УДК 658.512.011.056

А.В. Булыгин, В.Ю. Гранин,
О.С. Гурова, М.Л. Тюрина

Архитектура мультиагентной платформы для решения задач структурно-параметрического синтеза объектов. Постановка задачи

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

Проанализированы архитектуры существующих платформ для решения задач структурно-параметрического синтеза. Обосновано использование именно агентной платформы. На основе проведенных исследований предложен вариант применения агентных сетей для решения различного рода задач.

Ключевые слова: интеллектуальная система проектирования, агент, агентная сеть, структурно-парметрический синтез, мультиагентная платформа.

Введение

В индустрии разработки современных средств автоматизации различных процессов большинство разработчиков начинают изучать возможность применения агентных платформ. Причины этого детально описываются в книгах и статьях из библиографического списка, посвященных агентным платформам. Агентному подходу к решению задач уже около 20 лет, но до сих пор нет однозначного определения агента. В некоторой степени это обусловлено тем, что через некоторое время после появления такого подхода разработки замерли и активизировались только в конце 90-х годов [1]. Второй толчок к развитию агентного подхода обусловлен совершенствованием программного обеспечения среднего уровня, которое позволяет строить масштабируемые, относительно надежные распределенные системы [2]. Это позволило разработчикам мыслить более абстрактно, опираясь на уже созданный базис, обеспечивающий прозрачность относительно распределения кода и данных.

В настоящей статье приведено описание существующих достижений в области программного обеспечения среднего уровня (middleware), обеспечивающего механизмы распределения данных и вычислений при построении распределенных систем. Обоснована также необходимость в мультиагентной платформе совсем иного вида, отличного от существующих, – на основе гибких агентов, предложена укрупненная архитектура такого рода агентной платформы.

Подходы к созданию распределенных систем. Требования к распределенным системам

Рассмотрение различных способов распределения вычислений требует предварительного определения распределенной системы и перечисления основных требований к такой системе.

Распределенная система – это набор независимых компонентов, представляющихся их пользователям единой системой [2]. Следовательно

различия между компьютерами в системе и способы связи между ними для пользователя скрыты.

Преимущества использования таких систем следующие:

1. Сокрытие факта наличия распределения приложения.
2. Сокрытие способов связи между компонентами системы.
3. Единообразная работа. Схожие функции выплняемые подсистемами должны представляться пользователю одинаково и иметь один механизм выполнения команд.
4. Расширение и масштабирование.
5. Глобализация и увеличение степени интеграции

Естественно, что данные свойства в реальной жизни нельзя назвать абсолютно верными для конкретных распределенных систем. Например, если мы говорим о расширении и масштабировании, то невозможно сейчас представить себе уровень бесконечного масштабирования или расширения, хотя бы по той причине, что надежность соединения между современными компьютерами далека от идеальной. Кроме того большое влияние имеет географическая распределенность систем. Суть в том, что ожидание ответа от машины, находящейся в текущей локальной сети может занимать доли секунды, тогда как ожидание ответа от машины, находящейся на другой стороне земного шара уже будет в несколько раз дольше. Такая задержка обуславливается не столько ограничением по скорости распространения сигналов, сколько задержками на узлах маршрутизации и коммутации пакетов. Однако, несмотря на то, что достигнуть абсолютности перечисленных свойств невозможно, преимущества распределения вычислений и данных очевидны.

Одним из самых важных свойств таких систем является то, что они способствуют всемирной интеграции и глобализации. Вспомним самую большую распределенную систему управления документами WWW. Благодаря наличию такого рода системы за последние годы образ жизни среднестатистического человека изменился кардинально. На основании этого можно сделать вывод, что дальнейшее развитие информационных технологий направлено в сторону глобализации, интеграции, автоматизации, делегирования обыденных действий человека помощникам – программным агентам.

Развитие стандартов работы с удаленными компонентами прошло несколько стадий, на которых мы наблюдали появление новых технологий

- RPC – удаленный вызов процедур (рис. 1). Такую службу в наши дни имеет каждая операционная система. По своей сути технология RPC предлагает посредством IDL – язык определения интерфейса – абстрагироваться от механизмов удаленного доступа. На основе IDL генерируется интерфейс в нотации того или иного языка программирования, при этом программист просто работает с этим интерфейсом, вызывая методы так, как если бы он вызывал локальные методы. Во время вызова методов происходит упаковка и передача параметров на сервер, на котором эти параметры распаковываются и обрабатываются так, как будто такой метод вызвали локально. Передача ответа от сервера клиенту происходит подобным образом. Идея достаточно проста и в то же время гениальна.

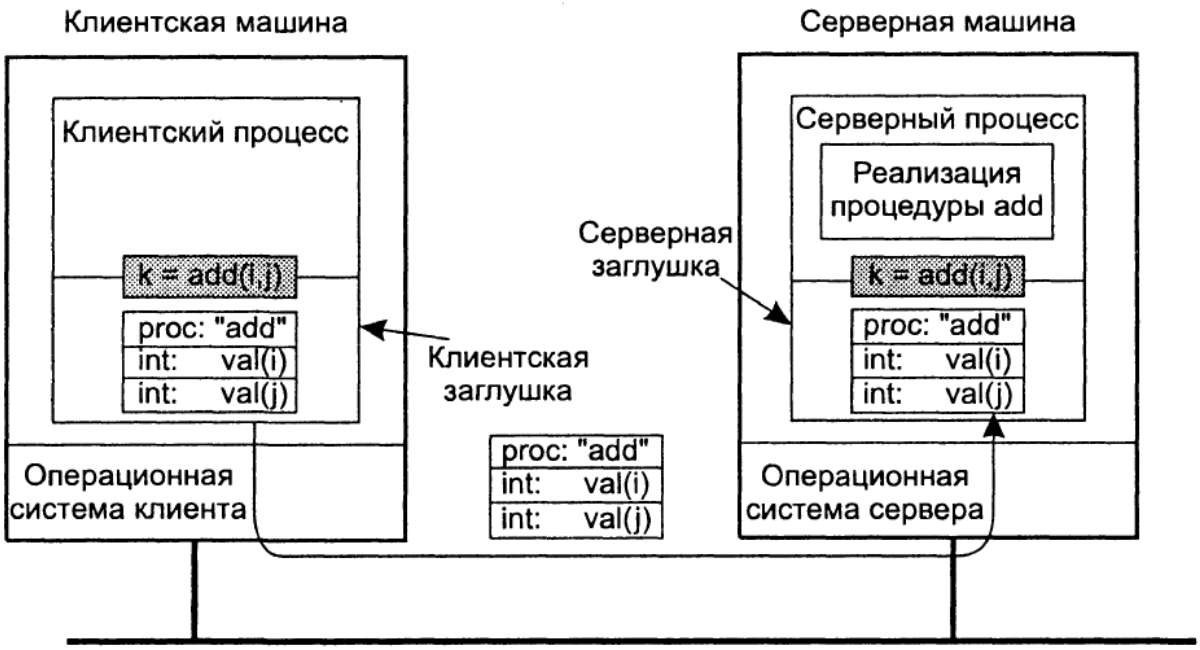

Рис. 1. Механизм реализации RPC

Несмотря на кажущуюся простоту данного подхода, при реализации возникают проблемы, в основном связанные маршалингом и демаршалингом различных типов (упаковка/распаковка значений, представление значений в другом адресном пространстве, на различных архитектурах). Эти проблемы с успехом решаются, и технология RPC заняла свое почетное место.

- Удаленные объекты (рис. 2). Удаленные объекты – идея, дополняющая RPC и в основном реализованная, наверное, благодаря подходам, опробованным в RPC. Суть заключается в том, что объект – сущность, инкапсулирующая в себе свойства (данные) и методы их обработки. При этом в объектах происходит сокрытие свойств, а доступ к ним достигается через внешний интерфейс. Описание таких интерфейсов происходит за счет наличия неких общих стандартов описания. Здесь, например, также используется IDL для описания интерфейсов. Различные реализации этого подхода представляют объекты по-разному. Однако в любом случае клиентская программа может использовать удаленные объекты так, как если бы она использовала локальные объекты.
- Когда клиент выполняет привязку к распределенному объекту, в адресное пространство клиента загружается реализация интерфейса объекта, называемая заместителем. Заместитель аналогичен клиентской заглушке RPC. На серверной стороне входящий запрос на обращение к методу сначала попадает на серверную заглушку, скелетон. Скелетон преобразует его в правильное обращение к методу через интерфейс объекта, находящегося на сервере.

Доступ осуществляется через ссылки на объекты. В общем случае ссылка должна инкапсулировать в себе как минимум локатор ресурса, на котором находится удаленный объект. Разрешение доступа к методам объекта происходит прозрачно для конечного программиста. Существует множество реализаций такого подхода – CORBA, DCOM, Global, .NET Remoting и другие реализации,

которые появлялись в рамках исследовательских проектов того или иного университета.

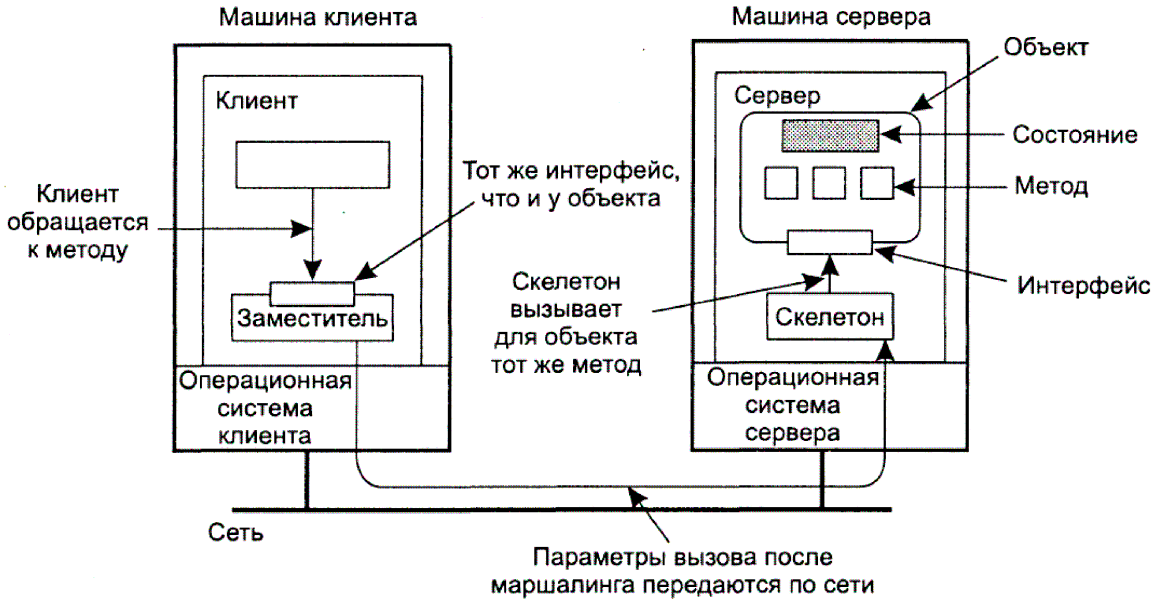

Рис. 2. Механизм работы удаленных объектов

- Агентный подход. Данный подход доступен в архитектуре, определяющейся спецификациями CORBA [2]. Агент, по определению – это самостоятельный, интеллектуальный, постоянный объект, который имеет набор сенсоров и эффекторов [2,5]. В программной реализации агент представляет собой объект, в качестве сенсоров и эффекторов которого являются каналы ввода/вывода. Сутью агентов является их независимость, мобильность, автономность. Это достигается различными способами, и единого подхода не существует. Стандартизация мультиагентных систем занимаются такие организации, как FIPA (Foundation of Intelligent Physical Agent)[7], OMG (Object Management Group)[8]. При этом упомянутые выше спецификации CORBA восходят к модели OMG, которая, в свою очередь, занимается спецификациями мобильных агентов.

Таким образом, можно сделать вывод о том, что агенты позволяют решить сложные вычислительные задачи на основании декомпозиции их на ряд более мелких обособленных задач, закрепленных за каждым агентом, не теряя целостности задачи и обеспечивая тем самым системе большую мобильность, наглядность, а следовательно, и простоту использования.

Мультиагентный подход в системах автоматизации структурно-параметрического синтеза в условиях многоаспектности данных

При решении задач автоматизации принятия проектных решений специалисты сталкиваются со сложными иерархическими структурами, расчет параметров отдельных элементов которых напрямую зависит от состояния смежных элементов. На нижнем уровне каждая из таких структур представляет собой набор простейших элементов, объединяющихся в более сложные объекты.

Программные комплексы, автоматизирующие процессы решения задач параметрического и структурного синтеза, должны скрывать в себе от конечного пользователя сложные структурные связи между элементами и в то же время должны позволять проводить сквозной контроль над процессами расчета и принятия решений. Как показывает практика, агентный подход к построению программных комплексов наделяет системы большей гибкостью, масштабируемостью, распределенностью. Данные свойства систем при решении задач автоматизации структурно-параметрического синтеза являются критически важными.

Рассмотрим преимущества использования агентной сети на *примере решения задачи выбора светильника для авиаприбора в кабине самолета* [7].

Требования к освещенности кабины экипажа достаточно противоречивы, так как пилоту во время ночных полетов необходимо следить как за показаниями приборов внутри кабины, так и за световыми сигналами вне летательного аппарата. Поэтому уровень освещенности отсчетных приспособлений должен быть с одной стороны достаточно высоким и не слишком ярким – с другой. Кроме того, освещенность должна быть достаточно равномерной для уменьшения утомляемости экипажа. Допускается неравномерность в освещении не более 3:1. Хорошие условия освещения значительно ускоряют выполнение работ, повышают производительность труда, способствуют повышению их качества.

Ограничим решение поставленной задачи предъявлением следующих требований:

1. Следует обеспечить неравномерность освещения не более 3:1.
2. Согласно нормам освещенности внутренних объектов, освещенность отсчетных приспособлений приборов должна быть в пределах 4,5 – 13,5 лк при белом освещении и в пределах 2,26 – 6,75 лк при красном освещении.
3. Необходимо чтобы видимость стрелки прибора обеспечивалась без дополнительных светильников.
4. Габаритный размер панели с двумя приборами, расположенными вертикально, не должен превышать некоторой высоты H, расстояние между приборами и до края панели не должно превышать некоторого значения h (рис. 3).
5. Тип приборов должен быть одинаковым (перечень типов см. ниже).

Критерий выбора – минимальная стоимость.

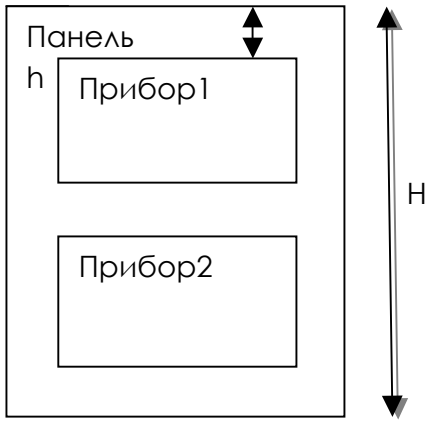

Рис. 3. Взаимное расположение элементов

Для решения поставленной задачи рассмотрим следующую упрощенную агентную сеть: в сеть входят три агента – Панель, Прибор1 и Прибор2 (рис. 3).

Из базы данных, основанной на накопленном опыте проектирования, система выбирает ряд возможных вариантов структуры для Прибора1 и Прибора2 (рис. 4, 5, таблице).

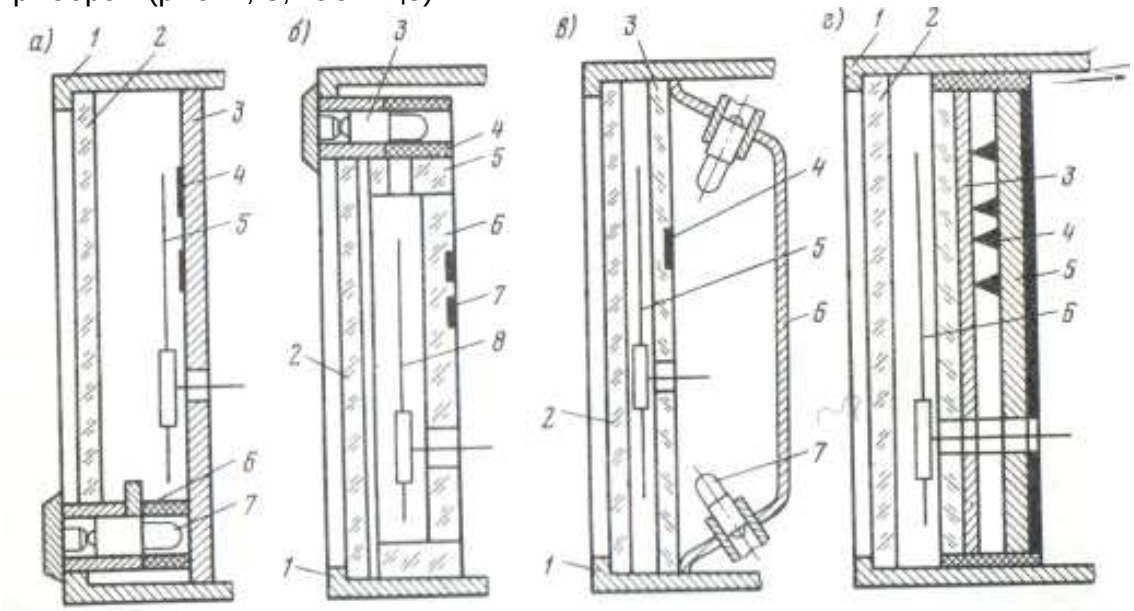

Рис. 5. Светильники, встраиваемые в приборы: а – светильник заливающего света, б – светильник с кольцевым светопроводом, в – светильник с освещением на просвет, г – светящаяся панель

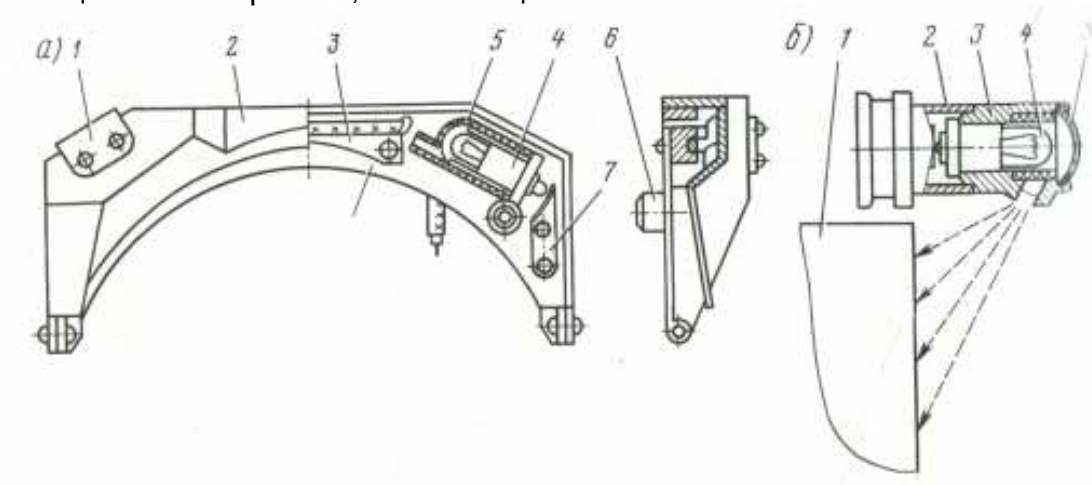

Рис. 4. Щелевой светильник (а) и столбиковый светильник (б)

Выбирая тип прибора, система проверяет выбранную структуру на соответствие требованиям, предъявленным постановке задачи.

Первые два типа светильников не могут быть использованы, так как не соответствуют требованию неравномерности освещенности, которая доходит до 8:1.

Последние два типа нельзя применять, так как для освещения стрелки необходимо иметь дополнительный светильник (условие 3).

Варианты 3 и 4 удовлетворяют условию 3.

Варианты типов светильников

Наименование	Неравномерность освещения	Освещенность, лк	Использовать дополнительные источники света
Щелевой светильник ПС	до 8:1	13	нет
Светильник типа С	до 8:1	12	нет
Светильник заливающего света	до 4:1, зависит от размера светильника	7	нет
Светильник с кольцевым светопроводом	до 3:1	2 - 7, зависит от размеров прибора	нет
Светильник с освещением на просвет	до3:1	10	да
Светящаяся панель	до 3:1	9	да

Удовлетворение условию 1 в светильнике типа 4 зависит от габаритных размеров светильника. Агенты Прибор1 и Прибор2 сообщают сети об изменении своих габаритных размеров, соответствующих необходимым нормам равномерности освещения. Агент Панель реагирует на событие изменения размера приборов, пересчитывает свои размеры и сообщает сети о подтверждении изменения размеров приборов и об изменении своих размеров.

Удовлетворение условию 2 в светильнике типа 3 зависит от габаритных размеров прибора: чем выше освещенность, тем больше размеры прибора. Агенты Прибор1 и Прибор2 устанавливают оптимальное значение освещенности, пересчитывают габаритные размеры и сообщают об этом своей сети. Панель реагирует на событие изменения размеров приборов и пересчитывает свои размеры в связи с изменением размеров приборов. Полученный размер нарушает требование 3. Агент Панель сообщает о неудовлетворительных размерах агентов Прибор1 и Прибор2 и пересчитывает свои размеры как допустимые максимально приближенные к «предложенным» размерам агентов Прибор1 и Прибор2 и сообщает сети о пересчете размеров. Агенты Прибор1 и Прибор2 реагируют на это событие сети и проверяют, удовлетворяет ли соответствующая освещенность требованию 2. Предположим, что освещенность допустимая. Тогда получаем два подходящих варианта. Из них согласно критерию выбора будет выбран вариант с меньшей стоимостью. Если результат отрицательный, тогда проектировщик может изменить требования и повторить решение задачи либо создать свое решение на основании предложенных системой, наиболее близких к оптимальному, существующих вариантов решения.

Следует отметить, что использование агентной сети для решения поставленной задачи позволило распределить выполнение расчетов и определение соответствия поставленным требованиям между конкретными агентами, что обеспечило гибкость системы, а механизм общей среды и событийности позволил избежать установления жестких взаимосвязей агентов и упростил механизм взаимных и цикличных вычислений.

Управление информацией с помощью агентных сетей

В условиях тенденции к глобализации разработки программного обеспечения и доступа к информации следует ожидать, что дальнейшее развитие агентного подхода позволит создать мощнейшую инфраструктуру глобального управления информацией. Пользователи агентных систем будут настолько интегрированы, что в принципе агентный подход к средствам коммуникации в дальнейшем может стать альтернативой непосредственному общению. В первую очередь агенты будут способствовать экономической выгоде производств, не связанных с индустрией разработки программного обеспечения, но являющихся потребителями этой индустрии.

Агент –некая сущность, которая может быть использована различными пользователями, как, например, это происходит с WEB-сервисами, которые предоставляют общий интерфейс доступа к своим вычислительным мощностям или как распределенные объекты (прямые предки автономных агентов). Таким образом достигается многократное использование ранее созданной реализации сервиса. Агенты как минимум должны включать в себя данное свойство. Если агентная индустрия получит должное развитие и поддержку со стороны ведущих корпораций, то разработка программного обеспечения может быть сведена к построению системы из готовых блоков-агентов. При этом сопряжение блоков в идеальном случае не должно требовать от человека знаний конкретных языков программирования, а только знания о структуре той предметной области, которую автоматизирует разрабатываемое программное обеспечение.

Опыт всемирной паутины WWW наметил следующую тенденцию: создателями всемирной паутины постепенно становятся её пользователи. Для создания личных интернет страниц не нужно навыков программирования, знания топологии сетевого взаимодействия, протоколов и среды передачи данных. Информация на таких ресурсах появляется сама по себе, её по собственному желанию отдают конечные пользователи. Естественно, что для создания сложных интернет приложений необходимы не только навыки верстки, но и системные знания, которые отсутствуют у конечного пользователя.

Идея автоматизации создания программ структурно-параметрического синтеза, например, объектов САПР схожа с подходом WEB: необходимо создать инфраструктуру позволяющую решать такого рода задачи и дать пользователю инструмент ее совершенствования. При этом необходимо принять основную парадигму, от которой нужно оттолкнуться в ходе разработки подобного рода системы. В случае WEB-парадигма звучит примерно так: работа ведется исключительно с документами, имеющими стандартизованную структуру описания информации, вся работа проводится с документами; можно привести также пример файловой парадигмы, которая используется, например, в UNIX – все вокруг файлы, и все операции сводятся к операциям чтения и записи данных в файл [2]. Так как мы декларируем агентный подход, то выберем следующего рода парадигму – все вокруг агенты.

Определим принципиальные, специфические для нашего подхода свойства агентов (неспецифические свойства много раз определялись и переопределялись многими авторами [3,4]):

1. Доступ к любому агенту выполняется посредством агентной ссылки. Здесь можно провести аналогию с распределенными объектами CORBA, DCOM, .NET Remoting.

2. Агент открыт к изменениям своей структуры и механизмов. Это значит, что любой желающий может изменить конкретную реализацию агента: добавить свойство, интерфейс, изменить реализацию какого-либо метода. В конечном счете сам агент может проделать это над собой. Таким образом, агент в процессе обучения может не только пополнять свою базу знаний, усложняя свое поведение, но также изменять и сам процесс обработки информации.
3. Отсутствует явно выраженная архитектура клиент-сервер. Каждый агент, если на него разрешается удаленная ссылка является сервером, если же сам агент разрешает удаленную ссылку, то он становится клиентом.

Первое свойство системы очевидно. Доступ к любой сущности, в конечном счете, должно осуществляться через интерфейс. Ссылка определяет канал связи с объектом, через нее осуществляется передача управляющих сообщений.

Второе свойство системы необходимо главным образом для реализации многоаспектности данных. Любое свойство либо метод агента может обладать произвольным количеством атрибутивных описаний. При анализе такого рода описаний конечный пользователь агента может определить то или иное назначение элемента агента. Точно такой подход применяется для описания метаданных в платформе .NET, ещё раньше эти описания появились в интерфейсах COM. Главное отличие нашей архитектуры – динамичность информационной модели агента. Одним важным следствием этого является возможность удаленного конструирования агентов, что при условии применения прозрачных ссылок на удаленные сущности существенно упрощает развертывание распределенных приложений. Разработчик в идеальном случае не должен даже замечать, что он работает в распределенной среде.

Свойство динамичности описания и реализации агента как следствие предполагает мобильность агентов. Например, все описание агента может быть перемещено с одного узла на другой узел, где агент будет развернут в памяти. При этом может быть восстановлено его состояние.

Необходимо кратко объяснить причину возникновения таких требований к агентной платформе. Системы автоматизации создания САПР являются частным случаем систем автоматизации создания «каких-угодно» программных продуктов. Такое свойство предполагает наличие механизмов создания ПО непрограммирующими специалистами предметной области. Для этого необходимо наличие инструментов создания информационной модели объекта проектирования, инструментов создания баз знаний, а также инструментов анализа знаний. Знания могут интерпретироваться, и тогда мы получаем интерпретирующую систему, либо могут компилироваться – и тогда у нас будет компилирующая система, возможно также наличие гибридных систем. Современные CASE-средства позволяют это сделать. Однако создание систем уровня предприятия требует наличия нового свойства – распределенности. Для непрограммирующего специалиста задача создания распределенной системы внутри, например, своего отдела крайне сложна. В этом случае обычно такие системы централизуются. Центральным местом может являться какая-либо объектная или реляционная база данных, которая содержит всю проектную информацию. Данная БД может быть распределенной, но при этом общий подход к системе не меняется. Для того, чтобы расширить возможности системы необходимо, если есть такая возможность, дополнить её новыми модулями.

Обычно такая задача выполняется программистами, которые поддерживают данное ПО.

Агент, обладающий свойством мгновенного изменения своих характеристик, по удаленному запросу может являться тем базисом, на котором будет построена агентная платформа для непрограммистов. Писать программы в такой системе можно, оформляя её в виде UML диаграмм, блок-схем, диаграмм состояний. Связи между элементами могут быть выражены в виде онтологий, например, И-ИЛИ графа [12]. На сегодняшний день существует много способов представления такого рода информации. При сохранении диаграмм агент автоматически подхватывает изменения и модифицирует свою структуру и поведение. Конечно, данный подход имеет большое количество явных и неявных препятствий к реализации, которые в дальнейшем будут рассматриваться более подробно.

Чтобы окончательно определить описанный в статье минимум требований, необходимо решить несколько важных задач:

1. Разработать формат хранения метаинформации об агенте (аналог IDL DCOM).
2. Определить механизм динамического изменения структуры агента.
3. Определить структуру модуля управления агентом.
4. Определить правила разрешения удаленных ссылок внутри агента.
5. Создать архитектуру модуля, который позволит пользователю, пользуясь принципами агентной платформы, а также определенными при решении первых трех задач, динамически определять функции и структуру агента.

В результате получим полнофункциональную систему, позволяющую строить распределенные агентные сети. Для адаптации таких агентных сетей непрограммирующий пользователь должен обладать набором средств, которые дают воможность представлять в понятном, стандартном виде знания, в том числе знания о структуре и функциях проектируемых изделий.

Выводы

1. Проведен краткий обзор существующих архитектур создания распределенных приложений.
2. Приведены аргументы, доказывающие необходимость применения агентного подхода при решении задач структурно-параметрического синтеза в условиях многоаспектности.
3. Выделены основные требования к такой агентной сети и обоснование, почему существующие средства не подходят к решению такого рода задач.
4. Определены также задачи, которые необходимо решить для того, чтобы агентную сеть смог моделировать непрограммирующий пользователь.

Список литературы

1. Иванов А. Что такое интеллектуальные агенты? /А. Иванов // http://aivanoff.blogspot.com/.
2. Таненбаум Э. Распределенные системы. Принципы и парадигмы. /Э. Таненбаум, М. Ван Стеен. – СПб.: Питер, 2003. — 877 с:
3. Рассел С.. – Искусственный интеллект: Современный подход / С. Рассел, П. Норвинг. – М.: Вильямс, 2006. – 1408 с.

4. Тарасов В.Б. Агенты, многоагентные системы, виртуальные сообщества: стратегическое направление в информатике и искусственном интеллекте / В.Б. Тарасов. – 1998. – №2. – С.40-86
5. The Foundation for Intelligent Physical Agents (FIPA) // http://www.fipa.org/
6. The Object Management Group (OMG) // http://www.omg.org/
7. Клейманов Г.Н.. Электрооборудование летательных аппаратов / Г.Н. Клейманов, Н.С. Курбатов, Н.В. Максимов. – М.: Транспорт, 1982 – 280 с.
8. Жмурко С.А. Основные принципы и модели построения многоагентных систем.: Технологический институт Южного федерального университета в г.Таганроге./ С.А. Жмурко // http://pitis.tsure.ru/files34/01.pdf
9. Акимов С.В. Проект, посвященный проблемам автоматизации структурно-параметрического синтеза / С.В. Акимов// http://www.structuralist.narod.ru/index.htm
10. Евгенев Г.Б. Системология инженерных знаний / Г.Б. Евгенев. – М.: Изд-во МГТУ им. Баумана, 2001. – 336 с.
11. Акимов С.В. Мультиагентная модель автоматизации структурно-параметрического синтеза /С.В. Акимов // Системы управления и информационные технологии. – 2005 – № 3 (20). – С. 45-48.
12. Crystaliz: исследование технологий разработки программного обеспечения// http://crystaliz.com/blog/about/.

Рецензент: д-р техн. наук, проф. Е.А. Дружинин, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков.

Поступила в редакцию 19.12.09

Архітектура мультиагентної платформи для вирішення задач структурно-параметричного синтезу об'єктів. Постановка завдання

Проведено аналіз архітектур існуючих платформ для вирішення задач структурно-параметричного синтезу. Обгрунтовано використання саме агентної платформи.У результаті проведених досліджень запропоновано варіант використання агентних мереж для вирішення різних задач.

Ключові слова: інтелектуальна система проектування, агент, агентна мережа, структурно-параметричний синтез, мультиагентна платформа.

Multiagent platform architecture for structure and parametrical synthesis design solution. Problem definition

In article the existing platforms architecture analysis for structure and parametrical synthesis problems decision is passed. The multiagent platforms use is founded. On the basis of the spent researches multiagent networks use variant for different problems decision is offered.

Keywords: intellectual design system, agent, agent network, structure and parametrical synthesis, multiagent platform.