Аннотация

Технология мобильных агентов достаточно новая, поэтому системы программирования мобильных агентов существенно различаются по архитектуре и реализации. Эти различия препятствуют интероперабельности и быстрому внедрению систем мобильных агентов. Учитывая уже существующие системы, как объектные, так и не объектные, в MAF выделены аспекты, которые необходимо стандартизировать для обеспечения интероперабельности систем мобильных агентов.

Содержание

1 Мобильные агенты **2**
- 1.1 Применение . . . 3
- 1.2 Свойства агентов . . . 4
- 1.3 Система мобильных агентов . . . 5
- 1.4 Проблемы мобильности . . . 5

2 Требования к реализации агентских систем **6**

3 Общие понятия технологии мобильных агентов **7**
- 3.1 Обзор . . . 7
- 3.2 Объекты стандартизации . . . 9
- 3.3 Основные понятия . . . 10

4 MAF IDL **18**

Литература **22**

A Существующие реализации агентских систем **23**

Цели

Основная цель документа *CORBA Fascilities: Mobile Agents System Interoperability Fascilities Submission* [1] (сокращенно MAF) – обеспечить интероперабельность между различными существующими и разрабатываеми агентскими системами.

Интероперабельность достигается при стандартизации таких аспектов, как передача агентов и служебных (используемых агентом) классов (или кода для необъектных систем) между агентскими системами, а также управление агентами. Под управлением агентами подразумеваются функции администратора агентской системы, такие как создание агента заданного класса, его уничтожение, прерывание и возобновление деятельности агента (т. е. соответствующем потоке управления). Кроме перечисленных выше аспектов, необходима также стандартизация синтаксиса и семантики различных параметров, например имен агентов и агентских систем, типов агентских систем.

На следующем этапе следует стандартизировать действия агентских систем при пересечении агентом нескольких доменов безопасности, а также формат представления кода и состояния агента при его перемещении между агентскими системами разных типов.

В MAF не затрагиваются вопросы языковой интероперабельности и не рассматриваются зависящие от реализации операции, такие как сериализация/десериализация, интерпретация агентов, а также взаимодействие агентов.

1 Мобильные агенты

Основной идеей технологии агентов является выполнение задач (администрирования, поиска данных и т.п.) непосредственно на локальной машине, что снижает до минимума взаимодействие между администартором и узлами системы.

Предпосылки

Рассмотрим пример. Клиент хочет стереть с удаленного сервера все файлы, которые были созданы более чем два месяца назад. Если решать эту задачу с помощью RPC, то клиенту придется выполнить $N+1$ вызовов (1 – для получения данных о файлах, N – для удаления N файлов). Таким образом, всего по сети будет передано $2(N+1)$ сообщений ($(N+1)$ запросов и $(N+1)$ ответов).

Определение мобильного агента

Рассмотрим другой подход к решению этой задачи – метод удаленного программирования (Remote Programming, RP). Клиент создает процедуру, которая анализирует файлы сервера и удаляет файлы, удовлетворяющие заданному в ней критерию. Затем клиент передает эту процедуру для выполнения на сервере. При таком подходе по сети передается два сообщения (передача процедуры и возврат результатов ее работы). Будем называть такую перемещаемую процедуру мобильным агентом. Удаленное взаимодействие заменяется локальным, и уменьшается зависимость от работы сети, уменьшается трафик. Второе преимущество этого метода – расширение функциональности сервера. Серверные компоненты приложения, работающего через RPC, должны быть статически инсталлированы, в то время как серверные компоненты приложения, использующего RP, динамически инсталлируются агентами.

Можно рассмотреть пример, в котором клиент хочет взаимодействовать с несколькими серверами. Пусть существует M таких серверов и к каждому из них клиент делает N_i обращений. Таким образом, при использовании первого подхода мы имеет $2((N_1+1)+(N_2+1)+\ldots+(N_M+1))$ сообщений.

Второй подход в этом случае предполагает, что агент перемещается от сервера к серверу и затем возвращает результаты клиенту. При этом возможен случай, когда совокупность серверов, которые должен обойти агент, формируется динамически в процессе его работы. Так, возможно, в процессе работы на i-ом сервере агент определяет свои дальнейшие действия, например, он решает вернуть результаты клиенту и закончить работу. В случае использования RPC, анализ получаемых от серверов результатов ложится на клиента. Если агент последовательно “обходит” все сервера, по сети будет передано $M+1$ сообщение.

Конечно, для оценки трафика нужно оценивать длину сообщений, передаваемых в случае обычного запроса и мобильного кода. Очевидно, что для выполнения запросов на одном удаленном сервере метод RP может не дать выигрыша в трафике, так как объем мобильного кода может превысить суммарный размер сообщений, содержащих запросы и ответы. Если же задействовано несколько удаленных серверов, каждому из которых к тому же нужно послать большое число запросов, то использование мобильного агента, содержащего в себе (инкапсулирующего) эти запросы, кроме отмеченных выше преимуществ, снижает трафик. Можно утверждать следующее: чем больше вычислений нужно выполнить, тем больший выигрыш дает эта технология.

1.1 Применение

В каких задачах могут применяться мобильные агенты:

- мобильные вычисления (миграция агентов может поддерживаться не только между постоянно подсоединенными к сети узлами, но и между мобильными платформами, подключаемыми к постоянной сети на некоторые промежутки времени и возможно по низкоскоростным каналам). Клиент подсоединяется к постоянной сети на короткий промежуток времени с мобильной платформы, отправляет агента для выполнения задачи и отсоединяется; затем клиент подсоединяется к другой точке сети и забирает результаты работы агента. Второй вариант – сервер, на который должен переместиться агент, подсоединяется к сети, а затем отсоединяется. В этом случае агент должен уметь переместиться на такой временно подсоединяемый сервер и вернуться в постоянную сеть.
- задачи управления информацией:
    1. Поиск информации (море информации – один человек не в состоянии найти необходимую ему информацию и проанализировать ее – использование агента, который странствует по сети в поисках информации, лучше всего удовлетворяющей потребности человека); поисковые агенты содержат сведения о различных информационных источниках (включая тип информации, способ доступа к ней, а также такие характеристики информационного источника, как надежность и точность данных);
    2. Отбор (обработка) информации. Из всех данных, приходящих к клиенту, выбирают только те данные, которые могут быть интересны клиенту. Используются в комбинации с поисковыми агентами (сначала поиск, затем – отбор);
    3. Мониторинг данных. Извещение пользователя об изменениях в различных источниках данных в реальном времени (например, мобильный агент перемещается на вычислительный узел, на котором расположен источник данных; это эффективнее, чем использовать статического агента, посылающего запросы источнику данных);
    4. Универсальный доступ к данным. Агенты – посредники для работы с различными источниками данных, имеющие механизмы для взаимодействия друг с другом (например, агент создает несколько агентов, каждый из которых работает со своим источником данных).

Примерами использования агентов может быть поиск информации (data mining, агенты взаимодействуют с серверами баз данных и хранилищами данных), электронная коммерция.

1.2 Свойства агентов

Некоторые свойства агентов являюыся ключевыми: автономность, взаимодействие, мобильность. Кроме этого существуют и другие свойства агентов. Рассмотрим эти и другие свойства более подробно:

1. *автономность* – агенты могут выполнять свои задачи без непосредственного вмешательства клиентов или других агентов;
2. *взаимодействие* – при возникновении потребности агенты взаимодействуют с другими агентами или людьми с целью получения или оказания помощи в решении задачи;
3. *мобильность* – способность перемещаться между различными узлами сети во время выполнения; агенты, не обладающие этим свойством, называются статическими;

   Перечень свойств агентов можно расширить:

4. *реактивность* – агенты реагируют на изменения среды в реальном времени, обычно их деятельность описывается следующим образом:

   ```
   WHEN <event> IF <condition> THEN <action>;
   ```

5. *проактивность* – способность решать задачи (достигать цели); в отличие от реактивных агентов, они не просто реагируют на изменения среды, но и сами ее опрашивают;
6. способность существовать как постоянно выполняющийся процесс, точнее, иметь собственный поток управления;
7. способность выполняться от имени некоторого *владельца*;
8. *гибкость* – действия агентов не фиксированы жестко;
9. *интеллектуальность* (обучаемость) – способность находить новые решения; такие агенты могут изменять свое поведение, используя как свой опыт, так и опыт других агентов.

Будем придерживаться определения, согласно которому первые 3 свойства являются обязательными для агента, а остальные – необязательными. Классификация агентов – от “слабых” к “сильным”, т. е. по степени, в которой проявляются эти свойства. Как минимум агент – это сущность, которая взаимодействует со своим окружением. Пример работы агентов разных типов: один “большой”, интеллектуальный агент посылает множество “маленьких”, мобильных агентов, которые осуществляют поиск данных, а он сам – ее обработку.

1.3 Система мобильных агентов

Клиент – сущность, пользующаяся услугами агента. В нашем случае клиент также является владельцем агента.

В случае мобильных агентов на каждом вычислительном узле должен существовать сервер (далее будем называть его агентской системой), то есть платформа, предоставляющая функциональность для создания/уничтожения, приема/передачи и среду для выполнения агентов. Эта среда должна быть безопасной. Для организации взаимодействия агентов должен существовать сервис именования, т. е. сервис, предоставляющий возможность работы с сущностью, зная только ее имя. Функции этого сервиса выполняет агентская система.

Агентская система может иметь своего владельца. Агентская система имеет определенный тип, то есть агенты, управляемые ей, имеют некоторый профиль (например, в понятие профиля агента могут входить: язык реализации агента, производитель, алгоритм сериализации агента). Профили необходимы для обеспечения взаимодействия различных систем – так, зная профиль агента, система в состоянии определить, может ли она обеспечить среду для выполнения этого агента.

Агентская система может поддерживать понятие Места ("плэйса",place). В этом случае сеть компьютеров представляется как набор мест, предоставляющих сервисы. С другой стороны, место является контейнером и фабрикой агентов. Агент порождается в месте и умирает в месте. В течение жизненного цикла агент перемещается между местами. Сервисы, предоставляемые местом, используются агентами, находящимися в нем. Плэйс может накладывать ограничения на ресурсы, которые используют агенты, находящиеся в нем. Плэйсы могут существовать как на стороне клиента, так и на стороне сервера. Плэйс может хранить "следы" посещавших его агентов. Агент может содержать "историю" своей жизнедеятельности (может использоваться для проверки поведения агента (например, для проверки ограничений безопасности) и для обучения агента на собственном опыте).

Если понятие места ("плэйса") не поддерживается, его функции берет на себя сама агентская система.

Агент может порождать других агентов. Таким образом, возникает проблема делегирования задач и полномочий, управления совокупностями агентов, имеющими общих предков ("поколениями" и "семьями" агентов).

Агент выполняет задание, если он располагает достаточными для этого ресурсами и имеет соответствующие права. В противном случае он может искать другого агента, способного выполнить это задание. Агент также может отказаться от выполнения задания, поскольку он перегружен в данный момент или очередь заданий этого агента переполнена.

1.4 Проблемы мобильности

Мобильные агенты создают дополнительные проблемы, такие как взаимодействие с агентами, меняющими свое расположение в сети (то есть взаимодействие с агентом в то время, как он перемещается), безопасность (мобильный код начинает выполняться на вычислитель-

ном узле, и агентская система должна защищать ресурсы вычислительного узла на котором она выполняется), взаимодействие с агентами в случае отказов сети (устойчивость к отказам), синхронизация времени (в случае, если задачи, выполняемые агентом, являются “time sensitive”, то есть должны быть выполнены в реальном времени).

2 Требования к реализации агентских систем

Обеспечение переносимости кода на различные платформы

Понятие мобильности неразрывно связано с понятием переносимости. Переносимость кода можно обеспечить двумя различными способами. Можно реализовать ее посредством использования интерпретируемых языков (Perl, Tcl и др.), обеспечив каждую вычислительную машину соответствующим интерпретатором, можно использовать один из языков, поддерживающий раздельную компиляцию (Java, Oberon и др.). Кроме того язык должен поддерживать механизмы посылки, передачи, получения и выполнения мобильного кода.

Доступность на множестве платформ

Это требование является продолжением предыдущего. Мобильные агенты должны осуществлять свою работу в гетерогенной компьютерной среде. Любой компьютер должен уметь принять и обеспечить работу мобильного агента.

Поддержка сетевого взаимодействия

Помимо операций непосредственно связанных с перемещением между агентскими серверами, агент должен обладать средствами для коммуникации с другими агентами и доступа к удаленным ресурсам. Поэтому поддержка сетевых услуг должна включать в себя широкий спектр возможностей (служба имен, RPC, OLE, CORBA, RMI и т.д.).

Многопоточная обработка

Для реализации одновременного выполнения нескольких действий агентская система должна включать в себя поддержку параллельного выполнения функций агента и поддержку средств синхронизации.

Безопасность

Мобильные агенты, приходящие из сети, могут содержать потенциально опасный, вредоносный код. Поэтому система должна поддерживать средства обеспечения безопасности достаточные для ее нормальной работы.

3 Общие понятия технологии мобильных агентов

3.1 Обзор

В MAF вводятся базовые понятия технологии мобильных агентов и предлагается архитектура агентской системы.

Агент определяется как программа, выполняющаяся от имени некоторого владельца. Агент имеет собственный поток управления и в объектных агентских системах рассматривается как активный объект. Агент имеет код и состояние, причем в понятие состояния включается состояние выполнения, являющееся либо регистровым контекстом данного агента, либо набором значений атрибутов, определяющим деятельность агента при возобновлении его выполнения после миграции. Агенты однозначно идентифицируются следующим набором значений: владелец агента, имя агента и тип агентской системы (создавшей данного агента).

Агентская система – это платформа, предоставляющая функциональность для создания/уничтожения, приема/передачи и выполнения агентов. Агентская система имеет владельца и обладает определенным типом. Тип агентской системы определяет профиль, или свойства агентов, создаваемых в данной системе. В понятие профиля входят тип агентской системы, язык реализации и метод сериализации агентов.

Агент всегда выполняется в некотором контексте агентской системы, называемом Место (Place).

Для обеспечения безопасности вводится понятие региона, или, другими словами, федерации агентских систем. Регион содержит агентские системы возможно разных типов, но имеющие одного владельца. Одна из функций региона – именование агентов и мест. Таким образом, клиент, запрашивающий взаимодействие с агентом, должен знать только имя агента и имя соответствующей агентской системы.

Все взаимодействия агентских систем производятся коммуникационной инфраструктурой.

С точки зрения обеспечения интероперабельности агентских систем выделены следующие виды взаимодействия агентов: удаленное создание, передача, вызовы методов (когда агент вызывает метод агента или объекта, если имеет объектную ссылку на этот объект и соответствующие права доступа).

Общими функциями агентской системы являются прием/передача агентов и служебных классов, создание агентов. Последовательность действий при выполнении этих операций специфицирована. Приведены различные стратегии передачи служебных классов (автоматическая передача всех классов, связанных с агентом, или передача по требованию).

Агентская система должна обеспечивать безопасное окружение для функционирования агентов, в том числе поддерживать концепцию региона. Для того чтобы обеспечить безопасность системных ресурсов, агентская система должна уметь идентифицировать и проверять права сущности (клиента), создавшей агента. Агент, также как и агентская система,

может определять собственную концепцию безопасности. Конкретная концепция безопасности выбирается в динамике. Концепция безопасности может ограничивать права доступа и используемые ресурсы. При миграции и вызове методов агент может специфицировать трубуемый уровень безопасности. Для безопасного взаимодействия агентов требуются: идентификация клиента, взаимная идентификация агентских систем, идентификация агентов и распространение их полномочий.

Также агентская система должна предоставлять сервис поиска мобильных агентов, именования агентов и мест.

Для мобильных агентов, являющихся объектами CORBA, приводятся сценарии использования стандартных сервисов CORBA для реализации агентской системы. Агент, являющийся объектом CORBA, может быть зарегистрирован в сервисе именования. Так как объектная ссылка CORBA содержит имя узла, в котором находится объект, и номер порта, то объектная ссылка на мобильный агент должна обновляться после каждой миграции. Это обновление может обеспечиваться самим ORB, сервисом именования, или в узле создания агента может существовать объект-представитель (proxy), перенаправляющий обращения к агенту. Для поддержки жизненного цикла агента могут использоваться LifeCycle и Externalization сервисы. Последний предоставляет средства для сохранения состояния объекта в потоке данных и восстановления из него. Хотя Security сервис CORBA не обеспечивает всех необходимых для агентских систем механизмов безопасности, он также может использоваться при реализации агентских систем.

MAF определяет набор интерфейсов, обеспечивающих интероперабельность агентских систем. Эти интерфейсы определены на уровне агентских систем. Как агенты, так и сами агентские системы могут быть объектами CORBA. Поскольку агенты мигрируют между агентскими системами, поддерживающими один и тот же профиль агентов, нет необходимости в стандартизации интерфейса агента – для обеспечения интероперабельности агентских систем одного типа стандартизируются операции управления мобильными агентами. В спецификации MAF определены интерфейсы объектов `MAFAgentSystem` и `MAFFinder`.

В интерфейсе `MAFFinder` определены операции регистрации/разрегистрации и разрешения имен агентов, мест и агентских систем, то есть объект этого типа является сервисом именования. Один такой объект может разделяться между несколькими регионами. Определена структура имен агентов и используемых им классов, формат преставления адреса агента (либо URI, содержащий составное имя, либо URL, содержащий адрес Internet).

В интерфейсе MAFAgentSystem определены операции приема/передачи, создания/уничтожения, прерывания/возобновления агентов. Следует отметить, что параметрами данных операций являются имена агентов, а не объектные ссылки. Для обеспечения механизмов взаимодействия агентов различных профилей (различных типов) определена операция получения адреса агентской системы, “ближайшей” к заданному профилю.

3.2 Объекты стандартизации

Как уже говорилось, важнейшей целью технологии мобильных агентов является достижение взаимодействия между системами агентов различных производителей. Возможность взаимодействия становится более реальной, если стандартизировать такие общие для всех агентских систем операции как перемещение агента, перемещение класса агента, и управление агентами.

Таким образом, в первую очередь стандартизуются следующие моменты:

- управление агентами
- перемещение агентов
- имена агентов и агентских систем
- типы агентских систем
- синтаксис местоположения

Рассмотрим их более подробно.

Управление агентами

Вопрос стандартизации управления агентами вызывает большой интерес в среде разработчиков систем мобильных агентов. Необходимо предусмотреть возможность управления различными системами мобильных агентов посредством стандартных операций, чтобы облегчить работу администратора подобных систем. Должен быть определен стандартный путь создания агентов из данных для агента классов, операция приостановки потока управления агента, операция восстановления работы приостановленного потока управления и операция завершения работы агента. Определение общих управляющих функций позволит одному администратору управлять системами мобильных агентов разных типов.

Перемещение агентов

Существуют две причины, по которым локальное взаимодействие двух агентов предпочтительней сетевому: количество сетевых транзакций и мониторинг данных. Эффективность локального взаимодействия достигается посредством перемещения агента из локальной агентской системы в удаленную систему.

Если участок сети между двумя агентами обладает низкой пропускной способностью, поиск информации через сеть становится накладным занятием по сравнению с локальным перемещением данных. Например, в случае, если два агента используют технологию RPC для коммуникационных нужд, то передача информации между агентами требует многократных сетевых транзакций.

Мониторинг данных (например, наблюдение за состоянием определенной машины и группы машин) это задача, которая выполняется ежечасно. Предпочтительней послать агента

на платформу, которая предоставляет необходимые данные, чем использовать локального агента, который будет периодически слать запросы к удаленной платформе. Для подобного типа приложений мобильность – ключ к снижению затрат и более эффективному использованию ресурсов.

Имена агентов и агентских систем

Помимо стандартизации операций необходимых для достижения способности к взаимодействию между агентами, следует определить семантику различных параметров агента. Особо следует отметить такие параметры, как имя агента, имя агентской системы и местоположение.

Для применения управляющего воздействия к агентам, необходимо идентифицировать агента, к которому данное воздействие применяется. Поэтому нужно стандартизировать синтаксис имен агентов. Введение подобного синтаксиса позволит получить еще несколько положительных моментов, например, агентская система сможет сразу определить, поддерживает ли она тип входящего агента, а два агента смогут идентифицировать друг друга по именам.

Тип и местоположение агентской системы

Необходимо также стандартизировать синтаксис местоположения, так чтобы агент смог получить информацию о типе удаленной системы, а локальная и удаленная агентская система смогли идентифицировать друг друга. Если удаленная система поддерживает данный тип агента, то можно производить операцию перемещения агента в эту систему.

Также важно присвоить каждому типу агентской системы уникальное имя (присваивается фирмой разработчиком). Гарантия уникальности подобных имен позволит избежать путаницы между различными типами агентских систем.

3.3 Основные понятия

В [1] вводятся понятия следующих сущностей:

Агент

Агент – компьютерная программа, которая выполняется асинхронно соответственно поведению, заложенному в нее конкретным лицом или организацией.

Стационарный агент

Стационарный агент выполняется только на той системе, на которой был запущен. Если агенту требуется получить информацию, находящуюся вне данной системы или необходимо

установить контакт с удаленным агентом, то для взаимодействия обычно используется один из механизмов передачи данных, например метод удаленного вызова процедур (RPC).

Мобильный агент

Мобильный агент не привязан к системе, на которой он был запущен. Он обладает способностью перемещаться посредством сети из одной системы в другую. Это представление является основополагающим для мобильных агентов.

Способность к перемещению позволяет мобильному агенту попасть в систему адресат, содержащую объект, с которым агент должен осуществить взаимодействие. Кроме того, агент может воспользоваться сервисами, предоставляемыми системой адресатом.

Тем не менее, современные распределенные системы ориентированы на коммуникационные потребности стационарных агентов и не отвечают потребностям мобильных агентов. Мобильные агенты обладают большими возможностями и потребностями, чем какая-либо из существующих распределенных систем.

Состояние агента

Во время перемещения агента, его состояние и код транспортируются вместе с ним. В данном контексте, состоянием агента может быть или его рабочее состояние (состояние выполнения) или значения атрибутов агента, определяющие последовательность действий, которые должны быть выполнены после возобновления работы агента в системе адресате. Значения атрибутов агента включают системное состояние, ассоциированное с агентом (т.е. время жизни агента).

Рабочее состояние агента

Рабочее состояние агента, это состояние его выполнения, включая счетчик команд и стековый фрейм.

Полномочия Агента

Полномочия агента представляют персону или организацию, от имени которых выполняется агент. Технологией предоставляется возможность проверки полномочий агента системой адресатом.

Имена агентов

Агентам назначаются имена, которые можно использовать при выполнении управляющих операций и при поиске агента через систему имен. Агенты именуются исходя из их полномочий, уникального идентификатора и типа агентской системы. Идентификатор агента –

уникальное значение в рамках полномочий, определяющее конкретную копию агента. Комбинация полномочий, идентификатора и типа агентской системы является глобально уникальным значением. Поскольку имя агента является глобально уникальным и не может быть изменено, его можно использовать в качестве ключа в операциях, ссылающихся на конкретную копию агента.

Местоположение агента

Местоположением агента является адрес места (см. “Место”). Место находится внутри агентской системы. Следовательно, местоположение агента должно содержать имя агентской системы, внутри которой находится агент и имя места. Надо заметить, что если местоположение не содержит имени места, то адресуемая агентская система выбирает место по умолчанию.

Агентская система

Агентская система – это платформа, которая может создавать, интерпретировать, запускать, перемещать и уничтожать агенты. Также как и агент, агентская система ассоциируется с полномочиями, которые определяют организацию или персону, от имени которых работает система. Агентская система с полномочиями конкретного пользователя, реализует политику безопасности этого пользователя в плане защиты его ресурсов.

Агентская система однозначно идентифицируется именем и адресом. На одной машине могут располагаться несколько агентских систем. Схема агентской системы отображена на рисунке 1.

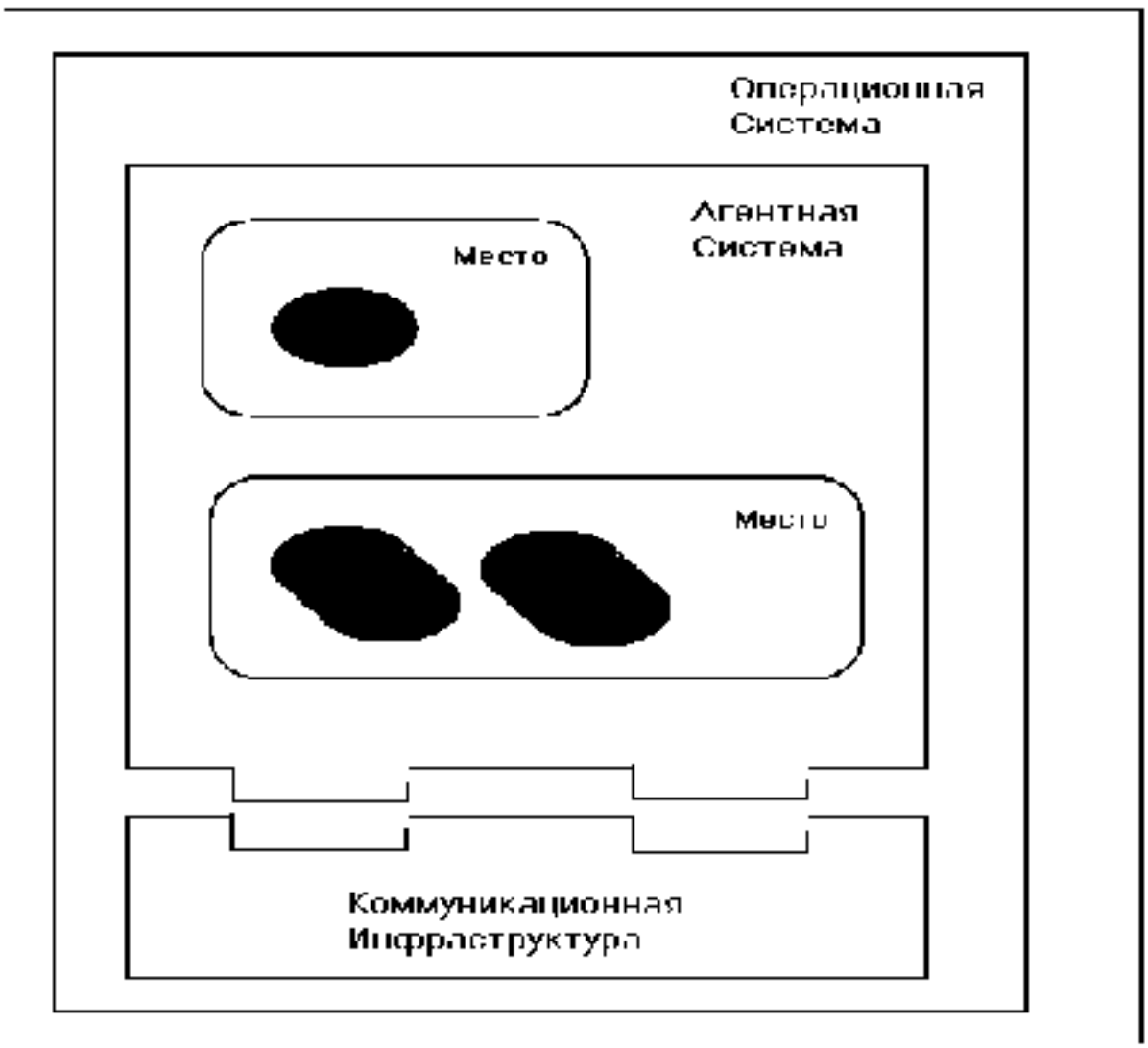

Рисунок 1. *Агентская система.*

Тип агентской системы

Тип агентской системы описывает совокупность параметров агента. Например, если типом агентской системы являются “аглеты”, то это означает, что агентская система создана компанией IBM, поддерживает язык Java в качестве языка реализации агентов, и использует Java Object Serialization для преобразования агентов в последовательную форму.

Связь между агентскими системами

Все общение между агентскими системами происходит через коммуникационную инфраструктуру (КИ). Администратор сетевого региона определяет службы коммуникации для внутрирегионных и межрегионных взаимодействий. Рисунок 2 показывает связи между агентскими системами.

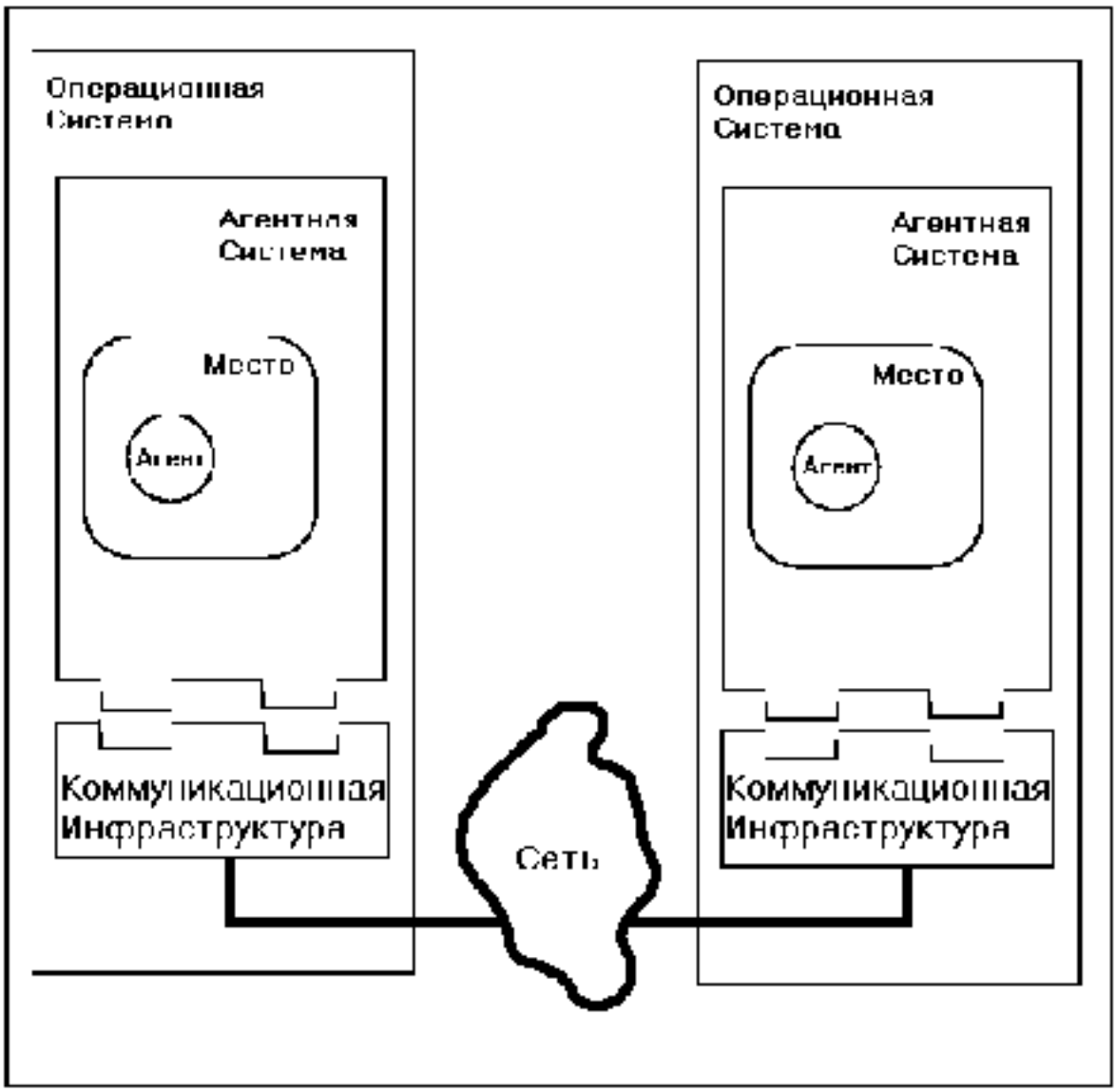

Рисунок 2. *Связь между агентскими системами.*

Место

Когда агент перемещается, он движется между стационарными процессами, называемыми местами. Место – контекст внутри агентской системы, в котором может быть запущен агент. Этот контекст может предоставлять набор функций, таких как контроль доступа, например. Исходное место и адресуемое место могут находиться как в пределах одной агентской системы, так и в различных агентских системах, поддерживающих одинаковую совокупность параметров агента. Место сопоставляется с местоположением, агентская система может содержать одно или несколько мест и место может содержать одного или более одного агентов. Хотя место определяется как окружение, в котором запускается агент, если агентская система не реализует понятие места, то место определяется по умолчанию.

Когда клиент запрашивает местоположение агента, то в качестве ответа он получает адрес места, где выпоняется агент.

Регион

Регион – совокупность агентских систем обладающих общими полномочиями, но не обязательно принадлежащих к одному типу агентских систем. Концепция региона позволяет несколько агентских систем представлять одной персоной или организацией. Регионы позволяют реализовать возможность масштабирования, поскольку можно распределить нагрузку между несколькими агентскими системами.

Регион реализует уровень абстракции для клиентов, связывающихся из других регионов. Клиент, пожелавший установить контакт с агентом или агентской системой, может не знать местоположения агента. Вместо этого, у клиента есть адрес региона (по существу, адрес агентской системы назначенной в качестве точки доступа региона), и имя агента или места. Этой информации достаточно для организации коммуникации с агентом или агентской системой.

Агент может обладать теми же полномочиями, что и регион, в котором он находится и выполняется. Это означает, что агент представляет туже организацию или персону, что и регион. Обычно, настройка региона позволяет наделить подобные агенты большими правами, по сравнению с другими резидентными агентами, обладающими отличными полномочиями. Например, агенту с теми же полномочиями, что и у системы, могут быть назначены административные привилегии.

Регион полностью связывает агентские системы, находящиеся внутри его границ и позволяет осуществлять обмен информации между ними посредством модели сетевого соединения point-to-point. Каждый регион содержит одну или несколько точек доступа, посредством которых регионы соединяются в некоторую форму сети. Архитектура регионов отображена на рисунке 3.

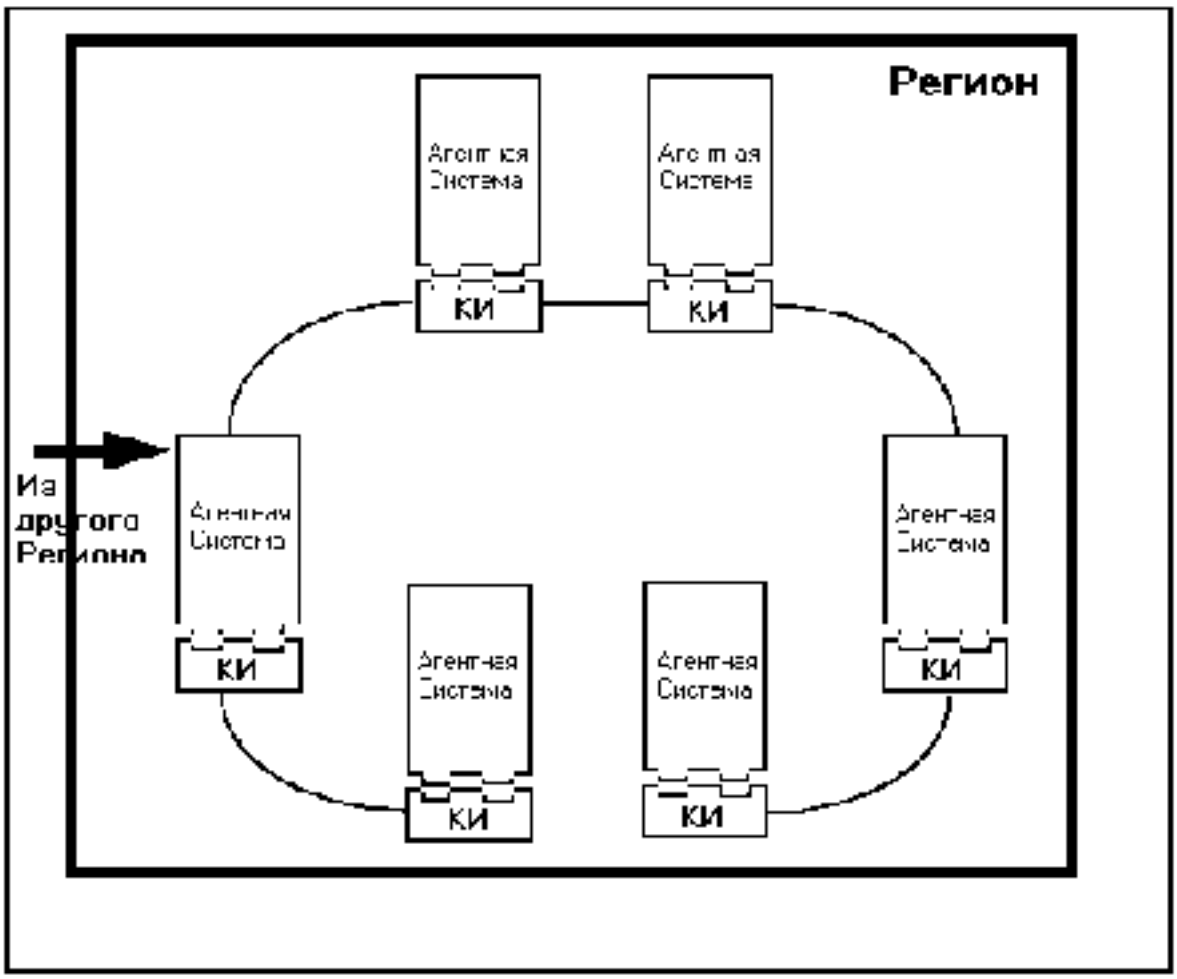

Рисунок 3. *Регионы.*

Связь между регионами

Регионы соединяются посредством одной или более сетей и могут совместно использовать службу имен, организованную на основе соглашений между администраторами регионов и особенностях реализации регионов. Неагентские системы могут также связываться с агентскими системами в переделах региона, если обладают соответствующими разрешениями.

Регион содержит одну или несколько агентских систем. Агентские системы и клиенты вне региона получают доступ к ресурсам внутри региона посредством агентских систем открытых для внешнего мира, подобно варианту с работой брандмауэра. Эти агентские системы определяются как точки доступа к региону.

Агенты наделяются правами доступа соответствующими полномочиям региона, в котором они запущены. Используя это определение, можно рассматривать регион как домен безопасности в нотации CORBA. Рисунок показывает связь между регионами.

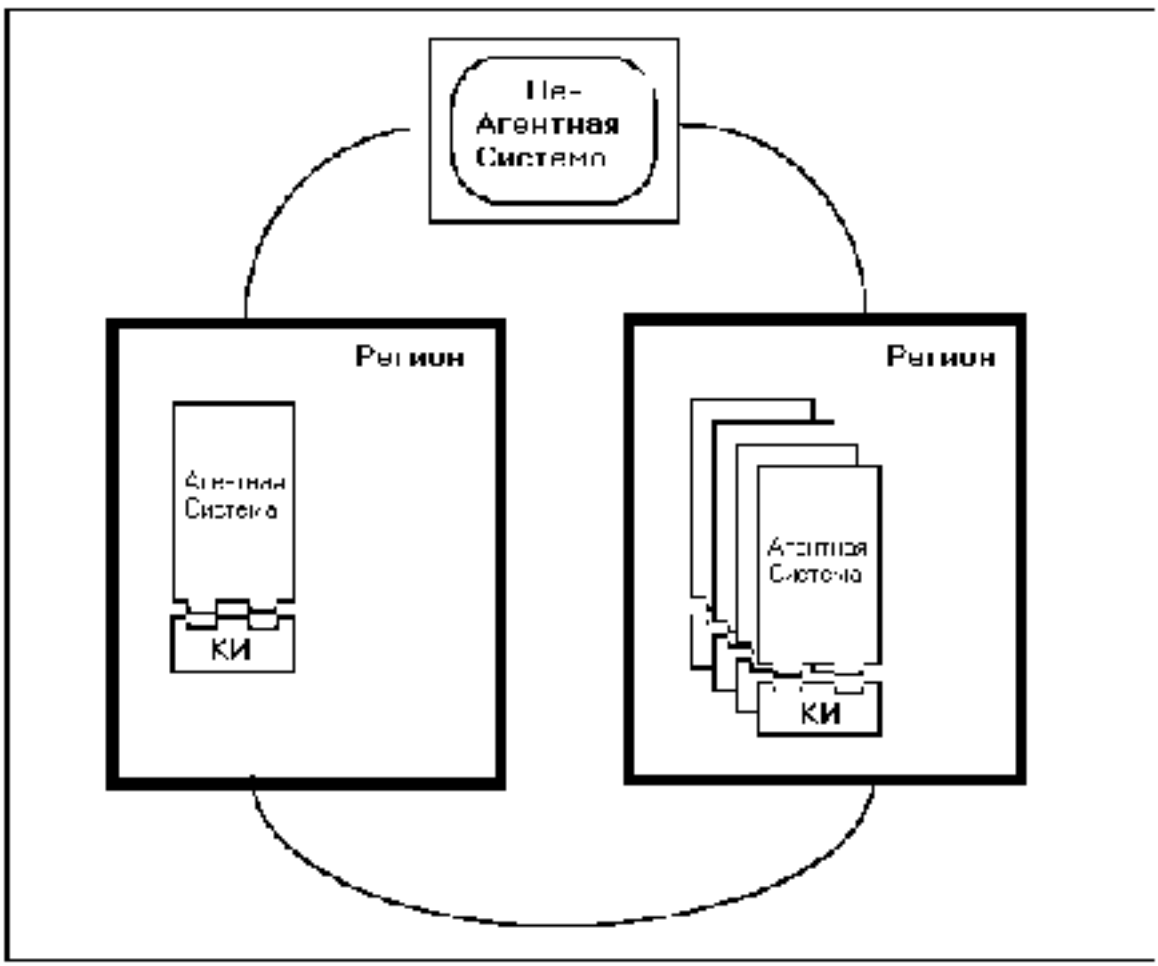

Рисунок 4. *Связь между регионами.*

Сериализация / десериализация

Сериализация – процесс сохранения агента в последовательной двоичной форме. Де сериализация – процесс восстановления агента из последовательной формы.

Ключ к перемещению агента лежит в сохранении его состояния в последовательной форме, достаточной для его последующего воссоздания. Надо заметить, что последовательная форма должна обладать возможностью определения и проверки классов, поля из которых в ней содержатся.

Источник кода

Источник кода определяет местоположение классов используемых агентом. Это может быть как агентская система, так и не распределенный объект, такой как Web-сервер. Если агентская система ответственна за предоставление необходимых классов, источник кода должен содержать достаточный объем информации для нахождения агентской системы. Такая агентская система называется провайдером классов.

Коммуникационная инфраструктура

Коммуникационная инфраструктура обеспечивает транспортные службы связи (например RPC), службу имен и службу безопасности для агентских систем.

Локальность

В случае мобильных агентов, локальность определяется как близость к адресату информации в пределах отдельной машины или сети.

Итак, были рассмотрены базовые понятия, относящиеся ко всем реализациям агентских систем.

4 MAF IDL

Итак, имеется три типа взаимодействия с (между) агентами:

- создание удаленного агента
- перемещение агентов
- вызов метода у агента

В функции агентской системы входит:

- перемещение агентов, который может включать в себя посылку агента, получение агента или пересылку классов
- создание агента
- обеспечение уникального имени каждому агенту и его местоположению
- поддержка сущности Регион
- поиск мобильных агентов
- обеспечение безопасности при выполнении агентами операций

В MAF описывается два интерфейса: MAFAgentSystem и MAFFinder. Интерфейс MAFAgentSystem описывает такие операции с агентами как создание, удаление и управление, а MAFFinder описывает операции для регистрации и поиска агентов, мест и агентских систем.

Интерфейс MAFFinder представляет собой аналог naming-сервиса. Каждый регион имеет свой собственный MAFFinder.

Для получения ссылки на MAFFinder клиент может воспользоваться как стандартными средствами (например CORBA Naming Service), так и методом `AgentSystem.get_MAFFinder()`.

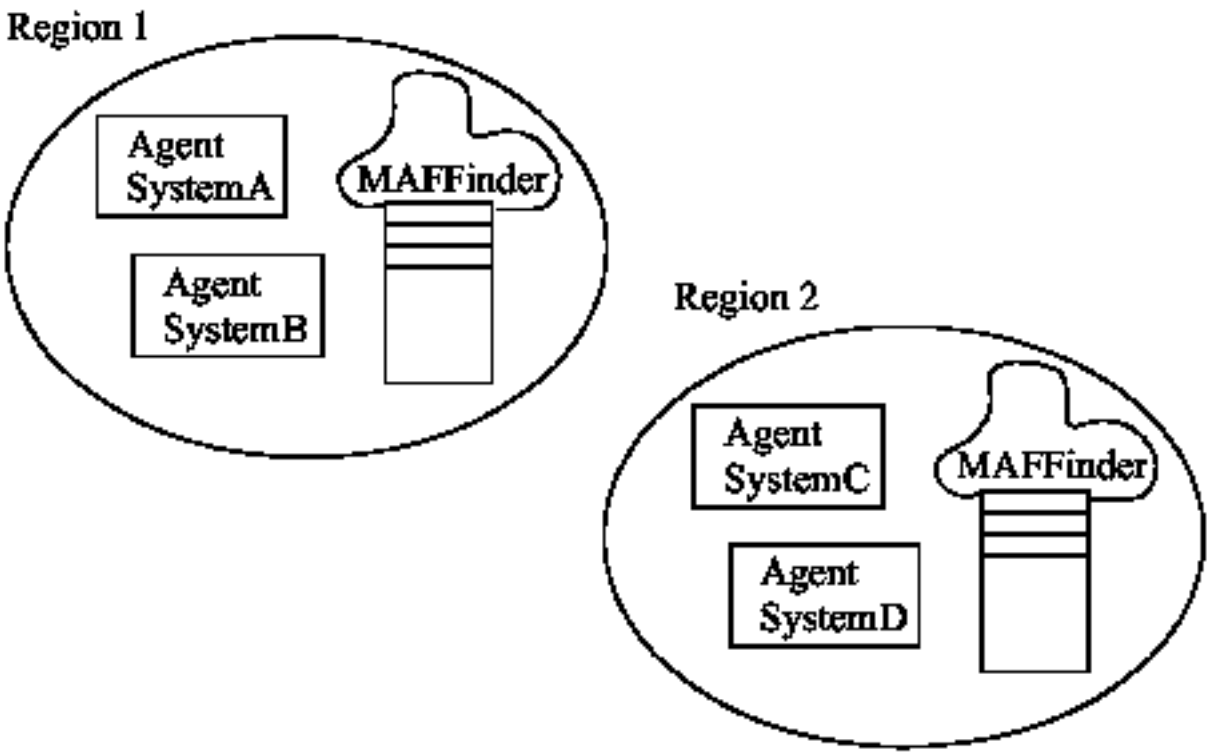

Рисунок 5. *Два региона.*

Name, Class Name и Location

```
typedef short AgentSystemType;

typedef sequence<octet> OctetString;
struct ClassName {
    string name;
    OctetString descriminator;
};
typedef sequence<ClassName> ClassNameList;
typedef sequence<OctetString> OctetStrings;
typedef OctetString Authority;
typedef OctetString Identity;

struct Name {
    Authority auhtority;
    Identity identity;
    AgentSystemType agent_system_type;
};
typedef sequence<Name> NameList;

typedef string Location;
```

OMG Naming Authority Identifiers

```
typedef short LanguageID;
typedef short AgentSystemType;
typedef short Authenticator;
typedef short SerializationID;
typedef sequence<SeriaizationID> SerializationIDList;

enum AgentStatus {
    CfMAFRunning, CfMAFSuspended, CfMAFTerminated
};

typedef any Property;
typedef sequence<Property> PropertyList;

struct LanguageMap {
    LanguageID language_id;
    SerializationIDList serializations;
};
typedef sequence<LanguageMap> LanguageMapList;

struct AgentSystemInfo {
    Name system_name;
    AgentSystemType system_type;
    LanguageMapList language_maps;
    string system_description;
    short major_version;
    short minor_version;
    PropertyList properties;
};

struct AuthInfo {
    boolean is_auth;
    Authenticator authenticator;
};

struct AgentProfile {
    LanguageID language_id;
    AgentSystemtype agent_system_type;
    string agent_system_description;
    short major_version;
    short minor_version;
    SerializationID serializations;
    PropertyList properties;
};
```

Интерфейс MAFAgentSystem

```
interface MAFAgentSystem {
```

```
Name create_agent(
    in Name agent_name,
    in AgentProfile agent_profile,
    in OctetString agent,
    in string place_name,
    in Arguments arguments,
    in ClassNameList class_names,
    in string code_base,
    in MAFAgentSystem class_provider) raises (ClassUnknown,
        ArgumentInvalid, DeserializationFailed, MAFExtendedException);

OctetStrings fetch_class(in ClassNameList class_name_list, in string code_base,
    in AgentProfile agent_profile) raised (ClassUnknown, MAFExtendedException);

Location find_nearby_agent_system_of_profile(in AgentProfile profile)
    raises (EntryNotFound);

AgentStatus get_agent_status(in Name agent_name) raises (AgentNotFound);

AgentSystemInfo get_agent_system_info();

AuthInfo get_authinfo(in Name agent_name) raises (AgentNotFound);

MAFFinder get_MAFFinder() raises (FinderNotFound);

NameList list_all_agents();

NameList list_all_agents_of_authority(in Authority authority);

Locations list_all_places();

void receice_agent(
    in Name agent_name,
    in AgentProfile agent_profile,
    in OctetString agent,
    in string place_name,
    in ClassNameList class_names,
    in string code_base,
    in MAFAgentSystem agent_sender) raises (ClassUnknown,
        DeserializationFailed, MAFExtendedException);

void resume_agent(in Name agent_name) raises (AgentNotFound, ResumeFailed,
        AgentIsRunning);

void suspend_agent(in Name agent_name) raises (AgentNotFound, SuspendFailed,
        AgentIsSuspended);

void terminate_agent(in Name agent_name) raises (AgentNotFound, TerminateFailed);

void terminate_agent_system() raises (TerminationFailed);
};
```

Интерфейс MAFFinder

```
interface MAFFinder {

void register_agent(
    in Name agent_name,
    in Location agent_location,
    in AgentProfile agent_profile) raises (NameInvalid);

void register_agent_system(
    in Name agent_system_name,
    in Location agent_location,
    in AgentSystemInfo agent_system_info) raises (NameInvalid);

void register_place(
    in string place_name,
    in Location place)location) raises (NameInvalid);

Locations lookup_agent(
    in Name agent_name,
    in AgentProfile agent_profile) raises (EntryNotFound);

Locations lookup_agent_system(
    in Name agent_system_name,
    in AgentSystemInfo agent_system_info) raises (EntryNotFound);

Location lookup_place(in string place_name) raises (EntryNotFound);

void unregister_agent(in Name agent_name)  raises (EntryNotFound);

void unregister_agent_system(in Name agent_system_name)  raises (EntryNotFound);

void unregister_place(in string place_name)  raises (EntryNotFound);
};
```

Литература

[1] *CORBA Facilities: Mobile Agents System Interoperability Fascilities Submission*, OMG document 98-03-09.

[2] William T. Cockayne, Michael Zydd, *Mobile Agents*, Manning Publication Co., 1998.

A Существующие реализации агентских систем

Ниже приведен наиболее полный список существующих реализаций систем мобильных агентов, с указанием названия системы, версии и названия организации данную систему разрабатывающей.

Название системы	**Версия**	**Организация**
AgentSpace (UK)	V0.9b3	University of Hull / University of Sunderland
AgentSpace	1.2	Universidade Tйcnica de Lisboa (IST/INESC)
Aglets	1.0.3/1.1 beta 1	BM Tokyo Research Laboratory
Ajanta	Alpha 1.0	University of Minnesota
AMASE	1.1	AMASE Consortium (ACTS project AC346)
AMETAS	1.0	Johann Wolfgang Goethe-Universitдt, VSB
Anchor Toolkit	1.0beta	Lawrence Berkeley National Laboratory
Ara – Agents for Remote Action	1.0a	University of Kaiserslautern
ARCA – Autonomous Remote Cooperating Agents	2.3	Institute of Telecommunications and Computer Science, University of Catania – ITALY
Bee-gent – Bonding and Encapsulation Enhancement aGENT	Beta1.2	TOSHIBA Corp.
Bond	release Nov.99	Computer Sciences Dept.,Purdue Uni-versity
Cborg	3.0	Vrije Universiteit Brussels
Concordia	V1.1.3	Mitsubishi Electric ITA
D'Agents	2.0	Dartmouth College
Dejay	0.9.4	Hamburg University
Evolutionar Agent Societies (EAS)	1.0 Alpha	ANSER
FarGo	0.2.2	Technion – Israel Institute of Technology
FfMAIN	2.0	Fachbereich Informatik (ABVS/Telematik), Johann-Wolfgang-Goethe-Universitaet Frankfurt
Grasshopper	1.2.2	IKV++ GmbH
Gypsy	0.5	Technical University of Vienna
Hive	0.6	MIT Media Lab
IMAJ	alpha release	UNINOVA/FCT + ISEL
J-SEAL2	1.0 Beta 3	CoCo Software Engineering GmbH
JAE – Java Agent Environment	1.2	Aachen University of Technology (RWTH), Computer Science Department (i4)
JAM	0.61+0.79i	Intelligent Reasoning Systems (Company name)
JAMES	v2.0.1	University of Coimbra, Portugal, Siemens S.A, Portugal
JavaNetAgents	0.9	LIP6/ONERA
JavaSeal	1	University of Geneva
JCAFE – Jini Compositional Agent Framework for the Enterprise	1.0	MITRE Corporation

Название системы	**Версия**	**Организация**
Jumping Beans	1	Ad Astra Engineering, Inc.
Kaariboga	1999-12-04	None
Klaim (Kernel Language for agent interaction and mobility)	1.1	University of Florence and University of Pisa
Knowbot Operating Environment	1.0 alpha 2	CNRI
M0 Messengers	0.27.11	Centre Universitaire d'Informatique. University of Ge-neva, Switzerland, Department of Computer Systems, Uppsala University, Sweden
MAgNET	1.0	University of California, Santa Barbara
MAP	2.0	University of Catania – IIT
MATS – Mobile agent teams	Beta	British Telecom
MESSENGERS	2.0	University of California, Irvine
MILLENNIUM	Version 1.0	University of Oviedo
MIPLACE	2.1.2	NEC Corporation
Mobidget	1.1	NEC Corporation
Mogent1.0	1.0	State Key Lab. for Novel Software Technology,NanJing University,P.R.China
Mole	3.0	University of Stuttgart, IPVR
muCode	1.0	Politecnico di Milano
Nomadic Pict	alpha	University of Cambridge, Computer Laboratory
NOMADS	0.3	University of West Florida
Odyssey	None.	General Magic, Inc.
Pathfinder	0.9	National Chung-Hsing University, TAIWAN
Planet	alpha	University of Tsukuba, Japan
Plangent	1.0beta7	TOSHIBA Corp.
rmi64	1.0.0	Technische Universitдt Wien
SeMoA (Secure Mobile Agents)	0.3	Fraunhofer IGD
SOMA	2.0	DEIS – University of Bologna
Telescript	None.	General Magic, Inc.
TuX (Tacoma UniX)	2.0	University of Troms
Voyager ORB Professional	3.1.1	ObjectSpace, Inc.
WASP	2.0	Darmstadt University of Technology, Distributed Systems Group