

Прикладные многоагентные системы группового управления

Аннотация. В работе рассматривается технология построения прикладных систем группового управления, состоящих из большого числа автономных подсистем, организованных в сеть, узлы которой могут работать под управлением различных операционных систем и в различных коммуникационных средах. Технология интегрирует подходы распределенного принятия решений, многоагентных систем, ориентированной на сервис архитектуры и вычислений на основе парных взаимодействий. Технология поддерживается инструментальными средствами, которые обеспечивают эффективную разработку агентов и механизмов их взаимодействия. Приводятся примеры использования технологии в ряде приложений, в частности, для автономного управления воздушным движением в районе аэропорта.

Ключевые слова: групповое управление, многоагентные системы, автономные системы, парные взаимодействия, инструментальные средства, агентская платформа.

Введение

Проблема коллективного поведения и группового управления не является новой и исследуется уже в течение нескольких десятилетий, начиная с классических работ Дж. Фон Неймана, М. Цейтлина [7], В. Варшавского и Д. Поспелова [2, 3] и др. Ее актуальность возрастает год от года, причем в настоящее время проблема группового управления ставится уже по-новому, а для ее решения требуются новые парадигмы концептуализации, новые модели формализации и новые архитектуры построения, что, в свою очередь, требует создания новых технологий проектирования и программной реализации. Обусловлено это, прежде всего, особенностями современных прикладных задач группового управления, в частности, значительным возрастанием их масштабности, необходимостью обеспечить согласованное функционирование большого количества гетерогенных подсистем, организованных в сеть, различные узлы которой могут работать в различных операционных средах и использовать различные коммуникационные протоколы. В таких приложениях самыми трудными оказываются две очень разные задачи, а именно:

обеспечение эффективной координации группового поведения и создание программной инфраструктуры (платформы) поддерживающей взаимодействие большого количества разнородных устройств и программ.

С точки зрения архитектуры современные приложения характеризуются такими свойствами, как *открытость* и *автономность*. Открытость понимается как способность прикладной системы группового управления изменять в процессе работы свою структуру (за счет изменения количества автономных подсистем и структуры их связей), а автономность понимается как способность объектов приложения формировать и изменять при необходимости свои текущие цели, функционировать без вмешательства человека и осуществлять самоконтроль над своими действиями и внутренним состоянием. Как правило, автономные объекты приложения имеют свои индивидуальные цели, отличные от целей других объектов системы и от целей системы в целом, так что согласование этих целей является дополнительной проблемой.

Особенно трудными являются задачи управления группами подвижных объектов, когда они по тем или иным причинам должны или координировать свое поведение, например, для

избегания столкновений, или кооперироваться для совместного решения задач. Характерными примерами приложений такого типа являются управление коллективом роботов, групповое управление полетом беспилотных летательных аппаратов (БПЛА) и управление выполнением миссии (целевого назначения) группы БПЛА, автономное управление воздушным движением в районе аэропорта и др. Другой большой класс прикладных задач группового управления связан с использованием мобильных систем и управлением ими. В качестве примера можно указать задачи управления силами и средствами при техногенных авариях, крупных пожарах, природных катастрофах и т.п. с целью минимизации их вредных последствий. В процесс управления здесь вовлекаются мобильные группы различного назначения и динамического состава (разведывательные, пожарные, обеспечения эвакуации и медицинской помощи, информационной поддержки, связи и т.п.), используются мобильные каналы связи и мобильные устройства. Иногда может потребоваться оперативно развертывать беспроводную коммуникационную инфраструктуру (например, в труднодоступной местности) и сеть сенсоров.

Для этих и других аналогичных приложений общими характерными чертами, оказывающими первостепенное влияние на требования к информационным технологиям, которые способны справиться с их разработкой, являются следующие:

1. Сетевая организация с очень большим числом узлов.

2. Большое число источников, генерирующих огромные потоки данных, которые технически невозможно хранить в каком-либо централизованном хранилище. Для последующего решения задач анализа и принятия решений в архитектуре с централизованной обработкой могут потребоваться вычислительные средства нереальной производительности.

3. Открытость систем, когда состав узлов сети и ее топология постоянно изменяются. Действительно, мобильные устройства могут включаться и выключаться, сенсоры сети могут выходить из строя и прекращать работу из-за истощения источников энергии. Беспилотные летательные аппараты, формирующие распределенную систему наблюдения, могут прекращать свое существование в любой момент вре-

мени. В компьютерной сети могут включаться в работу и выключаться различные приложения и вычислительные устройства и т.п. По этим причинам в ряде случаев архитектура приложения типа "клиент-сервер" оказывается неработоспособной.

4. Конфиденциальность данных отдельных источников, поскольку данные могут попадать под понятие военной или государственной тайны, носить коммерческий характер и т.п., и тогда централизованная обработка невозможна по организационным соображениям.

5. Сочетание автономности задач, которые решаются программными сущностями, установленными в различных узлах сети, с необходимостью координации поведения и кооперации соответствующих автономных программ. Во многих приложениях для совместного решения задач необходимо или формировать временные либо устойчивые коалиции узлов сети и установленных на них программных сущностей, и/или координировать распределение общих ограниченных ресурсов. Это может потребовать использования специальных алгоритмов формирования коалиций, например, с помощью механизмов распределенного обучения.

6. Гетерогенность операционной среды, в которой функционируют программы, установленные в различных узлах сети. Например, в мобильной вычислительной и коммуникационной средах различные узлы могут работать под управлением различных операционных систем (MS Windows, Linux, Symbian, TinyOS и др.), а также использовать совместно различные коммуникационные каналы и протоколы (проводные и беспроводные каналы, TCP/IP, UDP, WiFi, Bluetooth и др. протоколы). Это требует использования специальных механизмов взаимодействия программ, установленных в узлах сети, которые были бы инвариантны к типу операционной и коммуникационной сред и обеспечивали бы совместную работу ("interoperability") распределенных программных компонент.

В настоящее время исследования и разработки в области систем группового управления ведутся во многих странах, в них вовлекаются большие коллективы людей, исследования и разработки проводятся во многих направлениях. Тем не менее, в настоящее время пока не создано (информационной) технологии, которая была бы в состоянии справиться с реализа-

цией приложений в области группового управления с отмеченными выше особенностями.

Совместный анализ возможностей современных информационных технологий (ИТ) и требований к ним со стороны приложений рассматриваемого типа показывает, что наибольшими перспективами здесь обладают ИТ, интегрирующие *идеи распределенного принятия решений* (они смещают основной объем вычислений по обработке данных на уровень источников данных, отдельных подсистем), *концепцию многоагентных систем* (она очень удобна для концептуализации и декомпозиции распределенных проблем), *архитектуры, ориентированной на сервис* (в ней агенты могут кооперироваться при минимуме начальной информации о топологии сети, об агентах, присутствующих в ней, и о доступных сервисах) и *вычислений на основе парных (Peer-to-Peer, P2P) взаимодействий* (эта технология поддерживает реализацию свойства открытости, поскольку она не требует использования централизованного сервера для поддержки взаимодействия агентов).

Вопросы интеграции этих технологий и их использования при создании многоагентных приложений в области группового управления составляют основное содержание данной работы. В разделе 1 кратко анализируются особенности многоагентных систем, которые существенны для координации и управления групповой работой автономных агентов. Описываются классы методов координации, которые наиболее часто используются в многоагентных системах группового управления и указывается их связь с классическими методами координации, принятыми в теории сложных иерархических систем [6].

В разделе 2 приводятся основные сведения о разработанной технологии создания многоагентных систем группового управления и инструментальных средствах ее поддержки. Описывается разработанная распределенная агентская платформа, которая поддерживает парное взаимодействие агентов, работающих в компьютерной сети. Особенность этой платформы состоит в том, что она не предполагает использования какой-либо централизованной компоненты и, таким образом, реализует концепцию полной открытости. Платформа поддерживает совместную работу системы с динамическим составом узлов сети и установленных на ней агентов в условиях, когда топологии се-

ти динамически меняется. В этом же разделе кратко описывается предложенная технология разработки агентов и многоагентных приложений и инструментальное средство ее поддержки, MASDK 4.0. Это средство реализует графический стиль создания интеллектуальных агентских приложений и поддерживает все этапы жизненного цикла разработки приложения от концептуализации проблемы до генерации кода и развертывания готовой программной системы в компьютерной сети.

В двух последующих разделах описываются конкретные примеры разработанных многоагентных приложений в области задачи группового управления. В разделе 3 дается описание разработанной модели и программного прототипа многоагентной системы (МАС) автономного управления воздушным движением в районе аэропорта, которая минимизирует участие диспетчера в процессе управления. Этот пример демонстрирует также использование инструментального средства MASDK 4.0 в процессе разработки. В разделе 4 дается описание программного прототипа МАС, моделирующей групповую работу распределенных средств наблюдения, установленных на беспилотных летательных аппаратах, предназначенных для обнаружения объектов на земной поверхности.

В заключении резюмируются основные результаты работы и указываются некоторые направления будущих работ применительно к МАС группового управления.

1. Общие принципы и модели координация группового поведения

В парадигме многоагентных систем координация поведения агентов является неотъемлемой функциональностью агентов. Ситуации и причины, которые вызывают необходимость координации группового поведения, очень разнообразны. Уже сама модель агента в МАС предполагает, что

- агенты существуют в *общей внешней среде*, где имеются, например, пространственные ограничения (избежание столкновений роботов, соблюдение дистанций безопасности при управлении воздушным движением, учет глобального контекста, ограничение на общее время выполнения проекта и др.);

- агенты имеют ограниченные общие ресурсы (средства транспортировки в логистике, компьютерные ресурсы, доступные программным агентам, и т.д.);

- агенты существуют и принимают решения в условиях неопределенности, когда каждый агент обладает ограниченной информацией, что влечет необходимость информационного обмена между ними;

- агенты обладают ограниченной компетенцией и возможностями, что может быть восполнено путем привлечения знаний и функциональных возможностей других агентов;

- агенты должны синхронизировать свои действия при решении общей проблемы, в частности, в приложениях реального времени (управление воздушным движением, футбол роботов, проведение гуманитарной операции и др.).

Все эти особенности модели автоматически подразумевают необходимость координации, причем достаточно разнообразной.

С содержательной точки зрения *координация* предназначена для *согласования индивидуальных целей и вариантов поведения агентов*, при которых каждый агент улучшает или не ухудшает значение своей функции полезности, или система в целом улучшает качество решения общей задачи (если система имеет общую глобальную цель).

Задача координации поведения агентов является центральной и наиболее сложной задачей в области МАС и потому привлекает большое внимание исследователей и разработчиков. С точки зрения теории, эта задача не является новой, как и многие другие, которые составляют теоретический базис МАС. Ее истоки следует связывать с теорией систем [6], классической теорией управления и исследованием операций, теорией двойственности, теорией игр, планированием и другими областями математики и кибернетики, становление и развитие которых относится еще к 1950-60-м годам. Именно тогда сформировались многие ключевые идеи, методы и модели, которые в настоящее время явно или неявно эксплуатируются или "открываются" заново при построении формальных моделей и методов координации поведения агентов МАС.

В [6] сформулированы три базовых принципа координации в сложных многоуровневых системах. Они хорошо известны. К ним относятся *прогнозирование взаимодействий, развязывание взаимодействий и оценка взаимодействий*. Эти

принципы принимают конкретную алгоритмическую форму в рамках конкретной математической модели и конкретного приложения. Среди них наиболее известные алгоритмы координации базируются, например, на назначении и согласовании локальных показателей качества, оптимизация которых ведет к оптимизации глобального показателя, на распределении общего ресурса, согласовании значений общих переменных и др. В рамках сложной системы эти принципы могут дополнительно комбинироваться. Они же являются базовыми и для МАС. Возможно, единственный механизм, который не вполне попадает в число названных принципов координации, используемых специалистами в области МАС, это так называемый механизм "*конструирования правил*" ("*Mechanism Design*"). Он используется в игровых моделях конкурентного взаимодействия агентов и предполагает искусственное конструирование таких правил "игры" участников (агентов), которые ведут к ситуации *равновесия с заранее заданными свойствами группового поведения*. Его отличие от принципа назначения локальных показателей состоит в том, что "правила игры" здесь формируют *требуемый* механизм самоорганизации коллективного поведения, который реализуется без вмешательства "сверху".

Следует отметить одно важное обстоятельство. Механизмы координации в МАС, как правило, являются приближенными и не имеют цели получить оптимальное совместное поведение. Именно за счет *отказа* от обязательного стремления к *оптимальности* задачи координации в МАС решаются эффективно с вычислительной точки зрения.

В основе большинства известных методов координации МАС, явно или неявно, лежит понятие "*обязательства*" (*commitments*) агента, которое постулирует необходимость выполнения агентом последовательности действий, ведущей к достижению предопределенной цели в интересах сообщества агентов. Значимость этого понятия в задачах координации МАС обусловлена его сильным влиянием на процесс практических рассуждений агента. Действительно, во-первых, агент имеет всегда ограниченные вычислительные ресурсы, а потому не имеет возможности постоянно переоценивать свои намерения, обновлять свои убеждения (знания о внешнем мире) и уточнять свои текущие цели. Выбрав намерение в некоторый момент, агент берет на себя *обязательство* достичь соответствующей (высоко-

уровневой) цели. Во-вторых, обязательства агента являются основой для планирования им своих будущих действий. Агент использует также свои знания об обязательствах других агентов для того, чтобы учесть при планировании "общественный контекст" и ограничения, которые он должен принимать во внимание. Заметим, что одной из форм обязательств агента является его *роль*, которую он взял на себя или которая ему назначена.

Вторым важным понятием, связанным с координацией, является понятие общественных "*соглашений*" (*conventions*), которое фиксирует условия, определяющие обстоятельства, при которых обязательства выполняются, и обстоятельства, когда агент может или должен отказываться от исполнения взятых на себя обязательств.

В работе [25] выдвинута важная гипотеза, которая утверждает, что "*все механизмы координации в многоагентных системах могут быть выражены в терминах (совместных) обязательств агентов и отвечающих им соглашений*". Анализируя различные известные методы координации в МАС, которые изначально не были сформулированы в терминах обязательств и соглашений, автор показывает, что они действительно могут быть выражены в этих терминах.

Конкретные методы координации, используемые в теории и практике МАС, достаточно разнообразны и хорошо освещены в литературе. Перечислим некоторые из них.

1. *Координация с помощью удовлетворения общих правил группового поведения (social rules)*. Обычно такой подход используется (совместно с другими) в системах с заданной организационной структурой, в которой правила общественного поведения должны строго выполняться [35]. Эти правила формируют свойства среды, в которой функционируют агенты. Биологическим прототипом такого варианта координации поведения агентов являются пчелиный рой или сообщества муравьев, термитов и т.п. В таких сообществах каждый агент является простейшим автоматом. Однако множество правил группового поведения формирует систему, которая проявляет высокоорганизованное поведение, которая по существу является новым "организмом". Здесь координация содержит два вида деятельности: поддержание организационной структуры сообщества (обеспечение невозможности нарушения *общественных правил*, например, принятых междуна-

родных правил поведения самолетов в воздушном пространстве аэропорта) и использование правил, принятых в этой общественной структуре, для вычисления конкретных действий агента с использованием его локальных знаний (его "ментального состояния"). Первый аспект для каждого агента определяет, когда, кому и какая информация должна посылаться (по каким протоколам выполняется информационный обмен), а второй определяет важность целей агента, соотношение важности целей, сгенерированных извне (для выполнения общественных правил), и локальных целей агента.

2. *Координация поведения на основе обмена информацией на метауровне*. Типичным примером такой координации является задача распределенного планирования [17], например, планирование с использованием метода "Частичного глобального планирования" ("*Partial Global Planning*", PGP [12]). В нем агенты информируют друг друга о своих локальных планах (не детализируя их) и ведут переговоры для согласования своих обязательств и разрешения конфликтов. Обмен частичными планами выполняется агентами для того, чтобы построить представление о поведении партнеров по МАС: установить, какие цели будут достигаться, в каком порядке, какие будут получены результаты и сколько времени ориентировочно займет процесс достижения каждой из целей. Далее агенты, вовлеченные в выполнение некоторой части плана, строят модель собственного поведения и модели поведения партнеров по группе и далее по частичным планам устанавливают, когда их планы являются частью общего плана группы агентов. Если такие части планов имеются, то они на верхнем уровне объединяются в единый план совместной деятельности группы агентов. Этот план анализируется и агенты, если необходимо, пересматривают свои локальные планы, чтобы добиться их согласования в общем плане, например, чтобы добиться более эффективного использования общих ресурсов или чтобы синхронизировать свое поведение. Этот процесс реализуется с использованием метаправил. Например, если некоторый агент формирует информацию, которая потребуется нескольким другим агентам, то ему отдается приоритет в выделении ресурсов (ему может быть выделено больше вычислительных ресурсов для ускорения получения нужной информации). На основании анализа частичных пла-

нов может быть установлено, что некоторые агенты перегружены работой, в то время как другие – недогружены. Это может послужить основанием для корректировки частичных локальных планов агентов путем перераспределения задач. Такой вариант координации обычно реализуется с помощью ролевого подхода, когда агенты должны выполнять некоторые роли, (выбранные агентами или назначенные им), которые взаимодействуют в соответствии со строго определенными протоколами. Заметим, что в задаче автономного управления воздушным движением в районе аэропорта (раздел 3) используется именно такой метод координации.

По существу, описанный метод координации является комбинацией координации путем назначения локальных показателей качества и их согласованием на метауровне, и координации путем перераспределения ресурсов [6].

Существует также много других частных подходов, методов и вариантов координации поведения агентов МАС. Более детальное представление о них может быть почерпнуто из работы [11], где дается также обширный список работ по данной тематике и которая имеется в свободном Интернет-доступе.

Особое место занимают две крайние ситуации взаимодействия агентов. Одна из них – это так называемая "*командная работа*", а вторая – это "*координация в условиях конкуренции агентов*".

3. *Командная работа.* Про агентов, которые прилагают совместные усилия (сотрудничают) для достижения общей *долговременной* цели, функционируют в динамической внешней среде в присутствии шума и противодействия со стороны соперника (или команды соперников) принято говорить, что они образуют *команду агентов*. Наличие соперничающей команды или "*враждебной*" внешней среды является одной из специфических особенностей, которая отличает командную работу от обычной кооперации агентов в МАС. Таким образом, командное поведение агентов есть нечто большее, чем просто скоординированное множество индивидуальных действий агентов МАС. Основная проблема организации командной работы агентов состоит в том, чтобы обеспечить работу агентов как единой команды в ситуации, когда каждый агент имеет *ограниченную информацию* о собственной команде, о внешней среде и о сопернике. При этом он должен реализовывать собственные намерения с помощью индивидуальных действий, исполняемых

параллельно или последовательно с действиями других агентов. Теория командной работы – это особая область теории МАС. Она достаточно специфична. Отметим, что основополагающие результаты в этой области содержат "Теория общих намерений" [9, 10], "Теория общих планов" [23] и "Комбинированная теория" [38]. Обширная литература по данной теме может быть найдена в Интернет.

4. *Координация в условиях конкуренции агентов.* Существует много приложений многоагентных систем, когда агенты, так или иначе, конкурируют. Про таких агентов говорят, что они являются "*самоинтересованными*" (*self-interested*), или "*эгоистичными*". Формально, агенты называются эгоистичными, если каждый из них стремится максимизировать свою функцию полезности и при этом *полностью игнорирует* интересы других агентов. В таких ситуациях взаимодействие агентов реализуется в форме *переговоров*, цель которых, вообще говоря, разрешение конфликтов.

В процессе переговоров агенты стремятся достичь взаимовыгодного соглашения. При этом предполагается, что правила ведения переговоров предварительно установлены и известны всем агентам. Обычно правила ведения переговоров формализуются в виде *протокола переговоров*. При этом к протоколам переговоров предъявляются следующие основные требования [1]:

Эффективность. Под эффективностью понимается не только эффективность по Парето для каждого агента, но и эффективность по Парето по отношению к группам (коалициям) агентов.

Устойчивость. Протоколы переговоров не должны позволять агентам извлекать дополнительную пользу при отклонении от установленных правил, т.е. правила ведения переговоров должны обеспечить достижение состояния равновесия.

Простота. Протокол переговоров не должен предъявлять дополнительных требований к архитектуре агентов и усложнять структуру коммуникационных протоколов.

Децентрализованность. Процесс переговоров желательно выполнять без использования централизованного управления, хотя часто в той или иной форме присутствует некий посредник, ответственный за реализацию самого протокола.

Симметричность. Протокол переговоров должен предоставлять всем агентам одинаковые возможности по участию в переговорах.

Механизм переговоров является объектом исследования в теории игр с начала 50-х годов, когда были опубликованы классические работы Дж.Нэша [32]. Как правило, протоколы многоагентных переговоров основываются либо на теоретико-игровой арбитражной схеме Нэша [32], либо на модели аукциона [26], а теоретические свойства аукциона обычно устанавливаются с помощью теоретико-игровых моделей. Для каждого протокола переговоров должна быть задана модель, состоящая из трех компонент. Первая компонента задает *пространство возможных соглашений*. Оно должно быть фиксировано и известно всем агентам. Вторая компонента вводит *правила взаимодействия*. В теории MAC практически все протоколы переговоров используют модель "*предложение–контрпредложение*", которая легко представляется в терминах речевых актов [34, 9], а потому не создает дополнительных проблем для коммуникационных компонент агентов MAC. Третья компонента протокола задает *оптимальные стратегии поведения агентов*. В "хорошем" протоколе оптимальные стратегии агентов образуют равновесие по Нэшу, хотя для случая, когда число агентов более двух, задача определения состояния равновесия является непростой [39, 40, 33].

В настоящее время литература, описывающая различные схемы построения протоколов переговоров, их теоретические свойства, области целесообразного использования и т.п. достаточно обширна, и описать имеющиеся здесь результаты не представляется возможным. Кроме работ, цитированных выше, дополнительные результаты могут быть найдены в работах [36, 37, 16, 26, 8], а также в большом количестве более поздних работ, которые могут быть легко найдены в Интернет.

2. Инструментальная поддержка процессов разработки MAC группового управления

2.1. Подходы к разработке

Любая многоагентная система состоит из программных агентов и агентской платформы, которая поддерживает взаимодействие агентов. Наиболее совершенные из существующих инструментальных средств поддержки технологии

разработки MAC обычно интегрируют в себе средства разработки компонент обеих типов. Обычно принято использовать агентскую платформу, которая построена в соответствии с *абстрактной архитектурой FIPA* [14]. Поэтому различные реализации, в основном, различаются интерфейсной частью, связывающей агентов и агентскую платформу. Значительно больше разнообразия в средствах разработки агентов. По этому признаку все инструментальные средства создания агентов можно разделить на два базовых класса, а именно, инструментари, ориентированные на разработку MAC *BDI (Belief–Desire–Intention)-архитектуры*, и инструментари, ориентированные на разработку агентов с *архитектурой, ориентированной на явное описание группового поведения*.

BDI-архитектура (по смыслу – «архитектура, управляемая намерениями и знаниями», или «архитектура, ориентированная на достижение заданных целей») представляет знания, механизмы вывода и внутренние состояния агента в такой форме, которая позволяет ему при поиске решений рассуждать аналогично человеку. Обычно в этой архитектуре информационные, мотивационные и поведенческие аспекты модели агента представляются в терминах его *ментальных* понятий, к которым относятся знания агента, его убеждения (знания о внешнем мире), желания и намерения. Эти понятия формализуются с помощью тех или иных расширений языков модальных и темпоральных логик. При всем разнообразии вариантов построения таких агентов общее в них состоит в том, что в процессе принятия решения о поведении агент выбирает конкретную цель, стоит план ее достижения, и выполняет этот план. BDI-агенты должны выполнять этот процесс самостоятельно на основе собственных "интеллектуальных" возможностей. Можно говорить, что BDI-архитектура ориентирована на описание модели *индивидуального поведения* агента в групповом поведении, но не на описание группового поведения MAC в целом. "Групповое поведение" здесь должно возникать как результат интеграции поведения агентов сообщества и обмена сообщениями между ними. Заметим, что механизмы вывода при этом обычно оказываются достаточно сложными, а в сложных случаях групповое поведение MAC может оказаться и непредсказуемым. Однако существует широкий класс приложений, где такая архитек-

тура оказывается эффективной. Можно также отметить, что BDI-архитектура является весьма популярной среди исследователей, склонных больше к теоретической работе.

Другой класс архитектуры – это *архитектура, ориентированная на явное описание группового поведения*. При ее разработке сначала на основании содержательного анализа приложения определяется множество возможных сценариев группового поведения МАС в целом. Далее для каждого такого сценария определяются роли, которые в нем должны играть агенты, а также протоколы, в соответствии с которыми эти роли должны в этих протоколах взаимодействовать. Иначе говоря, в этой архитектуре внимание акцентируется на "внешнем поведении" агента, т.е. на поведении, которое "видно" другим агентам и которое явно задается в терминах ролей и протоколов. В этой архитектуре поведение агента управляется событиями. Источниками событий могут быть сообщения других агентов, сигналы сенсоров, воспринимающих состояние внешней среды и др. События могут инициироваться самим агентом при переходе в некоторое внутреннее состояние. В этой архитектуре групповое поведение агентов реализуется проще, описание ментальных понятий агента и процессов выработки решений выглядит более простым и понятным, отладка и оценка свойств результирующей системы также могут быть выполнены проще. Методологической основой технологии разработки МАС такой архитектуры является методология Gaia [41].

Применительно к приложениям в области группового управления, которые, например, перечислены во вводной части работы, последняя из описанных архитектур является намного более предпочтительной. В таких приложениях самый сложный аспект модели МАС, каким является групповое поведение, описывается явно, а функции самих агентов (их роли) обычно описываются в более простой форме. В этой архитектуре проще выполнить все те требования к процессам координации группового поведения, которые были сформулированы в начале раздела 1.

Для поддержки технологии создания прикладных МАС с архитектурой, ориентированной на явное описание группового поведения, авторы разработали два базовых инструментальных средства. Одно из них – это инстру-

ментальное средство *Multi-Agent System Development Kit* (MASDK), реализующее упомянутую выше методологию Gaia [41]. Оно рассчитано на поддержку технологии разработки широкого круга прикладных многоагентных систем, в частности, систем группового управления. Вторая компонента предназначена для генерации программной инфраструктуры, обеспечивающей открытость МАС на основе *архитектуры, ориентированной на сервис*. В последующих двух подразделах дается описание названных инструментальных средств.

2.2. MASDK 4.0 – инструментальное средство поддержки технологии разработки

Инструментальная среда MASDK для разработки многоагентных приложений разрабатывается и тестируется в СПИИРАН уже в течение последних восьми лет, а описываемая далее реализация соответствует ее версии 4.0. Эта версия инструментальной среды поддерживает полный жизненный цикл разработки прикладных МАС, а именно, создание концептуальной модели, разработку архитектуры и спецификацию компонент приложения, генерацию исходного кода, а также развертывание приложения в конкретной компьютерной сети.

Среда включает в себя пять основных компонент (Рис. 1). Компоненты *Язык описания прикладных МАС* и *Визуальная среда проектирования* поддерживают графический стиль спецификации компонент приложения в терминах разработанного авторами графического языка *ASML (Agent-based System Modeling Language)*. Этот язык содержит множество понятий и графических примитивов, необходимых для описания моделей прикладных МАС. В частности, язык содержит такие понятия, как *роль агента*, *цель*, *метасценарии* и *сценарии поведения ролей*, *правила интеллектуального ("проактивного") поведения*, *протоколы взаимодействия* и другие. Все эти понятия–примитивы, а также связи между ними представляются диаграммами, которые далее автоматически преобразуются во внутренние представления на языке типа XML. Компоненты модели МАС–приложения описываются с помощью графических диаграмм шести типов. Они соответственно предназначены для описания *метамодели прикладной МАС*, *схем ролей*, *протоколов*

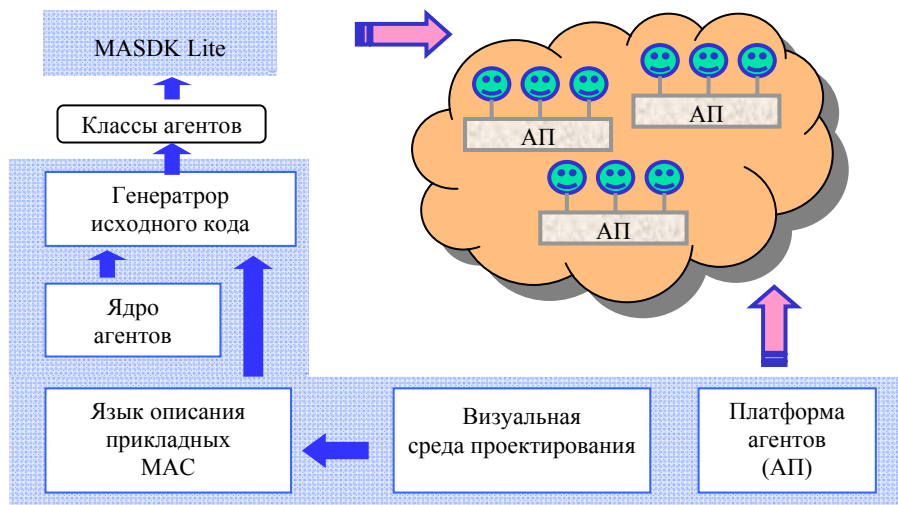


Рис.1. Структура компонент среды MASDK 4.0

взаимодействия ролей, сценариев поведения ролей и поведения агентов, и, наконец, онтологии предметной области.

Формирование и редактирование диаграмм поддерживается компонентой MASDK *Визуальная среда проектирования*. Совместно с компонентой *Язык описания прикладных MAC* она используется для описания приложений на языке *ASML*. Используя построенное описание, компонента MASDK *Генератор исходного кода* автоматически формирует тексты программ на языке C++, описывающие поведение классов агентов. При генерации этих программ используется библиотека классов *Ядро агентов*, описывающая инвариантные (проблемно-независимые) сценарии поведения агентов. Пятая компонента *Платформа агентов* (следующий подраздел) используется для формирования инфраструктуры для функционирования и взаимодействия агентов.

Итоговое модельное описание MAC–приложения на языке *ASML*, которое используется в процессе генерации кода, состоит из моделей множества классов агентов и моделей онтологий. Для каждой онтологии автоматически генерируется библиотека, обеспечивающая агентам объектно-ориентированный доступ к понятиям онтологии. Для каждого класса агентов, описанного в метамодели MAC, генерируются две программные библиотеки. Одна из них, которая описывает поведение агентов (внутреннее и внешнее), генерируется полностью автоматически и не требуют дополнительного редактирования со стороны програм-

миста. Другая библиотека содержит описания отдельных действий агента в рамках его поведения. Эти действия указаны в модельном описании MAC. В нем для каждого действия специфицированы типы входных/выходных переменных и параметров в терминах понятий онтологии, а также даны содержательные пояснения действий в виде текстов. Эта библиотека генерируется в среде частично: она содержит только шаблоны программных процедур, соответствующих отдельным действиям. Исходный программный код последней библиотеки далее разрабатывается программистом в обычной среде программирования, в нашем случае–в среде C++. При этом среда MASDK 4.0 обеспечивает поддержку взаимодействия между разработчиком модели MAC и разработчиком программного кода, обеспечивая автоматическую генерацию шаблонов программных процедур, соответствующих отдельным действиям на основе модельного описания. Такой подход позволяет эффективно управлять разработкой (модификацией) программного кода в процессе итеративного развития модели прикладной MAC. Таким образом, среда MASDK 4.0 обладает принципиально важным свойством, которое состоит в том, что для модельного описания MAC, которое содержит в себе как статические, так и поведенческие аспекты агентов, автоматически выполняется генерация исходного кода, практически готового к компиляции. Это влечет существенное сокращение трудоемкости разработки прикладных MAC.

Еще одной важной функцией, которая реализована в среде MASDK 4.0, является контроль и поддержание целостности (согласованности, непротиворечивости) компонент модельного описания, а следовательно, и программного кода МАС–приложения. Среда обладает встроенными средствами анализа разрабатываемых моделей прикладных МАС для выявления тех элементов в описании моделей, которые должны быть доработаны. Эта же функциональность включает также автоматическую генерацию описания шаблонов сценариев поведения агентов, полностью соответствующих протоколам взаимодействия, на основе описания протоколов. Описание протоколов взаимодействия играет ключевую роль в описании систем со сложным групповым поведением. При этом шаблоны сценариев поведения являются «вторичными» по отношению к протоколам взаимодействия, т.е. шаблоны сценариев представляют поведение агентов, которое следует из описания протоколов взаимодействий. Поэтому функция контроля и поддержания целостности (согласованности, непротиворечивости) компонент модельного описания МАС–приложения является эффективным средством поддержки корректности всего процесса разработки. Дополнительные пояснения к возможностям инструментария MASDK 4.0 и технологии его практического использования могут найдены далее в разделе 3 при описании прототипа МАС автономного управления воздушным движением в районе аэропорта.

Детальное описание инструментальной среды MASDK 4.0 и языка *ASML*, которое фактически является практическим руководством к их использованию, находится в свободном доступе на веб-сайте лаборатории интеллектуальных систем СПИИРАН по адресу: <http://space.iias.spb.su/products/>. Runtime–версия среды также находится в свободном доступе на этом веб-сайте.

2.3. P2P агентская платформа – программная инфраструктура для поддержки взаимодействия гетерогенных агентов в открытых системах

Как уже было отмечено в подразделе 2.1, прикладная МАС состоит из программных агентов и агентской платформы, которая поддерживает взаимодействие агентов. Агентская платформа

является "системообразующей" программной инфраструктурой, которая превращает множество индивидуальных агентов в многоагентную систему. Она должна быть построена таким образом, чтобы, несмотря на разнообразие агентов, несмотря на их гетерогенность, они, тем не менее, были бы в состоянии работать совместно. Поэтому с самого начала развития МАС–технологии была осознана необходимость стандартизации агентской платформы.

FIPA [13] разработала абстрактную архитектуру платформы, которая дает единое представление среды взаимодействия агентов, одинаковое для различных транспортных протоколов, для различных языков коммуникации агентов и описания содержания их сообщений.

Абстрактная архитектура FIPA [14] включает в себя описание следующих компонент:

- модели сервисов, доступных агентам МАС, и методов, позволяющих агентам получать информацию о доступных сервисах;
- средств поддержки совместного использования ("интероперабельности") различных транспортных протоколов;
- средств поддержки различных форм представления ACL¹–стандартного языка коммуникаций агентов;
- средств поддержки различных форм языка описания содержания («контента») сообщений;
- структур описания сервисов в различных директориях.

Абстрактная архитектура описана в терминах языка UML с использованием шаблонов проектирования, приведенных в работе [14]. Ее компоненты представлены на Рис. 2.

В общем случае, для того чтобы конкретная архитектура была совместима с абстрактной архитектурой FIPA, она должна обладать некоторыми обязательными свойствами. В частности, она должна включать в себя три вида механизмов (сервисов), а именно, (1) механизм для регистрации агентов; (2) механизм, с помощью которого любой зарегистрированный агент может получить информацию о других агентах и предоставляемых им сервисах, а также (3) механизм передачи сообщений от одного

¹ ACL — Agent Communication Language (Коммуникативный язык агентов). Специфицирует формат сообщений, протоколы взаимодействия, язык описания содержания сообщений.

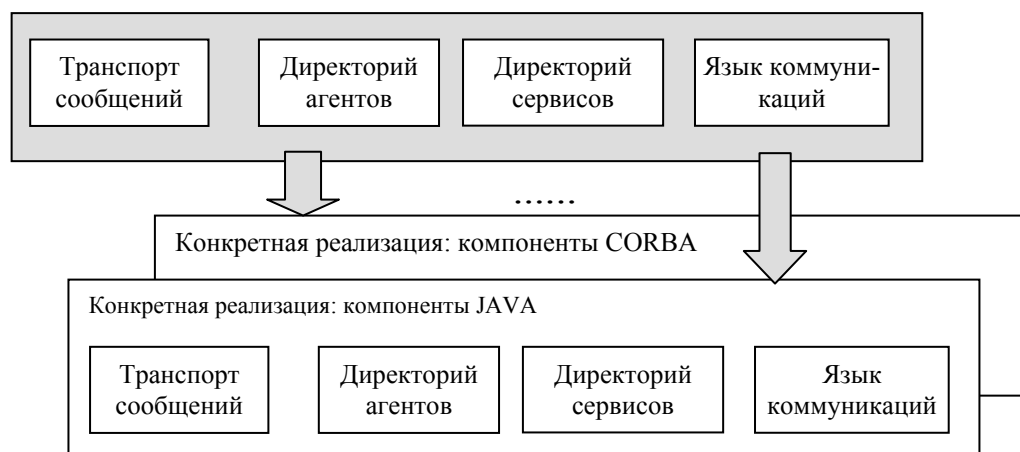


Рис. 2. Абстрактная архитектура и примеры вариантов ее реализации

агента к другому. Эти сервисы должны быть явно описаны в терминах соответствующих элементов абстрактной архитектуры.

Подобная абстрактная архитектура к настоящему времени имеет много различных реализаций. Одна из них разработана в рамках инструментального средства MASDK 4.0, описанного в предыдущем подразделе.

Имеющиеся реализации FIPA архитектуры платформы являются централизованными: в них информация об агентах и сервисах, а также механизмы доступа к сервисам и агентам, хранятся в одном месте. Поэтому чтобы воспользоваться этими сервисами, все агенты системы должны иметь постоянный доступ к компонентам платформы. Но в тех приложениях, которые рассматриваются в данной работе, использовать централизованную платформу невозможно. Причин для этого много, например, динамика состава агентов сети и узлов, в которых они установлены, а также динамика связей между узлами сети. Мобильные сети могут включать в себя десятки миллионов узлов, и тогда проблема хранения и обновления содержания белых и желтых страниц, как и проблема маршрутизации сообщений в динамической P2P коммуникационной среде, могут привести к полной загрузке ресурсов сети.

В течение последних лет предпринимались различные попытки создать программную инфраструктуру для P2P взаимодействия агентов, установленных на мобильных устройствах. Для этих целей было предложено несколько программных инфраструктур поддержки парных взаимодействий. Однако большинство из них, такие как Napster [30, 31], Kazaa [27], Gnutella

[18, 19], ICQ [24] и др., были созданы различными разработчиками и оказались несовместимы друг с другом, несмотря на то, что функциональности систем иногда совпадают [5]. Например, Napster поддерживает обмен музыкальными файлами, Gnutella – обмен файлами произвольного типа, ICQ-система предназначена для обмена мгновенными сообщениями между пользователями и поддерживает обмен файлами произвольного типа. Опыт их использования показал, необходимо искать пути совместимости подобных приложений, и выход состоит в том, чтобы создать стандарты для P2P взаимодействия в агентских системах.

В 2006 г. рабочей группой FIPA был выпущен рабочий документ, описывающий абстрактную архитектуру агентской платформы для P2P сетей [15], которая рассчитана на использование, в частности, в мобильных устройствах (КПК, сотовые телефоны, и т.п.). В 2007 г. авторами данной работы в соответствии с этим документом была разработана первая программная реализация P2P агентской платформы [20]. В настоящее время эта платформа используется совместно с инструментарием MASDK 4.0. Дадим ее краткое описание.

В соответствии с архитектурой FIPA, программные компоненты разработанной платформы и ее окружения построены по трехуровневой схеме (Рис. 3). На нижнем уровне находится P2P провайдер, т.е. узел P2P сети, который предоставляет потребителям (клиентам) коммуникационный сервис. В разработанной реализации P2P провайдер предоставляет своим клиентам механизм отправки сообщений другим клиентам P2P провайдера, расположен-

ным в том же или в другом узлах сети, а также реализует механизм управления списком контактов своих соседей. Список контактов узла содержит имена соседних P2P провайдеров и их адреса, т.е. список узлов P2P сети, с которыми он имеет прямые коммуникационные каналы.

Средний уровень реализует собственно P2P агентскую платформу, которая предоставляет сервисы белых и желтых страниц агентам, зарегистрированным на ней, а также обеспечивает (через P2P провайдера) обмен сообщениями между агентами. В разработанной реализации агентская платформа поддерживает распределенный поиск агентов, присутствующих в P2P сети, и доступных сервисов. При этом поиск реализуется как распределенная процедура, которая использует протокол, известный под названием *gossiping*, разработанный для P2P маршрутизации сообщений.

На *верхнем уровне* располагаются типовые сервисы, доступные прикладным агентам, расположенным на экземпляре платформы.

По существу сеть P2P провайдеров образует «оверлейную» сеть, установленную поверх TCP/IP сети. Аналогично, множество экземпляров P2P агентской платформы образует «оверлейную сеть», установленную поверх P2P сети провайдеров. В этой сети множество экземпляров P2P агентской платформы играет роль распределенной базы метазнаний, в которой каждый экземпляр платформы содержит знания о прикладных агентах, которые на ней установлены, и о предоставляемых ими сервисах. В частности, таблица данных сервиса желтых страниц содержит информацию о сервисах, предоставляемых агентами, которые зарегистрированы на экземпляре платформы, представленную в стандартной форме в терминах общей онтологии сети агентов, а также об именах агентов, которые предоставляют соответствующие сервисы. Экземпляр агентской платформы может содержать информацию о других экземплярах платформы. Таблица данных сервиса белых страниц содержит информацию об адресах агентов, предоставляющих сервисы, указанные в таблице сервисов желтых страниц.

Сервис P2P агентской платформы «*Белые страницы*» реализует следующие функциональности:

- Регистрация агентов на белых страницах;
- Регистрация агентов на белых страницах с присвоением имени;

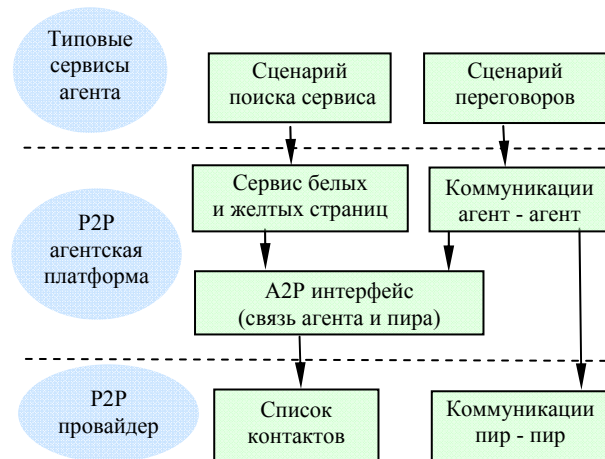


Рис. 3. Функциональная архитектура P2P агентской платформы и P2P провайдера, установленных в узле сети

- Отмена регистрации агентов на белых страницах;
- Определение состояния агента (активен/неактивен);
- Поиск агентов (их адресов).

Аналогично оверлейной сети экземпляров агентской платформы, множество агентов, установленных на экземплярах агентской платформы, образует «оверлейную сеть» третьего уровня, которая установлена поверх сети экземпляров агентской платформы. Топология этой третьей «оверлейной сети» определяется списком контактов каждого агента.

Важно отметить, что в общем случае в начальном состоянии таблицы данных сервисов желтых и белых страниц могут быть вообще пустыми, а их заполнение может выполняться уже в процессе функционирования сети, когда агенты объявляют о своих сервисах, запрашивают тот или иной сервис или получают ответы на свои запросы о сервисах. Таким образом, содержание желтых и белых страниц может постоянно обновляться в процессе работы сети. **Рис. 4** демонстрирует взаимодействие трех перечисленных оверлейных сетей.

В соответствии с современным пониманием, любая функциональность, которую агент может предоставить в распоряжение других агентов, интерпретируется как сервис, предоставляемый агентом другим агентам системы. Разработанный формат описания сервисов базируется на стандарте FIPA, в соответствии с которым сервис характеризуется именем, типом и набором

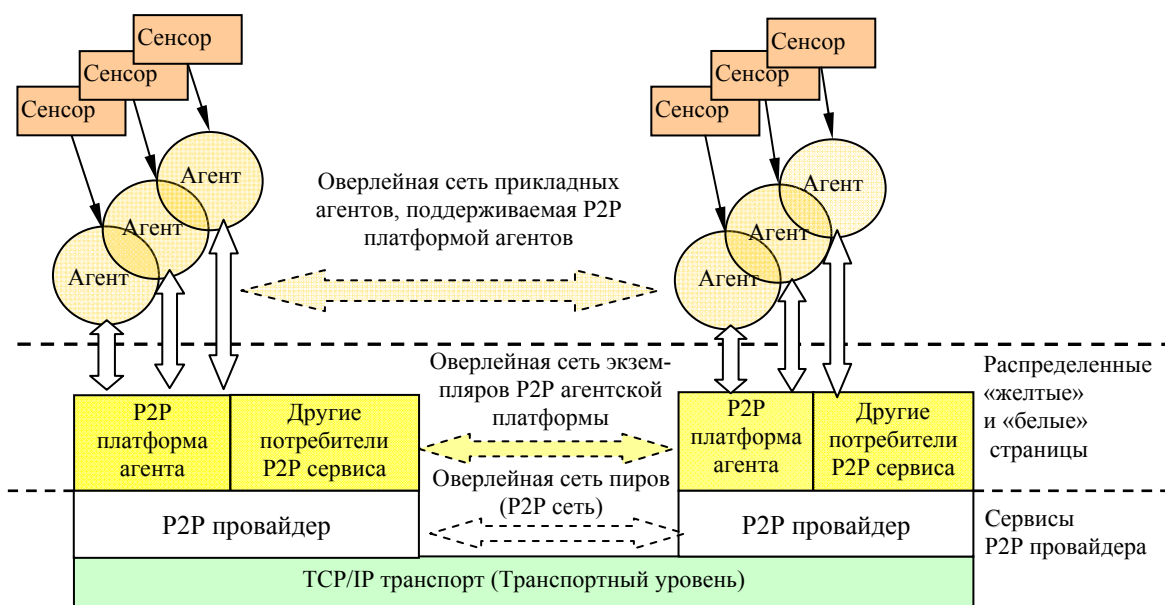


Рис. 4. Оверлейные сети, построенные над TCP/IP, обеспечивающие “прозрачность” взаимодействия прикладных агентов

свойств. Имя сервиса является уникальным именем в агентской среде и служит однозначным его идентификатором. Типом сервиса является некоторое концептуальное описание сервиса, задающее его назначение. Свойства сервиса описывают его на более детальном уровне, различая однотипные сервисы, которые представляются в терминах экземпляров понятий из онтологии агента.

Совместно с платформой могут использоваться также два других разработанных авторами инструментария. Один из них – это генератор сетей заданной топологии, который позволяет автоматически генерировать конфигурационные файлы для экземпляров агентских платформ и P2P провайдеров. Второй инструментарий представляет собой эмулятор виртуальной P2P коммуникационной среды, который позволяет, совместно с разработанной P2P агентской платформой, запускать готовые P2P агентские приложения в интересах тестирования прикладных агентских систем, функционирующих в сетях с переменной топологией [20]. Исчерпывающая информация о P2P платформе, включающая подробную документацию, а также свободно распространяемый исполняемый код, может быть найдена на веб-странице лаборатории интеллектуальных систем СПИИРАН по адресу <http://space.iias.spb.su/products/>.

3. Автономное групповое управление воздушным движением в районе аэропорта

В связи с возрастанием интенсивности трафика воздушного движения и ограниченной пропускной способностью зон воздушного пространства, которое используется самолетами для взлета и посадки с одного или нескольких аэродромов, все более актуальным становится вопрос о совершенствовании принципов управления воздушным движением и повышения его безопасности. Уже сейчас нагрузка на диспетчеров воздушного движения чрезвычайно велика и она имеет тенденцию увеличиваться. Кроме того, время от времени в районе аэропорта возникают нештатные ситуации, обусловленные как техническими причинами, так преднамеренным их созданием, например, ситуации, возникающие при захвате воздушного судна (ВС). Подобные ситуации обладают высокой динамикой и зачастую непредсказуемостью развития. Поэтому естественным направлением совершенствования процессов управления воздушным движением является делегирование как можно большего числа функций диспетчерского управления бортовым средствам управления ВС, переход к парадигме автономного управления воздушным движением.

Что касается движения ВС на маршруте, то здесь уже с середины 1990-х г. развивается концепция "свободного полета", т.е. полета, в которой бортовые средства помогают пилоту прокладывать маршрут без участия наземных служб, например, диспетчеров маршрута. При этом потенциальные конфликты в воздушном пространстве, которые могут возникать на маршруте, должны обнаруживаться и предупреждаться бортовыми средствами воздушных судов на основании парных переговоров между ними. В настоящее время обсуждается вопрос о переносе концепции автономного управления также и на управление в районе аэропорта, предлагается делегировать бортовым средствам ВС функции формирования безопасных траекторий взлета и посадки.

Предлагаемая многоагентная модель группового управления воздушным движением в районе аэропорта имеет целью распределенное автономное управление воздушным трафиком. Вначале рассматривается, каким образом в настоящее время обеспечивается безопасность воздушного движения, а затем описывается разработанная многоагентная модель и программный прототип группового автономного управления, которые делегируют основные функции управления бортовым средствам ВС.

Система обеспечения безопасности воздушного движения в районе аэропорта включает в себя несколько мер. Рассмотрим только те из них, которые связаны с обеспечением безопасности за счет управления воздушным

трафиком. В этой части существенными являются следующие меры:

1. Структуризация воздушного пространства в районе аэропорта, которая включает в себя фиксацию зон возможного входа ВС в воздушное пространство аэропорта, а также зон выхода из него; задание секторов и коридоров воздушного пространства, внутри которых, главным образом, допускается движение ВС; задание допустимых последовательностей прохождения этих коридоров; задание кругов ожидания, внутри которых ВС могут находиться в процессе ожидания разрешения (от диспетчера) на продолжение движения. Еще один путь структуризации воздушного пространства – это эшелонирование воздушного пространства по высоте. Пример структуризации воздушного пространства в районе г. Нью-Йорк, где находятся три крупных аэропорта, дан на **Рис. 5**. В нем в зоне прибытия (она выделена серым цветом) имеется 4 сектора, которые состоят из последовательных коридоров, формирующих древовидные структуры, из кругов ожидания (каждый из них имеет уникальное имя), зоны подхода (она выделена белым цветом и структурирована, в основном, аналогично зоне прибытия), а также из взлетно-посадочных полос.

2. Использованием норм безопасных расстояний между самолетами по трем координатам, которые зависят от относительных параметров движения каждой пары ВС. Эти нормы определяются официальными документами.

В настоящее время диспетчеры воздушного движения полностью ответственны за разреше-

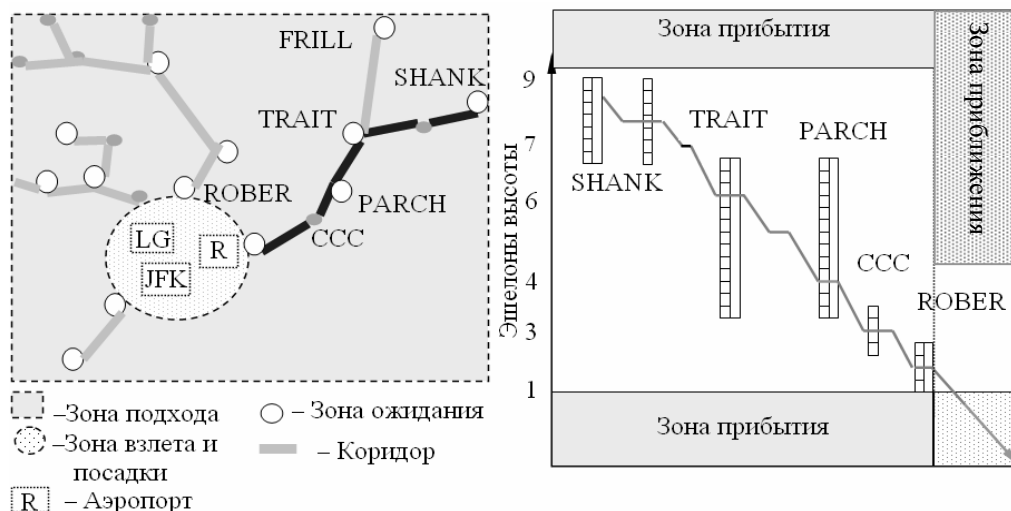


Рис. 5. Схема воздушного пространства аэропортов г. Нью-Йорка

ние на взлет и на вход в зону аэропорта, за выбор траектории движения ВС, и соблюдение норм безопасности. Управление выполняется ими последовательной выдачей команд, основные из которых перечислены ниже.

Команда ВС, приближающемуся к зоне аэропорта

А. Выдача разрешения на вход в зону.

Команды ВС, находящемуся в зоне прибытия

Б. Разрешение на переход в следующий коридор.

В. Выдача директив на переход в другой высотный эшелон (как правило, более низкий).

Г. Координация маневров ВС при обгоне.

Д. Разрешение на вход ВС в зону подхода для посадки.

Е. Команда на изменение скорости ВС.

Ж. Команда на выполнение маневра векторения.

Команды, выдаваемые ВС, находящемуся в зоне подхода

З. Разрешение на взлет ВС, находящемуся на взлетно-посадочной полосе.

Многоагентная модель группового управления воздушным движением в зоне аэропорта, рассматриваемая далее, позволяет реализовать схему автономного управления, в которой все функции управления в *зоне прибытия* (и только в ней) делегируются бортовым средствам ВС. В этой модели бортовые средства воздушных судов решают задачи планирования траекторий посадки, прогнозирование потенциальных конфликтов с другими ВС (возможного нарушения дистанций безопасности), переговоры с целью разрешения конфликта на основе парных взаимодействий с бортовыми средствами ВС, с которым возможен потенциальный конфликт, разрешение конфликта и выбор, при необходимости, новой траектории движения. Принципиальным моментом в этой стратегии автономного управления воздушным движением является использование *распределенной политики обеспечения безопасности* (правил поведения ВС, при соблюдении которых нормативы дистанций безопасности не нарушаются). При этом предполагается, что выбор решений, которые должны быть сформированы для выдачи команд А, Б, В, Г, Д и Е, перечисленных выше, выполняется либо бортовыми средствами ВС автономно, либо на основании парных (Р2Р) переговоров с другими ВС, если автономно выбранное решение может привести к конфликту.

В данной работе детали распределенной политики безопасности, протокол Р2Р переговоров для разрешения конфликтов и информационный обмен, который этим переговорам сопутствует, не рассматривается в деталях, поскольку здесь этот пример имеет целью только проиллюстрировать модель группового управления с использованием многоагентной технологии и инструментальных средств ее поддержки, описанных в предыдущем разделе. Отметим лишь, что политика безопасности представляет собой множество правил, с помощью которых потенциально конфликтующие ВС вычисляют приоритеты участников конфликта на использование "конфликтного" участка траектории. Аргументами правил могут быть задержки ВС по отношению к расписанию, остаток топлива, класс ВС, близость к зоне подхода, занимаемый высотный эшелон и др. Далее перечисляются классы агентов, участвующих в процессе группового управления воздушным движением, их роли в различных сценариях управления и разрешения конфликтов и модель взаимодействия ролей, которая является центральным моментом группового управления.

Архитектура МАС автономного управления воздушным движением формально описывается в терминах, принятых в методологии Gaia [41]. В соответствии с ней архитектура МАС описывается с помощью множества диаграмм, специфицирующих метамодель системы, которая, главным образом, направлена на представление модели взаимодействия агентов, на представление группового ("внешнего") поведения агентов. Внутреннее поведение агентов моделируется с помощью соответствующего *множества сценариев* ("liveness expressions"), которые описывают это поведение агентов в различных протоколах. Метамодель прикладной МАС описывается с помощью ролей, исполняемых агентами в том или ином протоколе, протоколов взаимодействия, а также распределением ролей на множестве классов агентов.

В метамодели МАС автономного управления воздушным движением (Рис. 6) выделены следующие роли:

- *Ассистент пилота (АП-агент)*. Агент этого класса assiste пилоту в процессе решения задач обеспечения безопасности при управлении движением в воздушном пространстве при взлете и посадке.

– *Ассистент диспетчера (АД-агент)*. Агент этого класса assisteрует диспетчеру при решении задач управления в пределах зоны (сектора) подхода, а также assisteрует ему при формировании команд на вход ВС в зону прибытия аэропорта.

– *Агент, моделирующий поведение захваченного ВС (ЗВС-агент)*. Полагается, что движение этого ВС может быть представлено отрезками прямых линий в трехмерном пространстве. Его движение не подчиняется принятым правилам и командам диспетчера.

Заметим, что число экземпляров агентов каждого класса определяется числом соответствующих сущностей, присутствующих в контуре управления воздушным трафиком, и при этом, очевидно, для каждого класса агентов число его экземпляров постоянно изменяется. Соответственно, система должна автоматически генерировать новые их экземпляры и удалять те, которые заканчивают свое существование.

– *Сервер симуляции*; Эта компонента архитектуры не является агентом. Она предназначена для симуляции и отображения воздушной обстановки. Кроме того, она ответственна за генерацию событий типа появление нового ВС в воздушном пространстве и уход ВС из него. Эта компонента обеспечивает также пользовательский интерфейс, который необходим для задания различных ситуаций, в том числе ситуаций захвата воздушного судна, управления средствами отображения, визуализации возникающих конфликтов и др.

Формальная спецификация внутреннего поведения агентов выполняется описанием сценариев поведения роли в том или ином протоколе взаимодействия. Например, внутреннее поведение агента класса АП-агент в различных протоколах описывается 14 сценариями, а именно: *Инициализация*, *Цикл симуляции*, *Группирование ВС*, *План прибытия*, и др., которые частично представлены на Рис. 6. Агент класса АД-агент в различных протоколах, где он участвует, выполняет два сценария: *Запрос разрешения* и *Выдача разрешения*. Поведение агента класса ЗВС-агент также описывается двумя сценариями: *Инициализация* и *Прогноз траектории*. Здесь не рассматриваются детально сценарии внутреннего поведения, представленные на Рис. 6, поскольку цель данного примера проде-

монстрировать спецификацию группового поведения ВС в процессе движения в воздушном пространстве аэропорта.

Разработанная модель и программный прототип многоагентной системы автономного управления воздушным движением формализует семь сценариев использования системы, а именно (Рис. 6):

(U1) Инициализация экземпляров агента класса АП-агент и экземпляра агента класса ЗВС-агент;

(U2) Выполнение цикла симуляции;

(U3) Группировка ВС (экземпляров агента класса АП-агент), предназначенная для декомпозиции множества ВС на группы самолетов, которые потенциально могут конфликтовать. Это делается с целью снижения нагрузки на вычислительные мощности ВС. Информационный обмен выполняется только "внутри" каждой группы ВС;

(U4) Автономное планирование собственного движения ВС в зоне прибытия, выполняемое экземплярами агента класса АП-агент;

(U5) Перепланирование собственного движения в зоне прибытия, выполняемое экземплярами агента класса АП-агент для избежания конфликта как с "нормальным" ВС, так и с захваченным;

(U6) Управление выполнением взлета;

(U7) Управление прибывающим ВС в зоне подхода (от момента получения разрешения на вход в зону подхода до посадки).

Конкретный сценарий выполнения соответствующего действия – это и есть то, что здесь называется внутренним поведением агента. Следует различать два типа сценариев внутреннего поведения агента. В сценариях одного типа такой сценарий поведения инициируется входным сообщением от другого агента, а в сценариях другого типа инициатором сценария поведения экземпляр агента является сам агент. В последнем случае агент инициирует события, запускающие сценарии внутреннего поведения других агентов. Источники таких событий могут быть различными. Например, событие может генерироваться при достижении ВС некоторого пространственного состояния (например, ВС достигает конечной точки коридора), или переход агента в некоторое внутреннее состояние, где соответствующее событие генерируется самим агентом.

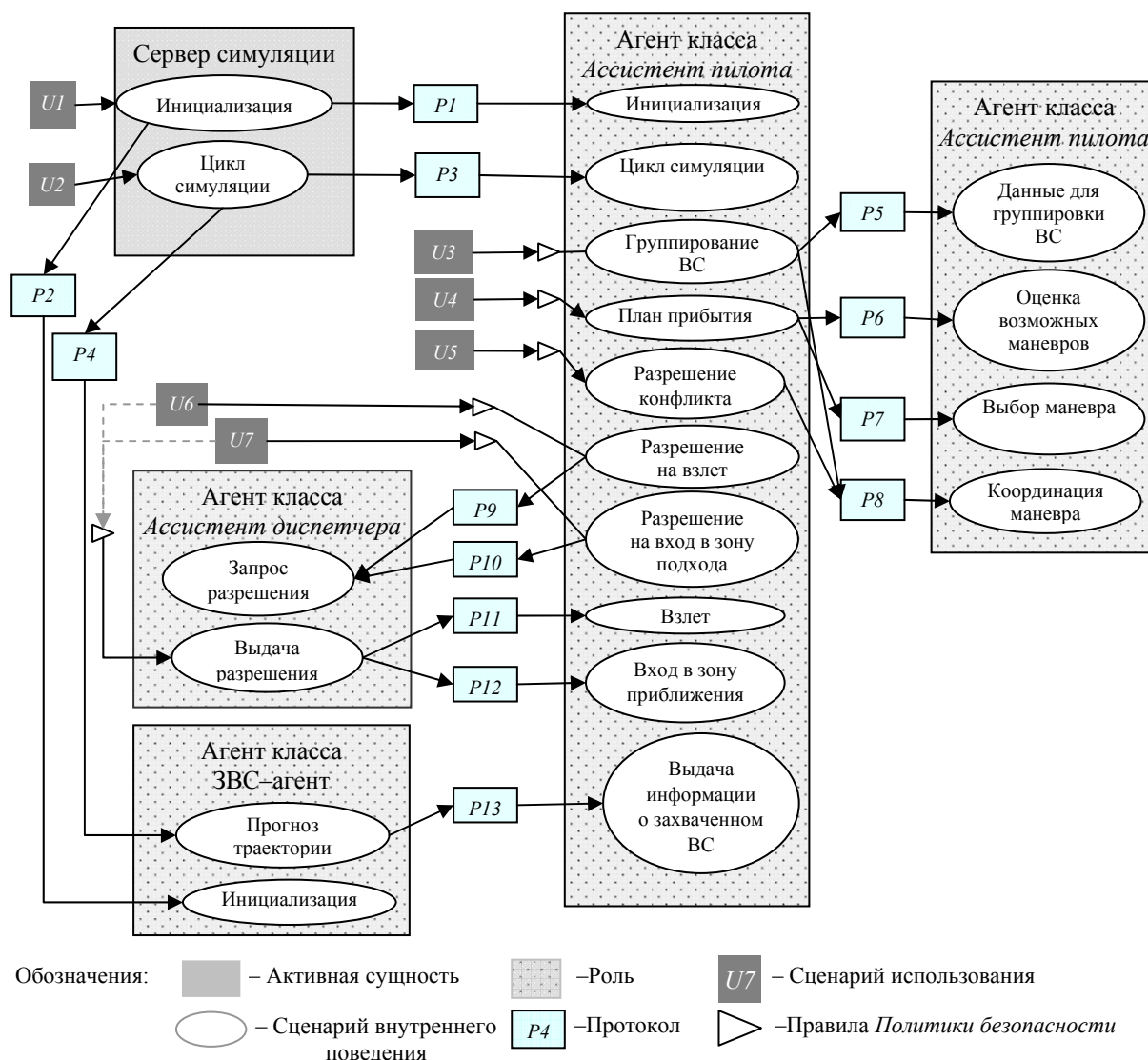


Рис. 6. Мета-модель системы автономного управления воздушным движением

Пользовательский интерфейс в разработанном программном прототипе дает возможность вводить модель топологии воздушного пространства аэропорта, расписание прибывающих ВС, вводить характеристики всех самолетов и данные о моменте появления и начальном положении захваченного ВС, а также о его дальнейшей траектории и др. Главное окно пользовательского интерфейса представлено на Рис. 7.

Разработанная модель автономного управления воздушным движением в воздушном пространстве аэропорта, а также соответствующий программный прототип могут найти различное практическое использование, причем не только непосредственно при управлении воздушным движением. Среди других приме-

нений важным является, например, использование многоагентной модели группового управления воздушным движением в рамках интеллектуального диспетчерского тренажера, который поддерживает функции симуляции, отображения и оптимизации воздушного движения в районе аэропорта, а также дает оценку качества работы диспетчера. Еще одно применение – анализ оптимальности принятой или вновь разрабатываемой структуры воздушного пространства в районе существующего или проектируемого аэропорта, выявление ситуаций, которые могут оказаться сложными для диспетчеров, с последующей поддержкой процессов принятия решений при возникновении аналогичных ситуаций, проведение тренировок

по управлению воздушным движением в нестандартных ситуациях. Возможны и другие варианты применения разработанной модели группового управления.

4. Групповое обнаружение наземных объектов распределенной системой наблюдения

В отличие от задачи группового управления, в данном примере рассматривается задача группового принятия решений о классе наземных объектов, в которой отдельные решатели (агенты) взаимодействуют в P2P стиле для кооперативного выполнения миссии. Предполагается, что система наблюдения наземной обстановки является распределенной. Она состоит из множества сенсоров, роль которых играют инфракрасные видеокамеры, установленные на беспилотных летательных аппаратах (БПЛА). В данном примере использованы исходные данные, которые сформированы искусственно на основании серий инфракрасных изображений реальных наземных сцен.

Предполагается, что на борту каждого БПЛА имеется один или несколько агентов, каждый из которых обучен распознаванию наземного объекта одного конкретного типа при его наблюдении под определенным ракурсом, а именно, или спереди, или сзади, или справа, или слева. В поле зрения видеокамеры каждого летательного аппарата попадает определенная область земной поверхности, причем эти области для различных БПЛА, которые летят близко друг от друга, могут быть перекрывающимися. Таким образом, различные видеокамеры могут получать информацию об одних и тех же объектах, которые, как правило, в таких случаях будут наблюдаться ими под различными углами. Следовательно, одни и те же наземные объекты могут обнаруживаться агентами, и эти агенты могут находиться как на борту одного и того же БПЛА, так и на борту различных БПЛА.

В рассматриваемой здесь модели и в соответствующем программном прототипе МАС P2P системы обнаружения наземных объектов каждый отдельный БПЛА интерпретируется как узел сети («пир»), который может связываться с соседними пирами (БПЛА) в P2P стиле на основе беспроводной связи и обмениваться с

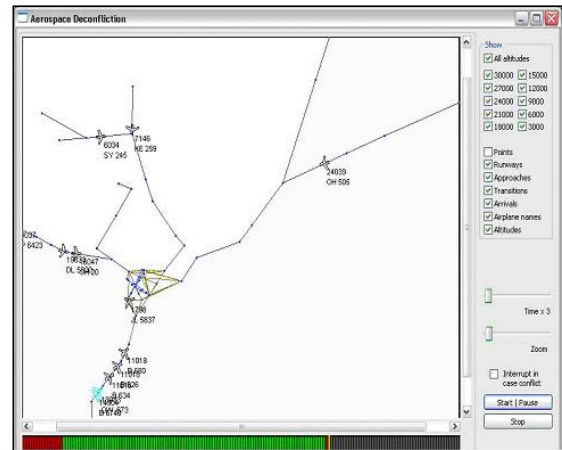


Рис. 7. Главное окно симулятора

ними информацией, а множество всех БПЛА образует P2P сеть. Предполагается, что на каждом БПЛА установлен экземпляр агентской платформы. Все экземпляры агентской платформы (они имеют сервисы белых и желтых страниц) вместе с сервисами пиров, обеспечивающими P2P коммуникации в сети БПЛА, реализуют «прозрачное» взаимодействие агентов различных БПЛА в соответствии со схемой, представленной на Рис. 4.

Функциональность каждого агента по обнаружению наземных объектов определенного класса, наблюдаемых под заданным ракурсом, рассматривается как сервис, который агент может использовать сам или предоставлять его по запросу любого другого агента P2P сети. В программном прототипе P2P сеть состоит из восьми пиров (БПЛА). На Рис. 8 представлена топология сети, которая отражает отношение «соседства» пиров (БПЛА) в некоторый момент времени, а также указывает состав агентов, установленных на экземпляре агентской платформы каждого пира. Всего в примере распределенной системы обнаружения используются 22 агента. Из них 21 агент предназначен для обнаружения наземных объектов (они являются агентами-детекторами объектов), и еще один агент реализует интерфейс с пользователем (Рис. 8).

Сценарий взаимодействия агентов в процессе обработки информации следующий (предполагается, что желтые и белые страницы всех экземпляров P2P агентской платформы в сети содержат необходимую информацию). Когда агент обнаруживает объект «своего» типа (т.е. объект, обнаружению которого он обучен), он

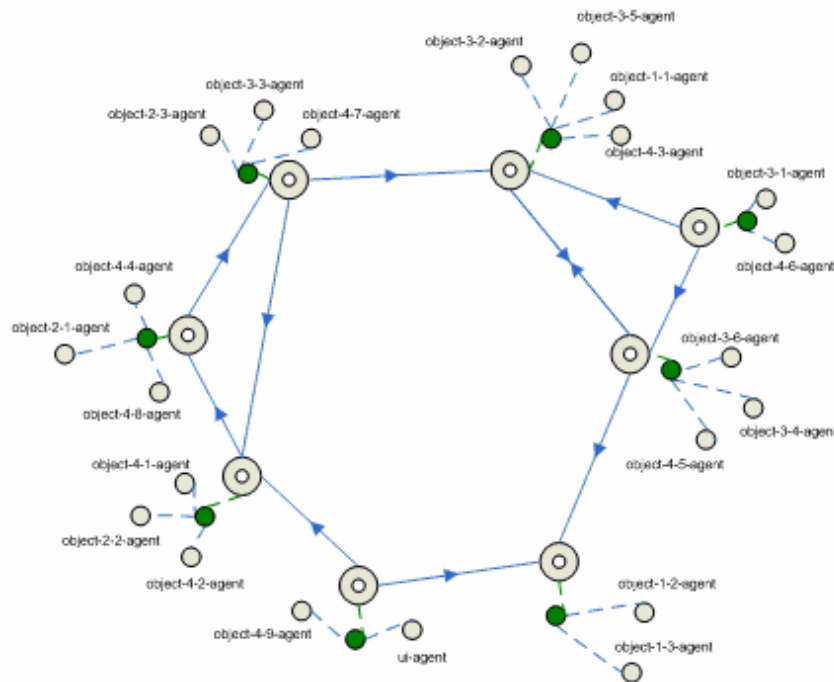


Рис. 8. Открытая P2P сеть агентов, предназначенная для группового обнаружения наземных объектов, наблюдаемых распределенной системой наблюдения

посылает другим агентам координаты соответствующей прямоугольной области. Агенты-получатели этой информации могут быть обучены обнаружению объектов такого же типа или близкого типов. Агент-инициатор запроса запрашивает у последних информацию о том, обнаружили ли они в указанной им или близкой области объект «своего» класса, наблюдая тот же район, возможно, под другим ракурсом. После получения ответов агент объединяет решения, полученные в ответ, используя метод голосования на основе простого большинства, и в случае положительного решения отправляет информацию агенту, реализующему интерфейс пользователя, который, в свою очередь, отображает полученные результаты на экране монитора. На Рис. 9 дан пример, в котором все четыре БПЛА обнаружили один и тот же объект, но каждый из них указывает свой «прямоугольник», привязанный к наблюдаемой поверхности, в котором он предположительно обнаружил объект того же класса.

Результаты численного моделирования процессов распределенного обнаружения наземных объектов описанной выше системой показали, что даже при столь простом алгоритме объединения решений, каким является метод

голосования на основе простого большинства, последний дает удовлетворительные результаты по точности работы системы распределенного обнаружения наземных объектов в целом.

Отметим, что обнаружение объектов агентами выполняется с использованием метода, известного под названием SIFT (Scale Invariant Feature Transform [28, 29]), который в настоящее время достаточно популярен. Программный код, задающий структуру связей в P2P сети, распределение экземпляров агентских платформ и распределение агентов по экземплярам платформ, был сгенерирован автоматически с помощью специального инструментального средства, которое было разработано одним из авторов данной работы [21].

Заключение

В работе проанализированы особенности современных приложений в области группового управления и предложена многоагентная модель и технология разработки таких приложений. Предложенная модель интегрирует в себе идеи распределенного принятия решений, архитектуры, ориентированной на сервис, и вычислений на основе парных взаимодействий.

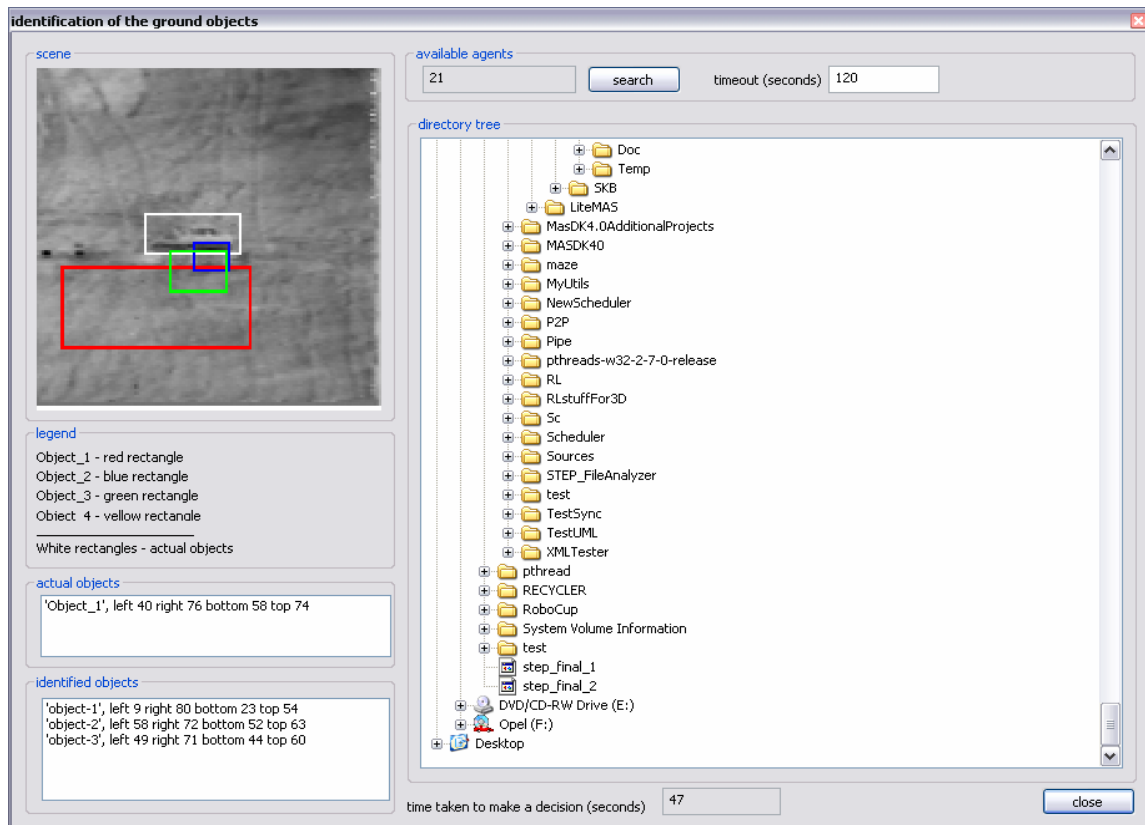


Рис. 9. Интерфейс пользователя, который отображает наблюдаемую область земной поверхности и объекты, обнаруженные различными агентами системы наблюдения

Разработанная технология создания много-агентных систем базируется на хорошо обоснованной методологии Gaia [41] и поддерживается двумя инструментальными средствами. Первое из них, инструментальная среда для разработки многоагентных приложений MASDK 4.0 (Multi-agent System Development Kit, версия 4.0) поддерживает полный жизненный цикл создания приложений, а именно: концептуальное проектирование, формальную спецификацию приложения, генерацию кода, а также развертывание приложения в конкретной компьютерной сети. Второе инструментальное средство, которое используется совместно с первым, – это P2P агентская платформа. Эта платформа обеспечивает взаимодействие распределенных программных сущностей приложения (агентов) в полностью распределенном варианте. Последнее достигается использованием распределенных сервисов белых и желтых страниц, реализующих распределенный поиск агентов и сервисов. Платформа поддерживает совместное функционирование агентов, установленных на различных операционных плат-

формах. В настоящее время имеется реализация платформы, которая обеспечивает взаимодействие агентов, работающих под управлением операционных систем Windows и Linux. В дальнейшем планируется ее расширение для мобильных устройств, где смогут использоваться программы, работающие под управлением операционной системы Symbian. Детальная информация об инструментальном средстве MASDK 4.0, об агентской P2P платформе, а также документация и исполняемый код программ могут быть найдены на веб-странице лаборатории интеллектуальных систем СПИИРАН по адресу <http://space.ias.spb.su/products/>.

Оба инструментальных средства совместно формируют прототип новой информационной технологии для поддержки процессов разработки широкого класса современных приложений в области многоагентных систем, в том числе решающих задачи группового управления. Эта технология проверена экспериментально. В частности, в работе приведены примеры ее использования при разработке программных прото-

типов систем автономного управления воздушным движением в районе аэропорта и группового обнаружения наземных объектов распределенной аппаратурой наблюдения.

Разработанная технология и инструментальные средства использовались также при разработке приложений в других предметных областях, в частности, при разработке прототипа системы распределенного обнаружения атак на компьютерные сети (в ней агенты, осуществляющие мониторинг различных информационных потоков компьютерной сети, кооперируются для совместного обнаружения атак), системы планирования и управления проектами в области микроэлектроники, где осуществляется управление группой разработчиков, и в ряде других приложений.

Дальнейшие исследования будут направлены на расширение возможностей разработанной технологии создания многоагентных приложений, а также на разработку конкретных приложений. Одной из главных целей при этом является доведение разработанной технологии и имеющихся инструментальных средств до индустриального уровня.

Литература

- Брайнов С. Разработка и исследование моделей устойчивых коопераций в мультиагентных системах. Кандидатская диссертация. ВЦ РАН, 1998.
- Варшавский В.И. Коллективное поведение автоматов - М. Наука, 1973
- Варшавский В.И., Поспелов Д.А. Оркестр играет без дирижера, М: Наука, 1984.
- Городецкий В. И. , Карсаев О. В. , Самойлов В. В. , Серебряков С.В. Инструментальные средства для открытых сетей агентов Известия РАН. Теория и Системы Управления. — Москва: Наука, 2008. — Вып. 3, С. 106-124.
- Ли Г. Почему JXTA? 2007, www.javable.com/columns/P2P/solutions/03/.
- Месарович М., Мако Д.и Такахара И. Теория иерархических систем. "Мир", 1973.
- Цетлин М.Л. Исследования по теории автоматов и моделированию биологических систем. М: Наука, 1969.
- Chavez A., Maes P., and Kasbah B. An Agent Marketplace for Buying and Selling Goods. In Proceedings of the First International Conference "The Practical Application of Intelligent Agents and Multi-Agent Technology", London, UK, 1996, C. 75-90.
- Cohen P., Levesque H. Communicative Actions for Artificial Agents, In Proceedings of ICMAS'95, 1995, C. 65-72.
- Cohen P., Levesque H. Teamwork. Nous, vol. 35, 1991.
- Durfee E. Scaling Up Agent Coordination Strategies. Computer, July 2001, pp. 39-46, 2001. Статья доступна также по адресу [ftp://ftp.eecs.umich.edu/people/durfee/computer01.pdf](http://ftp.eecs.umich.edu/people/durfee/computer01.pdf).
- Durfee E. Coordination of Distributed Problem Solvers. Kluwer Academic Publishers, 1988.
- FIPA: The Foundation of Intelligent Physical Agents. <http://www.fipa.org/>
- FIPA abstract architecture. <http://www.fipa.org/specs/fipa00001/>.
- FIPA P2P NA WG6. Functional Architecture Specification Draft 0.12 <http://www.fipa.org/subgroups/P2PNA-WG-docs/P2PNA-Spec-Draft0.12.doc>.
- Fisher K, Muller J.P., Heimig I., Scheer A-W. Intelligent Agents in Virtual Enterprises. In Proceedings of the First International Conference "The Practical Application of Intelligent Agents and Multi-Agent Technology", London, UK, 1996, C. 205-224.
- Gasser L. DAI approaches to coordination. In N.Avouris and L.Gasser (Eds.) Distributed Artificial Intelligence: Theory and Practice., Kluwer Academic Publishers, 1992, C. 31-51.
- Gnutella at wiki pages, 2007. <http://ru.wikipedia.org/wiki/Gnutella>.
- Gnutella web page, 2007. <http://www.gnutella.com/>.
- V.Gorodetskiy, O.Karsaev, V.Samoylov, S.Serebryakov, S.Balandin, S.Leppanen, M.Turunen. Virtual P2P Environment for Testing and Evaluation of Mobile P2P Agents Networks. The Second International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM 2008), Valencia, Spain, pp. 422-429.
- Gorodetskiy, V. O. Karsaev, V. Samoilov, S.Serebryakov. P2P Agent Platform: Implementation and Testing. The AAMAS Sixth International Workshop on Agents and Peer-to-Peer Computing (AP2PC 2007). Honolulu, Hawaii, USA, 2007, C. 21-32.
- Gorodetskiy, V. O. Karsaev, V. Samoilov, S.Serebryakov. Agent-based Service-Oriented Intelligent P2P Networks for Distributed Classification. International Conference "Hybrid Information Technologies" (ICHIT-2006). Cheju Island, Korea, November 9-11, 2006, C. 224-233.
- Grosz B.and Kraus S. Collaborative Plans for Complex Group Actions. Artificial Intelligence, 86, 1996, C. 269-358.
- ICQ web page, 2007. <http://www.icq.com/>.
- Jennings N. Commitments and Conventions: The Foundation of Coordination in Multi-Agent Systems, The Knowledge Engineering Review, 8 (3), 1993, C. 223-250.
- Jennings N., Paratin P., Jonson M. Using Intelligent Agents to Manage Business Processes. In Proc. of the First Intern. Conference "The Practical Application of Intelligent Agents and Multi-Agent Technology", London, UK, 1996, C. 345- 376
- Kazaa web page. 2007. <http://www.kazaa.com/us/index.htm>.
- Lowe D. Object Recognition from Local Scale-Invariant Features. Proceedings of the International Conference on Computer Vision, Corfu (September 1999)
- Lowe D. Distinctive Image Features from Scale-Invariant Key points, 2004, <http://www.cs.ubc.ca/~lowe/papers/ijcv04-abs.html> (Also published in the International Journal of Computer Vision, 2004).
- Napster at wiki pages, 2007. <http://ru.wikipedia.org/wiki/Napster>.

31. Napster web page, 2007. <http://free.napster.com/>
32. Nash J. The Bargaining Problem, *Econometrica*, vol. 18, 1950, C. 155-162.
33. Sandholm T. Negotiation among self-interested computationally limited agents. Ph.D.Thesis, University of Massachusetts Amherst, 1996.
34. Searle J. Speech acts: An Essay in the philosophy of language. Cambridge University Press, 1970.
35. Shoham Y. and Tennenholtz M. On the synthesis of useful social laws for artificial agent societies. In *Proceedings of 10th National Conference of Artificial Intelligence*, San Jose, USA, 1992, C. 276-287
36. Smith G. Contract Net Protocol: High-level Communication and Control in a Distributed Problem Solver. *IEEE Transactions on Computers*, C-29 (12), 1980, C. 1104-1113.
37. Smith R., and Davis R. Framework for Cooperation in Distributed Problem Solving, *IEEE Transactions on Systems, Man and Cybernetics*, SMC-11, 1981, C. 61-70.
38. Tambe M. Towards Flexible Teamwork. *Journal of Artificial Intelligence Research*, 7, 1997, C. 83-124.
39. Zlotkin G., and Rosenshtein J. Negotiation and Goal Relaxation. In Y.Demazeau and J.-P.Muller (Eds.), "Decentralized A.I.", *Proceedings of the Second European Workshop on Modelling of Autonomous Agents in a Multi - Agent World*, North Holland, Amsterdam, 1991, C. 273 - 286.
40. Zlotkin G., and Rosenshtein J. Mechanisms for Automated Negotiation in State Oriented Domain. *Journal of Artificial Intelligence Research*, 5, 1996, C. 163-238.
41. Wooldridge M., Jennings N., and Kinny D. 2000. The Gaia Methodology for Agent-Oriented Analysis and Design. *International Journal of Autonomous Agents and Multi Agent Systems*, 3(3), C. 285-312.

Городецкий Владимир Иванович. Главный научный сотрудник лаборатории интеллектуальных систем Санкт-Петербургского института информатики и автоматизации РАН. Окончил Ленинградскую военно-воздушную инженерную академию им. А.Ф.Можайского в 1960 году и математико-механический факультет Ленинградского государственного университета в 1970 году. Доктор технических наук, профессор, заслуженный деятель науки Российской Федерации. Опубликовал более 250 работ, 9 монографий и учебных пособий. Область научных интересов: теория и технология многоагентных систем, инструментальные средства для проектирования, программирования и развертывания многоагентных систем, многоагентные приложения в области защиты компьютерных сетей, объединения данных из гетерогенных источников, оценки ситуаций, управления воздушным движением, методы и средства интеллектуальной обработки данных и извлечения знаний, методы распределенного обнаружения знаний в данных, peer-to-peer системы, распределенное принятие решений. Адрес электронной почты: gor@iiias.spb.su.

Карсаев Олег Владиславович. Заведующий лабораторией Санкт-Петербургского института информатики и автоматизации РАН. Окончил Военно-инженерный институт им. А.Ф.Можайского в 1981 г., кандидат технических наук. Опубликовал более 50 работ. Область научных интересов: многоагентные системы, инструментальные средства для проектирования, программирования и развертывания многоагентных систем, машинное обучения, планирование и составление расписаний. Адрес электронной почты ok@iiias.spb.su.

Самойлов Владимир Владимирович. Научный сотрудник Санкт-Петербургского института информатики и автоматизации РАН. Окончил Высшее военно-морское инженерное училище им. Ф.Э. Дзержинского, 1993 г. Опубликовал более 30 работ. Область научных интересов: многоагентные системы, системы объединения данных из разных источников, методы распределенного обучения, базы данных, программирование. Адрес электронной почты samovl@iiias.spb.su.

Серебряков Сергей Валерьевич. Старший научный сотрудник Санкт-Петербургского института информатики и автоматизации РАН. Окончил Санкт-Петербургский государственный политехнический университет в 2004, кандидат технических наук. Победитель чемпионата мира по футболу роботов RoboCup в симуляционной лиге двухмерного футбола (команда STEP, Португалия, 2004). Опубликовал 16 работ. Область научных интересов: искусственный интеллект, многоагентные системы, машинное обучение, объединение информации, сервис ориентированные архитектуры, peer-to-peer системы, программирование. Адрес электронной почты: sergey_s@iiias.spb.su.