняется в формате SWF. Объект SWF встраивается на страницу, при этом необходимо обеспечить обмен информации между указанным объектом и агентами, запущенными в контексте веб-приложения,

для обеспечения динамической визуализации онтологических моделей.

Программные решения рассмотренные выше полностью обеспечивают возможности интеграции мультиагентных систем и онтологических моделей знаний[2]. Направлениями использования данных технологий можно считать не только автоматизированные системы диспетчерского управления, но и различные информационные системы, база знаний которых подразумевает формализацию в виде онтологической модели, а также существует конечная достижимая результативная составляющая.

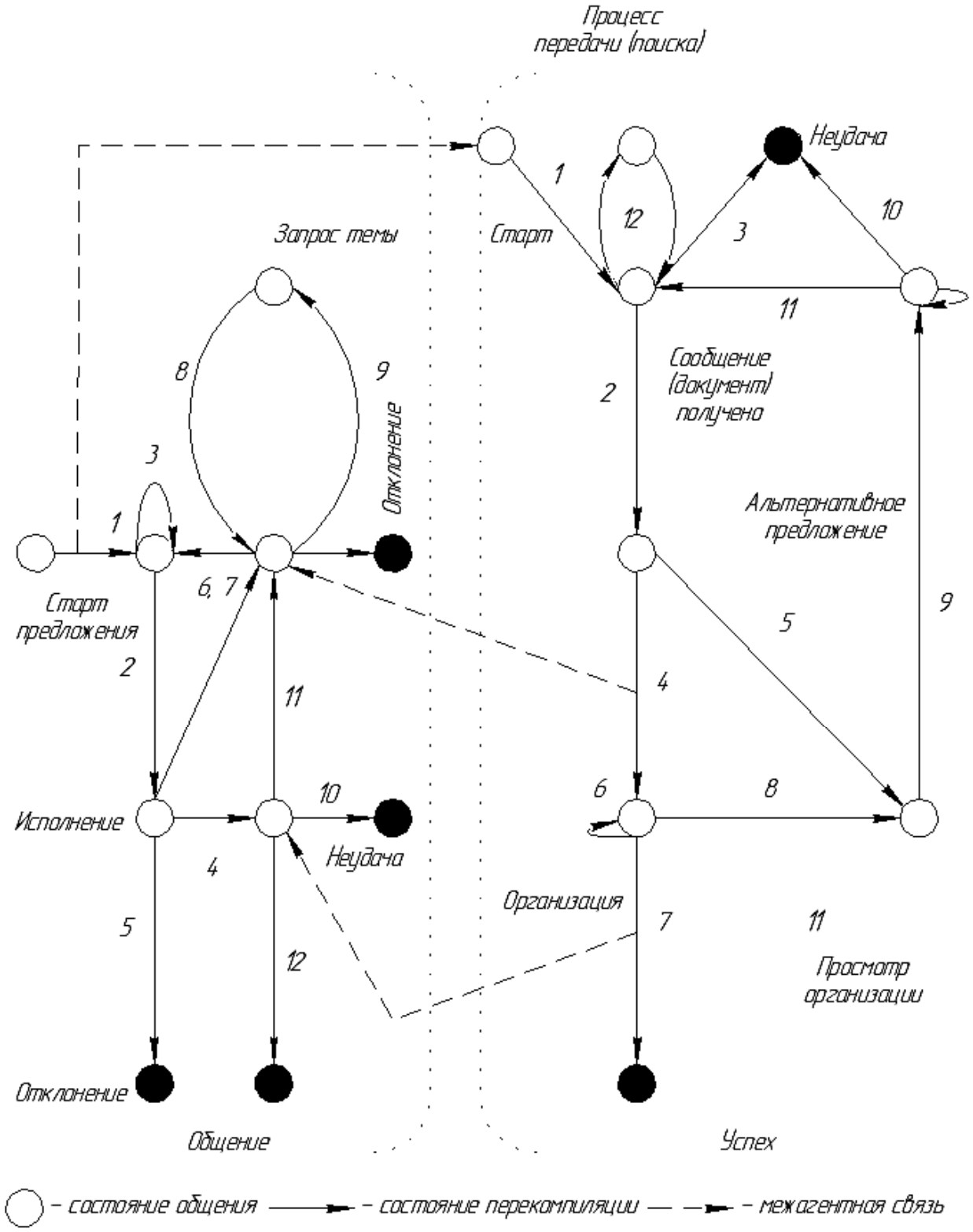

Рис. 1. Модель передачи сообщения в мультиагентной системе

При создании мультиагентной системы (МАС) следует учесть требования, обусловленные поставленными задачами. Используя методы построения абстрактной логической модели, представим многоагентную распределенную систему.

$$E = \{O, S, \Omega\}$$

где O – объекты агентной среды;
S – связи между объектами среды O;
Ω – действия над объектами в пространстве E.

На примере рассмотрим создание математической модели распределенной мультиагентной сети, основанной на технологии JADE, для объединения на уровне доступа к информации баз данных, являющихся частями существующих автоматизированных систем.

Фактически имеется набор агентов на базе одной мультиагентной платформы JADE. Все агенты работают по своим определенным задачам, собирая статистику с функционирующих узлов и создавая элементарные базы данных на каждом узле. По запросу пользователя в определенный момент времени каждый агент пересылает набор данных который определяет функционирование узла, работоспособность сервисов и сетевых протоколов[1].

Для задачи, которая рассматривается, в качестве объектов модели будут выступать компоненты современной информационной системы – разные типы хранилищ данных, в том числе реляционные базы данных

$$O = \{db_n \mid n = \overline{1, N_O}\}$$

Отношение

$$S = \{s_n \mid n = \overline{1, N_S}\}$$

позволяет определить стабильную систему взаимодействий между элементами распределенной мультиагентной E, ориентированной на определенного пользователя или агента

Тогда множество пользователей

$$G = \{g_n \mid n = \overline{1, N_G}\}$$

Все действия в данной системе могут быть выполнены только на основании множества операций в зависимости от предъявленной цели

$$\Omega = \{g_n \mid n = \overline{1, N_\Omega}\}$$

Элементарными операциями $\Omega = \{q_1, q_2, .., q_n\}$ на примере работы с базами данных будут являться операции обработки информации в виде простых запросов SQL

$$\{\text{«}select\text{»}, \text{«}insert\text{»}, \text{«}update\text{»}, \text{«}delete\text{»}\}$$

Для обеспечения доступа к информации и обмену информацией в информационном пространстве предложено использовать мультиагентную систему, математическое представление которой имеет вид

$$U_A = \{A, S_A, \Omega_A\}$$

где A – множество программных агентов;

S_A – отношения в мультиагентной системе;

Ω_A – множество операций, которые могут выполнять агенты.

Отношения S_A в мультиагентной системе делятся на взаимоотношения

S_{AA} между самими агентами и отношения S_{AE} между агентами A и объектами

информационного пространства

$$S_A = S_{AA} \cup S_{AE}$$
$$S_A = S_{AA} \cap S_{AE} = \varnothing$$

Для мультиагентной системы, которая не привязана к конкретной конфигурации информационного пространства

$$S_{AE} = \varnothing$$

Определим понятие мультиагентного пространства как расширения информационного пространства. Тогда модель мультиагентной среды будет иметь вид

$$E_A = E \cup U_A = \{O_g, S_g, \Omega_g\}$$

При этом множество объектов информационного пространства можно представить в виде

$$O_g = O \cup A$$

Множество отношений представим

$$S_g = S \cup S_A = S \cup S_{AA} \cup S_{AE}$$

где

$$S_{AE} \neq \varnothing$$

Множество операций будет иметь вид

$$\Omega_g = \Omega \cup \Omega_A$$

В мультиагентной среде множество объектных операций над объектами в случайные моменты времени $t_1, t_2, ..., t_m, t_{m+1} \geq t_m$ проводит актор g_n мультиагентной среды на основе последовательности действий, которая зачастую будет являться формализованной

$$\dot{z}_{n_1}(t_:), \dot{z}_{n_2}(t_-), \ldots \dot{z}_{nm}(t_m)$$

либо унифицированного действия $\dot{z}_1(t_1)$.

Взаимодействие пользователя с мультиагентным пространством реализуется через агента пользователя a_C, который взаимодействует с соответствующими системными агентами

$$A_{Si} = \{a_{S1}, a_{S2}, ..., a_{Sn}\}$$

каждый из которых сопоставлен с некоторым информационным ресурсом пространства E.

Множество агентов A мультиагентного пространства будет состоять из множества агентов пользователя A_C и множества системных агентов A_S

$$A = A_C \cup A_S$$

Поскольку, рассматриваемая модель мультиагентного пространства не предусматривает взаимодействие между однотипными агентами, то компонент отношений в мультиагентной системе будет иметь следующий вид

$$S_A = S_{CS} \cup S_{SE}$$

где S_{CS} – отношения между разными типами агентов (агент-агент);

S_{SE} – отношения между агентами множества AS и объектами информационного пространства (агент-система).

Соответственно, множество операций будет иметь вид

$$\Omega_A = \Omega_{CS} \cup \Omega_{SE}$$

где Ω_{CS} – множество операций, выполняемых агентами в отношениях S_{CS};

Ω_{SE} – множество операций в отношениях S_{SE}, для доступа агентов множества A_S к объектам информационного пространства.

Формализованная задача z_i, поставленная пользователем, преобразуется агентом пользователя a_c в последовательность команд, которые отсылаются системным агентам A_{Si}, и последующую обработку информации после получения ответа от агентов A_{Si}.

Например, задача

$$z_i = \{find\_data\_in\_database\}$$

которую ставит пользователь, будет реализована последовательностью действий агента пользователя q_{ac} и системных агентов q_{As} (рис. 2)

$$q_{a_C} = q_{a_C 1} \cup q_{a_C 2}$$

где q_{ac1} – (get_search_parameters, search_available_agents, form_query, send_query);

q_{ac2} – (collect_all_results,merge_results, show_final_result)

$$q_{A_C} = q_{A_S 1} \cup q_{A_S 2}$$

$$q_{A_S} = q_{A_S 1} \cup q_{A_S 2}$$

где q_{AS1} – (receive_query,execute_query,get_result);

q_{AS2} – (form_result, send_result).

Рис. 2. Схема взаимодействия пользователя с мультиагентным пространством

Как указано выше, в системе используется два типа агентов a_C и a_S.

Основной задачей агента a_C является взаимодействие с пользователем, передача запросов, получение и консолидация информации от группы агентов A_{Si}. Агент a_S выступает в роли системного агента, основная задача которого – взаимодействие с базой данных на узлах системы и передача данных агенту a_C в ответ на его запрос.

На каждом из узлов информационного пространства, на котором находится база данных к которой необходимо иметь доступ, должен быть установлен один a_S. *Агент* a_C устанавливается на тех узлах, из которых необходимо предоставить доступ пользователю к распределенным данным.

Согласно спецификации FIPA реализация мультиагентной системы предусматривает наличие дополнительного агента в системе, агента Directory Facilitator (DF), который внедряет сервис « желтых страниц» [5].

С помощью этого сервиса агенты системы могут узнать, какие еще агенты есть в этой системе и какие сервисы они предлагают. После включения агентов, все агенты a_S, которые есть в системе, регистрируются в сервисе «желтых страниц», то есть в DF. Это нужно для того, чтобы a_C при посылке запросов мог иметь актуальную информацию о существующих на данный момент активных агентах в системе. Чаще всего a_S предлагает сервис доступа к базам данных, а a_C использует этот сервис для решения задач поставленных пользователем.

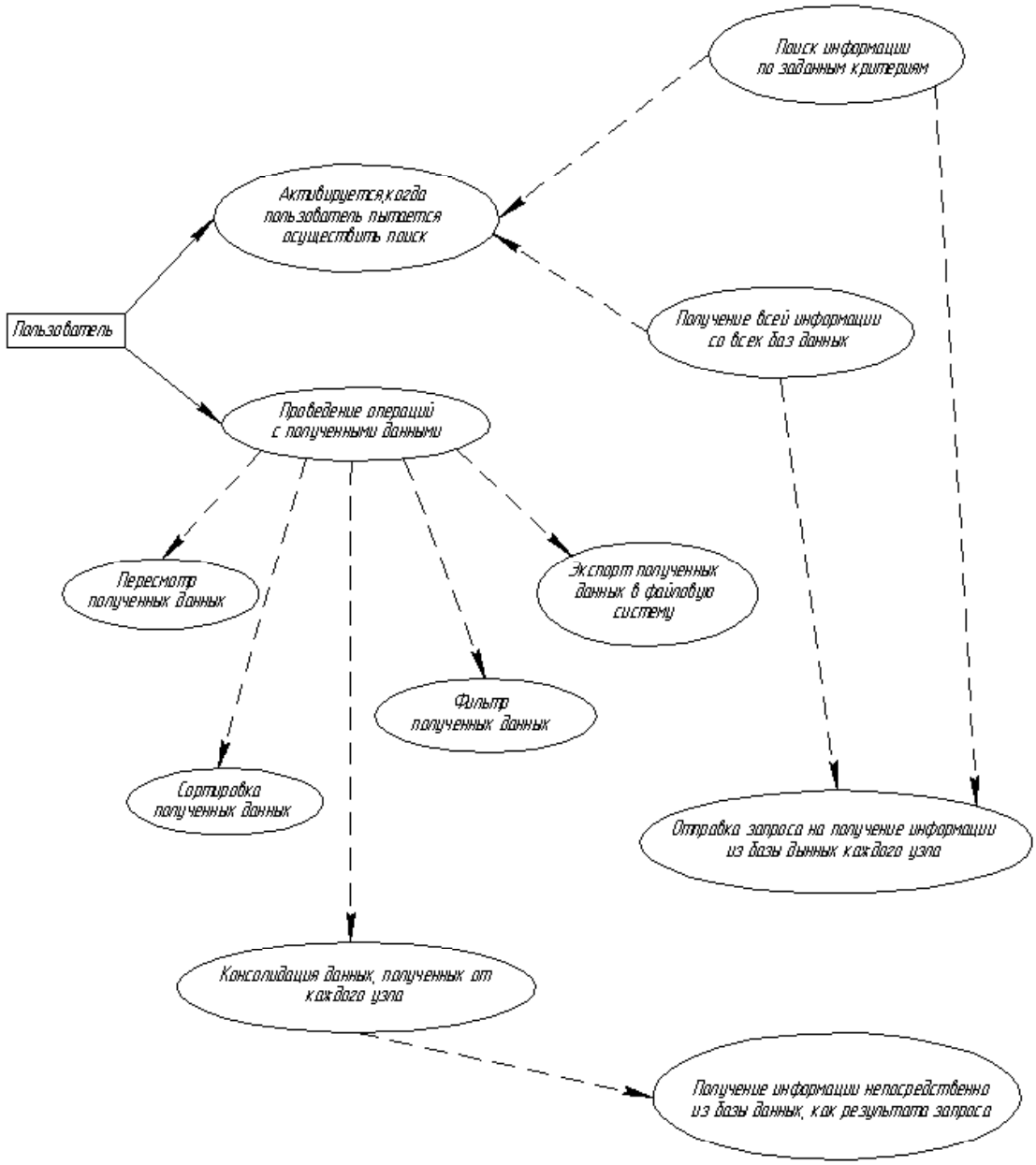

Рис. 3. Диаграмма прецедентов взаимодействия пользователя с мультиагентным пространством согласно полученной математической модели

Для четкого понимания схемы взаимодействия между объектами в смоделированной мультиагентной системе была составлена диаграмма агентов (рис. 4).

Смоделированная система легко адаптируется к изменениям в среде. В случае, если в системе появляется новый узел, его подключение к системе не требует дополнительных усилий и в основном заключается в правильной конфигурации подключения агента.

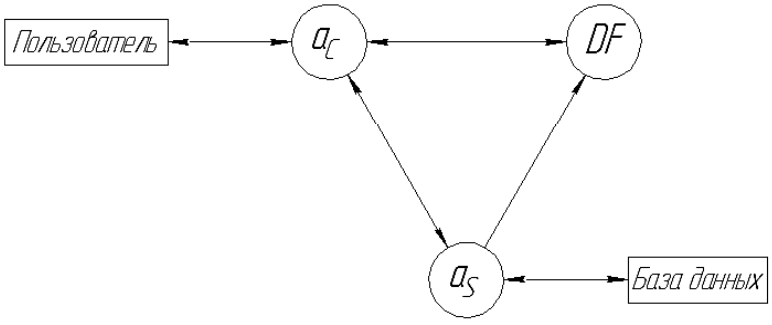

Рис. 4. Диаграмма агентов МАС получения и обмена информации

Выводы. Предложенная в работе математическая модель мультиагентной системы предоставляет формальный аппарат для построения распределенной информационной системы на основе агентных технологий. Рассмотренный подход к реализации позволяет получить достаточно гибкую, независимую и устойчивую к изменениям в информационной среде систему. Реализованный прототип многоагентной системы показал адекватность модели и возможность ее использования в реальных распределенных системах.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Поляков В.М., Буханов Д.Г., Синюк В.Г. Базовые структурные модели распределенных вычислительных систем в многоагентной диагностической среде. // Журнал «Информационные системы и технологии». 2012. №4(72). С. 52-56

2. Белоусов А.В., Глаголев С.Н., Быстров А.Б., Кошлич Ю.А. Демонстрационная зона по энергосбережению БГТУ им. В.Г. Шухова – база для развития энергоэффективных проектов в регионе // Энергосбережение. Энергетика. Энергоаудит. 2013. No 10 (116). С. 10–17.

3. Письменный В.Ю., Терновой М.Ю. Организация доступа к гетерогенным базам данных с использованием агентных технологий // Международный научный журнал «Компьютинг». 2011. Т.10, Вып. 2. С. 183–191.

4. Филатов В.А. Мультиагентные технологии интеграции гетерогенных информационных систем и распределенных баз данных : Дис. д-ра техн. наук: 05.13.06 // ХНУР. Х., 2004. 341с. с. 313-336.

5. Международный Стандарт FIPA. Агентные платформы [Электронный ресурс]. Систем. требования: Веб-браузер. URL: http://www.fipa.org/specs/fipa00023/SC00023K.html#_Toc75951006 (дата обращения: 25.09.2015)

6. Международный Стандарт FIPA. Спецификация [Электронный ресурс]. Систем. требования: Веб-браузер. URL: http://fipa.org/specifications/index.html (дата обращения: 13.10.2015).

7. Агентная платформа JADE. [Электронный ресурс]. Систем. требования: Adobe Acrobat Reader.URL: http://jade.tilab.com/doc/tutorials/JADE_methodology_website_version.pdf (дата обращения: 18.10.2015).

Gvozdevskiy I.N.

AGENT-ORIENTED APPROACH FOR THE EMPOWERMENT OF AUTOMATED SUPERVISORY SYSTEMS WITH THE USE OF ONTOLOGIES

Information systems of modern enterprises is a complex hierarchical structure, including a huge number of heterogeneous elements. The variety of existing components due to the specific approaches to building such systems. In analyzing the issues of interaction between elements of the environment in a geographically distributed information systems, there is a question of creating and adapting to the communication interfaces between the various hardware and software resources. It is necessary to take into account the possibility of unification of the interaction of these media reports, the specifics of methods, algorithms, receiving, processing and storing information. Modernization of dispatching systems and constant updating of or addition to the element base requires the use of the approaches for constructing multi-agent systems that allow you to use a variety of innovative methods of artificial intelligence, parallel programming.

Key words: *agent, agent systems, ontology, distributed computing systems, automated dispatch control system.*

Гвоздевский Игорь Николаевич, аспирант кафедры программного обеспечения и автоматизированных систем.
Белгородский государственный технологический университет им. В.Г. Шухова.
Адрес: Россия, 308012, Белгород, ул. Костюкова, д. 46.
E-mail: igorek@intbel.ru