Моделювання технологічних процесів

УДК 621.513

А.В. Прохоров,
Е.М. Пахнина

Национальный аэрокосмический университет им. Н.Е.Жуковского «ХАИ»

КОНЦЕПЦИЯ АГЕНТНО-ОРИЕНТИРОВАННОГО ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ ПРОИЗВОДСТВЕННЫХ ПРОЦЕССОВ

Введение. Современные автоматизированные производственные системы характеризуются: сложной структурой объектов и процессов многоцелевого функционирования; распределенностью в пространстве и времени; многоуровневостью; сложными и разнообразными материальными, финансовыми и информационными потоками; необходимостью принятия решений, связанных с оптимальным распределением имеющихся и, как правило, ограниченных ресурсов. Кроме того, в реальных условиях эксплуатации производственных систем действуют внешние факторы (риски), которые могут изменить ход работы всей системы, поэтому очень важно предусмотреть все эти факторы, с учетом возможной неопределенности, а также соответствующую реакцию системы управления на них. Наиболее эффективно подобный анализ и прогнозирование динамических характеристик производственных систем обеспечивают средства, основанные на методах имитационного моделирования.

Однако рассмотренные особенности приводят к тому, что модель исследуемого объекта или процесса должна иметь возможность динамической перестройки за счет создания/удаления элементов и связей между ними, пополнения или уточнения «на ходу», включения различных сценариев поведения с механизмами адаптации. В традиционном подходе к имитационному моделированию в случае возникновения изменения требуется остановка процесса моделирования, ручное внесение изменений в модель и перезапуск. Автоматизация данных задач может быть реализована с помощью интеллектуальных информационных технологий, реализуемых в виде агентно-ориентированных систем, имеющих возможность реализации динамического поведения, автономности и адаптации отдельных компонентов модели. Таким образом, наиболее актуальным и перспективным направлением исследования в настоящее время является создание интеллектуальных систем имитационного моделирования на основе агентного подхода.

Анализ последних исследований и публикаций. Примерами агентно-ориентированных систем моделирования являются Repast, Swarm, NetLogo, MASON, AnyLogic и др. Repast представляет собой визуальную среду для агентного моделирования, которая включает дискретно-событийный планировщик и адаптивные средства поведения, такие как нейронные сети и генетические алгоритмы. Swarm (*Swarm Development Group*, США) представляет собой набор библиотек с открытым кодом, для разработки приложений агентно-ориентированного моделирования систем [2,3]. В отличие от Repast, Swarm-планировщик поддерживает только продвижение времени через фиксированные промежутки. NetLogo (*Center for Connected Learning, Northwestern University*, США) также представляет собой кросс-платформенную среду агентного моделирования, однако в отличие от предыдущих систем, где используется объектно-ориентированный язык программирования Java, NetLogo является продолжением языка Logo. MASON (*George Mason University*, США) – программный продукт, в котором реализованы наиболее известные агентно-ориентированные модели Life, SugarScape и др. Однако,

направленность, для которой разрабатывались эти системы и созданные в настоящее время библиотеки моделей, охватывают в основном моделирование биологических и социальных процессов. Кроме того, базис средств имитационного моделирования в этих системах представлен только поддержкой дискретно-событийного механизма продвижения времени.

Отдельный интерес представляет разработка российской компании *XJ Technologies* система AnyLogic. AnyLogic объединяет в себе преимущества моделей системной динамики, дискретно-событийного моделирования и мультиагентных технологий [5]. В AnyLogic представление модели является визуальным и иерархическим. Графический язык моделирования (основанный на концепции UML-RT) оперирует понятиями активных объектов и связей между ними – дискретными (отправка сообщений произвольной структуры) и непрерывными (отслеживание показателей). Преимуществами этой инструментальной среды является возможность описания сложных систем с динамической структурой, изменение параметров модели в ходе имитации. Для описания сложного поведения можно использовать графические диаграммы переходов и состояний. Описание поведения объектов производится с помощью фрагментов кода на языке Java. Пользователю необходимо определить существенный код действий в специальных полях свойств элементов объектов, а весь рутинный код генерируется пакетом автоматически. Тем не менее, если модель исследуемой системы содержит значительные фрагменты описаний в виде процедурной логики, то это ведет к написанию значительного объема программного кода. Одной из причин этого и явным недостатком системы на данном этапе развития является отсутствие средств для представления и манипулирования знаниями, которые в мультиагентных системах служат для создания онтологий. Онтология является формальным описанием (концептуализацией) предметной области и правил принятия решений, которое служит для упрощения программирования поведения агентов и используется ими при взаимодействии.

Таким образом, анализ работ, посвященных применению агентного подхода в задачах имитационного моделирования производственных систем, показывает, что основными проблемами в настоящее время являются следующие:

- определение состава и распределение ролей агентов среди основных компонент системы имитационного моделирования;
- формирование распределенной базы знаний агентов и построение общей онтологии, разделяемой всеми агентами;
- создание интеллектуальных агентов с механизмами логического вывода решений;
- организация и планирование действий интеллектуальных агентов;
- разработка механизмов взаимодействия агентов, включая такие как, кооперация, конкуренция, компромисс, конформизм, уклонение от взаимодействия, выработка стратегий агентов при коллективном поведении.

Цель исследований. Целью данной работы является разработка знаниеориентированной системы имитационного моделирования производственных процессов, на основе агентного подхода, в составе которой функционируют интеллектуальные агенты, осуществляющие принятие решений и взаимодействие с помощью онтологической базы знаний и механизма логического вывода. Рассматриваются вопросы построения системы с учетом механизмов организации взаимодействия интеллектуальных агентов для моделирования производственных процессов с учетом информационного взаимодействия исполнителей в организационной структуре управления и факторов риска.

Основные особенности использования агентного подхода в имитационном моделировании производственных систем. Агентное моделирование предполагает, что модель включает множество взаимодействующих между собой и с внешней средой агентов – информационных (программных) элементов, которые имеют свои цели и задачи, внутреннее состояние и правила поведения [6]. Агент может обладать такими свойствами

как инициативность и реактивность, ориентация в пространстве, способность обучаться, общаться, “интеллект” и т.д. Отличительной особенностью агентных моделей является то, что, они децентрализованы и в них отсутствует централизованное поведение системы в целом [1]. Эти возможности радикально отличают агентные системы от существующих «жестко» организованных моделирующих программных систем, обеспечивая им такое принципиально важное новое свойство, как способность к самоорганизации. При этом отдельные автономные «части» моделирующей программы – агенты – получают возможность самостоятельно принимать решения и договариваться о том, как должна решаться задача, они приобретают собственную активность и могут вступать в различные отношения между собой, инициировать диалог с пользователем в заранее не предписанные моменты времени и т.д. Таким образом, определяется поведение на индивидуальном уровне, а глобальное поведение возникает как результат деятельности многих агентов, каждый из которых следует своим собственным правилам, функционирует в общей среде и взаимодействует со средой и с другими агентами.

Рассматривая основные преимущества агентного подхода при имитационном моделировании [4,5,8] процессов управления производственных систем с учетом рисков, следует отметить следующее:

- принцип автономности различных частей моделирующей программы (агентов), совместно функционирующих в распределенной системе, где одновременно протекает множество взаимосвязанных процессов;
- наличие элементов индивидуального поведения (от простых условий и ограничений, до сложных, которые учитывают цели и стратегии);
- агенты имеют возможность обучаться, адаптироваться и менять свое поведение, иметь динамические связи с другими агентами, которые могут формироваться и исчезать в процессе функционирования и др.

Одной из центральных задач при имитационном моделировании производственных систем является формирование множества альтернативных вариантов реализации проектов, описанных в виде комплекса работ, ограниченных во времени и требующих затрат ограниченных же ресурсов. В этом случае простейшие варианты организации мультиагентного сообщества при решении задач по распределению ресурсов могут быть основаны на взаимодействии агентов-заказов и агентов-ресурсов, выполняющих поиск соответствия на рынке предприятия. Эти агенты, в обобщенном виде представляющие собой некоторые «возможности» и «потребности», ищут себе подходящие пары. При примерном совпадении, например, некоторых свойств или условий, устанавливается мягкая связь, если же наблюдается более точное совпадение всех свойств – устанавливается более жесткая связь (каждому агенту-заказу выделяется часть рабочего времени агента-ресурса). Установление и поддержание связи может требовать некоторых расходов, вследствие чего невостребованные связи могут с течением времени разрушаться и исчезать. Такой способ позволяет осуществлять планирование в реальном времени, т.е. учитывать внезапные остановки оборудования или возникновение новых ресурсов и заказов простым добавлением, удалением или изменением агентов и их параметров в модели.

При этом агенты могут не только решать различные задачи и для этого обеспечивать свое функционирование, но и заботиться о своем развитии, т.е. изменении представляемых ими сущностей, будь-то люди, оборудование или абстрактные понятия, или установлении новых связей между ними. Конкурируя и кооперируясь между собой при заключении «сделок» для совместного решения возникающих задач (для чего агенты могут использовать развитые экономические механизмы, включая договора на выполнение работ или оказание услуг, долевое участие в прибыли, кредитование, страхование и т.д.), агенты могут обеспечить системе новые возможности в самоорганизации для постоянного приспособления к изменяющейся ситуации.

Агентная имитационная модель. Архитектура предлагаемой системы моделирования – основанная на знаниях (имеет базу знаний и принятие решений о действиях агентов осуществляется на основе механизмов логического вывода) и иерархическая, поскольку имеются агенты «метауровня», осуществляющие координацию распределенного решения задач другими агентами.

Агенты системы с учетом их ролей при взаимодействии в процессе моделирования представлены на рис. 1.

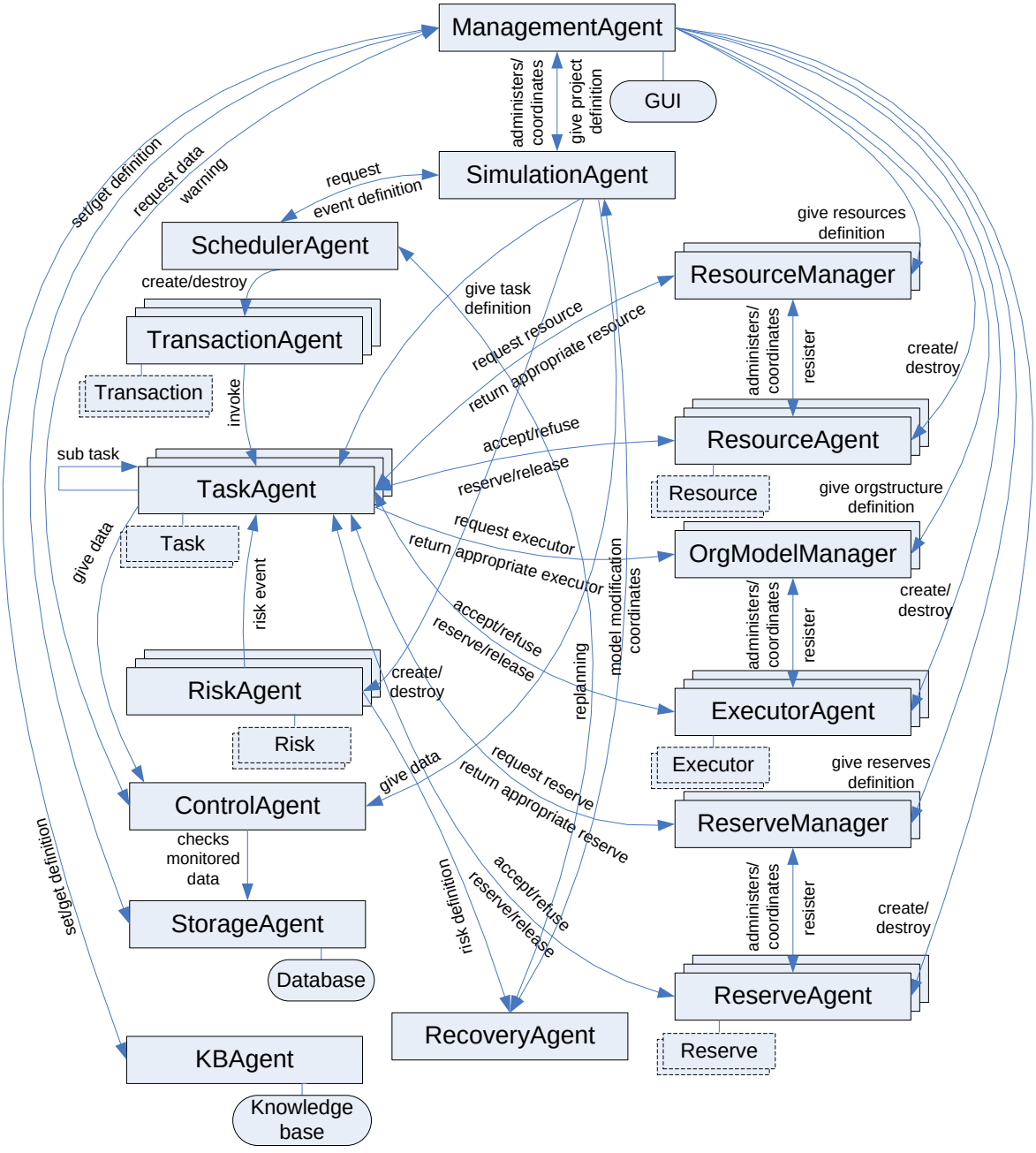

Рис. 1. Агенты системы

Концептуальную основу построения агентной имитационной системы составляет система дискретно-событийного имитационного моделирования [9]. Основным элементом модели в ней, является блок задач или работа, каждому из которых ставится в соответствие свой агент *TaskAgent*, основной целью которого будет успешное завершение задачи (в срок, за выделенные средства, в полном объеме и с требуемым качеством).

Класс *TaskAgent* в онтологической базе знаний имеет следующие атрибуты: тип задачи (с фиксированной длительностью, с фиксированным объемом, растяжимые, контрольные события); длительность исполнения; объем работ на задачу; трудоемкость; прямые затраты на задачу; ограничения на сроки исполнения задачи, признак вложенности, приоритеты, условия синхронизации, ресурсы, директивные сроки и условия выполнения, риски, возможность прерывания и т.д.

Динамические процессы, в дискретно-событийной имитационной модели существуют в виде взаимодействия ряда составляющих [9]. При переходе от дискретно-событийной модели к агентному представлению имитационной модели необходимо исходить из выделения элементов с индивидуальным поведением [1].

При данном подходе все моделируемые динамические «сущности» (изделия, детали, заказы, запросы, требования, документы и т.д.) являются агентами-заявками *TransactionAgent*, а генерация очередной заявки в системе будет соответствовать созданию нового агента.

В дискретно-событийной системе моделирования устройства осуществляют обслуживание заявок и предназначены для ограничения исполняемых работ на основе объема и состава наличного технологического оборудования и различного рода ресурсов.

Под ресурсами понимаются люди, оборудование и материалы, необходимые для выполнения задачи. Ресурсы являются элементом с индивидуальным поведением, поэтому представляются с помощью агентов двух типов агенты-исполнители *ExecutorAgent* (узлы организационной структуры управления) и агенты-ресурсы *ResourceAgent* (все остальные виды ресурсов).

Выделение отдельного класса агента-исполнителя, обусловлено тем, что в данной системе решается задача моделирования информационного взаимодействия узлов организационной структуры (отдельные работники, отделы и другие звенья аппарата управления). Особенности взаимодействия агентов-исполнителей обусловлены в первую очередь тем, что отношения между ними поддерживаются благодаря связям, существующим в организационной структуре, которые принято подразделять на горизонтальные (носят характер согласования) и вертикальные (связи подчинения). Кроме иерархии подчиненности, необходимо также учитывать структуру информационных каналов связи между узлами организационной структуры; характер и динамику информационного обмена между ними в процессе управления на всех уровнях. Кроме того, от роли узла организационной структуры по отношению к задаче (инициатор, лицо, принимающее решение, координатор, руководитель, непосредственный исполнитель, посредник) будут зависеть его сценарии поведения и протоколы взаимодействия с другими агентами.

Для координации доступа к ресурсам необходим какой-то механизм, реализованный либо централизованно (диспетчер), либо децентрализовано через взаимодействие агентов. Агент управления ресурсами *ResourceManager* осуществляет подбор ресурсов для выполнения задач, контроль параметров ресурсов, подсчет затрат за хранение, наблюдая изменяющиеся потребности в ресурсах, может закупать (дозаказывать) пользующиеся спросом материалы и комплектующие или принимать решения по сокращению или увеличению складских помещений, тем самым осуществляя постоянную адаптацию и эволюционное развитие. Каждый агент-ресурс в системе зарегистрирован и приписан к одному из агентов управления ресурсами, общее количество которых может определяться, например, группированием ресурсов по каким-либо признакам.

Для агентов исполнителей ту же роль выполняет агент управления организационной структурой *OrgModelManager*.

Агент резерва *ReserveAgent* создается в целях обеспечения устойчивых к рискам проектов, когда необходимо создавать собственный дополнительный запас страховых ресурсов или резервов, предназначенных для парирования комплекса возможных рисков.

Координацию доступа к резервам осуществляет агент управления резервами *ReserveManager*.

Агент-риска *RiskAgent* предназначен для моделирования наступления рисковых ситуаций. Для каждой задачи на этапе идентификации рисков (указывается возможность присутствия риска для задачи из классификационного перечня) задается вероятность данного риска, опасность данного риска, т.е. насколько существенными окажутся последствия наступления неблагоприятного события, важность риска как произведение вероятности на опасность его наступления. Кроме того для каждого риска могут быть заданы следующие параметры: фаза выполнения задачи, где проявляется риск, тип и размер убытка (например, в виде требуемых дополнительно затрат или ресурсов), признак застрахованности для данного риска (указывается источник (собственный резервный фонд или третье лицо), сумма, порядок выплаты и размер процента (для внешнего страхования)). Создание нового агента-риска в системе будет соответствовать наступлению рискового события.

Агент моделирования *SimulationAgent* осуществляет общий контроль и управление моделированием комплекса работ, обеспечивает поддержку кооперативного решения возникающих конфликтов, если это находится в его компетенции, помогает агентам в координации действий, распределяет привилегии, предлагает разнообразные возможные общие решения, может переключать переговоры агентов (например, с избежания штрафа за счет выполнения директивных сроков проекта, на резервирование времени по ремонту оборудования собственными силами).

Агент планирования *SchedulerAgent* осуществляет планирование событий в системе, связанных с выполнением задач по проекту, синхронизацию, обеспечивает генерацию заявок.

Агент мониторинга *ControlAgent* осуществляет сбор всех данных и отправку их агенту хранения, выявление неблагоприятных ситуаций, которые определяются для различных аспектов модели проекта: комплекса работ (задачи с малым свободным резервом, продолжительные задачи с большим числом ресурсов, задачи с внешними зависимостями), ресурсов (ресурсы с большим объемом работ, исполнители с уникальными навыками, материалы с единственными поставщиками), финансов, резервов (уровни резервов, скорость потребления) и т.д. Так выявляются события, осуществление которых может помешать завершить проект в срок или создать нехватку ресурсов или финансовых средств в определенный момент его выполнения. Обо всех выявленных нарушениях он сообщает агенту менеджеру.

Агент менеджер *ManagementAgent* обеспечивает функциональные возможности по управлению всем потоком задач по проекту, создание и удаление новых задач в ходе моделирования (перепланирование), создание новых агентов ресурса, определение ролей, инициирует диалог с пользователем, в случае, когда для продолжения моделирования необходимы его решения. Если при получении сообщения от агента мониторинга в случае неблагоприятной ситуации становится ясно, как уменьшить степень риска от ее наступления, то происходит сразу же внесение соответствующих изменений в модель или же происходит инициация диалога с пользователем.

На агента восстановления *RecoveryAgent* возлагаются обязанности, связанные с координацией действий других агентов (в частности, *SimulationAgent, SchedulerAgent*) при разрешении (перепланирование работ, восстановление/ремонт ресурсов, использование резервов и др.) возникающих рисковых событий в системе. Кроме того, обращаясь к исторической базе данных, где сохраняется вся информация о ходе моделирования, он может осуществлять возврат к предыдущим шагам моделирования по требованию пользователя.

Агент хранения *StorageAgent* управляет сохранением всей статистики по моделированию в базе данных.

Агент базы знаний *KBAgent* управляет взаимодействием с базой знаний системы, которая содержит онтологию, разделяемую всеми агентами системы и базы правил для принятия решений в ходе моделирования и на этапе интерпретации результатов.

Архитектура агентной системы имитационного моделирования. Для построения прототипа системы (рис. 2) была выбрана платформа JADE (*Java Agent Development Framework*), которая упрощает создание мультиагетных систем с помощью реализованных FIPA (*The Foundation for Intelligent, Physical Agents*)-спецификаций, а также поддержки фаз отладки и внедрения.

Каждый запущенный экземпляр JADE runtime-окружения называется контейнером и может содержать нескольких агентов. Набор активных контейнеров называется платформой.

Архитектура платформы JADE включает: AMS (Agent Management System) – служба имен и сервис авторизации, DF (Directory Facilitator) – сервис “желтых страниц”, который позволяет агентам производить поиск других агентов по типу предоставляемого сервиса.

Все агенты системы создаются как классы, расширяющие *jade.core.Agent*. Поведение агентов – классы, унаследованные от *jade.core.behaviors.Behavior*. Поведение агента может быть одноразовым *jade.core.behaviors.OneShotBehavior* (исполняются только один раз), циклическим *jade.core.behaviors.CyclicBehavior* (никогда не заканчиваются), общим (требуют определения *done()* метода самостоятельно).

Преимуществом JADE является то, что планировщик поведений использует кооперативную, а не вытесняющую многопоточность, что дает более высокую производительность и не требует синхронизации потоков.

Агенты в JADE могут взаимодействовать друг с другом, посылая сообщения. Для передачи сообщений используется язык FIPA ACL.

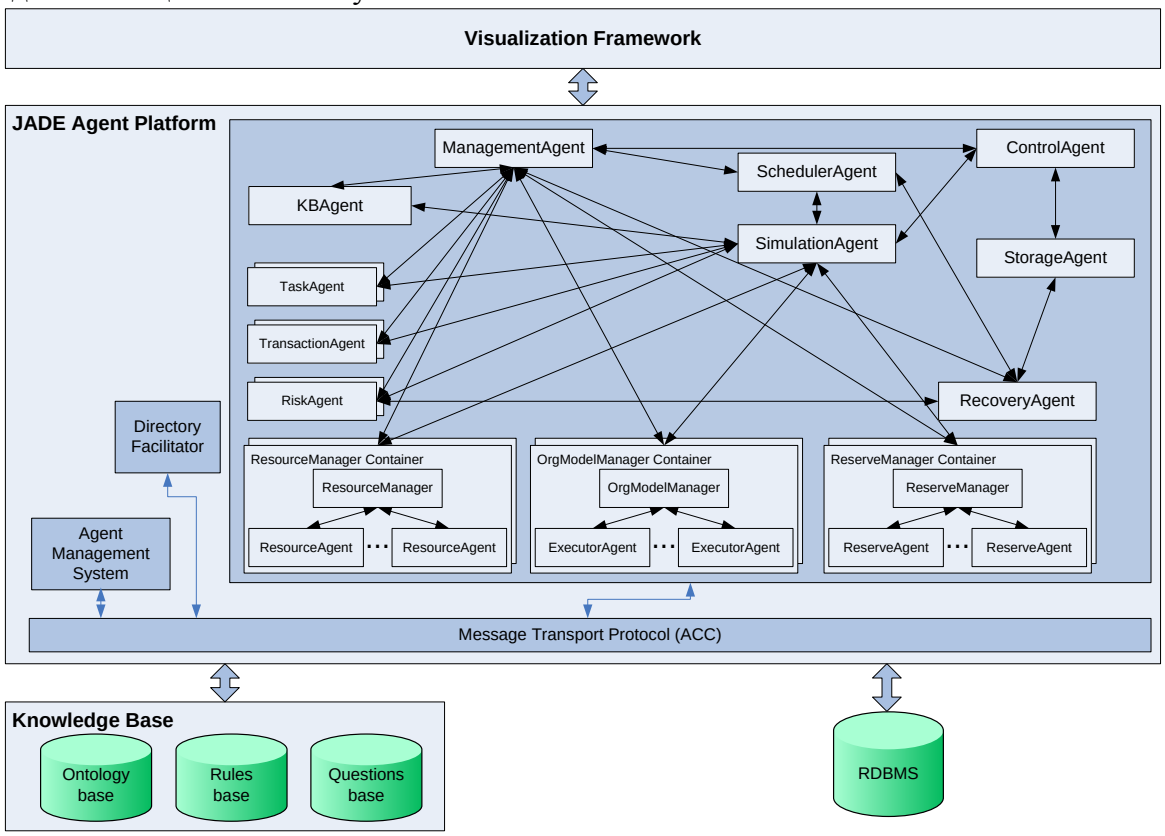

Рис. 2. Архитектура системы моделирования

Для всех агентов реализован язык коммуникаций и транспортный уровень для передачи сообщений. Для агентов менеджеров, координирующих работу других агентов, созданы каталоги агентов DF.

На рис. 3. представлена типовая структура интеллектуального агента системы моделирования. База знаний агента включает онтологию, базу правил и вопросов для принятия решений агентом в ходе моделирования. Основным элементом интеллектуального агента, дающим ему возможность принимать решения, планировать действия, взаимодействовать с другими агентами, является онтологическая база знаний, содержащая модели концептуальных понятий, отношений и правила для анализа и ситуативной ориентации. Онтология служит для упрощения программирования поведения агентов и используется ими при взаимодействии. Для формального описания онтологий, в работе используется язык RDF/OWL. Онтология для данной системы разрабатывалась в системе Protégé.

Парсер онтологии производит преобразование онтологии из формата owl в внутренний формат базы знаний агента на языке исчисления предикатов. Подсистема принятия решений осуществляет логический вывод решений на основе метода резолюций для исчисления предикатов первого порядка.

Управляющий модуль агента производит анализ его текущего состояния, осуществляет общий контроль и управление событиями от различных агентов и программных модулей агента, управляет запуском обработчика событий, обеспечивает выполнение основных функций агента в зависимости от текущей ситуации. Планировщик действий агента осуществляет диспетчеризацию действий агента направленных на работу с внешней средой и с пользователем. Обработчик событий под общим контролем управляющего модуля осуществляет просмотр необработанных записей о сгенерированных событиях, и производит их обработку. Для реализации сложных моделей поведения агентов используются модели знаний на языке исчисления предикатов первого порядка, логический вывод по которым, осуществляется подсистемой принятия решений. При выполнении действий агента доступ к локальной базе данных агента осуществляется посредством специального SQL-модуля.

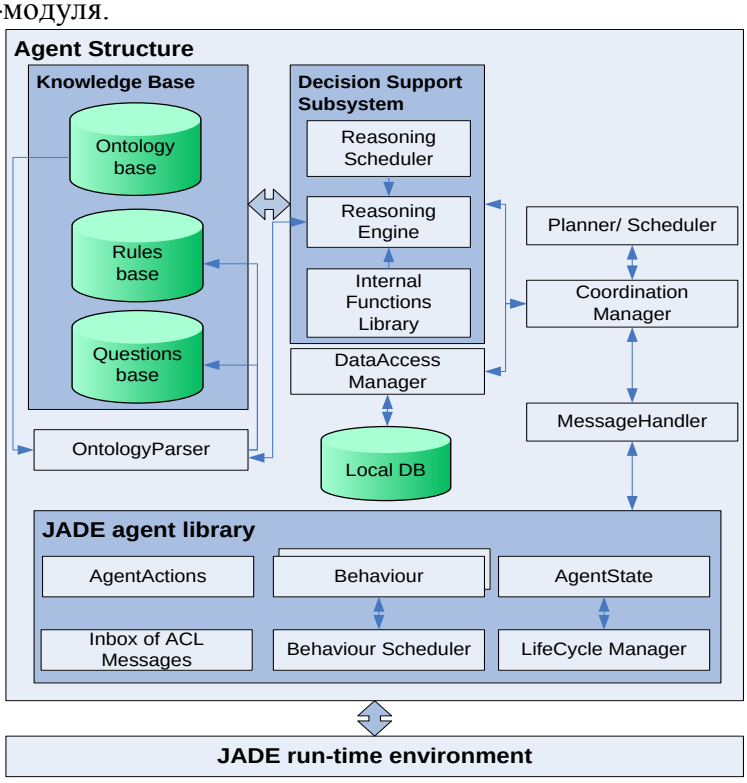

Рис. 3. Типовая структура интеллектуального агента

Заключение. Таким образом, предлагаемый подход позволяет достичь следующих преимуществ: автономность различных частей моделирующей программы (агентов), совместно функционирующих в распределенной системе, где одновременно протекает множество взаимосвязанных процессов; наличие элементов индивидуального поведения,

заложенных в систему в виде моделей знаний; агенты имеют возможность обучаться, адаптироваться и менять свое поведение, иметь динамические связи с другими агентами, которые могут формироваться и исчезать в процессе моделирования и др.

Литература

1. Borshchev A., Filippov A. . From System Dynamics and Discrete Event to Practical Agent Based Modeling: Reasons, Techniques, Tools. The 22nd International Conference of the System Dynamics Society, 2004, Oxford, England.
2. Macal C.M., North M.J. Tutorial on Agent-based Modeling and Simulation // Proceedings of the 2005 Winter Simulation Conference. WSC'05. 2005.
3. Marietto M., David N., Sichman J.S., Coelho H. Requirements Analysis of Agent-Based Simulaton Platforms: State of the Art and New Prospects // Multi-Agent-Based Simulation II, Vol. 2581 of LNAI series, Springer-Verlag. 2002.
4. N. Ruiz, A. Giret and V. Botti An Agent-Supported Simulation Architecture for Manufacturing Systems Agent-Directed Simulation 2007 .Vol. 1 pp. 63-70.
5. Yilmaz, L., T.I. Ören. Discrete-Event Multimodels and their Agent-Supported Activation and Update. Proceedings of the Agent-Directed Simulation Symposium of the Spring Simulation Multiconference (SMC'05), San Diego, pp. 63-72.
6. Инструментальная среда разработки многоагентных приложений MASDK. Основные характеристики среды / В. И. Городецкий, О. В. Карсаев, В. . Конюший, С. П. Ющенко // Телекоммуникации. — 2005. — №7. — С. 2-5.
7. Карпов Ю. Имитационное моделирование систем. Введение в моделирование с AnyLogic 5. – СПб.: БХВ-Петербург, 2005. – 400 с.
8. Котенко И.В., Уланов А.В. Агентно-ориентированное моделирование поведения сложных систем в среде Интернет // КИИ-2006. X Национальная конференция по искусственному интеллекту с международным участием. Труды конференции. Том 2. М.: Физматлит, 2006. С.660-668.
9. Федорович О.Е., Прохоров А.В., Жигулина Е.М. Имитационная модель анализа процессов управления проектами с учетом рисков // Авиационно-космическая техника и технология: Науч.-техн. журнал. Вып. 1 (37). – Харьков: ХАИ, 2007. – С. 75–84.