Агентская платформа для повсеместных вычислений [1]

В.И.Городецкий, О.В. Карсаев, В.В.Самойлов, С.В. Серебряков

Аннотация. Анализируется проблема создания программной инфраструктуры для поддержки сетевых интеллектуальных приложений, функционирующих в среде повсеместных вычислений и коммуникаций, и предлагается программная платформа, поддерживающая совместную работу сети гетерогенных устройств и программ с динамическим составом узлов сети и ее топологии. Особенностью предлагаемого решения является ориентация на технологию автономных агентов и технологию парных взаимодействий.

Введение

Динамика развития современных интеллектуальных информационно–управляющих систем характеризуется пересмотром ряда базовых положений, определяющих как архитектуры построения систем, так и концепции их практического использования. Хотя современные информационно–управляющие системы достигли уже достаточно высокого уровня зрелости и, в основном, успешно справляются с решением достаточно сложных задач управления, однако их очевидный недостаток состоит в том, что они "живут" и используются независимо друг от друга. Например, в современном автомобиле существуют десятки компьютерных и информационных систем, решающих задачи диагностики агрегатов, контроля и оптимизации расхода топлива, навигации с использованием системы GPS, задачи предотвращения столкновений и информирования о трафике по пути следования и т.д. Водителю доступны мобильные сервисы, например, доступ в сервисную службу, когда ему необходима техническая помощь. Водитель чувствовал бы себя гораздо комфортнее, если бы все эти системы были интегрированы в единую среду. Тогда при возникновении технической неисправности система диагностики автоматически по каналу Bluetooth передавала бы соответствующую информацию в мобильный телефон водителя. Мобильный телефон далее самостоятельно транслировал бы эту информацию в сервисный центр вместе с координатами автомобиля (они могли бы быть получены очень точно, например, с помощью навигационной системы GPS и данных о расстояниях до ближайших WiFi точек, координаты которых могут храниться в базе данных сервисного центра). Далее, сервисный центр мог бы найти ближайшую сервисную станцию, передать ей задание на обслуживание. Водителю неисправной машины оставалось бы только ждать обслуживания, находясь в ближайшем ресторане, информация о котором могла быть получена им через мобильный телефон, как и всякий другой мобильный web–сервис. Однако в настоящее время перечисленные системы автомобиля независимы. Поэтому, чтобы реализовать описанный сцена-

[1] Исследования по проблеме, рассмотренной в данной работе, поддерживаются проектом программы фундаментальных исследований Отделения нанотехнологий и информационных технологий РАН "Фундаментальные основы информационных технологий и систем", задача 2.4, и проектом программы Президиума РАН № 14 "Фундаментальные проблемы информатики и информационных технологий", Направление 1: "Интеллектуальные технологии и математическое моделирование", проект № 236.

рий, водитель должен переключаться между системами вручную, выполнять много дополнительных действий, если он обладает необходимыми знаниями. Таких примеров можно привести много.

Уже сейчас в своей повседневной жизни человек использует все больше и больше компьютеров. Эти компьютеры достаточно разнообразны, они обладают различными ресурсами памяти и процессоров, работают в различных операционных средах, используют различные физические каналы связи, соединяются с различными источниками данных по различным протоколам, предоставляют пользователю различные сервисы. Но они работают изолированно, и пользователь может работать с ними только поочередно, что его не устраивает. Потребность интеграции разнообразных компьютерных, информационных и коммуникационных ресурсов в единую среду в настоящее время является одним из главных факторов, определяющих современные тенденции развития в этой области. Эти тенденции характеризуются переходом от "интеллектуальных устройств" к "интеллектуальному пространству", от использования множества сетей различного назначения к их интеграции и кооперации. В идеале это означает переход к такой организации взаимодействия окружающих человека компьютерных и информационных сред, когда их работа "становится невидимой для человека, неотличимой от окружающей его среды" [22].

Научный и технологический базис, который потенциально может обеспечить такой переход, в настоящее время разрабатывается в рамках исследований по трем новым направлениями:

1) *повсеместные вычисления* (*ubiquitous computing*[2] [22]),

2) *повсеместные коммуникации* (*ubiquitous communication*) и

3) *многомодальный пользовательского интерфейс*.

По существу эти разработки совместно формируют новую информационную технологию, новую мощную парадигму построения информационных систем, круг потенциальных приложений которой весьма обширен.

Целью данной работы является краткое введение в существо названных выше направлений исследований (описание, потенциальные приложения, новые проблемы, имеющиеся разработки, оценка перспективности), а также представление программной инфраструктуры, разработанной авторами, которая реализует прототип среды повсеместных вычислений и коммуникаций для открытых сетей гетерогенных автономных агентов. Особенностью предложенного решения является ориентация на приложения, в которых отсутствует иерархия, когда распределенные прикладные агенты взаимодействуют на равноправной основе. Такой стиль взаимодействия агентов (и вообще–программ) в англо–язычной литературе называют термином Peer–to–Peer (сокращенно–P2P), что переводится как "соединение равноправных узлов". Заметим, что в таких сетях любой агент может выступать как в роли сервера, так и в роли клиента.

Далее работа организована таким образом. В разделе 1 дается краткое описание существа технологий повсеместных вычислений и повсеместных коммуникаций, а также их потенциальных приложений. В разделе 2 приводится краткий обзор известных разработок в области моделей и программных инфраструктур, поддерживающих совместную работу гетерогенных устройств и установленных на них программных компонент в среде повсеместных вычислений и коммуникаций, а также описываются некоторые из разрабатываемых приложений. Раздел 3 посвящен описанию разработанной и программно реализованной программной инфраструктуры, предназначенной для работы с агентскими приложениями P2P архитектуры. Эта инфраструктура строится как полностью распределенная FIPA[3]–совместимая P2P агентская платформа, обеспечивающая открытость сети, т.е. способность сети адаптироваться к динамике состава ее узлов и топологии их связей, а также поддерживаю-

[2] От латинского слова"*ubiqui*", означающего "*везде*".

[3] FIPA (Foundation for Intelligent Physical Agents) – это объединение специалистов в области многоагентных систем, разрабатывающее предложения по стандартизации языков, моделей, и интерфейсов в области многоагентных систем. С 2004 года FIPA функционирует как один из комитетов IEEE по стандартизации (http://www.fipa.org).

щая "прозрачность" взаимодействия гетерогенных агентов сети. Раздел 4 показывает, какими алгоритмическими средствами могут быть обеспечены адаптационные возможности гетерогенных агентских сетей, взаимодействующих через P2P агентскую платформу. Эти алгоритмические средства, использующие механизмы P2P обучения и принятия решений, демонстрируются на примере распределенной P2P системы обнаружения вторжений, в которой сеть распределенных прикладных агентов в процессе работы изменяет свою конфигурацию таким образом, чтобы минимизировать вероятности ложных тревог, генерируемых различными агентами обнаружения вторжений. В заключительной части статьи приводятся основные выводы и формулируются направления будущих исследований.

1. Повсеместные вычисления и коммуникации

1.1. Существо технологий

Поясним кратко существо технологий, о которых шла речь во введении, опираясь на материалы работ [1, 22].

Повсеместные вычисления — это парадигма организации вычислений, которая предполагает повсеместное использование компьютеров в форме, незаметной для пользователя [22]. Этот термин в настоящее время применяется, в основном, к встроенным системам, т.е. к системам, которые, как правило, имеют в своем составе большое количество распределенных компонент. Эти компоненты обычно выполняют простые функции в реальном времени и взаимодействуют для принятия решений и управления в масштабе системы в целом. Обычно эти задачи решаются с помощью процессоров, которые имеют весьма ограниченные ресурсы (мощность процессора, объем памяти, запас энергии). Примером такой системы является множество распределенных вычислителей современного автомобиля, общее число которых может достигать сотен и больше.

Повсеместные коммуникации — это парадигма построения коммуникационной среды, в которой одновременно используются различные каналы связи между устройствами–узлами сети, например, проводные и беспроводные. В этой среде могут совместно использоваться различные коммуникационные протоколы (TCP/IP, WiFi, Zigbee, и др.), по которым связываются различные устройства сети, например, настольные и карманные компьютеры, сотовые телефоны, устройства бытовой техники, сенсоры и т.п. Эта среда организуется таким образом, что взаимодействующие сущности (агенты) "не замечают" этих особенностей.

Интеллектуальный пользовательский интерфейс — это интерфейс, который использует *многомодальный вход* от пользователя (некоторую размытую комбинацию интерфейсов различных модальностей в форме письменного текста, разговорного языка, жестов, выражения лица и положения глаз и т.п., потенциально допускающую различное толкование) и генерирует *многомодальный выход* (координированное, понятное пользователю, представление реакции компьютера в виде текста, речи, графики и жестов на обычном дисплее или с использованием анимации человекоподобного агента) [1].

Среди новых приложений, для которых совместное использование названных технологий просто необходимо, следует, прежде всего, отметить класс приложений, объединяемых термином "*интеллектуальное пространство*" (*smart space*). Как правило, эти приложения включают в себя такие компоненты, как:

– *гетерогенные сенсоры*, объединенные в единую сеть; они воспринимают свойства среды и формируют ее целостный "образ";

– *гетерогенная коммуникационная среда;*

– *средства обработки данных*, генерируемых сенсорами, которые формируют пространственный и временной контексты интеллектуального пространства;

– *средства анализа* поведения сущностей (людей, устройств, и т.д.) в контексте интеллектуального пространства и *принятия решений*.

Две последние компоненты обычно трудно разделить. Они совместно решают множество типовых задач, как, например, *оценка состояния* различных объектов интеллектуального пространства, *оценка и прогнозирование текущей ситуации* в среде в контексте текущих целей, *принятие решений* на конкретные

действия и выдача *управляющих команд* на их выполнение.

Второй класс близких по сути приложений - *интеллект окружения* (*ambient intelligence*, *AmI*) - решает во многом сходные задачи. Однако приложения этого класса в большей мере ориентируются на удовлетворение интересов и запросов человека, в то время как приложения первого типа фокусируются больше на работе с устройствами и обслуживании интересов задач, которые могут включать в свой контур человека, а могут его и не включать. В [1] *интеллект окружения* характеризуется как "всесторонний, но ненавязчивый интеллект окружающей человека среды, который поддерживает постоянное активное взаимодействие с ним". *AmI*–системы должны обладать такими свойствами, как незаметность для пользователя, понимание и учет контекста среды, предвидение, использование человеко-машинного многомодального интерфейса и адаптивность (способность к самоконфигурации, самодиагностике и самовосстановлению, самоорганизации и самозащите). Одной из наиболее важных черт *AmI*–систем является учет персональных предпочтений пользователя, причем в такой форме, когда человек не должен заботиться о том, чтобы свои предпочтения как-то вводить в систему. Окружающие объекты должны, анализируя поведение человека, сами обучаться, когда и какой информацией, сервисами и решениями его обеспечивать.

1.2. Потенциальные приложения

Приложения, объединяемые названными выше терминами - "интеллектуальное пространство" и "интеллект окружения" - охватывают очень широкий круг приложений, некоторые из которых относятся к классу критических. Приведем отдельные примеры приложений, которые в настоящее время активно разрабатываются и уже имеют, по крайней мере, экспериментальную реализацию.

Управление в чрезвычайных ситуациях. Решение таких задач всегда основано на огромных потоках данных реального времени, которые генерируются сенсорами, различными службами и системами, мобильными телефонами, и т.д.

Мониторинг окружающей среды в интересах задач природопользования, экологии (анализ загрязнения воздуха, воды, и т.д.), обнаружения пожаров и др.

Здравоохранение. Круг потенциальных приложений здесь очень широк. В первую очередь к ним относят мониторинг состояния *тяжело больных* пациентов, удаленное слежение за состоянием *хронических* больных, *сервис для пожилых* людей (в названных случаях используются сенсоры, встроенные в одежду и в окружающую среду). Такие системы снижают нагрузку на медперсонал, чье внимание требуется только в ситуациях, которые представляют опасность для здоровья или жизни пациента, а эти ситуации обнаруживаются по сигналам "тревоги".

Управление автомобилем и дорожным движением. В этот процесс, кроме водителя, вовлекаются различные системы, которые используют информацию от сенсоров, отслеживающих состояние двигателя, состояние и работу системы торможения и др., от видеокамер, отслеживающих внешнюю обстановку, и т.д.

Управление интеллектуальным домом и его устройствами, имеющее целью обеспечить наилучшие условия для проживания человека, обеспечить его различными сервисами, экономить энергоресурсы, обеспечить безопасность и т.п.

Наблюдение и охрана периметра. Приложения данного типа решают задачи обеспечения безопасности критических инфраструктур (например, аэропортов), опасных производств, военных объектов, и т.п.

Распределенная логистика. В настоящее время эта задача является очень важной. Ее цель — обеспечить бесконтактное глобальное сопровождение и отслеживание грузов с использованием RFID технологий, мобильных систем, встроенных систем и т.п.

Комплексное управление городом. Концепция глобальной системы управления городом, которая уже получила общепризнанное название *ubiquitous city* (*u-city*), была предложена компаниями Южной Кореи. В настоящее время эта концепция уже находится в стадии тестирования в 22 городах страны.

Распределенная оценка энергопотребления в больших электрических сетях.

Список потенциальных приложений можно было бы значительно продолжить.

1.3. Основные проблемы

Технология повсеместных вычислений и коммуникаций в настоящее время находится в стадии активного исследования и развития. Рассмотрим основные проблемы, которые для этой технологии являются ключевыми и требуют наибольшего внимания исследователей и разработчиков.

Одна из центральных проблем рассматриваемой области - *создание программной инфраструктуры* (промежуточного программного обеспечения) для поддержки взаимодействия гетерогенных устройств и программ, установленных в узлах компьютерной сети. Такая инфраструктура является основой этой технологии. Не случайно исследования и разработки в области программных инфраструктур для повсеместных вычислений и коммуникаций финансируются приоритетно в проектах 6-й и 7-ой рамочных программ Европейской комиссии по разделу информационных и коммуникационных технологий.

Вторая трудная проблема–это *адаптация системы* к динамически изменяющемуся *контексту* приложения и поддерживающей его инфраструктуры. На *сетевом уровне* этот контекст характеризуется появлением новых узлов сети и выходом из нее ранее присутствовавших, а также изменением топологии сети, например, за счет перемещения мобильных устройств, их выключения, за счет выхода из строя сенсоров и т.п. В этих условиях коммуникационная инфраструктура должна адаптироваться к текущему состоянию сети, имея целью оптимизацию трафика. В P2P системах, где специально выделенная инфраструктура, поддерживающая маршрутизацию, вообще не задается, эта проблема особенно остра. Динамика контекста на *уровне сервисов* обусловлена изменением во времени доступности сервисов. Здесь адаптация необходима для ускорения процедур распределенного поиска сервисов. Наконец, важной проблемой является адаптация структуры взаимодействия компонент (агентов) приложения, что необходимо для повышения качества и скорости решения целевых задач. Проблема адаптации систем повсеместных вычислений и коммуникаций иногда называется также проблемой *само–конфигурирования системы* на сетевом уровне, на уровне сервисов и на уровне приложений.

Следующая проблема – это *обеспечение компьютерной безопасности* и *защита информации частного характера*. Если первая часть этой проблемы более или менее исследована, то вторая ее часть очень специфична и исследована намного слабее. Работая с ресурсами сети и с сервисами, человек поневоле оставляет в ней свои следы, например, в логах систем. Информация частного характера остается в учреждениях здравоохранения, в Интернет–магазинах и т.д. В настоящее время для обеспечения безопасности частной информации пока, совместно с криптографированием, используются только два самых простых подхода для скрытия содержания информации, а также ее принадлежности. Первый — это использование псевдонимов вместо имен владельцев, а второй — использование управления точностью представления информации со стороны пользователя.

Важной проблемой является *организации вычислений* в беспроводных сетях *с учетом ограниченности энергетических ресурсов* сенсоров. Решения, которые предлагаются в этой части, достаточно разнообразны, но все они основаны на использовании специализированных протоколов для управления временем работы сенсоров, на выборе маршрутов передачи информации и т.п. [6].

Следует также отметить, что традиционная проблема — *обработка гетерогенных данных*, слияние данных и информации для принятия решений и оценивания ситуаций, не только остается актуальной в рассматриваемой технологии, но и становится еще более острой. Причины этого кроются в огромных размерностях данных, в их потоковом характере, в жестких временных границах для принятия решений в условиях неопределенности, связанной с поведением человека, и т.п. Интеграция в систему многомодального пользовательского интерфейса дополнительно усложняет рассматриваемую проблему.

Имеются и другие проблемы, которые хотя и достаточно традиционны для распределенных

систем, но применительно к системам, работающим в среде повсеместных вычислении и коммуникаций, имеют свою специфику.

2. Взаимодействие гетерогенных устройств и программ. Состояние исследований

2.1. Требования к функциональным возможностям

Как было отмечено в предыдущем разделе, создание программной инфраструктуры для поддержки взаимодействия гетерогенных распределенных компонент систем, работающих в среде повсеместных вычислений и коммуникаций, является ключевой проблемой в этой области. Требования к ее функциональным возможностям представляют собой предмет активного обсуждения. Сформулируем основные из них, которые инвариантны к специфике приложений. При этом будем опираться на материалы проектов 6-й рамочной программы Европейской комиссии, в частности, на материалы проектов [2, 4, 5, 7, 12, 16, 18-21], которые в настоящее время находятся в процессе выполнения (они запланированы на 2006–2009 гг.). В этих проектах разработка функциональных требований к инфраструктуре поддержки взаимодействия распределенных гетерогенных компонент составляет важнейший предмет исследований.

1. *Поддержка совместного функционирования множества гетерогенных устройств* (компьютеров, карманных компьютеров, сотовых телефонов, встроенных систем и др.) и установленных на них программ, работающих под управлением *различных операционных систем.* Это связано с необходимостью поддержки большого количества протоколов и адаптации к появлению новых протоколов. При этом инфраструктура должна иметь средства рассуждения о своем состоянии (например, чтобы принимать решение о своих действиях в случае, если батареи устройства приближаются к порогу отключения по заряду).

2. *Понимание многоуровневого контекста* приложения для адаптации к нему, если это необходимо, с использованием некоторых механизмов рассуждений о своем состоянии. Например, система должна принимать во внимание состояние батарей различных устройств (мобильных устройств, сенсоров), должна учитывать текущее использование полосы пропускания, знать о присутствующих в сети в текущий момент узлах, агентах и доступных сервисах.

3. Наличие *эффективных механизмов поиска сервисов и адресации сообщений при переменном составе узлов и изменяющейся топологии сети*. В большинстве выполненных работ используется подход, в котором сервисы регистрируются на центральном сервере. Но в крупномасштабных сетях такой подход не только проблематичен, но и практически невозможен. В качестве альтернативы рассматривается вариант "публикации" сервисов путем широковещательной рассылки. P2P агентская платформа предлагает использовать для этого *распределенные* белые и желтые страницы и механизмы распределенного поиска [9].

4. *Обеспечение безопасности* информации, которая передается через инфраструктуру. Это является одним из ключевых требований к инфраструктуре. Источниками возможного нарушения безопасности могут быть ненадежные коммуникационные каналы, успешные попытки неавторизованного доступа через инфраструктуру к информации и сервисам, а также атаки того или иного типа. Важнейшей функцией инфраструктуры должна быть идентификация источника информации, аутентификация автора запроса, поддержание целостности данных и криптографирование информации, которая передается через инфраструктуру. Эта проблема достаточно глубокая и трудная [17]. В данной работе она не является предметом исследований

Существуют и другие требования, однако перечисленные здесь являются наиболее существенными.

2.2. Примеры инфраструктур

Рассмотрим примеры разработок в области инфраструктур для поддержки приложений, работающих в среде повсеместных вычислений и коммуникаций. Описываемые приложения и инфраструктуры являются объектами исследований и разработок, проводимых в рамках наиболее интересных проектов 6-й и 7-й рамочных

Рис. 1. Архитектура платформы, рассматриваемая в проекте Angel

программ Европейской комиссии по данной тематике.

Проект "Angel" [2] (2006–2009) имеет в качестве основной цели разработку платформы для поддержки совместной работы систем, использующих проводные и беспроводные коммуникации для обеспечения доступа к сенсорной информации в интересах мониторинга состояния здоровья и условий жизни человека под общим лозунгом "улучшение качества жизни". Предлагается инфраструктура, которая должна поддерживать совместную работу IP и беспроводных сетей, работающих по стандарту Zigbee [23] при передаче сенсорной информации. Предполагается, что сенсоры могут быть встроены в одежду пациента ("*wearable body sensor network*") с целью мониторинга различных параметров его состояния (температуры, пульса, дыхания и т.д.). Другие сенсоры используются для мониторинга параметров среды в помещении, где находится пациент, и для наблюдения за ним самим (микрофоны, камеры, измерители температуры и влажности в помещении, локаторы местонахождения пациента и др.). Предполагается, что эти сенсоры могут работать в двух режимах. Один их них соответствует мониторингу параметров с периодической передачей параметров и некоторых результатов их обработки внешнему потребителю (медперсоналу). Второй режим предназначен для обнаружения состояний пациента и окружающей его среды, которые могут представлять опасность для его здоровья или для медицинского заведения в целом. Эти сенсоры вырабатывают сигналы тревоги, если по результатам пороговой обработки данных обнаруживается, что с пациентом произошло что–то серьезное, требующее вмешательства медперсонала. Аналогично, если при мониторинге условий в помещении обнаруживаются явления, требующие немедленного вмешательства, например, при обнаружении пожара, то сигналы тревоги немедленно передаются медперсоналу и в пожарную часть, например.

Требования к функциональным возможностям платформы, которые принимаются во внимание разработчиками, стандартны: она должна обеспечить исполнение программ, написанных на разных языках, работающих под управлением различных операционных систем, установленных на устройствах, среди которых могут быть мобильные устройства, использующие различные коммуникационные протоколы и физические каналы связи. Абстрактная архитектура, предложенная в проекте Angel, представлена на Рис. 1. Она не требует особых пояснений, за исключением уровня "Расширение компоненты платформы на хосте". По существу эта компонента, построенная на основе Java RMI или CORBA, расширяет возможности компоненты платформы на хосте, поддерживая автоматизацию задачи сетевого программирования, в частности, программирование сетевого взаимодействия, управления памятью, назначения приоритетов, синхронизации и т.п. Это позволяет разработчику программировать приложение как единое целое, не заботясь о местонахождении объектов при их вызове.

Детальное описание программной инфраструктуры Angel можно найти в [3]. В целом эта платформа представляется достаточно громоздкой и сложной в использовании, хотя это может быть оправдано тем, что она рассчитана на достаточно широкий круг приложений. К сожалению, авторы не рассматривают вопрос обеспечения безопасности информации в инфраструктуре.

В проекте AWARE [4] (2006–2009) разрабатывается программная инфраструктура для

управления быстрым развертыванием беспроводной коммуникационной среды, которую далее предполагается использовать в автономном режиме для обеспечения связи с сенсорной сетью в районах, где коммуникационная среда отсутствует или она разрушена. Примером является труднодоступная местность, где произошло стихийное бедствие, или территория противника, где необходимо развернуть разведывательную сенсорную сеть, и т.п.

Предполагается, что сеть сенсоров и коммуникационная инфраструктура разворачиваются с помощью беспилотных малогабаритных воздушных объектов, например, вертолетов, в функции которых входит также перевоз оборудования, необходимого для работы коммуникационной инфраструктуры. Эти объекты используются также как мобильные узлы сети связи, которые при дальнейшей работе взаимодействуют между собой и с наземными узлами для совместного решения прикладных задач. Программная часть коммуникационной инфраструктуры должна решать задачи поддержания надежной связи путем самодиагностики с целью определения возможного нарушения связности сети и ее самовосстановления. Кроме того, она должна решать функции самоконфигурации узлов сети с целью минимизации числа передаваемых пакетов, чтобы таким образом экономить энергию. В этой сети одни узлы (наземные компьютеры) используют широкополосные каналы связи, в то время как другие (узлы на вертолетах) пользуются узкополосной связью. Обе сети связываются между собой через специальные шлюзы, и одна из основных задач инфраструктуры – обеспечить эту связь в форме, которая максимально "прозрачна" для приложения. Отметим, что эта платформа не является платформой общего назначения. Более подробная информация может быть найдена в [10].

Проект "Hydra" [12] рассматривает программную инфраструктуру для встроенной системы гетерогенных физических устройств распределенной архитектуры. Задача рассматривается в достаточно общей постановке. Разрабатываемую инфраструктуру предполагается использовать для поддержки работы интеллектуальных приложений, главными компонентами которых являются гетерогенные устройства. В основу инфраструктуры положена архитектура, ориентированная на сервис, которая должна обеспечить "прозрачность" на уровне коммуникации устройств. Проблема обеспечения безопасности решается в ней с привлечением понятия распределенного доверия. В качестве потенциальных приложений она рассматривает приложения типа интеллектуального пространства и интеллекта окружения. Вопросы поддержки взаимодействия гетерогенных устройств, каналов связи и программ являются в этой инфраструктуре основными вопросами исследований и разработок.

Проект MORE [16] имеет целью создание инфраструктуры для обеспечения коммуникаций групп пользователей, которые работают в среде гетерогенных устройств, гетерогенной беспроводной связи и используют программы, работающие под управлением различных операционных систем. Примером такого приложения является мониторинг хронически больного пациента, когда доктор вызывает пациента для очной встречи только по мере необходимости. Другой пример приложения – это многоцелевой мониторинг лесного массива, когда в процесс управления и оповещения нужно вовлекать различные группы людей, например, владельцев земли, представителей администрации, рабочие и спасательные бригады. В целом, в проекте для инфраструктуры принята стандартная архитектура, ориентированная на сервис. Специфика разрабатываемой платформы состоит в том, что она обслуживает встроенные устройства, которые имеют ограниченные ресурсы процессоров и памяти. Поэтому все сервисы в ней упрощены настолько, насколько это можно, чтобы минимальных ресурсов встроенных устройств хватало для выполнения целевых задач.

Проект SMEPP [20] выполняет задачу разработки программной инфраструктуры для систем, использующих принцип парного взаимодействия (P2P взаимодействия) с акцентом на *обеспечение безопасности*. Как известно, особенностью P2P систем является отсутствие предварительно созданной коммуникационной инфраструктуры, отсутствие деления компонент на клиентов и серверов. Это предъявляет

весьма специфические требования к промежуточному программному обеспечению. Кроме того, такие системы в большей мере уязвимы по отношению к атакам изнутри сети. Цель проекта – разработка инфраструктуры для встроенных P2P систем, которая должна быть защищенной, типовой и легко настраиваемой под специфическое приложение, а также адаптируемой к различным устройствам и приложениям (от критических приложений до развлекательных систем).

Предполагается, что программная инфраструктура (платформа), разрабатываемая в рамках проекта SMEPP, будет способна обнаруживать новые узлы и добавлять их в сеть, адаптироваться к появлению в сети новых сервисов и к исчезновению сервисов из сети. Платформа должна поддерживать свойство самоорганизация в условиях изменяющегося состава узлов коммуникационной сети и топологии их связей, работать с гетерогенными ресурсами, которыми обладают узлы сети (по мощности процессоров, памяти и запасу энергии), обладать способностью к самоконфигурации для поддержания полносвязности сети.

Разработка программных инфраструктур, предназначенных для поддержки повсеместных вычислений и повсеместных коммуникаций в рамках различных приложений является темой ряда других проектов программы Европейской комиссии. Информация о них представлена в Интернет (например, [5, 18-21]).

Рассмотренные инфраструктуры во многом сравнимы с позиций архитектуры и функциональных требований, которым они, в конечном счете, должны удовлетворять. Отличия инфраструктур касаются, главным образом, классов приложений, на которые они ориентируются, классов устройств, которые используются приложениями (например, в случае, если приложение должно работать с автономными сенсорными сетями, когда необходимо удовлетворять ряду жестких ограничений на ресурсы устройств, на дальность связи беспроводных каналов и др.). Из общего ряда выпадает инфраструктура, разрабатываемая в рамках проекта SMEPP, которая ориентируется на P2P приложения и на соответствующую коммуникационную среду. В этом отношении эта разработка представляется более передовой, если принимать во внимание те темпы, с которыми системы P2P вычислений завоевывают все новые позиции.

2.3. Тенденции

Однако ни одна из рассмотренных платформ не предполагает работу с приложениями, которые построены в соответствии с концепцией автономных агентов и многоагентных систем. В то же время, агентская технология сегодня предлагает наиболее перспективные модели и технологические средства для реализации крупномасштабных приложений, состоящих из распределенных автономных сущностей, которые взаимодействуют для решения собственных и общих задач приложения. Для многоагентных систем разработана *стандартная* абстрактная архитектура, основанная на использовании сервисов белых и желтых страниц [8], которая поддерживает *семантический* поиск сервисов, а также агентов, которые владеют нужным сервисом, что очень важно в открытых сетях.

Быстрыми темпами завоевывают популярность многоагентные системы, которые используют P2P технологии как для взаимодействия агентов, так и для поиска адресатов сообщений. Такая концепция использования многоагентных систем, интегрированная с концепцией архитектуры, ориентированной на сервис, открывает новые возможности для реализации систем, работающих в среде повсеместных вычислений и коммуникаций. Но для практической реализации такого подхода необходимо создание стандартной программной инфраструктуры (платформы) со всеми теми свойствами, которые были сформулированы в подразделе 2.1. Последующий материал данной работы посвящен описанию разработанной P2P агентской платформы и демонстрации ее возможностей.

3. Распределенная P2P агентская платформа

Разработка функциональной архитектуры P2P агентской платформы (P2P АП) и связанного с ее работой промежуточного программного обеспечения в конце 2005 г. была начата рабочей группой FIPA, Nomadic Agent Working Group. В своём первом документе [9] эта рабочая группа отмети-

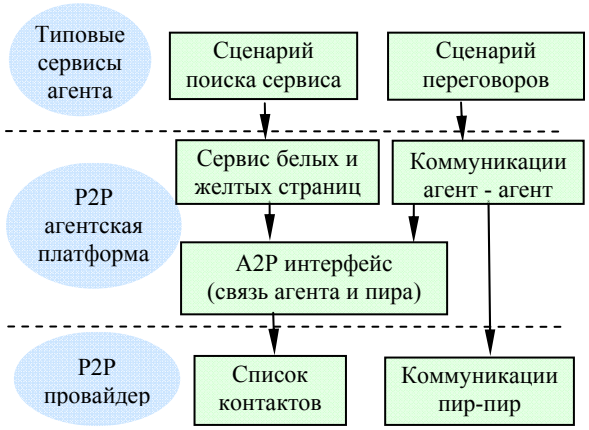

Рис. 2. Функциональная архитектура P2P агентской платформы и P2P провайдера ("пира")

ла, что в настоящее время "в мире не существует базовой P2P агентской платформы и что FIPA намерена сделать шаг в этом направлении, предоставив спецификации и ускорив разработку ее программной реализации". Усилия рабочей группы были сконцентрированы на разработке стандартов спецификации интерфейсов для P2P агентов, установленных на мобильных устройствах. Первый документ, описывающий абстрактную архитектуру P2PАП и ее интерфейсов приложением и с коммуникационной инфраструктурой, был выпущен в начале 2006 г. Первая программная реализация этой абстрактной архитектуры была выполнена авторами данной работы в 2007 г. [11]. Ее дальнейшее развитие, ориентированное на использование в среде повсеместных вычислений и повсеместных коммуникаций, описывается далее в данном разделе.

3.1. Функциональная архитектура

Основные обязательные функции P2P агентской платформы – представление всеми агентами P2P сети возможности семантического поиска нужных им сервисов, а также поиска агентов, которые нужными сервисами обладают. Другая функция – обеспечение агентов "прозрачной" связью с владельцами сервисов.

С точки зрения архитектуры, P2P АП представляет собой распределенное множество идентичных экземпляров платформы, каждый из которых устанавливается в узле P2P сети. Для выполнения из основной функции платформы – поддержки поиска сервисов, а также агентов, которые искомыми сервисами обладают, – каждый экземпляр платформы, установленный в некотором узле P2P сети, должен содержать метазнания об агентах, установленных на "своем" узле, и о тех сервисах, которые эти агенты могут предоставить. Значит, по своей сути множество экземпляров платформы, установленных в узлах P2P сети, являются *распределенной базой метазнаний* об агентах сети и их сервисах.

Архитектура разработанной реализации экземпляра P2P АП представлена на Рис. 2. Она в основных чертах следует архитектуре, предложенной FIPA [9], в частности, реализует все ее обязательные функции.

Программное обеспечение, содержащее экземпляр платформы, на котором установлены прикладные агенты, а также компоненту узла, обеспечивающую коммуникационный сервис, имеет трехуровневую архитектуру (Рис.2). *На нижнем ее уровне* находится программная компонента узла P2P сети, называемая *P2P провайдером*. В текущей реализации P2P провайдер предоставляет своим клиентам P2P каналы связи с другими узлами сети и реализует механизм управления списком своих соседей в P2P сети. Этот список называется "*списком контактов*" P2P провайдера. Напомним, что список контактов содержит имена и адреса соседних узлов, т.е. список узлов P2P сети, с которыми данный узел имеет *прямые* коммуникационные каналы.

Средний уровень соответствует экземпляру P2P агентской платформы, которая предоставляет агентам, зарегистрированным на платформе, *сервисы белых и желтых страниц*. Каждый их этих сервисов содержит некоторую таблицу и механизм поиска. В таблице сервиса желтых страниц содержится информация об именах и типах сервисов, которые доступны через данный узел P2P сети, т.е. о сервисах, которые могут быть предоставлены агентами, установленными поверх экземпляра платформы. В таблице сервиса белых страниц содержатся записи, которые отражают имена агентов, установленных в узле сети, и список сервисов, которые эти агенты могут предоставлять. Механизмы поиска, используемые сервисами белых и желтых страниц, поддерживают поиск агентов в P2P сети и сервисов соответственно. Поиск реализуется как распределённая процедура, которая

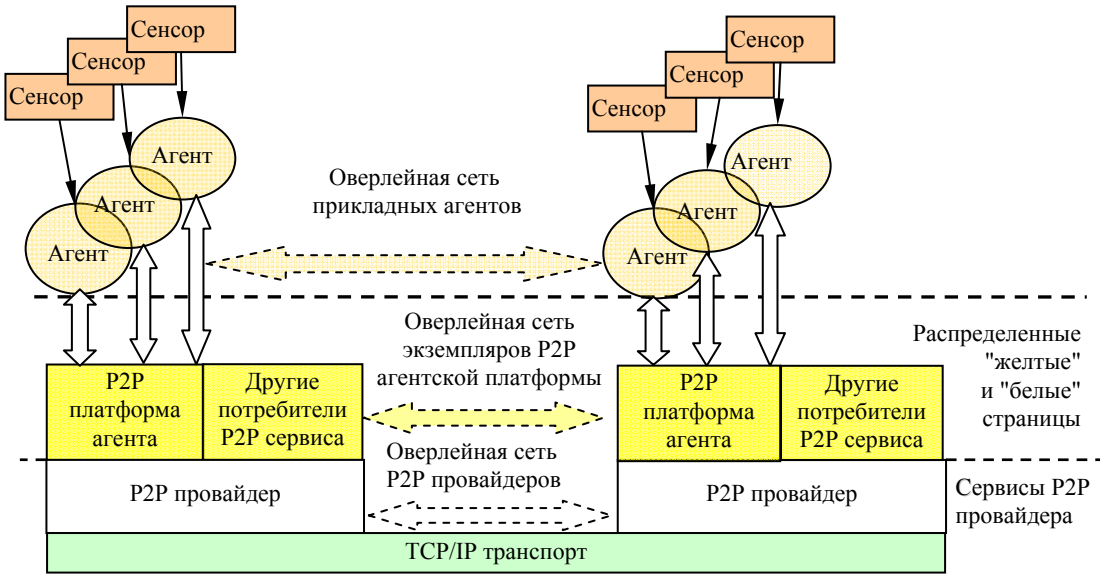

Рис. 3. Функциональная архитектура оверлейных сетей P2P провайдеров, экземпляров агентских платформ и прикладных агентов

использует протокол, известный под названием *gossiping* [14], т.е. тот же протокол, который используется в P2P сетях для маршрутизации сообщений.

На верхнем уровне программного обеспечения располагаются агенты прикладной системы, зарегистрированные на P2P АП.

Точно так же, как сеть P2P провайдеров образует "оверлейную сеть", установленную поверх, например, TCP/IP сети, множество экземпляро P2P АП образует "оверлейную сеть" установленную поверх сети P2P провайдеров (Рис. 3). В текущей реализации оверлейная сеть экземпляров агентской платформы имеет ту же самую топологию, что и оверлейная сеть P2P провайдеров. В следующей версии агентской платформы будет присутствовать возможность конфигурирования сети экземпляров агентской платформы иначе, чем сконфигурирована сеть P2P провайдеров. Эта возможность позволит автоматически (с использованием процедур обучения) выбирать конфигурацию экземпляров агентской платформы таким образом, чтобы оптимизировать распределенный поиск сервисов. Аналогично, множество прикладных агентов, установленных поверх экземпляров агентской платформы, образуют третью "оверлейную сеть" (Рис. 3). Заметим, что в разработанной версии уже присутствует возможность конфигурирования оверлейной сети прикладных агентов иначе, чем сконфигурированы сеть P2P провайдеров и сеть экземпляров агентской платформы. "Конфигурирование" здесь означает сопоставление с агентом списка его виртуальных соседей, с которыми агент, как это он себе представляет, может общаться "напрямую". На самом деле, конечно, эта связь реализуется напрямую только в TCP/IP сети. В других вариантах она реализуется в соответствии с топологией физических связей P2P сети, однако агент "думает", что он связывается напрямую. Именно в этом смысле используется выражение "платформа обеспечивает *прозрачность связи* агентов".

Важно отметить, что в начальном состоянии желтые и белые страницы могут быть вообще пустыми, а их заполнение может происходить уже в процессе работы сети, когда агенты объявляют о своих сервисах, запрашивают тот или иной сервис или получают ответы на свои запросы о сервисах. Содержание желтых и белых страниц может постоянно обновляться в процессе работы сети.

3.2. Взаимодействие агентской платформы с P2P провайдером

Общая схема взаимодействия компонент разработанной платформы с ее окружением демон-

стрируется на Рис. 2. С точки зрения P2P провайдера экземпляр агентской платформы - это просто один из потребителей предоставляемого им P2P коммуникационного сервиса. Если некоторое приложение желает получить доступ к P2P сети, то оно должно получить доступ к P2P провайдеру через специальный программный механизм, после чего зарегистрироваться на нём. При регистрации любой потребитель указывает свой уникальный идентификатор и тип. Идентификатор однозначно определяет потребителя в сети, а тип указывает на некоторое функциональное описание этого потребителя. Например, одним из типов потребителей P2P сервиса является "*agent-platform*" (агентская платформа), другим типом может быть "*file-storage*" (сервис хранения файлов) и т.п. Заметим, что прикладные агенты не могут регистрироваться на P2P провайдере.

После регистрации потребитель получает доступ к таким услугам провайдера, как отправка и получение сообщений, а также управление списком контактов (списком соседей) узла. Если потребитель больше не нуждается в услугах P2P провайдера, он может отменить свою регистрацию. В случае если некоторый потребитель P2P сервиса оказывается временно недоступным, то он может приостановить своё присутствие в сети, сохраняя регистрацию.

Как видно из описания, потребитель сервисов и P2P провайдер "слабо" связаны друг с другом, т.е. потребитель, в принципе, может функционировать и без использования P2P сервиса, например, в случае, если все агенты прикладной системы располагаются на одном экземпляре агентской платформы.

Рассмотрим взаимодействие потребителя с P2P провайдером более подробно (Рис. 4). Это взаимодействие выполняется не напрямую, а через специальную компоненту *Фабрику пиров* (*Peer Factory*). Эта компонента по запросу потребителя передает ему указатель на интерфейс P2P провайдера. Генератор P2P провайдеров по запросу потребителя может создавать их в двух вариантах: в варианте P2P провайдера уровня *приложения* (*in-proc-peer*) и в варианте уровня *системы* (*out-of-proc-peer*). Первый из них создаётся фабрикой пиров и находится в адресном пространстве процесса–потребителя сервиса. Преимуществом данного подхода является быстродействие, а недостаток состоит в том, что на таком P2P провайдере зарегистрироваться может только потребитель, создавший его. Второй тип P2P провайдера создаётся системой через механизм *COM*. Его преимущество состоит в том, что такой тип P2P провайдера может использоваться и другими приложениями системы. Однако он проигрывает в быстродействии, т.к. он оказывается задействованным в механизм межпроцессного взаимодействия между потребителем и P2P провайдером. Отметим, что этот механизм скрыт от потребителя и реализуется фабрикой P2P провайдеров.

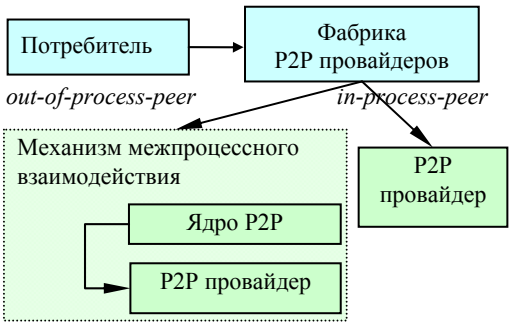

Рис.4. Функциональная архитектура P2P провайдера ("пира")

3.3. Взаимодействие агентов с агентской платформой

Связь между агентами и агентской платформой, в отличие от связи агентской платформы с P2P провайдером, является "сильной". Это означает, что агенты не могут функционировать вне агентской платформы. Платформа полностью управляет жизненным циклом агента: она создаёт его, инициализирует, инсталлирует на платформе, деинсталлирует и удаляет.

При создании агента экземпляром платформы происходит обмен интерфейсами между платформой и агентом. Через интерфейс платформы агент получает доступ к сервисам платформы, а через интерфейс агента платформа может, например, уведомлять агента о новых сообщениях. Агентам доступны такие сервисы агентской платформы, как отправка сообщений другим агентам, поиск конкретных сервисов и агентов, регистрация сервисов на жёлтых страницах, регистрация самого агента на белых страницах и т.д.

В разработанной платформе сервисы желтых и белых страниц реализованы как агенты. Эти агенты могут образовывать коалиции с агентами таких же типов, расположенными на "соседних" экземплярах платформы, которые, в свою очередь, расположены поверх P2P провайдеров соседних узлов. Такие коалиции позволяют прикладным агентам использовать алгоритмы поиска сервисов и агентов по протоколам, которые разработаны для маршрутизации сообщений в P2P сетях, например, по протоколу *gossiping* [14].

В заключение данного раздела ещё раз отметим суть интеграции P2P сети и сети экземпляров агентской платформы. Эту интеграцию можно рассматривать с двух сторон. С одной стороны, экземпляр агентской платформы использует P2P сеть для коммуникаций с другими экземплярами агентской платформы без использования сервера маршрутизации и далее, через экземпляры агентских платформ, – с установленными на них агентами. Таким образом, каждый P2P провайдер предоставляет стандартный коммуникационный сервис, связывающий различные экземпляры агентских платформ в одну распределённую агентскую платформу. С другой стороны, интеграция P2P сети и сети экземпляров агентской платформы расширяет возможности агентских систем, поскольку и экземпляры агентской платформы, и прикладные агенты могут быть структурированы иначе, чем узлы P2P коммуникационной сети. Эта интеграция предоставляет разработчикам приложений в среде повсеместных вычислений и коммуникаций принципиально новые возможности по реализации агентских приложений. Этот вопрос частично рассматривается на примере в следующем разделе.

4. Конфигурирование P2P сетей агентов и инфраструктуры взаимодействия

Как было отмечено ранее, прикладные агенты P2P сети образуют оверлейную сеть, установленную поверх сети экземпляров агентской платформы. Эта сеть может иметь структуру, отличную как от сети экземпляров агентской платформ, так и от сети P2P провайдеров коммуникационного сервиса. В реализованной архитектуре каждый агент имеет свой собственный "список контактов", определяющий его виртуальных соседей в P2P сети прикладных агентов, с которыми он может взаимодействовать напрямую. Естественно, что в этот список должны входить имена тех агентов сети, с которыми он взаимодействует чаще, чем с другими, что позволит снизить общий трафик в сети. Однако вначале агент может не знать, с какими агентами ему "полезнее" взаимодействовать чаще, а потому начальная конфигурация P2P сети прикладных агентов может быть далека от оптимальной. Процесс оптимизации списка виртуальных соседей, который выполняется одним, несколькими или всеми прикладными агентами сети, ведет к изменению ее топологии, и потому его называют процессом "(само)конфигурации" сети прикладных агентов. Например, в распределенной системе обнаружения объектов на земной поверхности, установленной на множестве беспилотных летательных аппаратов, зоны наблюдения отдельных аппаратов могут перекрываться. Поэтому целесообразно в первую очередь поддерживать взаимодействие агентов тех летательных аппаратов, которые имеют перекрывающиеся зоны наблюдения. Поскольку заранее такая информация агенту может быть недоступна, то агенты должны быть способны сами находить наилучших соседей в своей сети. Далее будет показано, что это может быть реализовано с помощью P2P алгоритмов (протоколов) обучения.

Другой аналогичный пример – система обнаружения вторжений в компьютерную сеть. Поскольку "следы" одной и той же атаки могут проявляться в большом количестве различных источников данных, которые собираются путем мониторинга различных процессов (трафика, логов операционной системы и приложений и т.д.), то одна и та же атака может обнаруживаться различными агентами. Если агент "знает" об агентах, решающих аналогичные задачи, то он может кооперироваться с ними для повышения качества обнаружения вторжений, например, за счет снижения вероятности ложных тревог. Но для этого каждый агент должен найти таких агентов, которые обучены обна-

ружению идентичных или "похожих" атак. Для каждого прикладного агента поиск таких агентов есть процесс формирования списка своих виртуальных соседей, т.е. он является процессом конфигурирования P2P сети. Легко заметить, что аналогичные процессы конфигурирования имеют большой практический смысл и для сети экземпляров агентских платформ (для ускорения поиска сервисов и агентов), и для сети P2P провайдеров коммуникационного сервиса (для снижения временных затрат на маршрутизацию). Рассмотрим проблему конфигурирования оверлейных P2P сетей на примере P2P сети прикладных агентов, совместно решающих задачу обнаружения вторжений.

Будем предполагать, что система обнаружения вторжений (СОВ) построена как сеть P2P агентов обнаружения вторжений, и в этой сети взаимодействие агентов поддерживается P2P агентской платформой, описанной в предыдущем разделе. При этих предположениях СОВ может рассматриваться как одно из приложений, которое работает в среде повсеместных вычислений и коммуникаций. Отметим, что состав узлов сети прикладных агентов, решающих задачу обнаружения вторжений, и топология их связей могут меняться в любой момент времени, если, например, в системе имеются мобильные устройства.

Предположим, что каждый из агентов СОВ осуществляет мониторинг некоторого источника данных. Заметим, что один и тот же источник данных может быть объектом мониторинга для нескольких агентов одновременно. Эти агенты могут быть обучены обнаружению различных типов атак или использовать различные алгоритмы для обнаружения атак одного и того же класса. В реальных системах число источников данных и число агентов СОВ может быть очень большим, причем их состав может изменяться во времени.

В рассматриваемой модели СОВ предполагается, что каждый агент обучен обнаружению атаки (или другой аномалии) только одного определенного класса. При этом агенты используют типовой сценарий работы. Обычно, если в процессе мониторинга агент не обнаруживает атаки "своего" класса, он "молчит", т.е. не посылает никаких сообщений. Если же он обнаруживает атаку "своего" класса, то он генерирует сигнал тревоги и посылает его администратору, ответственному за безопасность системы (далее, для краткости, "администратору"), который должен каким-то образом реагировать на этот сигнал.

Хорошо известно, что основной проблемой для администратора являются ложные тревоги, которые обычно генерируются достаточно часто. Получив такой сигнал, он должен решать, является ли полученный сигнал сигналом ложной тревоги или он соответствует атаке. В большинстве случаев администратор не останавливает систему, а, опираясь на свой опыт, чаще отключает некоторые компоненты системы защиты или меняет правила политики безопасности, которые слишком часто генерируют ложные тревоги. При этом возрастает риск пропуска атак, поэтому такой путь борьбы с ложными тревогами нельзя считать оправданным.

Хорошо известно, что если состав компонент системы защиты сети сохранять неизменным и не перенастраивать их атрибуты для улучшения качества работы (например, на основе процедур обучения), то единственным надежным путем борьбы с ложными тревогами является использование механизмов корреляции тревог, т.е. использование механизмов объединения решений (в данном случае– тревог), которые выдаются различными компонентами СОВ. Обычно для решения этой задачи выделяется специальный программный модуль, который собирает решения с различных средств защиты и принимает решение, используя всю эту информацию.

Однако в P2P сетях, работающих в среде повсеместных вычислений, такой подход невозможен ввиду изменчивости состава узлов и топологии сети. Последнее всегда имеет место, если в сети имеются мобильные устройства, если устройства могут свободно входить в систему и покидать ее, как это происходит в открытых сетях. Это может привести к тому, что сеть может покинуть узел, в котором помещена программа корреляции тревог. Кроме того, централизованная схема корреляции тревог невозможна часто и потому, что в сети может быть огромное количество источников данных, а при отсутствии сервера собрать всю инфор-

мации в одном месте в режиме реального времени практически невозможно.

Альтернативой такой модели корреляции атак является модель, в которой каждый агент системы обнаружения вторжений выступает также и в роли коррелятора атак, "заботясь" о достоверности своего решения. В этом варианте агент должен запускать свой механизм корреляции атак тогда и только тогда, когда он выработал сигнал тревоги на основании своего локального источника данных, чтобы подтвердить или опровергнуть факт атаки. Однако, поскольку в реальном времени агент может получить информацию только от ограниченного количества источников (других агентов), то он должен каким–то образом сделать выбор, с какими агентами ему наиболее полезно взаимодействовать. При этом агент не может использовать априорные экспертные знания, поскольку состав узлов сети и ее топология изменяются во времени. Поэтому агент вынужден пользоваться процедурой обучения, в которой, наряду с традиционной задачей (каким образом объединять решения своих виртуальных соседей?), он должен решать другую, более сложную задачу, а именно, чьи решения целесообразно использовать в процессе объединения решений.

Для агента более сложной является вторая задача. Наиболее "прямолинейный" выбор состоит в том, чтобы использовать решения только тех агентов, которые обучены обнаружению атак того же класса, что и он сам. Действительно, такие агенты, в принципе, могут существовать, но могут и отсутствовать в конкретной конфигурации или быть временно недоступными. Однако часто оказывается полезной информация и от агентов, которые настроены на обнаружение атак других классов, в чем-то похожих на атаки "собственного" класса. Рассмотрим, каким образом они могут быть найдены.

Предположим, что каждый агент СОВ в начальный момент имеет некоторый список своих виртуальных соседей. При выработке сигнала тревоги агент опрашивает их и пытается подтвердить или опровергнуть наличие атаки, обнаруженной им. Те виртуальные соседи, чьи решения агент использует, будем называть "коалицией".

Если коалиция агента и алгоритм корреляции их решений известен, то типовой сценарий работы *каждого агента* СОВ может быть таким. Когда агент обнаруживает (верно или неверно) атаку "собственного" класса, он начинает опрашивать агентов своей коалиции о решениях, принятых ими в некотором промежутке времени, близком к текущему моменту времени. Получив соответствующую информацию от агентов коалиции, он использует некоторый алгоритм объединения решений, принимает решение и посылает его администратору. Администратор уже принимает окончательное решение. В любом случае вектор решений агента и его коалиции с итоговым решением, которое выработано агентом, и с окончательным решением администратора записываются в базу данных. Этот вектор далее используется как экземпляр обучающей выборки агента СОВ для обучения решению задачи корреляции атак, в которой он должен: выбрать коалицию и построить алгоритм объединения своего решения и решений агентов коалиции.

Дадим описание сценария решения первой задачи, а именно, P2P обучения агента формированию коалиции.

Шаг 1. Этот шаг относится к начальной ситуации, когда агент имеет пустой список коалиции, когда агент должен как-то сделать начальный выбор коалиции, чтобы потом улучшать его в процессе обучения. Выбор можно делать различными путями, включая использование экспертной информации. В примере, который реализован программно в виде прототипа СОВ, для каждого агента исходная коалиция формировалась путем случайного выбора заданного количества агентов из числа тех, которые удовлетворяют некоторому условию. В качестве аргумента условия использовано понятие расстояния между узлами, измеряемое минимальным числом узлов, которые необходимо "посетить" в P2P сети при переходе из одного узла заданной пары в другой. Например, условие может выражать требование, чтобы расстояние было не более, чем заданное целое число N. Но на данном шаге выбранные агенты рассматриваются только как кандидаты на включение в коалицию, поскольку о них еще

не собрана обучающая информация, которая позволила бы провести процедуру обучения.

Шаг 2. В течение какого-то количества шагов агент работает в режиме мониторинга своего источника данных и обнаружения вторжений (см. выше). При этом он накапливает в базе данных информацию о тех решениях, которые он получал от своих "виртуальных соседей" в тех случаях, когда сам обнаруживал атаку и генерировал тревогу. Шаг 2 продолжается до тех пор, пока не выполнится условие перехода к процедуре обучения. Это условие может определяться, например, объемом накопленных обучающих данных. Частным случаем является обучение в реальном времени, когда процедура обучения реализуется после получения каждого очередного примера обучающих данных.

Шаг 3. На этом шаге выполняется *обучение*. Оно может выполняться либо самим агентом СОВ, либо другим агентом, специально выделенным для решения задачи обучения. Пусть эта задача решается выделенным агентом, *Агентом обучения,* который установлен на том же экземпляре агентской платформы. Этот агент использует накопленные данные в качестве обучающей выборки, решая две ранее названные задачи, а именно:

- из множества виртуальных соседей обучаемого агента выбирает по некоторому критерию заранее заданное число наиболее компетентных агентов, которые образуют его коалицию (один из вариантов функции компетентности агентов рассматривается далее в примере);
- обновляет значения атрибутов алгоритма корреляции тревог.

При решении первой задачи агент использует некоторую функцию, которая рассматривается им как мера компетентности виртуальных соседей. После оценки компетентности соседей и выбора состава коалиции агент удаляет наименее компетентных агентов из списка своих виртуальных соседей и с помощью некоторой процедуры выбирает новых кандидатов на включение в коалицию, добавляя их в свой список контактов (виртуальных соседей) вместо имен удаленных агентов. Далее решается задача обучения корреляции тревог (объединения решений) новой коалиции.

Шаг 4. После окончания процедуры обучения *Агент обучения* обновляет список виртуальных соседей и состав коалиции обучаемого агента, а также передает ему новый алгоритм корреляции тревог. Далее сценарий повторяется, начиная с шага 2.

Этот алгоритм P2P обучения агентов обнаружению атак был проверен на примере работы простого программного прототипа P2P агентской системы обнаружения вторжений. При этом генерация обучающей выборки (вектора решений агентов системы и его интерпретации в терминах "*тревога*"–"*ложная тревога*"), которые использовались далее для P2P обучения, генерировались с помощью специально разработанной вероятностной модели. "Истинная интерпретация" моделировала решение администратора. Таким способом накапливались данные для обучения агента.

Для корреляции тревог использовались два механизма объединения решений: "*Правило произведения*" и "*Правило максимума*" [13]. В обоих случаях использовалась одна и та же мера компетентности агентов, которая вычислялась по формуле:

$$Comp_{A\to"1"}(A_i,k) = P_{A\to"1"}[H_1(k)/A_i \to "1"] + P_{A\to"1"}[H_0(k)/A_i \to "0"]$$

где величина $Comp_{A\to"1"}(A_i,k)$ обозначает значение компетентности агента A_i из коалиции агента A; $P_{A\to"1"}[H_1(k)/A_i \to "1"]$ и $P_{A\to"1"}[H_0(k)/A_i \to "0"]$ обозначают вероятности гипотезы "*Агент A_i генерирует тревогу тогда, когда агент A генерирует тревогу и атака класса k действительно имеет место*" и "*Агент A_i не генерирует тревогу тогда, когда агент A генерирует ложную тревогу*" соответственно.

Смысл этой меры компетентности достаточно простой. Мера компетентности $Comp_{A\to"1"}(A_i,k)$ агента A_i равна сумме вероятностей совпадения решения агента коалиции с правильным решением при наличии и при отсутствии атак класса *k, когда агент, обученный обнаружению атак этого класса, генерирует тревогу*. Чем больше значение этой функции, тем чаще агент коалиции A_i выдает правильное решение, когда агент A гене-

рирует ложную тревогу или пропускает сигнал. Значения этой функции используются для упорядочения виртуальных соседей в соответствии с их компетентностью. Упомянутые ранее "*Правило произведения*" и "*Правило максимума*" [13] используются в качестве механизмов корреляции тревог агента *A* и агентов–членов коалиции.

Рис. 5 и Рис. 6 демонстрируют, каким образом в результате обучения изменяется топология оверлейной сети прикладных агентов обнаружения вторжений в данной конкретной задаче. Сеть прикладных агентов на этих рисунках включает в себя 10 СОВ. Рис.5 демонстрирует начальную конфигурацию оверлейной сети прикладных агентов, а Рис.6 - топологию этой же оверлейной сети после двух итераций обучения, на каждой из которых для различных агентов накапливались обучающие выборки размером от 800 до 2000. Тестирование системы после каждой итерации выполнялось на иной выборке данных. Результаты тестирования показали, что качество работы СОВ в целом существенно улучшается. В частности, в результате процедур P2P обучения вероятность ложных тревог уже за две итерации обучения снижается на 21%, что следует считать весьма хорошим результатом. Здесь не приводятся детально другие численные данные, поскольку цель, которая преследовалась при разработке программного прототипа СОВ и проведении последующих экспериментов с ней состояла не в том, чтобы получить какой-то новый результат в области компьютерной безопасности. Цель состояла только в том, чтобы проверить экспериментально работоспособность разработанного протокола обучения.

Важно отметить, что описанный выше алгоритм P2P обучения представлен в достаточно общей форме. Он лишь намечает общий протокол P2P обучения. Его конкретная реализация может отличаться выбором оптимизируемых функций, порогов генерации тревог и т.д.

Заметим, что в том виде, в каком протокол P2P обучения описан выше, он может использоваться для оптимизации конфигураций всех трех оверлейных сетей, а также в ряде других задач P2P обучения.

Заключение

В работе рассмотрена задача интеграции распределенных интеллектуальных систем и средств телекоммуникаций в рамках парадигмы повсеместных вычислений и коммуникаций. Эта парадигма, изначально ориентированная главным образом на реализацию встроенных систем, в настоящее время приобретает все более широкое применение. Она претендует на роль новой парадигмы построения и новой технологии реализации широкого круга современных приложений.

Среди множества новых проблем, которые инициированы технологией повсеместных вычислений и коммуникаций, одной из главных является проблема разработки программной инфраструктуры, предназначенной для под-

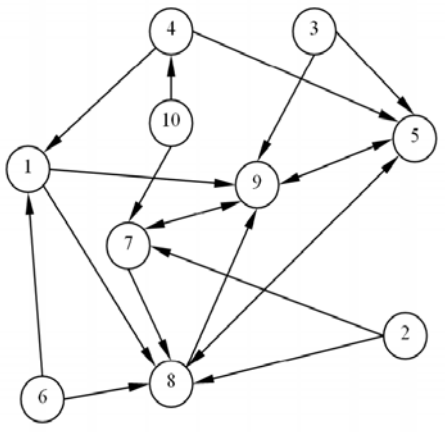

Рис.5.Начальная конфигурация P2P СОВ

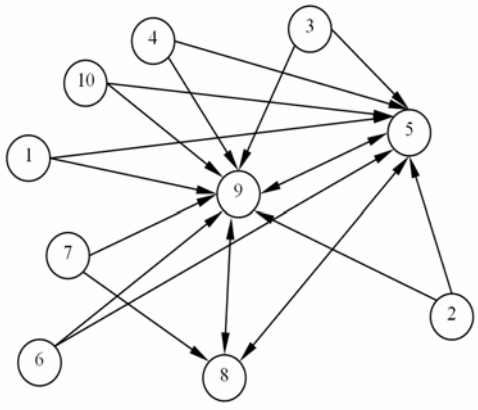

Рис.6 .Итоговая конфигурация P2P СОВ

держки совместной работы гетерогенных устройств и программных компонент. В работе проанализировано современное состояние исследований и разработок в данной области, перечислены основные требования к программным инфраструктурам, а также описаны конкретные разработки, которые ведутся по этому направлению в Европе в проектах 6-ой рамочной программы Европейской комиссии.

Основным результатом данной работы являются алгоритмы функционирования компонент P2P агентской платформы и их программная реализация. Эта платформа предлагает один из перспективных путей построения программной инфраструктуры для поддержки совместной работы гетерогенных устройств и программ в среде повсеместных вычислений и коммуникаций. Разработанная платформа к настоящему времени прошла достаточно всестороннее тестирование на различных приложениях.

В текущей реализации платформа поддерживает совместное функционирование программ, работающих под управлением операционных систем Windows и Linux. Дальнейшие исследования направлены на расширение множества допустимых операционных сред, в основном, операционных систем, используемых на мобильных устройствах (например, операционной системы Symbian). Кроме того, платформа нуждается в расширении возможностей для поддержки работы в гетерогенных коммуникационных средах.

Второй результат данной работы – это достаточно общий алгоритм (сценарий) P2P обучения, проверенный экспериментально в рамках частной задачи. Этот алгоритм предназначен для конфигурирования оверлейных сетей, формирующих открытые P2P сети агентов, совместно с инфраструктурой их взаимодействия и коммуникационной средой. Следует подчеркнуть, что способность систем, работающих в среде повсеместных вычислений и коммуникаций, поддерживать процедуры конфигурирования является практически обязательным свойством, которое необходимо для адаптации к изменению состава и взаимосвязей устройств и программ, формирующих систему и среду ее функционирования.

Полная информация о текущей версии P2P агентской платформы может быть найдена на веб–сайте Лаборатории интеллектуальных систем СПИИРАН по адресу http://space.iias.spb.su/ap. Здесь можно найти детальное руководство пользователя и разработчика, научные статьи, посвященные платформе, а также свободно распространяемый программный код (*runtime* версию), который может быть загружен и использован на практике.

Литература

1. M. Alcaniz, B.Rey. New Technologies for Ambient Intelligence. In Ambient Intelligence 3, G. Riva, F. Vatalaro, F. Davide, M. Alcañiz (Eds.) IOS Press, 2005, http://www.ambientintelligence.org.
2. Angel. FP-6 Project web site. http://www.ist-angel-project.eu.
3. Angel Project: Infrastructure for ubiquitous computing and communication. http://www. ist-angel-project.eu/docs/ ANGEL_D1_1_R_PU _AngelPlatformDefinition _20070205_01.pdf.
4. AWARE. FP-6 Project web site. http://www.aware-project.net/
5. CobiS. FP-6 Project web site. http://www.cobis-online.de
6. T.A. Elbatt, S.V.K. Murthy, D. Connors, and S. Dao. Power management for throughput enhancement in wireless ad-hoc networks. IEEE International Conference on Communications, 1503–1513, June 2000.
7. EMMA. FP-6 Project web site.http://www.emmaproject.eu.
8. FIPA abstract architecture. http://www.fipa.org/specs/fipa00001/XC00001J.html#_Toc8186403
9. FIPA P2P NA WG6, Functional Architecture Specification Draft 0.12., http://www.fipa.org/ subgroups/P2PNA-WG-docs/P2PNA-Spec-Draft0.12.doc
10. P.Gil, I.Maza, A.Ollero and P.J. Marron. Data centric middleware for the integration of wireless sensor networks and mobile robots.ROBOTICA 2007-7-th Conference on Mobile Robots and Competitions. Portugal, April 27, 2007.
11. V.Gorodetsky, O.Karsaev, V.Samoylov, S.Serebryakov. P2P Agent Platform: Implementation and Testing. The AAMAS Sixth International Workshop on Agents and Peer-to-Peer Computing (AP2PC 2007), Honolulu, 21-32, 2007.
12. Hydra. FP-6 Project web site. http://www.hydra.eu.com/
13. Kittler J., Hatef M., Duin R. P. W., Matas J. "On combining classifiers." IEEE Trans.on Patt. Anal. and Mach. Intellig., Vol. 20, No. 3, 226–239, 1998.
14. N.Lin., K.Marzullo, S.Masini. Gossip versus Deterministic Flooding. Technical report CS1999-0637, http://citeseer.ist.psu.edu/563854.html
15. LIS Agent Platform, http://space.iias.spb.su/ap/
16. MORE. FP-6 Project web site. http://www.ist-more.org.
17. D.Murphy, J.Kelly, K.Curley, J.Vickery, D.O'Keeffe. P2P Security. http://ntrg.cs.tcd.ie/ undergrad/4ba2.02-03/p10.html
18. Runes. FP-6 Project web site. http://www.ist-runes.org.

19. Sense. FP-6 Project web site. http://www.sense-ist.org.
20. SMEPP. FP-6 Project web site. http://www.smepp.org.
21. SOCRADES. FP-6 Project web site. http://www.socrades.eu.
22. M.Weiser. The Computer for the Twenty-First Century, Scientific American, 94-102, September 1991.
23. Zigbee standard. http://www.zigbee.org.

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 200 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 инструментальные средства распределенного обнаружения знаний в данных. P2P системы; распределенное принятие решений.

Карсаев Олег Владиславович. Заведующий лабораторией Санкт-Петербургского института ифнорматики и автоматизации РАН. Окончил Военно-инженерний институт им.А.Ф.Можайского в 1981 году. Кандидат технических наук. Опубликовал более 50 работ. Область научных интересов: многоагентные системы, задачи обучения, задачи планирования и составления расписаний.

Самойлов Владимир Владимирович. Научный сотрудник Санкт-Петербургского института информатики и автоматизации РАН. Окончил Высшее военно-морское инженерное училище им. Ф.Э. Дзержинского в 1993 году. Опубликовал более 30 работ. Область научных интересов: многоагентные системы, системы объединения данных из разных источников, методы распределенного обучения.

Серебряков Сергей Валерьевич. Младший научный сотрудник Санкт-Петербургского института информатики и автоматизации РАН. Окончил Санкт-Петербургский государственный политехнический университет в 2004 году. Награжден дипломом II степени на научной секции X Международной студенческой олимпиады по автоматическому управлению за практический вклад, победитель Чемпионата мира по футболу роботов RoboCup в симуляционной лиге двухмерного футбола (команда STEP, Португалия, 2004), завоевал Второе место на Открытом Европейском чемпионате по футболу роботов RoboCup в симуляционной лиге двухмерного футбола (команда Zenit-New ERA, 2004). Опубликовал 13 работ. Область научных интересов: искусственный интеллект, многоагентные системы, машинное обучение, объединение информации, сервис-ориентированные архитектуры, P2P системы.