Jet Info

ИНФОРМАЦИОННЫЙ БЮЛЛЕТЕНЬ № 3 (70)/1999

НА ПУТИ К УПРАВЛЯЕМЫМ ИНФОРМАЦИОННЫМ СИСТЕМАМ

КОРПОРАТИВНЫЕ СИСТЕМЫ

На пути к управляемым информационным системам

Максим Столяров,
Илья Трифаленков

СОДЕРЖАНИЕ

1. Введение

Управление информационными системами, как термин, имеет множество определений, в зависимости от того, что считать объектом и функциями управления. Разнообразие мнений и подходов велико, поэтому управление является одной из самых «идейных» областей информационных технологий.

Мы не станем приводить какое-либо определение, но просто постараемся показать, что должно быть осознано теми, кто намерен сделать свою информационную систему (сколь бы сложной она ни была) всегда готовой выполнять свои функции.

Для того, чтобы управлять, надо понимать, как именно выглядит управляемый объект, чем именно следует управлять, какова цель управления.

Известно, что многие администраторы информационных систем (ИС) с трудом могут определить, что же они действительно ожидают получить от управления. Отчасти это происходит потому, что они не знают точно, какая информация доступна о компонентах информационной системы, которой им предстоит управлять и (как крайний, но, к сожалению, не редкий случай) что должна система делать или не делать. Другими словами, они не имеют четкого системного представления о своей ИС.

Имеется также и распространенное мнение, что управлять информационными системами означает лишь контролировать и управлять конфигурациями активных сетевых аппаратных средств типа маршрутизаторов, концентраторов и обрабатывать сигналы ошибок (traps). Ничто выше сетевого уровня модели OSI не рассматривается как подлежащее управлению.

Широкое распространение графических пользовательских интерфейсов в современном компьютерном мире и естественное их проникновение в сферу управления создает ощущение значительного технологического прорыва. То, что имеет дружественный интерфейс, будь это аппаратура или программа, кажется управляемым!

Однако, как бы красиво и дружественно ни было раскрашено управление информационной системой, пользователи всегда четко распознают, когда система становится неуправляемой. К сожалению, они почти никогда не способны помочь администратору в решении проблем.

И вот здесь, когда речь идет о поддержке информационной системы в рабочем состоянии, у разумного администратора появляется мысль, что «интеллектуальный помощник» был бы полезнее «зоопарка красивых павлинов». Неразумный же предлагает тратить десятки и сотни тысяч долларов на аппаратное и программное обеспечение, называемое «Ваше Решение по Управлению», *прежде чем* сформулированы реальные требования к информационной системе и ее функциям[1].

Но что такое «интеллектуальный помощник»? Как его распознать? И дорого ли он стоит?

Информационные системы должны быть приспособлены к деловым требованиям корпорации, поддерживать ее бизнес-потоки. Поэтому если функциональные возможности управления не облегчают прямо или косвенно решение деловых задач, то такую систему управления следует признать полностью бесполезной как для администраторов ИТ-отдела, так и для корпорации в целом.

Современный бизнес требует сокращения стоимости информационной системы, увеличения ее производительности и надежности. Информационная система должна иметь четкую структуру и работать как хорошо отлаженный механизм, но при этом быть адаптируемой к новым, порой неожиданным задачам.

Реальность настоящего дня такова, что дорогостоящее «управляющее решение» не эквивалентно «интеллектуальному помощнику» администратора, несмотря на немалый возраст области управления информационными системами, наличие множества специализированных программных продуктов, управляющих каркасов (frameworks) и интегрированных сред, солидной базы стандартов.

Системы управления *почти* готовы к массовому внедрению, *почти* готовы стать «интеллектуальными помощниками» обычных рядовых администраторов ИТ-отделов, каких (в отличие от выдающихся, талантливых и гениальных) большинство. Можно ли избавиться от этого досадного «почти»?

В этой статье мы обсудим некоторые содержательные вопросы управления с указанной точки зрения.

В первой части обсуждаются основные идеи, которые должны быть заложены в реализации систем управления современными распределенными иерархическими многофункциональными информационными средами. Мы рассмотрим основные свойства информационной системы, делающие ее потенциально управляемой, определим, что мы хотели бы видеть в качестве управляемой системы, сформулируем требования, которым должна удовлетворять ИС, введем понятие каркаса управления и дисциплин управления.

Далее, во второй части, будет описан ряд стандартов управления.

[1] Интересно, что многие производители коммерческих систем управления называют свои продукты «решениями, или ответами». Однако не ясно, в чем заключались «задачи».

В третьей части в качестве примера функциональности современной системы управления рассмотрен программный продукт компании Computer Associates — Unicenter TNG, активно распространяемый конструктор для создания управляющих решений.

И, наконец, предложены ориентировочные критерии оценки полезности систем управления. Это опасная задача обобщения, но, как мы говорили выше, система управления должна помогать в решении бизнес-задач. Она хороша ровно настолько, насколько она это делает. Здесь мнения могут быть весьма противоречивы, но имеются некоторые общие свойства систем управления и информационных систем как объектов управления, которые позволяют ориентироваться в создании или заказе «собственной идеальной системы управления».

2. Распределенная информационная бизнес-среда, или пять Больших Проблем Администратора

Мы начнем рассмотрение проблем управления современными информационными системами с краткого неформального обсуждения их общих свойств. Информационные системы весьма разнообразны по назначению и решаемым задачам. Тем не менее, на некотором уровне абстракции мы легко сможем обнаружить в них одни и те же элементы.

2.1. Элементы информационной системы

В первом приближении информационная система может быть представлена в виде множества следующих взаимодействующих компонентов.

1. Во-первых, это некоторые элементы нижнего уровня, составляющие скелет системы. Обычно, это оборудование и программное обеспечение, его «оживляющее» (компьютеры, каналы связи, операционные системы и т.п.).
2. Далее, это данные и приложения, которые производят с данными некоторые действия. Для того, чтобы данные могли существовать, а приложения работать, необходим скелет.
3. Наконец, сервисы, которые должна предоставлять информационная система. Предоставление определенных, функционально полезных для бизнес-процессов корпорации сервисов является смыслом существования, назначением информационной системы.

Если устройства, данные, приложения могут существовать и работать сами по себе, то полезные для пользователя информационной системы сервисы получаются лишь в результате их (порой весьма тонких) взаимодействий.

Правильно устроенные взаимодействия определяют, по сути, всю функциональность информационной системы. Именно поэтому их наличие и работоспособность столь же важны для «жизни» информационной системы, как и грамотно подобранное оборудование, программное обеспечение, данные и сервисы.

2.2 Управляемые и неуправляемые элементы

Положим интуитивно ясным понятие «управляемый элемент». Будем считать, что это такой элемент, который способен изменять свое состояние некоторым заданным образом в ответ на заданный внешний сигнал, сигнал управления. Устройства, данные, приложения, сервисы должны быть управляемыми, если мы хотим добиться от них координированных действий.

Задача организации взаимодействия между управляемыми элементами с целью предоставления сервиса сложна, но решаема. Истинной же проблемой является способность контролировать состояние сервиса и предоставлять его на заданном уровне качества. Дело в том, что сервис, или функция, информационной системы появляется в результате взаимодействия достаточно сложных информационных объектов, причем все они могут иметь (или имеют) свои, совершенно различные технологии контроля состояния и управления.

Идеально управляемая информационная система должна полностью состоять из управляемых элементов и предоставлять управляемые сервисы. К сожалению, до этого идеала еще очень далеко.

Для того, чтобы объект был управляем, нам необходимы:

1. собственно объект управления (как это ни странно, не всегда даже ясно, что именно требует управления);
2. цели управления;
3. набор управляющих сигналов или управляющая информация.

Это, по сути, составляет формальную модель информационного объекта, на который возложена некоторая задача (функция) по обработке данных. Очевидно, от степени детальности описания самого объекта и связей с другими объектами зависит, насколько точно мы можем воздействовать на объект управления.

Здесь администратор сталкивается с Первой Большой Проблемой, проблемой описания «подопечной» информационной системы не как системы, состоящей из компонентов, а как совокупности сервисов, которые предоставляются пользователю, и их состояний.

На первый взгляд, кажется, будь решены задачи сбора всей доступной информации о состоянии компонентов информационной системы и взаимодействии между ними, а также передачи этой информации на управляющую консоль и ее представления в наглядном виде, идеальное управление будет достигнуто. Теоретически это верно. Практически же собранная «управляющая информация» должна позволить администратору принять некоторое решение. Например, «Все хорошо, можно расслабиться». Или «Все совсем плохо! Но ясно, в чем дело, и как это исправить!».

Задача сбора всевозможной информации о состоянии информационной системы не является сколь-нибудь трудной. Она технически громоздка, но реализуема при наличии специализированного оборудования и программного обеспечения. К сожалению, эта неструктурированная информация не имеет никакой ценности для администратора в силу ее гигантского объема при даже умеренных размерах системы. Эта информация *не представляет* состояния информационной системы и, как следствие, практически бесполезна для принятия решений.

Возникает весьма интересная ситуация. С одной стороны, для управления нам нужна детальная информация о состоянии управляемых объектов, а, с другой, чем этой информации больше, тем менее ясно, что она означает. Если мы не в состоянии охватить и интерпретировать данные о состоянии и поведении некоторого объекта, мы должны признать его неуправляемым.

Мы встретились со Второй Большой Проблемой Администратора — проблемой создания структуры управляющей информации согласно требованиям разумной достаточности, «управляющей информации должно быть ровно столько, сколько достаточно» для предоставления информационной системой заданного множества сервисов.

Таким образом, администратор должен создать модель не только информационной системы как структуры сервисов, но и необходимую и достаточную модель управления.

2.3 «Свои» и «чуждые» элементы

График закупок компонентов информационной системы всегда растянут во времени (см., например, [1], p. 7). В этом смысле от унаследованных систем, не поддерживающих каких-либо нововведений в области управления, избавиться невозможно в принципе. Однако унаследованные системы могут играть (и играют) важную функциональную роль в работе информационной системы просто потому, что они есть и используются. Более того, некоторые части корпоративных ресурсов, которые ранее никогда не осознавались как часть информационной системы, могут внезапно получить такой статус, например, вследствие перестройки бизнес-процессов или объединения информационных ресурсов корпораций в результате слияния.

С учетом этого обстоятельства, задача управления аппаратными и программными компонентами информационной системы является неразрешимой, поскольку многие из них либо попросту не содержат органов управления (морально устаревшее оборудование), либо органы управления слишком «продвинутые» (очень современное оборудование).

Администратор должен изучать документацию, привлекать свой опыт для того, чтобы справляться с каждым таким устройством или программой как с чем-то уникальным (чем они, впрочем, и являются на самом деле). Управление же информационной системой должно (или желательно) строить на основе «стандартных кирпичиков».

Поиск и/или создание средств унификации описания элементов разнородной информационной среды составляет суть Третьей Большой Проблемы Администратора.

2.4. Структурная иерархия элементов

Архитектура современных информационных систем является структурно многоуровневой, причем разные уровни относительно независимы. Эта независимость проявляется, например, в разной длительности жизненного цикла уровней. Множества сервисов и приложения меняются гораздо чаще, чем операционные системы или аппаратные платформы, не говоря уже о кабельной сети.

Вместе с тем, уровни тесно связаны, ибо взаимодействия между элементами различных уровней реализуют набор сервисов, или функций, информационной системы.

На самом нижнем иерархическом уровне (уровне оборудования) можно считать, что информационной моделью управляемого объекта служит специализированная «программа», или агент, который некоторым образом представляет непосредственно элемент информационной системы.

Управление на этом, нижнем уровне означает считывание состояний объекта (мониторинг) и выдачу команд (конфигурирование, запрос), изменяющих его состояние для достижения какой-либо цели.

При движении по иерархическим уровням снизу вверх, на этом же уровне впервые возникает понятие агента-посредника (proxy-agent) как информационной модели унаследованного оборудования или (еще пока) нестандартного оборудования. Агент-посредник является переводчиком с «языка команд» оборудования на «язык команд» системы управления.

С помощью абстракции «агент» на уровне оборудования возникает единообразный способ

описания элементов информационной системы и методов взаимодействия с ними.

На следующем шаге (иерархическом уровне) мы замечаем, что для работы приложения необходима согласованная работа некоторого количества элементов. При этом приложение является само по себе также компонентом информационной системы. Следовательно, разумно потребовать наличие информационной модели приложения, которая бы была способна представлять его текущее состояние. Возникает понятие агента уровня приложения.

Наконец, поднявшись на верхний уровень — уровень предоставляемых сервисов, мы обнаруживаем ту же ситуацию: группы приложений должны функционировать согласованно, чтобы информационная система могла предоставлять сервисы. Естественно потребовать наличие агента сервиса.

Процесс создания модели управления обретает конкретный вид: необходимо разумно подобрать агентов и построить иерархическую схему управляющих взаимодействий между ними.

Управляющие потоки в пределах уровня координируют действия соответствующих объектов, тогда как потоки между уровнями связывают их в единую структуру.

Задав свойства объекта (агент) и методы управления им (управляющая информация), требуется определить, как именно им надо управлять, а это зависит от задачи, которую объект выполняет сам по себе, либо во взаимодействии с другими элементами. Создание адекватных алгоритмов и программирование управления иерархическим объектом есть Четвертая Большая Проблема Администратора.

2.5. Распределенность элементов

Помимо многоуровневого строения современным информационным системам присуща территориальная распределенность. Это означает, что данные, требующие обработки, могут возникать в одних местах информационной системы, обрабатываться в других, а использоваться в третьих.

Преобразование входных данных в выходные (то есть решение прикладных бизнес-задач) должно быть устроено некоторым регулярным образом, чтобы имело смысл вообще говорить про информационную систему как объект обработки информации. Для распределенной системы это означает, что в процесс обработки прикладных данных могут оказаться включенными элементы всех уровней, причем разнесенные по разным частям системы.

Компонент современной информационной системы, подлежащий управлению и решающий некоторую полезную задачу, почти всегда будет распределен по двум «координатам»: структур-

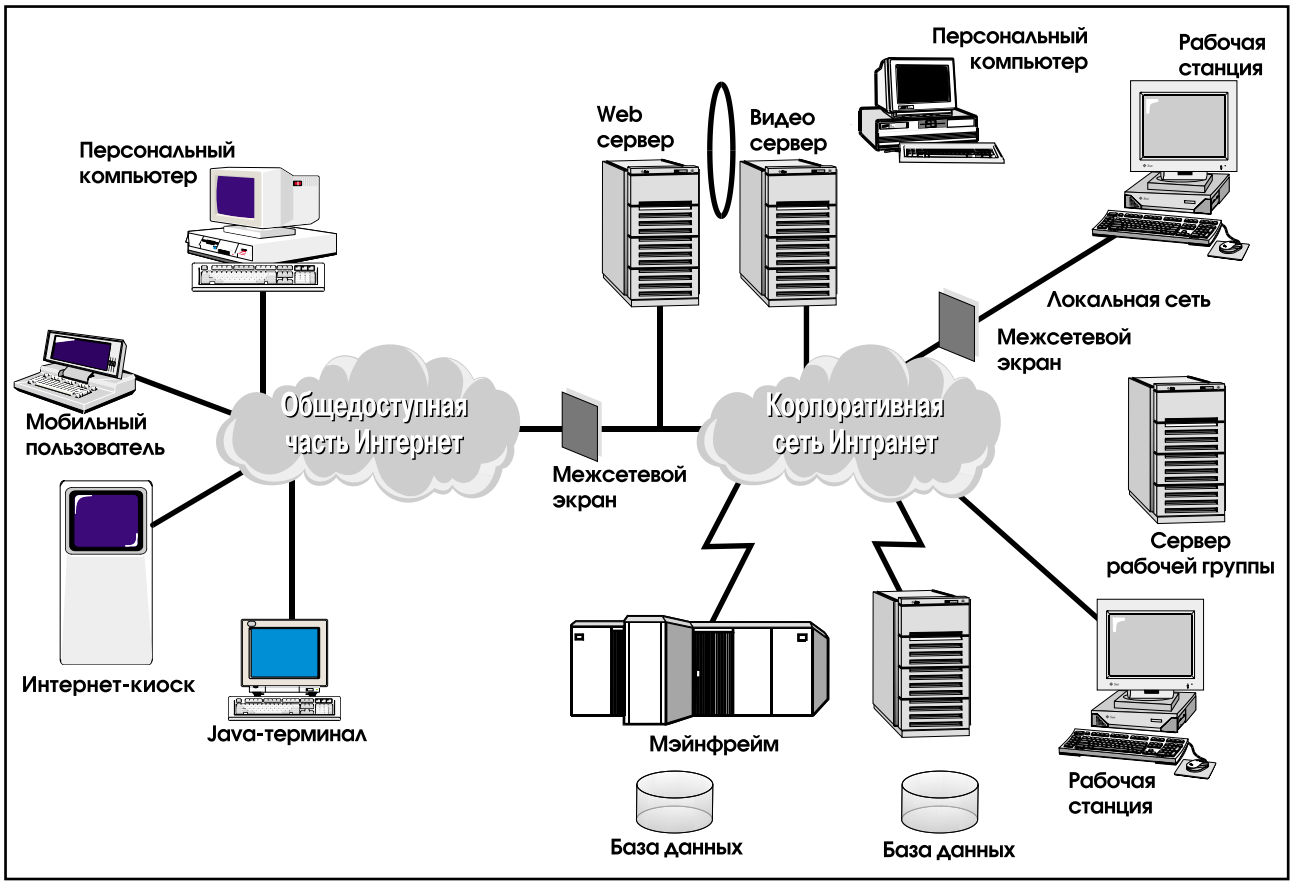

Рис. 1. Современная конфигурация, нуждающаяся в управлении.

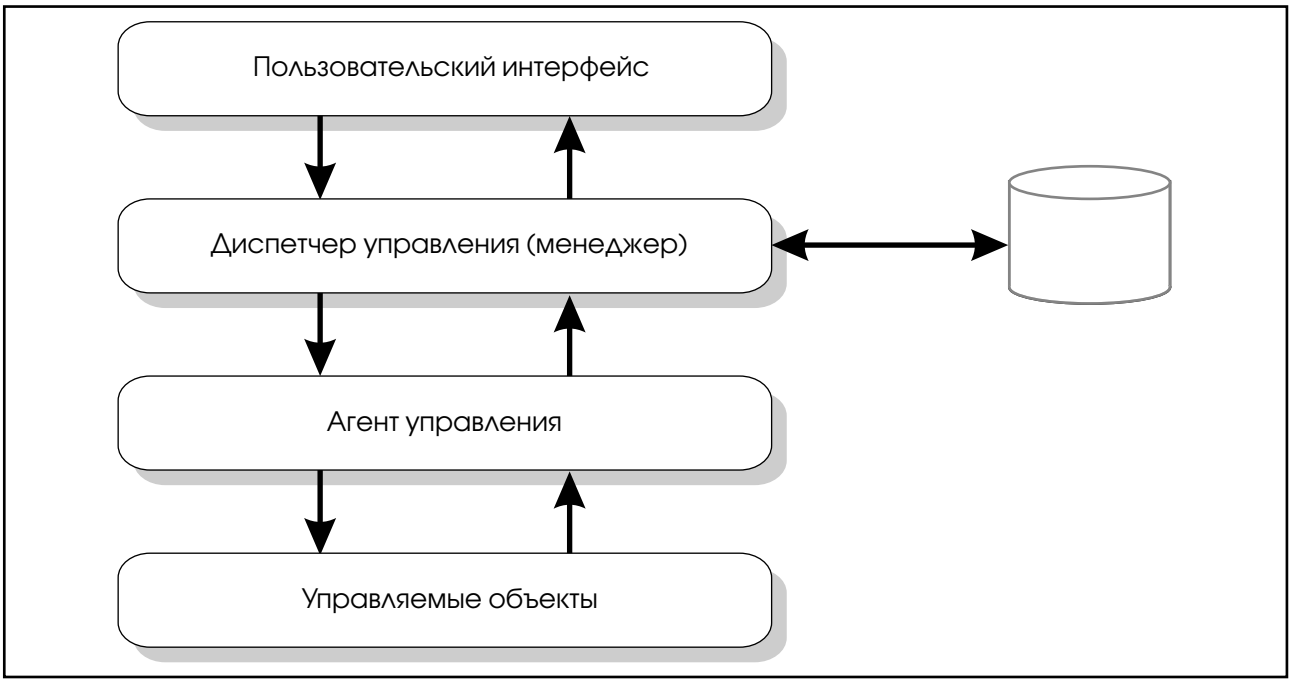

Рис. 2. Четыре уровня функциональных возможностей системы управления.

ные уровни (системы) и организационные уровни (функциональные задачи подразделений корпорации, которые в значительной степени определяют «топологию» информационной системы).

Каждый такой компонент оказывается иерархическим образованием иерархически же устроенных более элементарных составляющих, что качественно осложняет разработку процедур управления и составляет суть Пятой Большой Проблемы Администратора.

2.6. Типичная информационная бизнес-среда

Конфигурации информационных систем, основанные на мэйнфреймах или имеющие простую архитектуру «файл-сервер/клиенты» (и относительно просто формализуемые), остались в прошлом. Теперь приходится управлять иерархической сетью устройств, имеющих не только различные аппаратные и программные платформы, но и существенно разную функциональность и разный «интеллект»: от мощных серверов до клиентов («толстых» и «тонких»), от активного сетевого оборудования до устройств типа банкоматов (см. рис. 1).

То, что процесс превращения подобных распределенных информационных сред во всегда готовый к работе «автомат сервисов» весьма нетривиален, очевидно. Он требует не столько даже хорошего инструментария, сколько высоких интеллектуальных способностей и широких знаний администратора, способного решить Пять Больших Проблем в каждом конкретном случае. Хороших администраторов всегда не хватает, поэтому они всегда «на вес золота» в технологически развитых странах, где своевременно полученный результат есть победа в конкурентной борьбе.

3. Функциональная архитектура системы управления

3.1. Компоненты системы управления

Системы управления информационными ресурсами имеют четыре уровня функциональных возможностей (см. рис. 2). Каждый уровень решает множество задач, определенных так, чтобы обеспечить подготовку данных (сбор, обработка, представление), необходимых для управления элементами, составляющими существо уровня.

Управляемые объекты. Объектами управления являются устройства, системы и/или «что-нибудь еще», что может требовать некоторой формы контроля и управления. Примеры управляемых объектов включают маршрутизаторы, концентраторы, серверы и приложения, подобные СУБД Oracle, Lotus Notes или электронной почте. Важно понимать, что управляемый объект не тождественен фрагменту программных или аппаратных средств, а должен быть описан как некоторая функция, реализуемая информационной средой.

Агент управления. Для того, чтобы как-то взаимодействовать с объектом, необходим некоторый посредник. Маршрутизатор или приложение обычно не имеет «тумблеров». Однако имеется множество разных параметров, которые задают режим функционирования. Роль посредника между объектом и системой управления играет специальный компонент, на-

зываемый агентом. Его назначение — переводить команды системы управления на язык, понятный управляемому объекту, и наоборот.

Диспетчер управления. Агенты управления порождают некоторую информацию о состоянии управляемых объектов. Диспетчер (менеджер) управления собирает, обрабатывает фактическую информацию от отдельных управляемых элементов, при необходимости производя специальные виды обработок (корреляцию событий и т.п.) и определенным образом сохраняет ее для последующего использования.

Интерфейс пользователя. Простейшее использование собранной информации — это ее отображение и предоставление инструментария для работы администратора с ней (интерфейс пользователя). Задача этого интерфейса — обеспечить доступ к накопленной актуальной информации (сигналы тревоги реального времени, важные события, графики состояния системы и ее элементов, анализ тенденций, отчеты и т.д) сотрудникам ИТ-отдела с целью поддержки согласованного, коллективного представления о происходящем в системе. Если этого не сделать, то реальная цель создания (распределенной в общем случае) системы управления будет потеряна. Собранные данные не означают что-либо, до тех пор, пока не будут использованы для принятия обоснованных решений относительно работы информационной системы.

Эти четыре компонента информационной системы образуют то, что можно назвать областями управления, то, на чем система управления должна сосредоточить «свое внимание».

3.2. Место управления в информационной системе

Информационные ресурсы современных корпораций представляют собой *распределенную среду* критически важных данных, приложений и процессов различных типов. Главным вопросом в этих условиях является то, как обеспечить и контролировать состояние сервиса в условиях распределенной среды (клиент/сервер).

В соответствии с этим, задача управления информационной системой должна ставиться как задача управления предоставлением сервисов (обеспечение функциональности) посредством согласованного управления множеством компонентов информационной системы, принадлежащих разным структурным уровням и уровням функциональных задач (см. выше).

Управление информационной системой можно сравнить с нервной системой живого организма, одна из задач которой — обеспечить адаптацию внутреннего состояния к постоянно изменяющейся внешней среде.

Распространено мнение, что управление является хотя и важным, но все же дополнительным компонентом информационных систем. Нам представляется, что такой подход ошибочен. Во-первых, он предполагает, что существуют информационные системы, которые не управляются, что не соответствует действительности (ведь от того, что некоторые действия слишком просты и не воспринимаются как администрирование и управление, они таковыми не перестают быть). Во-вторых, предполагается, что любая информационная система может быть сделана управляемой, что также далеко от истины (см. рис. 3). И, наконец, управление является существенным функциональным интегрирующим компонентом, без которого система, претендующая на звание информационной, остается фрагментарной и быстро приходит в хаотичное и нефункциональное состояние.

Управление должно охватывать все критически важные части информационной системы (Вторая Большая Проблема Администратора) и, как следствие, иметь сильное воздействие на информационную систему. В то же время, оно должно быть незаметным и не мешать предоставлению функционально полезных сервисов.

Такая двойственность отражает тот факт, что управление есть чисто «системный эффект», возникающий в результате взаимодействия отдельных компонентов информационной системы.

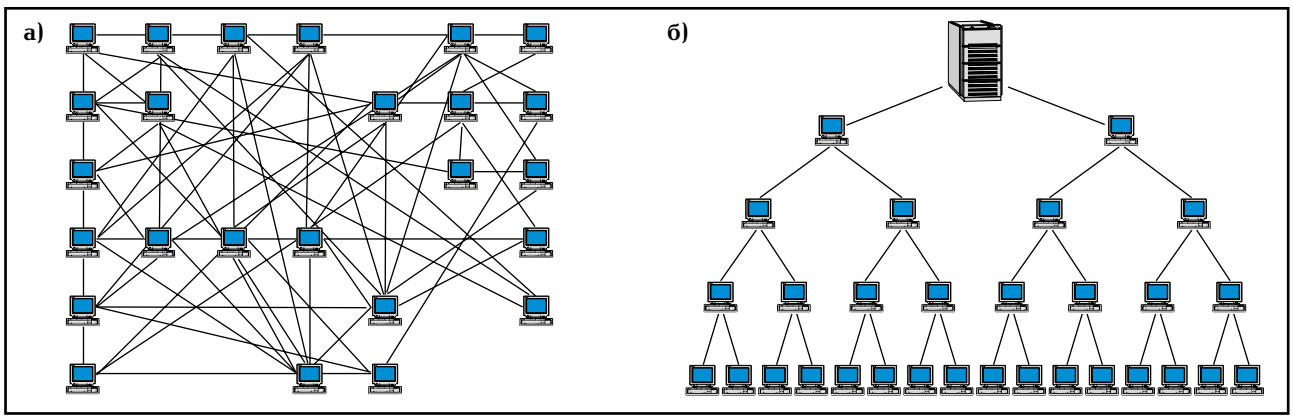

Рис. 3. Слева (а) явно неуправляемая система. Справа (б) — идеал упорядоченности.

Система управления (как мы несколько раз подчеркивали) не имеет дело собственно с компонентами информационной системы. Она работает с информацией, генерируемой «представителями» этих компонентов — агентами, программно или аппаратно реализованными информационными моделями объектов. Создание адекватного агента есть принципиально важная задача, которая предполагается здесь и далее решенной.

Описанная выше иерархия информационных агентов позволяет представлять состояние сервиса распределенной информационной системы в виде, аналогичном состоянию локализованного мэйнфрейм-сервиса.

Действительно, для пользователя безразлично, как именно предоставляются важные для него информационные услуги. Он может считать, что вся магия заключена в его настольной рабочей станции или портативном компьютере, его персональном «мэйнфрейме».

Администратор должен иметь в своем распоряжении обновляемое во времени представление состояния сервиса (информационной услуги) распределенной среды, в виде дерева текущих состояний сервис-образующих компонентов. В случае отказа сервиса имеется в наличии очевидный способ обнаружить вызвавший проблему компонент («прогулка» по дереву) и принять адекватные меры с целью его восстановления.

Насколько детально построено представление состояния компонентов на иерархических уровнях и описаны взаимодействия между уровнями, настолько управляемой будет функциональность информационной системы.

К сожалению, это условие является всего лишь необходимым, но не достаточным.

Таким образом, основные особенности системы управления распределенной информационной средой заключаются в следующем:

1. Она должна быть способна поддерживать иерархию состояний управляемых компонентов и управляющих воздействий.
2. Она должна поддерживать создание на основе этой иерархии представления состояния сервиса (или совокупности взаимосвязанных сервисов, называемой далее сервис-доменом).
3. Она должна сама иметь иерархическую структуру, поскольку является специализированным сервисом информационной системы.
4. Она должна поддерживать иерархическую структуру контрольных точек управляемой информационной системы (как множества сервис-доменов).
5. Она должна поддерживать структурированность управляющих воздействий, выражаемых в политиках управления.

Происхождение первых четырех пунктов обсуждалось выше; они описывают архитектурные основы системы управления. Последний пункт наделяет систему управления реальными полномочиями, превращая ее из несколько абстрактной теоретической конструкции в сурового исполнителя и блюстителя корпоративных информационных законов.

4. Политики управления и способы их проведения

4.1. «Горизонталь» и «вертикаль» управления

Горизонтальное слоение информационной системы (см. рис. 4) соответствует традиционному технологическому делению на приложения, базовые сервисы, системы и сети. Вертикальное — отражает скорее нетехнические аспекты — организационную структуру, территориальное размещение, функциональные задачи и т.п.; в принципе оно позволяет связывать управляемые объекты с бизнес-процессами и, тем самым, помогает найти общий язык управленцам и системным администраторам.

Употребленное нами в предыдущем абзаце осторожное сочетание «в принципе» не является случайным. Между горизонтальной и вертикальной структурой информационных систем есть существенная разница: первую можно поддерживать автоматически, вторую — нет. На практике это оз-

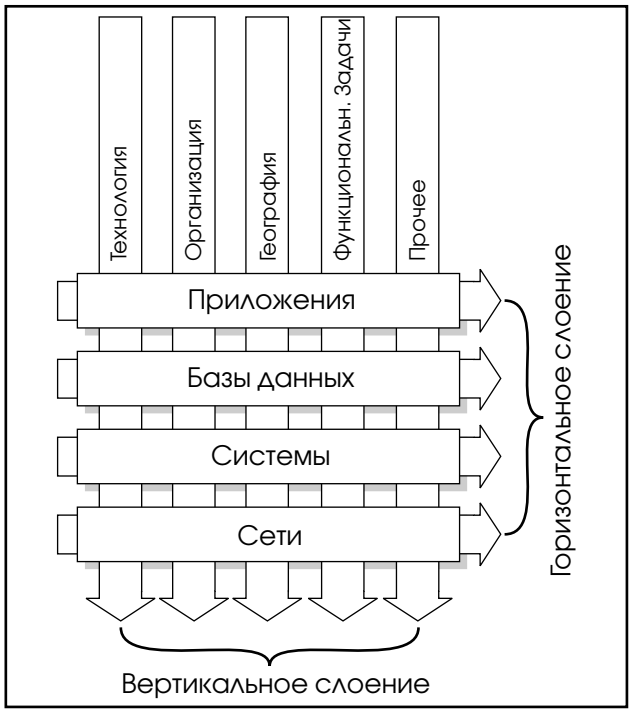

Рис. 4. Сочетание «горизонтального» и «вертикального» слоения управляемых объектов.

начает, что сохранение актуальности вертикальной структуры стоит дорого и чревато ошибками. Для не очень больших, относительно стабильных систем поддерживать ее можно, для по-настоящему корпоративных, динамичных — едва ли. Впрочем, для действительно больших систем в любом случае встает вопрос выбора подходящего уровня детализации представления, так что остается возможность поддержания вертикальной структуры для ресурсов, считающихся ключевыми, а также для важнейших групп более мелких объектов.

Политика (policy) управления представляет собой набор методов и стандартов, направленных на *предоставление* сервиса и поддержку его качества на *заданном уровне*. Политики управления являются своего рода описанием «правил игры», принятых в данной информационной системе. Эти правила регламентируют предоставление сервиса, или, быть может, более точно, правила доступа пользователя к сервис-домену.

В формировании политики управления сервис-доменом важную роль играет не только чисто технологический аспект, но столь же важный вклад дают и внутрикорпоративные стандарты организации бизнес-процессов.

Управление бизнес-процессами не относится непосредственно к сфере компетенции системы управления корпоративной информационной средой. Однако, поскольку основная функция сервис-домена — это поддержка именно бизнес-процесса, эти две системы взаимоотношений оказываются связанными и порой достаточно крепко.

Действительно, сервис-домен по своему происхождению охватывает информационные компоненты нескольких уровней. Это, скажем так, «вертикальное» образование. Сервис-домен можно считать ответом системы управления на потребность бизнес-процесса обеспечить доступ определенной категории пользователей, участвующих в нем, к некоторой совокупности информационных корпоративных ресурсов. Тенденция в создании управленческих подходов всегда бизнес-ориентирована. Такой концептуальный подход соответствует точке зрения бизнес-менеджера и называется, по понятной причине, «вертикальной моделью управления».

Исторически возникшие раньше «точечные» управленческие решения, как правило, компонентно и технологически ориентированы. Они отражают взгляд эксперта по информационным технологиям. Такой подход называется «горизонтальной моделью управления».

Упоминавшиеся выше «горизонтальные» и «вертикальные» потоки управляющей информации тесно связаны с обсуждаемыми концептуальными подходами к управлению.

Подвижность вертикальной структуры достаточно велика, поскольку постоянно должна происходить и происходит перестройка бизнес-процессов (слияние компаний, открытие новых филиалов, формирование новых направлений бизнеса и т.д.). Подвижность горизонтальной структуры также имеет место, однако она гораздо слабее и проявляется, как легко сообразить, в первую очередь на уровне сервисов и приложений. Поэтому неспособность «точечных» решений предлагать быстрые решения на этих «вертикалях» (в частности, из-за отсутствия способности генерировать вертикальные информационные и управляющие сигналы для объектов с групповым управлением) приводят к постоянному нарушению хрупкого баланса интересов двух групп (многочисленной «вертикальной» и малочисленной «горизонтальной») и постоянному естественному разрушению с трудом созданной «управляющей архитектуры».

Кроме того, на фоне ускоряющегося разрастания и усложнения информационных систем, как это часто бывает, ужесточаются требования по безотказности функционирования и удешевлению эксплуатации. Современные компании должны обслуживать своих клиентов 24 часа в сутки, 7 дней в неделю, не располагая штатом специалистов по всем имеющимся аппаратно-программным компонентам, поскольку это слишком дорого. Коротко говоря, следует работать как раньше, только больше, лучше и за меньшие деньги. Подобную постановку вопроса к оригинальным не отнесешь, но сейчас она актуальна, как никогда.

Таким образом, свод политик управления должен отражать общекорпоративный бизнес-интерес (который не выражается в терминах информационных технологий) в виде набора сервисов, направленных на информационную поддержку этого бизнес-интереса.

4.2. Понятие дисциплины управления

Политика управления декларирует, как что-либо должно быть устроено. Политик управления может быть множество. Однако что же является основой, позволяющей реализовывать не только разные виды политик, но и разные их системы в одной и той же информационной системе?

Такой основой является дисциплина управления, под которой следует понимать ту самую специализированную «бесполезную» функциональность информационной системы, предназначенную для поддержки предоставления «полезного» сервиса. Иными словами, дисциплина управления — это то, что остается в информационной системе, когда она не предоставляет никаких сервисов, не делает ничего. Дисциплина управления как специализированный сервис позволяет информационной системе оставаться в состоянии готовности к сервисному конфигурированию (согласно политикам) и работе. Это понятие крайне важно.

Несмотря на то, что политик управления бесконечно много, дисциплины управления можно пересчитать по пальцам. Вот основные:

1. Ведение репозитария политик.
2. Конфигурирование.
3. Мониторинг.
3. Анализ.
4. Управляющие воздействия.
5. Генерация отчетов.
6. Контроль проблемных ситуаций.

Имеются еще некоторые вспомогательные дисциплины (такие как Планирование производительности, Анализ тенденций и Моделирование), вариации (Анализ реального времени и Исторический анализ) и специализированные (Тестирование). Дисциплины управления должны образовывать минимальный набор независимых процедур, достаточный для однозначного описания любой политики управления (полнота дисциплин управления).

Заметим, что у абсолютно сервисно пустой информационной системы есть Репозитарий политик управления, и он хранит политики управления системой управления. Эти политики, по сути, имеют характер внутренних тестов состояния системы управления.

Приведем простой пример. Целесообразно регулярно делать резервную копию диска своей рабочей станции (персональной информационной системы), даже если никакой полезной работой система не занята (в особенности перед установкой нового программного обеспечения). Эта процедура является политикой, не снижающей готовность информационной системы к сервисному конфигурированию и предоставлению сервисов.

4.3. Структура дисциплины управления

Как мы уже знаем, результатом работы дисциплины управления (более точно, полного их набора) является наличие сервиса заданного качества. Поскольку качество сервиса можно, как правило, измерить и оценить количественно, появляется возможность количественно характеризовать и состояние информационной системы, предоставляющей сервисы. Ухудшение качества влечет за собой управляющее (корректирующее) воздействие (согласно политике реагирования), которое должно либо восстановить качество, либо возбудить проблемную ситуацию, которая является (специализированным) запросом на предоставление сервиса и т.д.

Этот сценарий напоминает выполнение некоторой программы, что, впрочем, достаточно точно соответствует действительности. Специальный «процессор» системы управления программируется на реализацию политик управления. Интересно, что основные задачи администратора такой информационной системы заключаются не в написании многочисленных скриптов для анализа файлов протоколов, но в содержательном анализе сервисов информационной системы и написании (в общем случае, сложных параллельных) программ на специализированных языках высокого уровня.

5. Стандарты в области управления

Как мы видели, система управления управляет не информационной системой, а ее моделью. Поэтому необходимо наличие структурированной управляющей информации и моделей управляемых объектов. Стандарты в области управления аккумулируют в себе знания, с которыми необходимо ознакомиться, чтобы уяснить концептуальные подходы. Рассмотрим основные идеи, заложенные в современных стандартах.

5.1. Общая информационная модель CIM группы DMTF

Группа DMTF (Desktop Management Task Force, группа управления настольными системами) разрабатывает Общую Информационную Модель (Common Information Model, CIM), призванную стать концептуальной основой систем управления (см. [9]). Ядро этой модели содержит понятия, общие для всех аспектов управления.

CIM является объектной моделью, активно использующей аппарат наследования, что позволяет справляться с колоссальным количеством и разнообразием управляемых компонентов и их атрибутов. Важной особенностью модели является то, что в число базовых понятий включено отношение между объектами. В результате стали выразимы отношения «состоять из», «реализовывать», «владеть», «зависеть» и т.п., необходимые как с технической точки зрения (без них не описать информационную систему), так и с точки зрения погружения системы в бизнес-среду, которую она (система) должна обслуживать.

В настоящее время (см. [9]) корнем объектной иерархии в CIM является Управляемый Элемент Системы (Managed System Element). На следующем уровне иерархии располагаются физические и логические элементы, далее — логические устройства, системы, сервисы, точки доступа к сервисам и т.д.

Весьма перспективным представляется планируемое наращивание этого дерева «вверх и вбок». Предполагается, что новым корнем станет Управляемый Элемент (Managed Element), в число преемников которого войдут Организация и Пользователь. В результате упомянутое выше по-

гружение ИС в бизнес-среду, уже реализованное в ряде коммерческих систем управления, станет выразимым и в модели CIM.

5.2. Стандарты ISO/OSI каркаса системы управления

Одними из первых с проблемами управления в полном масштабе и общем виде столкнулись организации, занимающиеся эксплуатацией глобальных сетей передачи данных. Соответственно, рекомендации, посвященные управлению, были развиты в рамках модели взаимодействия открытых систем (стандарты ISO/OSI) и в рамках семейства протоколов TCP/IP. В данном разделе мы рассмотрим серию «управляющих» рекомендаций X.700 (практически совпадающих с соответствующими стандартами ISO).

Прежде всего, необходимо отметить, что объектом стандартизации является управляющий каркас (см. [2]), который составляют:

1. сервисы и протоколы, используемые для передачи управляющей информации между открытыми системами;
2. абстрактный синтаксис и семантика информации, передаваемой управляющими протоколами.

Выделение понятия каркаса важно, поскольку системы управления должны быть рассчитаны на взаимодействие с разнородными компонентами. Стандартизация каркаса является основой обеспечения совместимости компонентов, обладающих определенной функциональной и/или предметной ориентацией.

Само управление в трактовке X.700 подразделяется на:

1. мониторинг компонентов;
2. контроль (то есть выдачу и реализацию управляющих воздействий);
3. координацию работы компонентов системы.

Системы управления должны:

1. позволять администраторам планировать, организовывать, контролировать и учитывать использование информационных сервисов;
2. давать возможность отвечать на изменение требований;
3. обеспечивать предсказуемое поведение информационных сервисов;
4. обеспечивать защиту информации.

Иными словами, управление должно обладать достаточно богатой функциональностью, быть результативным, гибким и информационно безопасным.

В X.700 выделяется пять функциональных областей управления:

1. конфигурационное управление (установка параметров для нормального функционирования, запуск и остановка компонентов, сбор информации о текущем состоянии системы, прием извещений о существенных изменениях в условиях функционирования, изменение конфигурации системы);
2. управление отказами (выявление отказов, их изоляция и восстановление работоспособности системы);
3. управление производительностью (сбор и анализ статистической информации, определение производительности системы в штатных и нештатных условиях, изменение режима работы системы);
4. управление безопасностью (проведение в жизнь политики безопасности путем создания, удаления и изменения сервисов и механизмов безопасности, распространения соответствующей информации и реагирования на инциденты);
5. управление учетной информацией (под этим понимается взимание платы за пользование ресурсами).

Безусловно, эта классификация носит фрагментарный и функционально несистематический характер. Внимательный читатель заметит, что указанные области частично пересекаются с описанным выше набором политик управления.

В рекомендациях семейства X.700 описывается модель управления, способная обеспечить достижение поставленных целей. Вводится понятие управляемого объекта как совокупности характеристик компонента системы, важных с точки зрения управления. К таким характеристикам относятся:

1. атрибуты объекта;
2. допустимые операции;
3. извещения, которые объект может генерировать;
4. связи с другими управляемыми объектами.

Рекомендации X.700 намечают объектно-ориентированный подход к управлению. Вместе с тем, модель множества управляемых объектов, составляющих информационную систему, строится на основе необъектно-ориентированной иерархической базы управляющей информации MIB (Management Information Base). Представляется, что главной проблемой является стандартизация логической структуры управляющей информации.

Согласно рекомендациям X.701 [3], системы управления распределенными ИС строятся в архитектуре менеджер/агент. Агент (как программная модель управляемого объекта) выполняет управляющие действия и порождает (при возникновении определенных событий) извещения от его имени. В свою очередь, менеджер выдает агентам команды на управляющие воздействия и получает извещения (см. рис. 5).

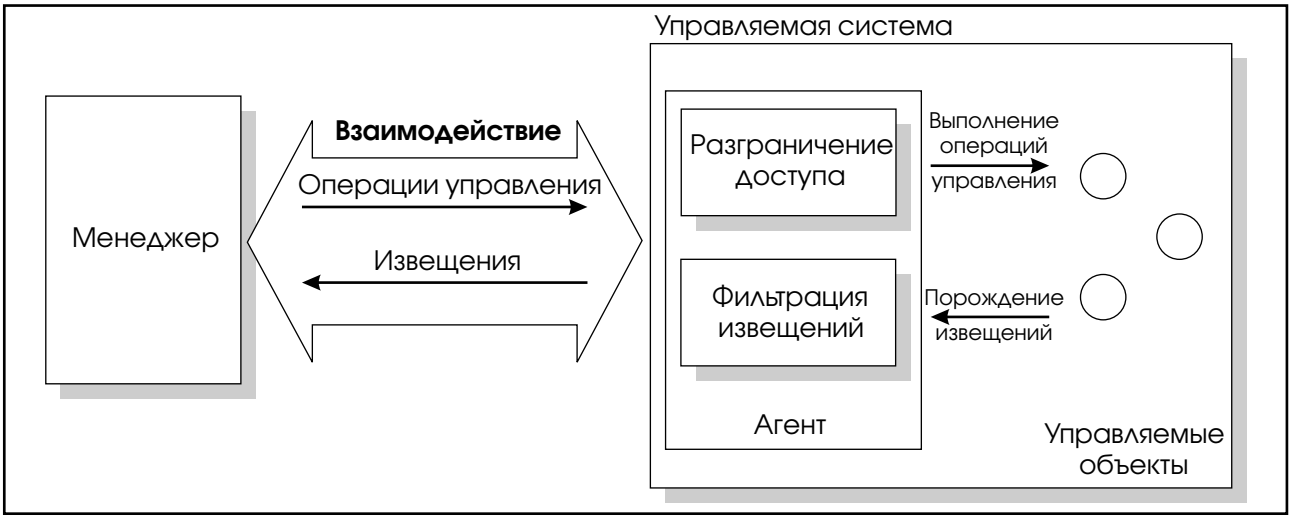

Рис. 5. Схема взаимодействия менеджер-агент.

С самого начала постулируется, что не все менеджеры имеют право на произвольные управляющие воздействия. При рассылке (а также при протоколировании для последующего использования) извещений существует механизм, определяющий адресатов и критерии, которым извещения должны удовлетворять.

Очевидно, иерархия взаимодействующих менеджеров и агентов может иметь несколько уровней. При этом элементы промежуточных уровней играют двоякую роль: по отношению к вышестоящим элементам они являются агентами, а для нижестоящих — менеджерами. Многоуровневая архитектура менеджер/агент — ключ к распределенному, масштабируемому управлению большими системами.

Логически связанной с многоуровневой архитектурой является концепция доверенного (или делегированного) управления. При доверенном управлении менеджер промежуточного уровня может управлять объектами, использующими собственные протоколы, в то время как «наверху» опираются исключительно на стандартные средства.

С точки зрения изучения возможностей систем управления следует учитывать полезное деление, введенное в X.701. Управление подразделяется на следующие аспекты:

1. информационный (атрибуты, операции и извещения управляемых объектов);
2. функциональный (управляющие действия и необходимая для них информация);
3. коммуникационный (обмен управляющей информацией);
4. организационный (разбиение на области управления).

Соответственно группируются и стандарты семейства X.700. На рис. 6 приведены основные стандарты и связи между ними. Целесообразно и в документации на коммерческие продукты выделять аналогичные компоненты.

Как видно из рис. 6, ключевую роль играет модель управляющей информации. Она описывается рекомендациями X.720 [4]. Модель является объектно-ориентированной с поддержкой инкапсуляции и наследования. Дополнительно введено понятие пакета как совокупности атрибутов, операций, извещений и соответствующего им поведения. Класс объектов определяется позицией в дереве наследования, набором включенных пакетов и внешним интерфейсом, то есть видимыми снаружи атрибутами, операциями, извещениями и демонстрируемым поведением.

Помимо концепций, рекомендации семейства X.700 специфицируют две вполне практичные вещи: сервис управления — CMIS (Common Management Information Service) и протокол управления — CMIP (Common Management Information Protocol). CMIS определяет, в частности, следующие базовые операции:

M-GET — запрос на выборку управляющей информации;

M-SET — запрос на модификацию управляющей информации;

M-ACTION — запрос на выполнение действия;

M-CREATE — запрос на создание нового экземпляра управляемого объекта;

M-DELETE — запрос на удаление экземпляра управляемого объекта;

M-CANCEL-GET — запрос на отмену ранее выданного, но ставшего ненужным запроса M-GET.

Прежде чем выполнить операцию над объектом, его необходимо выбрать. CMIS предлагает для этого механизмы областей действия и фильтрации. В соответствии с дисциплиной именования управляемые объекты организуются в виде дерева. Область действия может задаваться как часть некоторого поддерева в общем дереве. Фильтр — это логическая связка утверждений. Таким образом, для операции выбираются все «успешно отфильтрованные» объекты из заданной области действия.

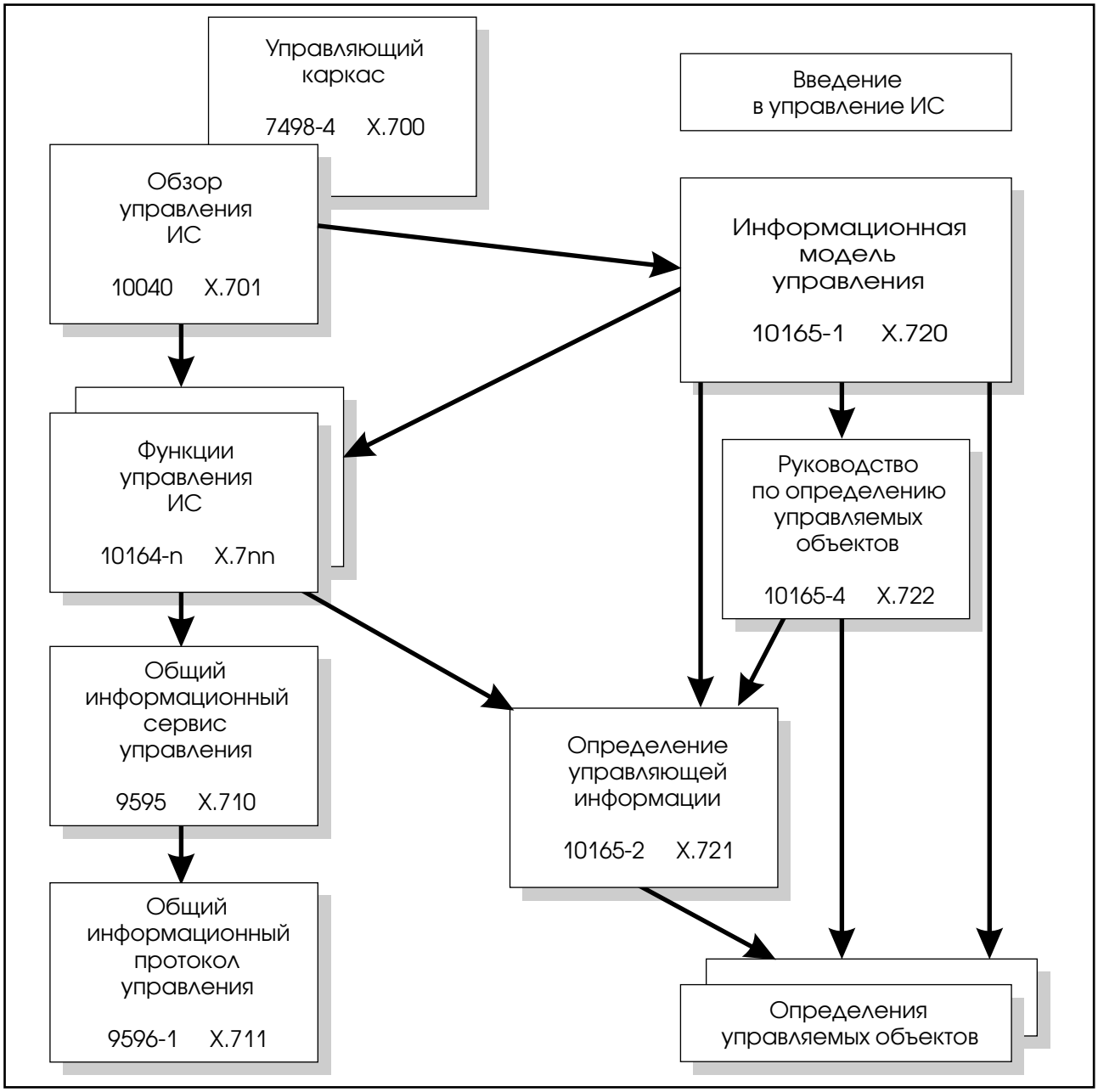

Рис. 6. Основные управляющие стандарты и связи между ними.

Поскольку, вообще говоря, операции являются групповыми, для них вводится режим транзакций (если какие-то действия закончились неудачей, откатывается вся транзакция) или «режим наибольшего благоприятствования» (прилагаются все силы, чтобы действия были успешными, но если где-то случилась неудача, никаких откаток не производится).

Приведенный минимальный набор операций, способов их синхронизации и средств выбора операндов образует базис для развитой функциональности современных систем управления.

5.3. Интернет-стандарты: SNMP-управление

В то время, как стандарты OSI/ISO являются «фундаментальными академическими» разработками и дают некоторое представление как правильно действовать, дела в практической области управления идут своей дорогой. В рамках Тематической группы по технологии Интернет (Internet Engineering Task Force, IETF) был разработан протокол SNMP (Simple Network Management Protocol).

Напомним, как выглядит модель SNMP-управляемой системы. Она состоит из следующих элементов:

- один или несколько управляемых узлов, каждый из которых содержит обрабатывающее приложение, называемое агентом;
- как минимум одна управляющая станция, содержащая один или более обрабатывающих элементов, называемых приложениями управления или менеджерами;

- дополнительно возможно наличие обрабатывающих элементов, способных одновременно выполнять роли агента и менеджера;
- управляющая информация на каждом управляемом узле, которая описывает конфигурацию, состояние, статистику и которая контролирует действия управляемого узла;
- протокол управления, который менеджеры и агенты используют для обмена управляющей информацией.

Эта модель управления универсальна и может быть применена для управления системами, где элементами являются компьютеры (рабочие станции, миникомпьютеры и мэйнфремы), маршрутизаторы, мосты, принтеры, терминал-серверы и т.д.

Именно концептуальная архитектурная простота и ясность модели привела к взрыву на рынке управляющих приложений в конце 80-х, начале 90-х годов[2], когда стали появляться десятки свободно распространяемых и коммерческих программных продуктов управления сетями передачи данных. Насыщение рынка привело к тому, что временное средство оказалось стандартом «de facto», и сейчас идет интенсивная работа уже над третьей версией протокола (SNMPv3, см., например, [5]).

Модель управления, однако, не может ограничиваться только технологией изменения переменных, поэтому она включает еще и описание управляющей информации и событий. Эти описания составляют определение структуры управляющей информации SMI (Structure of Management Information)[3].

И, наконец, третьим компонентом являются описания управляющей информации, событий и связанных с этим требований по реализации, которые специфицированы в документах, называемых MIB (Management Information Base). Тут свой ряд RFC.

Структура управляющей информации (MIB) такова, что в результате ее реального применения для описания и управления современными компонентами информационных систем порождается гигантский объем доступных данных даже в сравнительно небольших информационных системах.

В контексте данной статьи особого внимания заслуживают следующие особенности спецификаций SNMPv3:

1. Модульность. Имеется в виду модульность архитектуры как технических решений, так и спецификаций SNMPv3. Модульность позволяет сочетать в рамках одного решения компоненты от разных поставщиков, производить постепенные модернизации и развивать процесс стандартизации, что гарантирует живучесть SNMP.
2. Масштабируемость. Поддерживаются конфигурации по существу произвольного масштаба, причем стоимость системы управления оказывается пропорциональной этому размеру.
3. Информационная безопасность. Предусматривается защита управляющих сообщений и разграничение доступа к управляющей информации.

На рис. 7 представлены основные компоненты архитектуры SNMPv3. На верхнем уровне таких компонентов два:

1. SNMP-машина;
2. управляющие сервисы.

SNMP-машина присутствует во всех управляемых и управляющих компонентах. Она включает четыре компонента:

1. диспетчер;
2. подсистему обработки сообщений;
3. подсистему безопасности;
4. подсистему разграничения доступа.

Диспетчер занимается приемом и отправкой SNMP-сообщений.

Подсистема обработки сообщений, вообще говоря, поддерживает несколько моделей обработки, соответствующих, например, различным версиям протокола SNMP. Множественность моделей (равно как и унификация их интерфейса с диспетчером) является базовым средством обеспечения живучести спецификаций SNMP, а также систем, построенных на основе этих спецификаций.

В то время, как широко используемая SNMPv1 и менее распространенная SNMPv2 реализации протокола SNMP, по сути, не уделяли особого внимания вопросам информационной безопасности управления[4], спецификации SNMPv3 включают модель безопасности, которая предусматривает меры защиты против следующих возможных угроз:

1. модификация информации в процессе ее передачи;
2. «маскарад» как средство неавторизованного выполнения управляющих действий;
3. модификация потока сообщений (до уровня, превышающего обычные отклонения, возможные при использовании датаграммных транспортных протоколов);
4. несанкционированное ознакомление с сообщениями.

SNMPv3 не предусматривает специальных средств защиты против:

1. атак на доступность;
2. анализа трафика;

[2] Первое описание SNMP версии 1 появилось в 1988 г. (RFC1067) и стало Интернет-стандартом в 1990 г. (RFC1157).

[3] SMI соответствует свой ряд документов RFC.

[4] Протокол Secure SNMP остался в экспериментальной стадии и не используется.

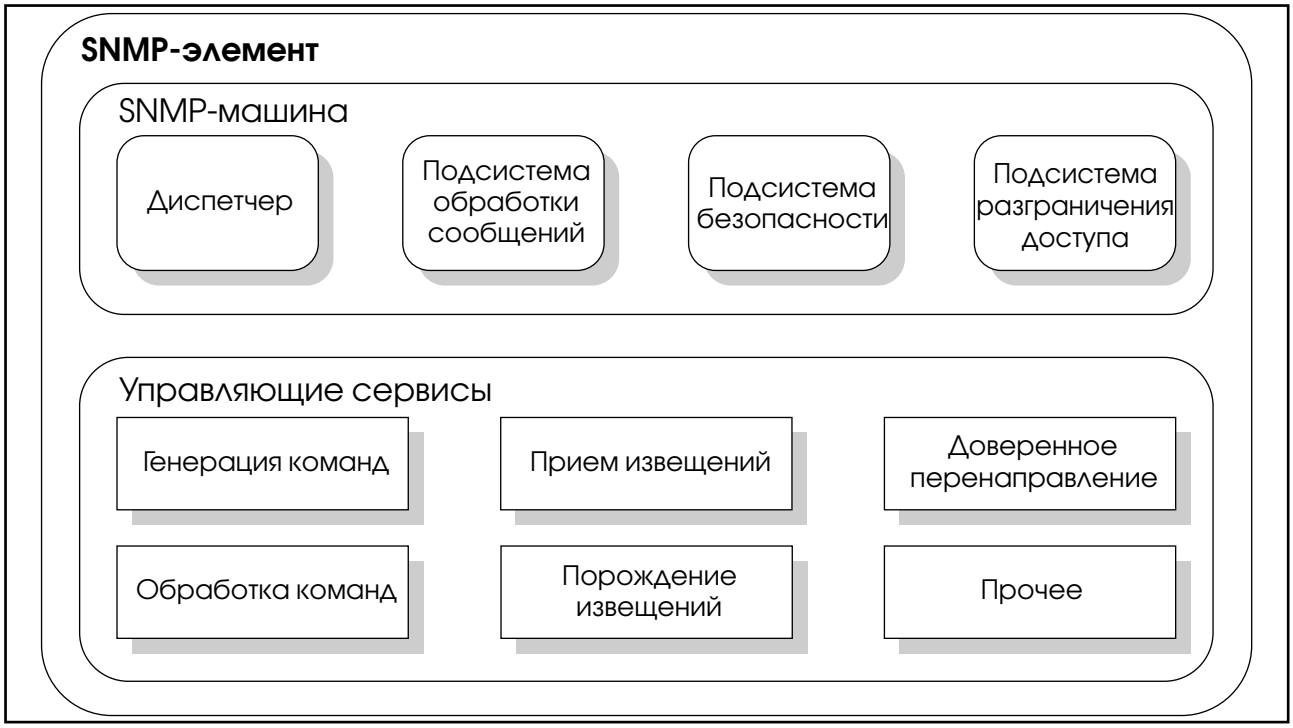

Рис. 7. Основные компоненты архитектуры SNMPv3.

поскольку считается, что во многих случаях проявления атак на доступность неотличимы от отказов в сети, с которыми любой жизнеспособный управляющий протокол должен уметь справляться. Что касается анализа управляющего трафика, то в нем нет большого смысла, поскольку характер подобного трафика является заведомо предсказуемым.

Подсистема безопасности SNMPv3 занимается защитой управляющих сообщений и обеспечивает:

1. контроль целостности сообщений;
2. аутентификацию источника данных;
3. шифрование/дешифрование, если это предусмотрено политикой безопасности.

Кроме того, за счет кэширования сообщений и установления временных границ достигается частичная защита от дублирования, переупорядочивания и кражи сообщений.

Для контроля целостности и аутентификации источника предусматриваются хэш-функции, вычисляемые на основании алгоритмов MD5 и SHA. Стандартным средством шифрования является CBC-DES.

Модель безопасности включает и понятие разграничения доступа к управляющей информации. Эта идея означает выдачу каждому субъекту своего представления (view) данных, а также подмножества управляющей информации, задаваемой спецификациями MIB. Таким образом, субъекты видят только то, к чему имеют доступ. Более того, при выполнении операций чтения, записи и выдачи извещений используются, вообще говоря, разные представления, так что разграничение доступа может быть довольно тонким. Разумеется, соответствие между субъектами и видимыми представлениями устанавливает политика безопасности.

Отметим, что спецификации SNMPv3, в соответствии с концепцией модульности, рассчитаны на одновременную поддержку нескольких моделей безопасности и разграничения доступа. Это, конечно же, гораздо более мощное средство настройки, чем поддержка различных криптографических алгоритмов.

В SNMPv3 предусмотрено пять стандартных управляющих сервисов:

1. генерация команд;
2. прием извещений;
3. обработка команд;
4. порождение извещений;
5. доверенное перенаправление,

которые функционально совпадают с рассмотренными в разделе, посвященном рекомендациям CCITT (см. выше).

В зависимости от поддерживаемых сервисов можно выделить несколько базовых типов программной конфигурации в узлах сети, управляемой по протоколу SNMPv3:

1. минимальный агент (поддерживаются обработка команд и порождение извещений);
2. доверенный агент (поддерживается доверенное перенаправление);
3. минимальный менеджер (генерация команд и прием извещений с возможностью управления из командной строки);
4. менеджер среднего уровня или дуальный узел, поддерживающий приложения ми-

нимального агента и минимального менеджера;

5. управляющая станция (менеджер верхнего уровня), поддерживающая приложения минимального менеджера и, возможно, другие средства, предназначенные для управления большим числом узлов.

Приведенное четкое разграничение функциональности — еще один элемент модульной архитектуры SNMPv3, средство обеспечения простоты этой архитектуры.

5.4. Распределенное управление по IETF

Под распределенным управлением понимается делегирование контроля от одной управляющей станции к другой. Распределенное управление не является новинкой в плане реализации (существует целый ряд коммерческих решений), однако до последнего времени оно находилось вне поля зрения органов по стандартизации. Тематическая группа по технологии Интернет (IETF) решила восполнить данный пробел и в августе 1998 года появился проект [6], в котором описывается каркас для распределенного управления.

Распределенное управление необходимо для достижения следующих целей:

1. Обеспечение масштабируемости до сверхбольших размеров. Распределенность позволяет снизить нагрузку на центральную управляющую станцию и на сетевую магистраль, к которой эта станция подключена. Иными словами, распределенность позволяет расшить узкое место, которым часто оказывается центр управления.
2. Обеспечение управления при отсутствии постоянной связности. Нередкой является ситуация, когда центральная управляющая станция не имеет постоянной связи с некоторыми компонентами информационной системы. Кроме того, отсутствие связи может быть вызвано отказом в сети. Распределенность управления помогает справиться с подобной ситуацией.
3. Отражение организационных границ и бизнес-процессов. Обычно структура управления информационной системой отражает организационную иерархию с ответственными лицами, распределенными по соответствующим подразделениям.
4. Обеспечение модульности архитектуры управления. Такая модульность дает гибкость, возможность строить разные части системы из продуктов разных производителей и т.п.

В соответствии с [6], каркас распределенного управления строится из приложений и сервисов.

Приложения выполняют управляющие функции, такие как автообнаружение, анализ регистрационной информации, слежение за превышением некоторого порога и т.д. Вероятно, со временем часть приложений будет стандартизована, но пока этого не произошло.

Сервисы образуют среду выполнения распределенного управления. Стандартизация сервисов позволяет обеспечить подлинно корпоративные масштабы управления, сэкономить усилия при написании приложений и повысить качество последних.

В [6] описываются следующие сервисы распределенного управления:

1. «Известные системы». Данный сервис предоставляет список всех компонентов, известных системе распределенного управления, а также атрибуты этих компонентов. Список может строиться вручную, с помощью средств автообнаружения или каким-либо иным способом.
2. «Области управления». Данный сервис выдает список всех компонентов, известных системе управления и удовлетворяющих определенным критериям. Сервис позволяет разбить информационную систему на (возможно, перекрывающиеся) области управления, что необходимо, например, для отражения организационной иерархии.
3. «Целевые компоненты управляющих операций». Сервис выдает список всех известных компонентов (возможно, расположенных в нескольких областях управления), к которым имеет смысл применить определенную управляющую операцию. Примером подобного списка может служить набор всех маршрутизаторов в сегменте локальной сети.
4. «Делегирование удостоверения». Сервис позволяет передать распределенному управляющему приложению удостоверение некоторого администратора, чтобы обеспечить выполнение управляющих действий от имени последнего. Отметим, впрочем, что проблема безопасности распределенного управления крайне сложна и до ее решения пока далеко.
5. «Контроль делегирования». Сервис позволяет ограничить количество ресурсов, потребляемых менеджером, которому делегировали управляющие функции. Могут быть лимитированы частоты опросов, посылки широковещательных сообщений и т.п.
6. «Планирование». Сервис дает возможность планировать выполнение распределенных управляющих приложений.
7. «Надежные уведомления». Сервис обеспечивает доставку многокомпонентных извещений и защищает от коммуникационных отказов.
8. «Адресаты извещений». Сервис позволяет конфигурировать набор получателей извещений, что особенно важно в распределенном случае, при делегировании управляющих действий.

Будет очень интересно проследить за развитием процесса стандартизации распределенного управления.

5.5. Расширяемые агенты, RFC 2257

Одному управляемому объекту соответствует один SNMP-агент. В то же время, если этим объектом является устройство, приходится считаться с его внутренней структурой. Устройства состоят из модулей, для каждого из которых определена своя управляющая информация. Более того, эти модули могут динамически добавляться или изыматься, так что от агентов требуется поддержка структурности и расширяемости, прозрачная с точки зрения SNMP-менеджера. Хотелось бы избавить управляющие станции от необходимости иметь дело с многочисленными разновидностями мелких модулей, «инкапсулировав» детали реализации управляемого объекта.

Приведенные соображения побудили авторов спецификаций [7] ввести понятие «расширяемого агента», обладающего внутренней структурой и состоящего из следующих компонентов:

1. главный агент, доступный по стандартному транспортному адресу и обеспечивающий взаимодействие с внешним миром по протоколу SNMP;
2. набор подчиненных агентов, управляющих отдельными модулями.

Для взаимодействия между главным и подчиненными агентами предложен протокол AgentX, напоминающий SNMP, но отличный от него (хотя бы из-за различия решаемых задач).

С внешней точки зрения протокол AgentX невидим; кажется, что расширяемый агент ведет себя так же, как и обычный, «монолитный». AgentX можно трактовать как попытку привнесения в SNMP элементов объектного подхода, правда, в ограниченной, необобщаемой форме.

5.6. Управление настольными системами

Рекомендации CCITT и спецификации Интернет-сообщества описывают инфраструктуру управления распределенными конфигурациями. Тематическая группа управления настольными системами (Desktop Management Task Force, DMTF), в которую входят представители таких компаний, как Sun, Intel, Computers Associates, Tivoli и др., в соответствии со своим названием сосредоточилась на вопросах управления (локального и удаленного) настольными системами и их физическими и логическими компонентами — платами, периферийными устройствами, операционными системами и приложениями.

С точки зрения DMTF, протоколы CMIP и SNMP являются протоколами нижнего уровня; управление настольными системами может использовать их, но не должно от них зависеть. Предлагаемые программные интерфейсы скрывают детали взаимодействия менеджеров и агентов; на первый план выходят архитектурные вопросы и формат управляющей информации.

Основным компонентом спецификаций DMTF (см., например, [8]) является Управляющий интерфейс настольных систем (Desktop Management Interface, DMI). В рамках DMI выделяется сервисный уровень (в более поздней версии спецификаций — поставщик услуг DMI), под которым располагаются управляемые компоненты, а над ним — управляющие приложения (см. рис. 8). Сервисный уровень реализуется на самой управляемой системе как резидентный фоновый процесс; управляющие приложения, вообще говоря, общаются с ним посредством механизма удаленного вызова процедур (Remote Procedure Call, RPC). Отметим, что в качестве управляющего приложения в смысле DMI может выступать SNMP-агент системы сетевого управления (или главный агент в терминологии предыдущего пункта). В результате появляется возможность естественной интеграции DMI-управления с системами, основанными на других стандартах.

Очевидно, в архитектуре DMI можно выделить две группы интерфейсов:

1. между управляющими приложениями и сервисным уровнем;
2. между сервисным уровнем и управляемыми компонентами.

Первый называется управляющим интерфейсом (Management Interface, MI), второй — компонентным интерфейсом (Component Interface, CI).

Таким образом, сервисный уровень в модели DMI служит связующим звеном между управляющими приложениями и компонентами, звеном, унифицирующим взаимодействие и предоставляющим удобный программный интерфейс, за счет чего достигается первая из поставленных DMTF целей.

Еще одна функция сервисного уровня — поддержание локальной базы данных в формате управляющей информации (Management Information Format, MIF). Каждому управляемому компоненту в этой базе соответствует свой файл. В настоящее время стандартизованы MIF-файлы практически для всех мыслимых компонентов — от плат до приложений. MIF — средство достижения второй цели DMTF.

Идея выделения сервисного уровня, разумеется, не нова. При желании можно усмотреть много общего в архитектуре DMI и, например, ODBC. Отметим также, что использование удаленного вызова процедур вместо более примитивных протоколов транспортного уровня делает спецификации DMTF вполне современными. По сути

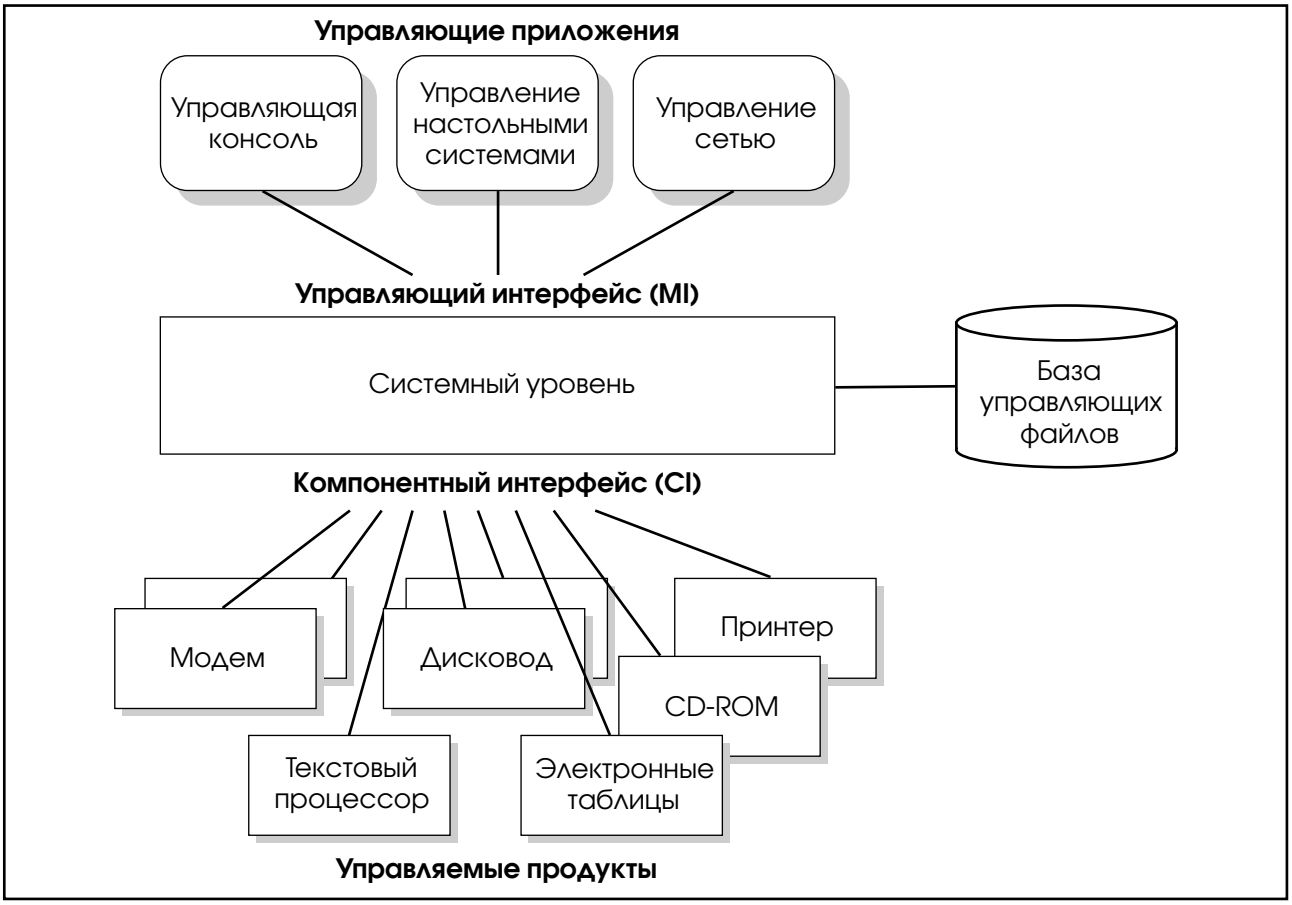

Рис. 8. Архитектура управления настольными системами.

бесплатно достаются средства RPC, обеспечивающие аутентификацию и конфиденциальность. Правда, проблема разграничения доступа в рамках DMI пока остается нерешенной.

5.7. Управление приложениями

Компания Tivoli Systems занимает ведущие позиции в области управления приложениями. По этой причине спецификации Application Management Specification (AMS, см. [10]), предлагаемые Tivoli, заслуживают внимания не меньшего, чем, например, стандарты DMTF.

Компания Tivoli Systems придерживается, если можно так выразиться, «аппликативно-центрического» подхода, полагая, что управление ресурсами является производным от управления приложениями. Некоторые основания для подобной точки зрения имеются, поскольку пользователям информационных систем нужна именно прикладная функциональность; все остальное — это просто «приложения к приложениям».

Основная идея AMS проста. Необходимо предоставить системе управления достаточную информацию о приложении, причем эта информация может иметь как статический, так и процедурный вид.

Важно отметить, что в спецификациях AMS приложения рассматриваются в привязке к решаемым бизнес-задачам. Вводится понятие бизнес-системы, под которой понимается совокупность приложений, ресурсов и связей между ними, обслуживающих определенную бизнес-функцию.

Важно и то, что спецификации AMS направлены на поддержание всех этапов и сторон жизненного цикла приложения, в число которых входят:

1. описание топологии приложения;
2. распространение программного обеспечения (ПО);
3. установка ПО;
4. проверка зависимостей;
5. отслеживание работы приложения;
6. конфигурирование приложения;
7. оперативное управление;
8. внесение изменений, установка новых версий;
9. управление безопасностью;
10. управление бизнес-системами;
11. управление временем отклика приложений.

С точки зрения AMS, архитектура управляемых систем строится с использованием шести базовых блоков (см. рис. 9):

Приложение. Под приложением понимается группа программных компонентов. Чтобы определить приложение, необходимо задать входящие в его состав компоненты, а также специфицировать некоторые общие харак-

теристики, относящиеся к безопасности или времени отклика.

Программный компонент. Функционально законченный компонент приложения, работающий на определенной аппаратно-программной платформе.

Бизнес-система. Это понятие было определено нами выше.

Компонент бизнес-системы. Он описывает роль, которую определенный программный компонент играет в бизнес-системе. Например, сервер СУБД может использоваться для операций с банковскими счетами или для резервирования авиабилетов.

Отображение бизнес-системы — отображение компонент бизнес-системы на программные компоненты.

Бизнес-подсистема служит для группирования компонентов бизнес-системы.

Спецификации AMS опираются на документы, подготовленные группой DMTF. Базовые элементы являются классами в смысле Общей информационной модели (CIM). Соответственно, в терминах CIM описываются характеристики и взаимосвязи этих элементов. Для задания статической информации используется MIF-формат.

Некоторые действия, такие как установка, проверка зависимостей и оперативное управление задаются командными процедурами, входящими в комплект поставки приложения. Отметим, что в основной своей части спецификации AMS устроены таким образом, что они не требуют изменения исходных текстов приложения. Управляющая информация может быть просто добавлена, причем сделать это могут как разработчик, так и третья сторона. Пожалуй, только для измерения времени отклика используется программный интерфейс (разумеется, специфицированный в рамках AMS), поддержка которого должна быть встроена в приложение.

Трудно сказать, как сложится коммерческая судьба спецификаций AMS, но идеи, заложенные в них, представляются весьма перспективными.

5.8. Управляющий интерфейс Java

В условиях разнородности и распределенности современных информационных систем буквально напрашивается применение Java-технологии для реализации инфраструктуры управления. В пользу такого выбора говорят следующие соображения:

- доступность виртуальной Java-машины практически на всех платформах — от потребительских до корпоративных;
- объектная ориентация Java, позволяющая удерживать сложность управляемых систем в разумных рамках;
- наличие готовых средств для компонентного программирования (JavaBeans);
- развитые средства обеспечения информационной безопасности;
- высокая мобильность Java-приложений и аплетов — «пишешь однажды, используешь везде»;
- масштабируемость Java-приложений как вниз (до встроенных систем), так и вверх (до серверных комплексов);
- наличие отработанного механизма для связи с базами данных — JDBC;
- наличие механизма удаленного вызова методов (Java RMI);
- возможность централизованного хранения и динамической загрузки приложений и апле-

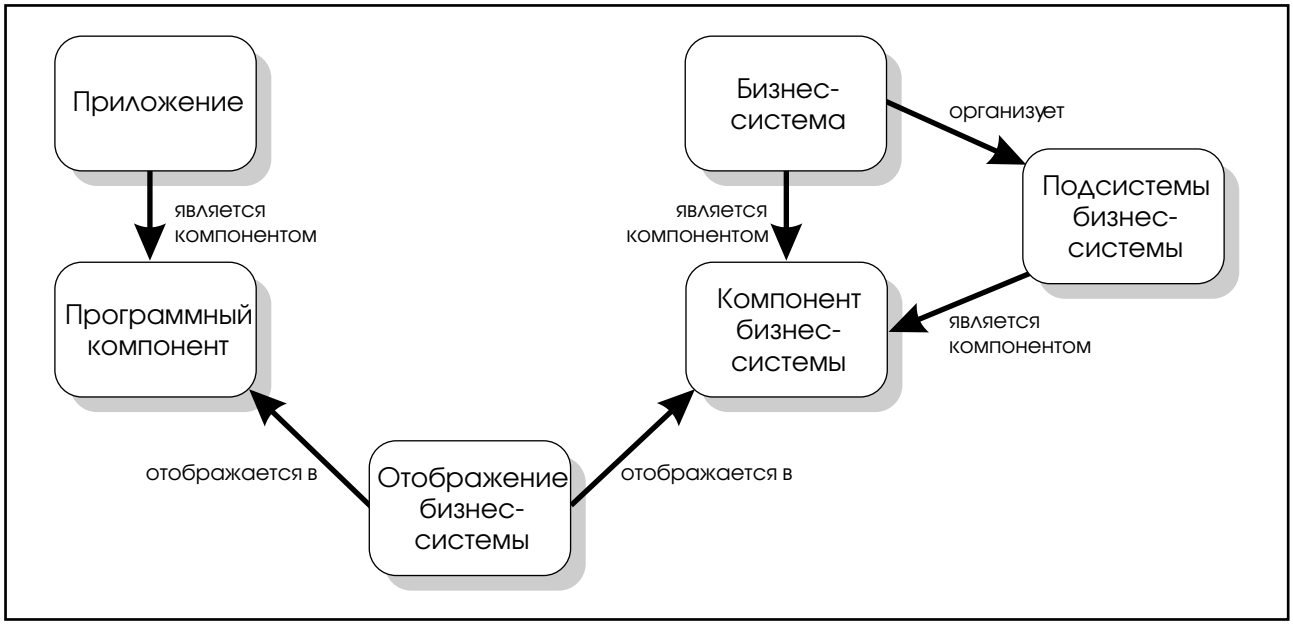

Рис. 9. Базовые элементы спецификаций AMS.

тов по сети, что облегчает администрирование самого управляющего программного обеспечения и, в частности, агентов;

- способность обеспечить функционирование всей цепочки управления — от агентов до графического интерфейса администратора (см. [11]).

Сам выбор Java для решения задач управления можно считать стандартом, причем весьма перспективным. Разумеется, компания JavaSoft на этом не остановилась. Разработаны спецификации Управляющего программного интерфейса Java (Java Management API, JMAPI, см., например, [14]). Правда, судьба у них довольно сложная. Первый вариант был выпущен в 1996 году, затем примерно в течение двух лет ничего не происходило, но в последнее время вновь наблюдается активность в этой области.

Как прикладной программный интерфейс, JMAPI состоит из следующих компонентов:

- модуль административного интерфейса (Admin View Module, AVM) — расширение Абстрактного оконного инструментария (AWT);
- базовые интерфейсы объектов — средство описания распределенных ресурсов и сервисов, подлежащих управлению;
- интерфейсы управляющих контейнеров — средство группирования управляемых объектов, позволяющее оперировать с ними как с единым целым;
- интерфейсы управляющих извещений — базовый механизм для работы с асинхронными событиями и извещениями о них;
- интерфейсы управляющих данных — средство связи с базой данных;
- интерфейсы протоколов управления — механизмы безопасного выполнения распределенных операций;
- SNMP-интерфейсы — средство связи с SNMP-агентами.

Мы не будем приводить здесь описания соответствующих Java-интерфейсов и классов. В данном случае нам достаточно уяснить, какие аспекты управляющей инфраструктуры стандартизуются. Предлагаемый набор представляется достаточно полным, хотя, быть может, пока он является слишком низкоуровневым, не учитывающим в достаточной степени специфику систем управления.

Помимо программных интерфейсов спецификации JMAPI предлагают архитектуру систем управления. Она (в несколько упрощенном виде) приведена на рис. 10.

Вероятно, наиболее принципиальным здесь является явное разделение компонента представления (его можно назвать управляющей консолью) и собственно управляющей станции, обеспечивающей, в свою очередь, взаимодействие с агентами. Реализация управляющей консоли с помощью аплетов, загружаемых по сети с HTTP-сервера, позволяет поддержать концепцию управления из любой точки (разумеется, при выполнении требований информационной безопасности).

Централизованное хранение и рассылка агентского ПО идейно близка развиваемой и активно продвигаемой концепции JINI (см. [11]). Представляется, что в контексте управления эти идеи вполне уместны.

Хотелось бы верить, что работа над JMAPI получит успешное продолжение.

6. Unicenter TNG

6.1. Основные понятия

Unicenter TNG предлагает подход к структуризации информационной системы с точки зрения управления ею, именно: «горизонтальное» и «вертикальное» слоения, обсуждавшиеся выше.

Вторая основополагающая концепция состоит в выделении в качестве самостоятельной сущности управляющего каркаса (который в рассматриваемом нами случае называется Unicenter TNG Framework). Системы управления должны иметь, если можно так выразиться, двумерную настраиваемость — на нужды конкретных организаций и на изменения в информационных технологиях. Системы управления живут (по крайней мере, должны жить) долго. За это время в различных предметных областях администрирования (например, в области резервного копирования) с высокой вероятностью появятся решения, превосходящие изначально заложенные в управляющий комплект. Последний должен уметь эволюционировать, причем разные его части могут делать это с разной скоростью. Никакая жесткая, монолитная система такого не выдержит. Единственный выход — наличие каркаса, с которого можно снимать старое и «навешивать» новое, не теряя (а, по идее, — повышая) эффективность управления.

Набор «точечных» решений, как бы хороши они ни были сами по себе, не позволит получить аналог вертикального слоения, не даст возможность построить целостную картину информационной системы с точки зрения управления бизнес-процессами. В Unicenter TNG принцип сквозного управления выполнен.

К числу концептуально важных можно отнести понятие «проактивного», то есть упреждающего управления. Упреждающее управление основано на предсказании будущего поведения системы на основе текущих данных и ранее накопленной информации. Простейший пример подоб-

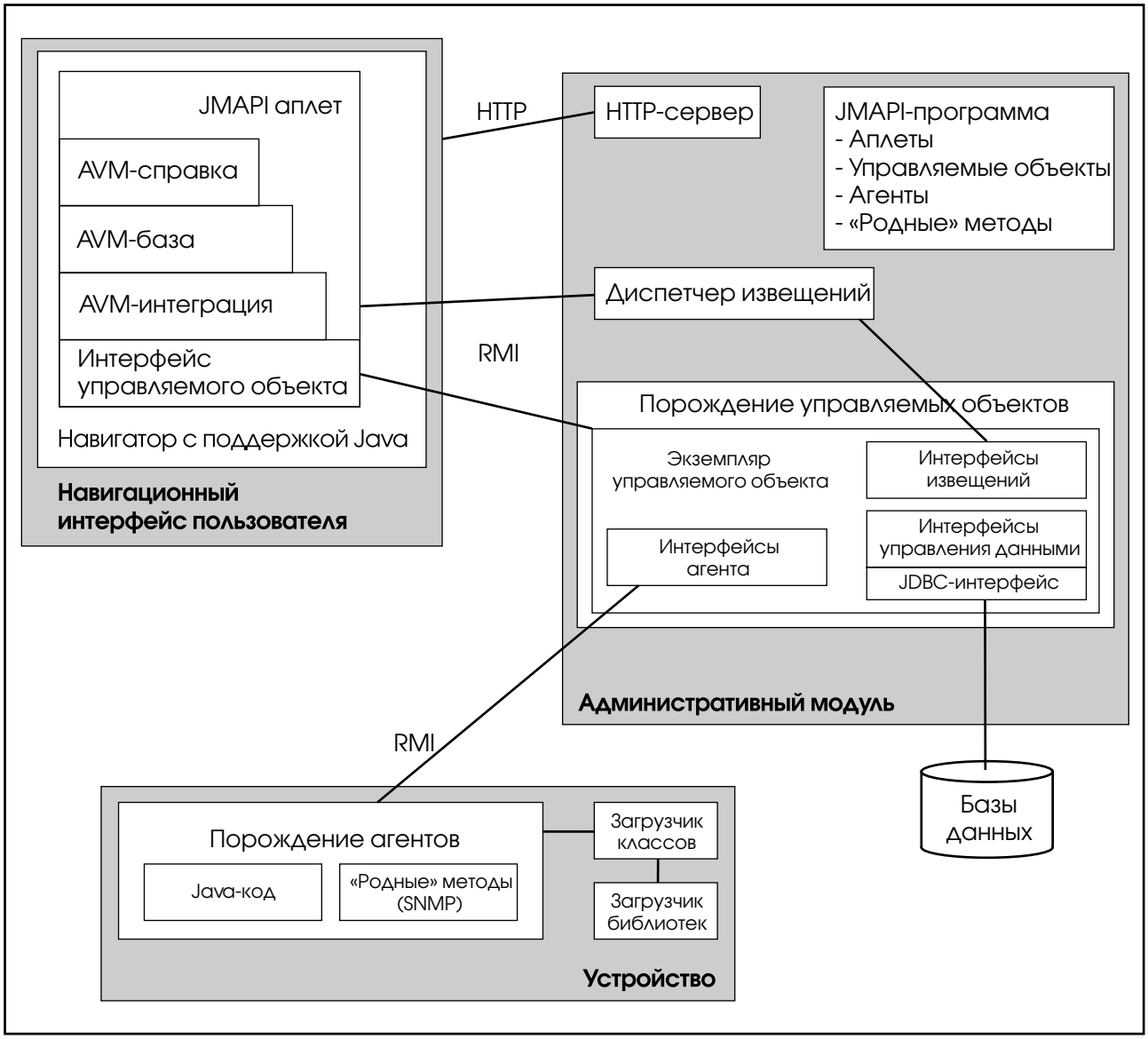

Рис. 10. Основные компоненты архитектуры JMAPI.

ного управления — выдача сигнала о возможных проблемах с диском после программно-нейтрализуемых ошибок чтения/записи. В более сложном случае определенный характер рабочей нагрузки и действий пользователей может предшествовать резкому замедлению работы системы; адекватным управляющим воздействием может служить понижение приоритетов некоторых заданий (не говоря уже об извещении администратора о приближении кризиса).

6.2. Архитектура системы управления

Типичная архитектура системы управления, реализованная в Unicenter TNG, представлена на рис. 11. Здесь следует обратить внимание на централизованное хранилище общих объектов. Вообще говоря, и у централизованных, и у распределенных хранилищ управляющей информации несложно отыскать свои плюсы и минусы. Тем не менее, нам кажется, что у централизованного решения плюсов больше. Если действительно пытаться управлять информационной системой как единым целым, учитывая связи и взаимное влияние компонентов, то в случае распределенного хранилища будет по меньшей мере индуцироваться значительный «управляющий» сетевой трафик (а практически будет требоваться постоянная связность всех компонентов, что далеко не всегда имеет место).

Возвращаясь к рис. 11, отметим, что, разумеется, для всех уровней (отображения, менеджеров, агентов) в Unicenter TNG определены открытые программные интерфейсы с возможностью встраивания продуктов третьих фирм, следующих этим спецификациям.

Централизация хранения общих объектов не противоречит распределенности обработки соответствующей информации. Как и положено современной системе, Unicenter TNG имеет объектную архитектуру, основанную на Общем интерфейсе взаимодействия (Common Communi-

cations Interface) и наборе распределенных сервисов:

1. Сервис баз данных. Отметим, что используемая СУБД может быть реляционной, но вовне все равно могут предоставляться объектные интерфейсы.
2. Сервисы визуализации и пользовательского интерфейса. Обычно требуется поддержка как графического интерфейса, так и режима командной строки.
3. Сервис оповещения о событиях. Предоставляет приложениям информацию о событиях в управляемой среде.
4. Менеджеры объектов/действий. Они имеют доступ к хранилищу объектов и на основе этой информации выполняют управляющие действия.

6.3. Функциональность системы управления

Вопрос о том, что, помимо каркаса, должно входить в систему управления, является достаточно сложным и спорным. Сложность эта коренится в двух обстоятельствах. Во-первых, многие системы управления имеют мэйнфреймовое прошлое и попросту унаследовали некоторую функциональность, которая перестала быть необходимой. Во-вторых, для ряда функциональных задач появились отдельные, высококачественные решения, превосходящие аналогичные по назначению «штатные» компоненты. Видимо, с развитием объектного подхода, многоплатформности важнейших сервисов и их взаимной совместимости, системы управления действительно превратятся в каркас. Пока же на их долю остается достаточно непокрытых областей, важных и нужных.

Если мы вновь обратимся к примеру Unicenter TNG, то увидим следующие функции:

1. управление защитой;
2. управление нагрузкой;
3. управление событиями;
4. управление хранением данных;
5. управление проблемными ситуациями;
6. генерация отчетов.

На рис. 12 показано, как эти функциональные задачи сочетаются с сервисами и интерфейсами Unicenter TNG в рамках архитектуры с распределенной обработкой управляющей информации. Сами функциональные задачи мы подробно рассмотрим далее, в разделе «Сервисы управления в Unicenter TNG».

На инфраструктурном уровне присутствует решение еще одной важнейшей функциональной

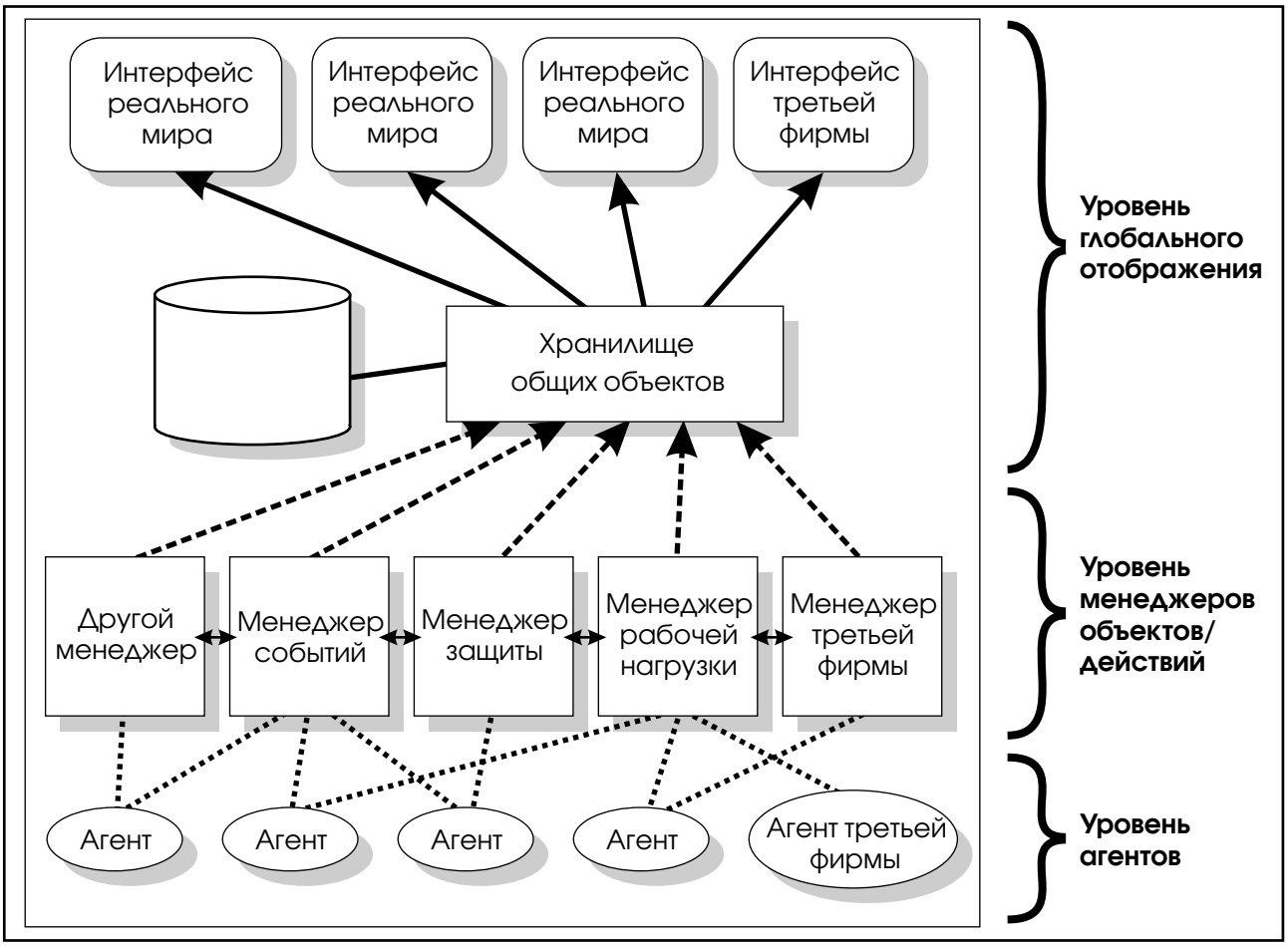

Рис. 11. Архитектура интегрированной системы управления на примере Unicenter TNG.

задачи — обеспечение автоматического обнаружения управляемых объектов, выявление их характеристик и связей между ними.

6.4. Пользовательский интерфейс

Пользователями систем управления являются администраторы различного уровня и предметной направленности, так что, возможно, в данном случае правильнее говорить об административном интерфейсе; мы, однако, сохраним привычное словосочетание.

Качество интерфейса для систем управления особенно важно по той причине, что с их помощью ведется работа с огромным числом объектов, обладающих большим числом характеристик и связей. Необходимо, чтобы существенная информация не просто была доступна «в принципе», но сама бросалась в глаза и воспринималась даже на уровне подсознания.

В этой связи очень важен реалистичный интерфейс, реализованный в Unicenter TNG. Когда сразу виден тип управляемого объекта и его основные характеристики (например, тип компьютера и операционной системы, его территориальное расположение и т.п.), не говоря о наличии патологий в его функционировании, администратору легко понять, с чем он имеет дело[5].

Объектно-ориентированные технологии, предназначенные, в частности, для удержания сложности системы в разумных рамках, полезны и в качестве идейной основы интерфейса системы управления. Unicenter TNG предоставляет навигаторы и визуализаторы классов, объектов, топологии и связей. Все это позволяет смотреть на управляемые объекты (устройства, приложения, сервис-домены) с разных точек зрения, с разной степенью детализации, фильтровать показываемые характеристики и т.д. Если администратора заинтересовала какая-либо часть управляемой системы, он сможет получить о ее состоянии полную (в рамках проведенного ранее построения модели) информацию, причем с помощью «дружественного» интерфейса.

6.5. Каркас системы управления Unicenter TNG Framework

Выше было введено понятие каркаса системы управления. Каркас как самостоятельный про-

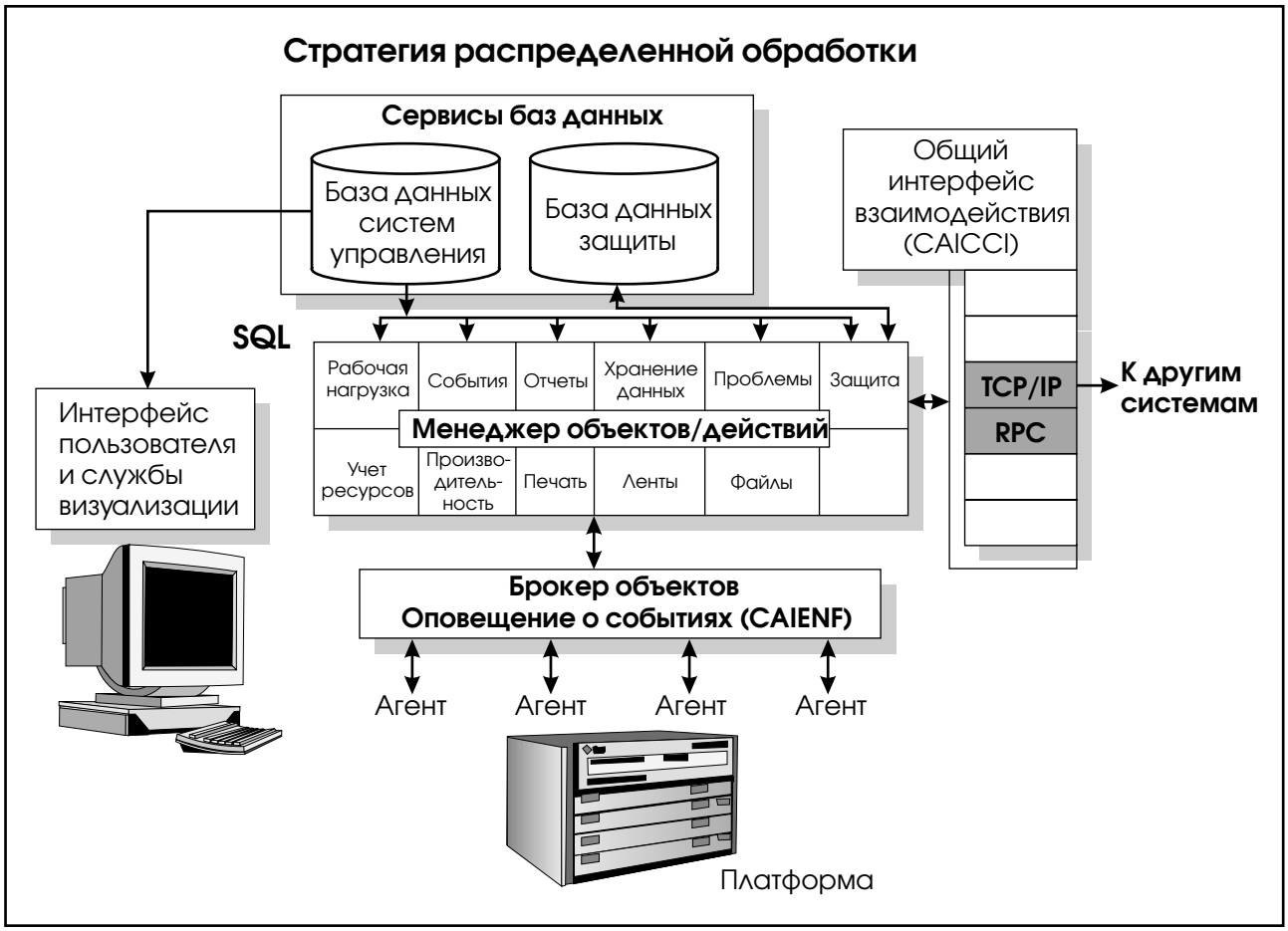

Рис. 12. Архитектура с распределенной обработкой управляющей информации.

[5] Известные почти анекдотические случаи, когда реалистичный интерфейс буквально открывал глаза администраторам, не знавшим, что у них в некотором сегменте сети присутствует два Unix-компьютера (предполагалось, что там стоит другая операционная система) связаны с некоторой «халатностью» в ведении дел. Тем не менее, хорошее представление данных есть, безусловно, подспорье в работе.

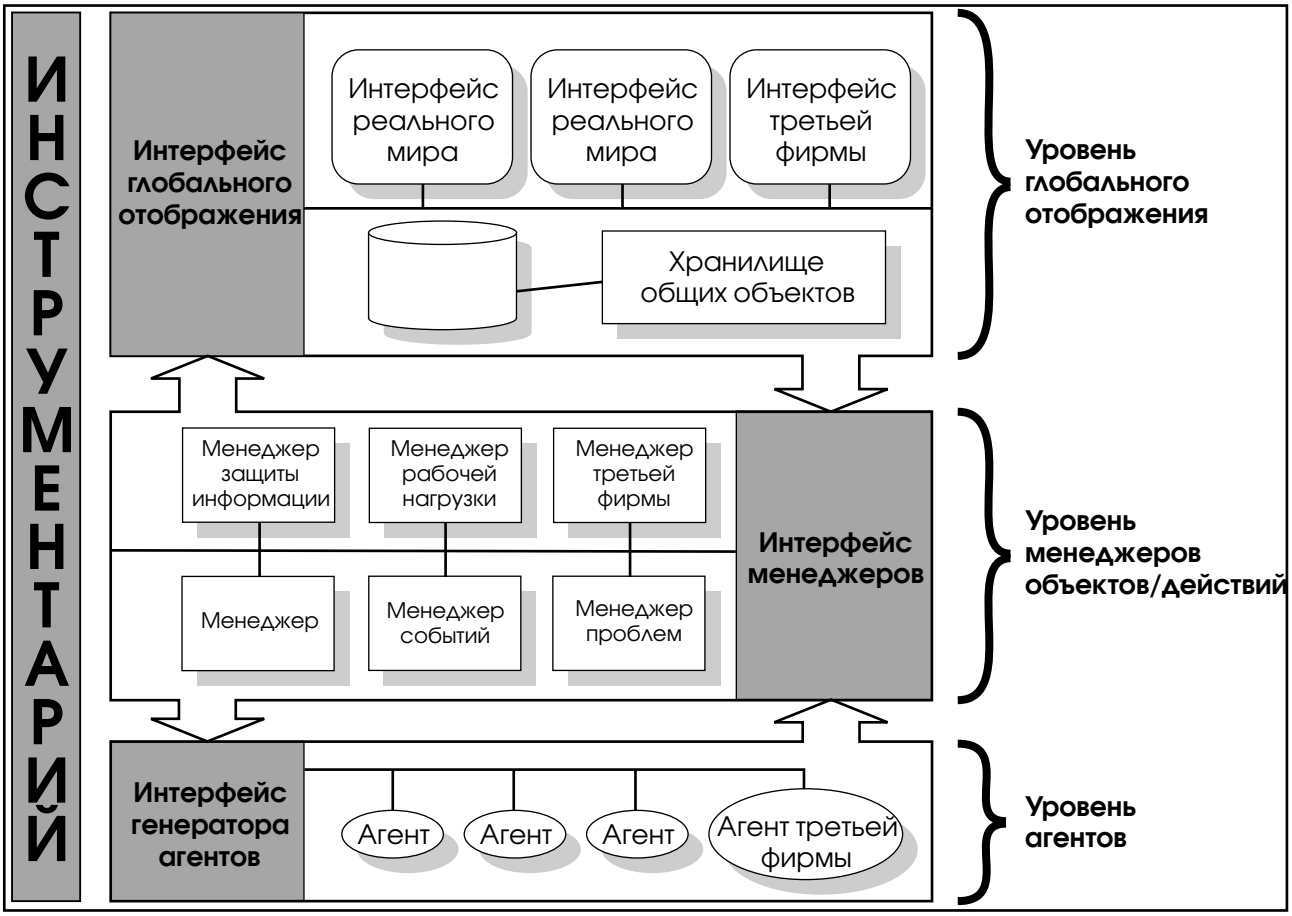

Рис. 13. Архитектура Unicenter TNG Framework.

дукт необходим для достижения по крайней мере следующих целей:

1. сглаживание разнородности управляемых информационных систем, предоставление унифицированных программных интерфейсов для быстрой разработки управляющих приложений;
2. создание инфраструктуры управления, обеспечивающей наличие таких свойств, как поддержка распределенных конфигураций, масштабируемость, информационная безопасность и т.д.
3. предоставление функционально полезных универсальных сервисов, таких как планирование заданий, генерация отчетов, антивирусная защита и т.п.

6.5.1. Архитектура Unicenter TNG Framework

Unicenter TNG Framework включает компоненты двух видов:

1. готовые элементы системы управления или заготовки таких элементов;
2. программные интерфейсы для подключения новых элементов.

Компоненты обоих видов распределены по трем уровням, что показано на рис. 13. Полезно сопоставить этот рисунок с рис. 11, изображающим архитектуру законченной системы управления Unicenter TNG. Конечно, каркас идейно близок со своим прародителем, но, в то же время, он представляет собой самостоятельный продукт, а не результат «выдирания по живому». Unicenter TNG Framework требует разумного количества ресурсов, его интерфейсы в достаточной степени универсальны, на его основе действительно можно строить собственные системы управления.

На верхнем уровне каркаса Unicenter TNG Framework, называемом уровнем глобального отображения, располагаются компоненты пользовательского интерфейса, хранилище объектов, аппарат автообнаружения и соответствующие программные интерфейсы.

Средний уровень соответствует корпоративным сервисам управления (управление загрузкой, безопасностью, консультационная служба и т.д., см. далее раздел «Сервисы управления в Unicenter TNG»). На этом же уровне располагается инфраструктурный сервис управления событиями, а также вспомогательные сервисы, такие как календарь, позволяющий планировать управляющие действия.

Нижний уровень можно назвать агентским. Здесь помещены агенты для различных операционных систем, заготовки для создания собственных агентов. Среди программных интерфейсов этого уровня следует выделить интерфейс с подсистемой генерации агентов.

Далее мы более детально рассмотрим каждый из уровней, но перед этим обратим внимание на важное свойство Unicenter TNG Framework — следование современным стандартам управления. Данный каркас поддерживает спецификации SNMP (см. выше раздел «Стандарты Интернет-сообщества») и интерфейс управления настольными системами DMI 2.0 (см. выше раздел «Управление настольными системами»).

6.5.2. Уровень глобального отображения

В этом разделе мы рассмотрим два элемента Unicenter TNG Framework, располагающиеся на уровне глобального отображения — пользовательский интерфейс и хранилище объектов.

Пользовательский интерфейс — один из самых впечатляющих компонентов Unicenter TNG. Мы уже описывали его выше, в разделе «Пользовательский интерфейс». Конечно, столь высокое искусство требует жертв — в данном случае ресурсов. Думается, что эти жертвы не напрасны, особенно в перспективном плане. Как само собой разумеющийся технический момент отметим наличие платформно-независимого Java-интерфейса (наряду с поддержкой «родных» оконных систем). Кроме того, упомянем, что с помощью механизма автообнаружения (функционирующего в рамках сетей IP и IPX) пользователь Unicenter TNG Framework получает двумерную карту своей информационной системы. На рис. 14 приведен пример подобной карты.

С концептуальной точки зрения более важными являются объектная модель, используемая в каркасе, способы хранения этой модели и поддерживаемые операции с объектами.

В Unicenter TNG Framework активно используется аппарат наследования, присущий объектному подходу. На рис. 15 показана часть иерархии классов, в которую входят «Управляемый объект», «Сеть», «Сегмент», «Маршрутизатор» и т.п. В принципе, в проведении объектного подхода в наше время нет ничего необычного, однако для систем управления, имеющих мэйнфреймовые корни, это очень большой шаг вперед.

Из операций над хранимыми объектами, доступных посредством программного интерфейса, отметим группирование объектов. Эта операция важна в первую очередь для обеспечения масштабируемости управляющих приложений. Критерий для объединения в группы может быть произвольным, и это тоже важно, так как у каждого приложения своя точка зрения на объекты.

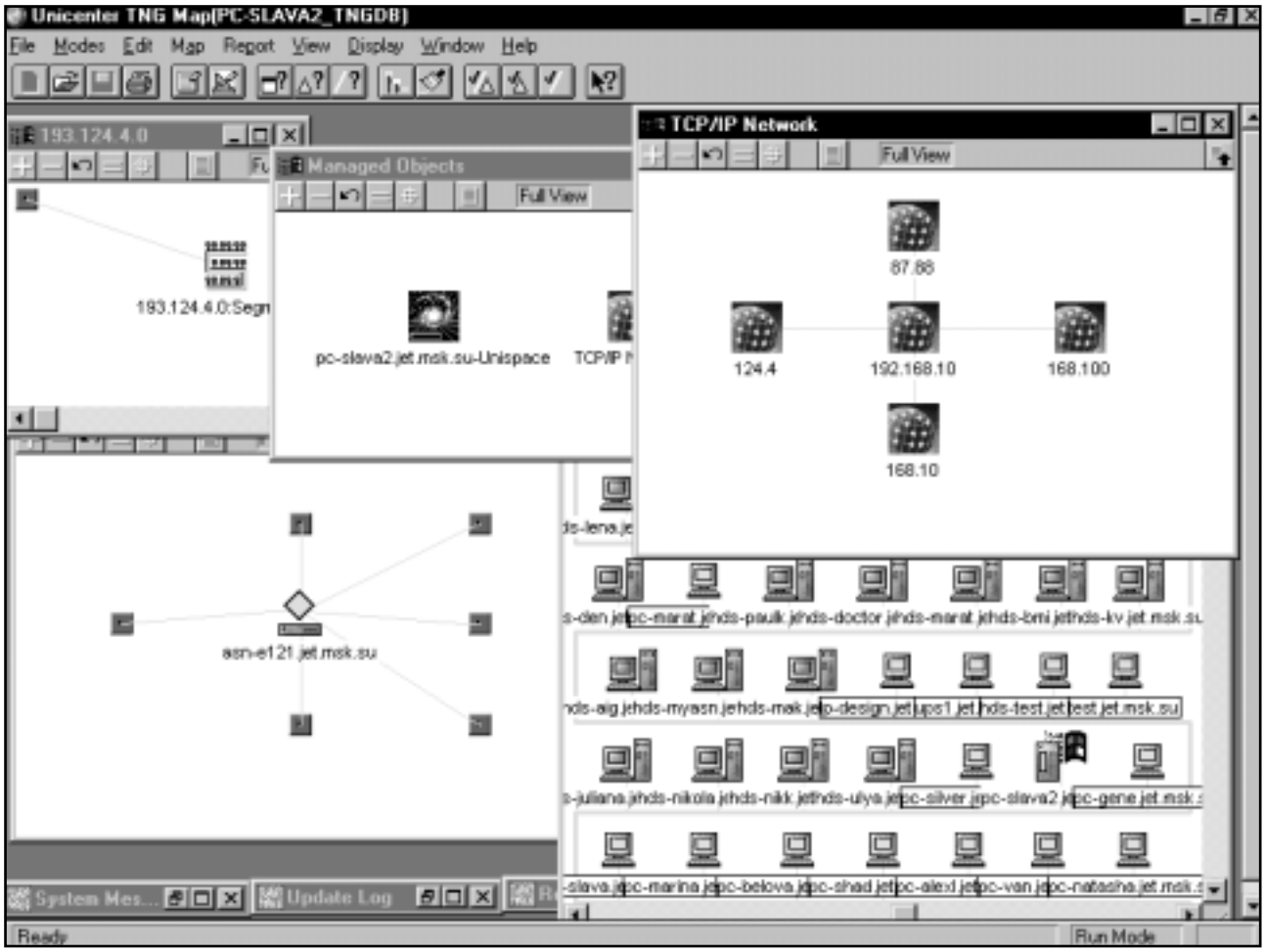

Рис. 14. Пример двумерной карты информационной системы.

6.5.3. Уровень менеджеров

На данном уровне можно выделить инфраструктурные компоненты и программные интерфейсы, обслуживающие сетевое взаимодействие и управление сообщениями, а также программные интерфейсы с управляющими приложениями, входящими в состав Unicenter TNG.

Общий коммуникационный интерфейс (Common Communication Interface, CCI) изолирует прочие компоненты системы управления от

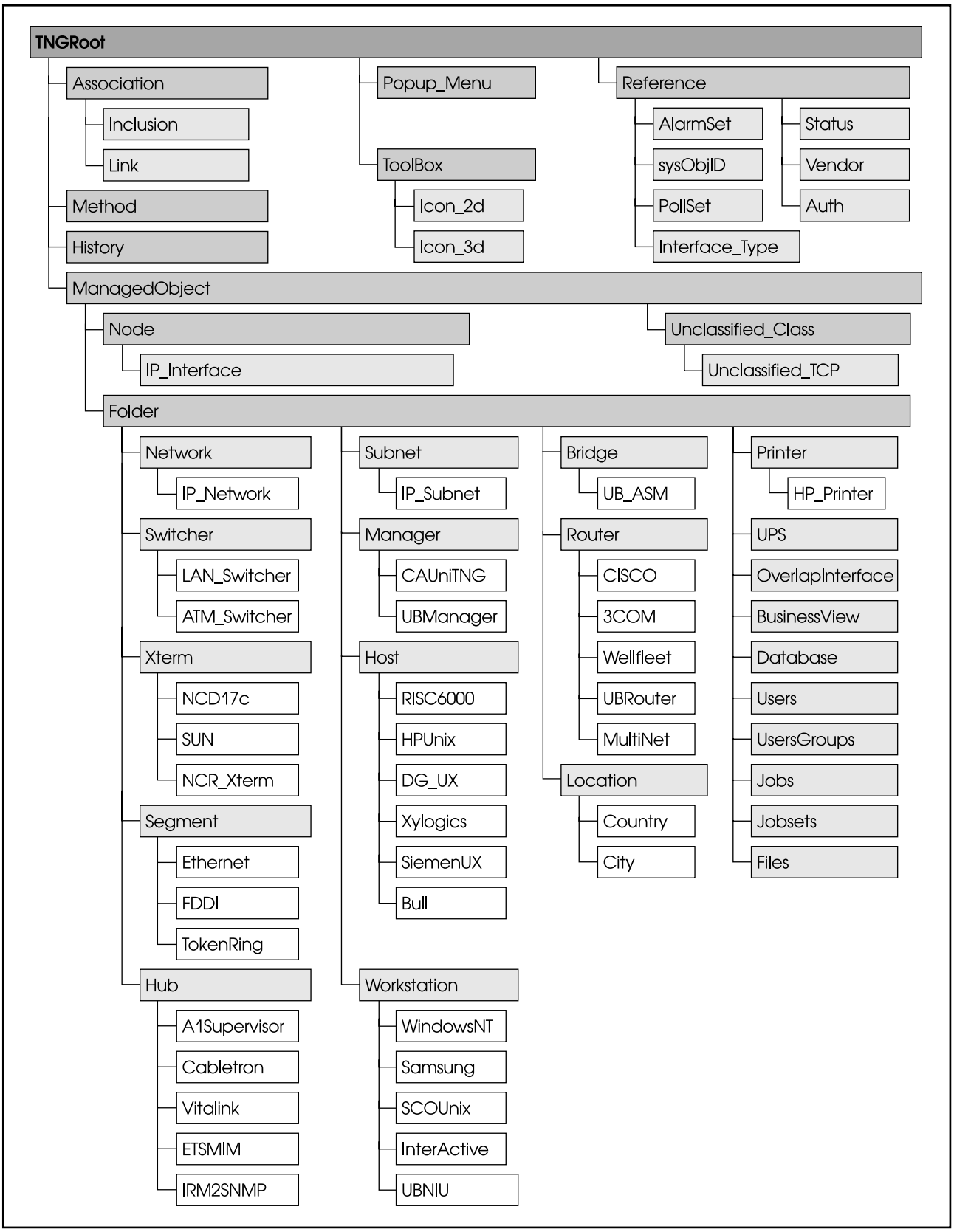

Рис. 15. Фрагмент дерева наследования классов в Unicenter TNG Framework.

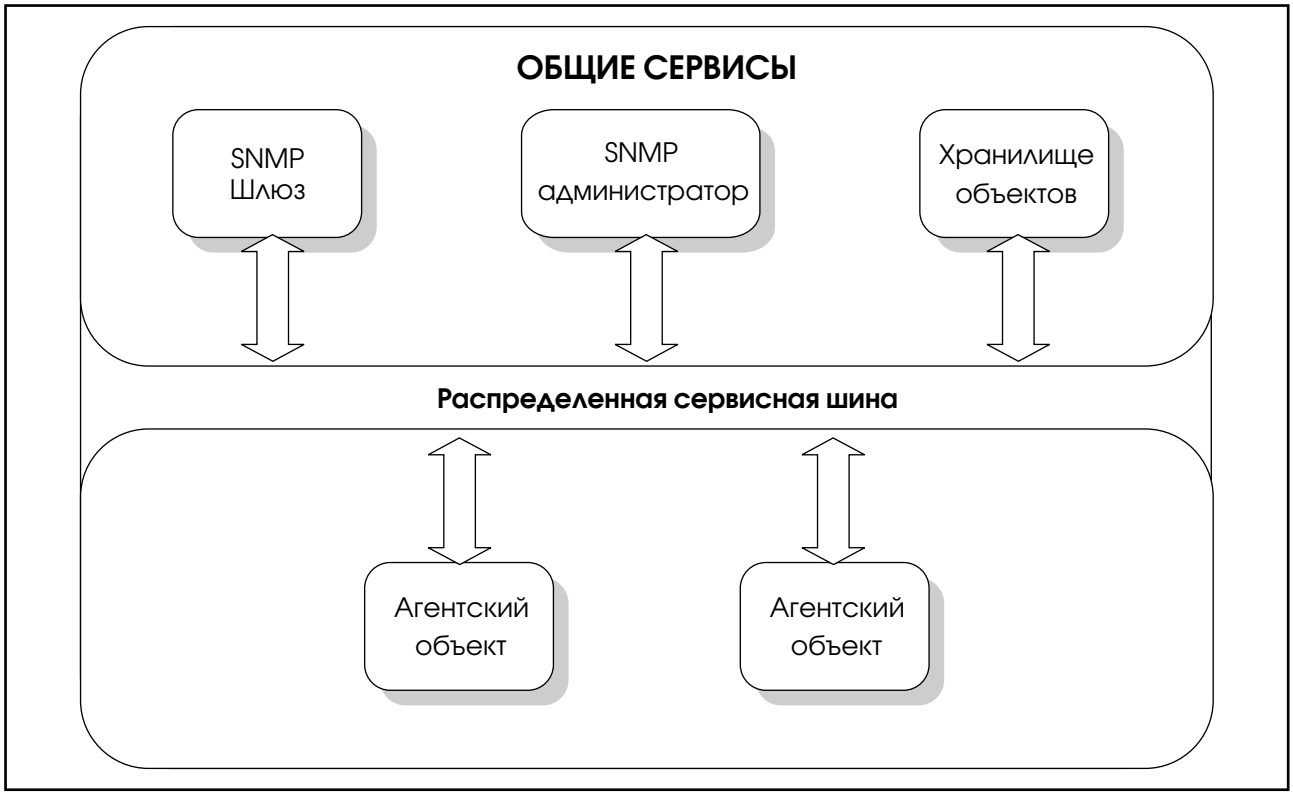

Рис. 16. Объектная архитектура генератора агентов.

особенностей конкретных сетевых протоколов, а также обеспечивает конфиденциальность и целостность передаваемых данных.

На базе CCI функционирует Служба уведомления о событиях (Event Notification Facility, ENF). Эта служба позволяет единообразно генерировать, пересылать и обрабатывать уведомления вне зависимости от платформы, на которой было зарегистрировано событие. Уведомления направляются на центральную консоль или, при необходимости, на специализированную — например, на консоль администратора безопасности. В механизм ENF встроены возможности выявления взаимосвязанных уведомлений, фильтрации несущественной информации, автоматического реагирования на определенные события.

Программный интерфейс ENF позволяет:

1. осуществлять доступ к регистрационным файлам Службы уведомления Unicenter TNG;
2. посылать и принимать сообщения;
3. выдавать программно-управляемые сообщения вида «SNMP-trap»;
4. воздействовать на фильтрацию сообщений.

Представляется вполне естественным, что Unicenter TNG Framework предоставляет программный интерфейс к управляющим приложениям Unicenter TNG. В число таких приложений входят:

1. управление загрузкой;
2. управление проблемными ситуациями;
3. автоматическое управление хранением;
4. управление накоплением вывода;
5. управление безопасностью

и некоторые другие.

Отметим, что реализованное в Unicenter TNG управление безопасностью в совокупности с соответствующим программным интерфейсом позволяют реализовать платформно-независимое разграничение доступа к объектам произвольной природы и (что очень важно) вынести функции безопасности из прикладных систем, освободив последние от несвойственных им функций. Чтобы выяснить, разрешен ли некоторый доступ текущей политикой безопасности, приложению достаточно обратиться к менеджеру безопасности Unicenter TNG.

Приведенный пример показывает, что Unicenter TNG Framework реально способен обеспечить интерфейс к управляющей функциональности при весьма умеренных затратах разработчиков приложений.

6.5.4. Уровень агентов

Наиболее интересными элементами данного уровня являются генератор агентов и программный интерфейс к нему.

Unicenter TNG поддерживает спецификации SNMP. Соответственно, чтобы некоторым объектом можно было управлять посредством Unicenter TNG, он (объект) должен быть снабжен SNMP-агентом. Из этого положения вытекает важность задач, решаемых генератором агентов.

Генератор агентов Unicenter TNG оставляет разработчику лишь написание кодов, специфичных для управляемого объекта; все остальное

он берет на себя. В частности, в генераторе агентов имеются готовые коммуникационные сервисы, обеспечивающие поддержку протокола SNMP.

С технической точки зрения генератор агентов представляет собой законченную среду разработки, состоящую из следующих компонентов:

1. библиотека, обслуживающая программный интерфейс к генератору;
2. готовые к употреблению сервисы, общие для различных агентов;
3. средства конфигурирования и тестирования агентов;
4. образцы программ и MIB-файлов.

В свою очередь, общие сервисы подразделяются на:

1. SNMP-шлюз;
2. SNMP-администратор;
3. хранилище.

На рис. 16 показана объектная архитектура генератора агентов. Отметим, что штатным языком разработки является язык C. Отметим также, что здесь хранилище предназначено для запоминания атрибутов одного или нескольких взаимосвязанных управляемых объектов; это хранилище агента, а не управляющих приложений.

Строго говоря, генератор агентов есть не только среда разработки, но и среда выполнения. Программный интерфейс генератора как раз и нацелен на использование возможностей, предоставляемых средой выполнения. Функции в библиотеке, обслуживающей программный интерфейс, подразделяются на довольно очевидные категории, такие как:

1. регистрация на уровне SNMP-администратора;
2. управление агентскими действиями и их планирование;
3. операции с событиями;
4. работа с хранилищем

и некоторые другие.

Архитектура агента, получающегося в результате генерации штатными средствами (см. рис. 17), напоминает объектную архитектуру самого генератора. Отличия выражаются в присутствии связей с управляющим приложением и управляемым объектом.

На наш взгляд, Unicenter TNG Framework является весьма зрелым продуктом, в достаточной степени «отвязанным» от родительской системы Unicenter TNG. С широким распространением этого каркаса связываются надежды на внедрение систем управления в организациях с умеренным бюджетом, составляющих абсолютное большинство.

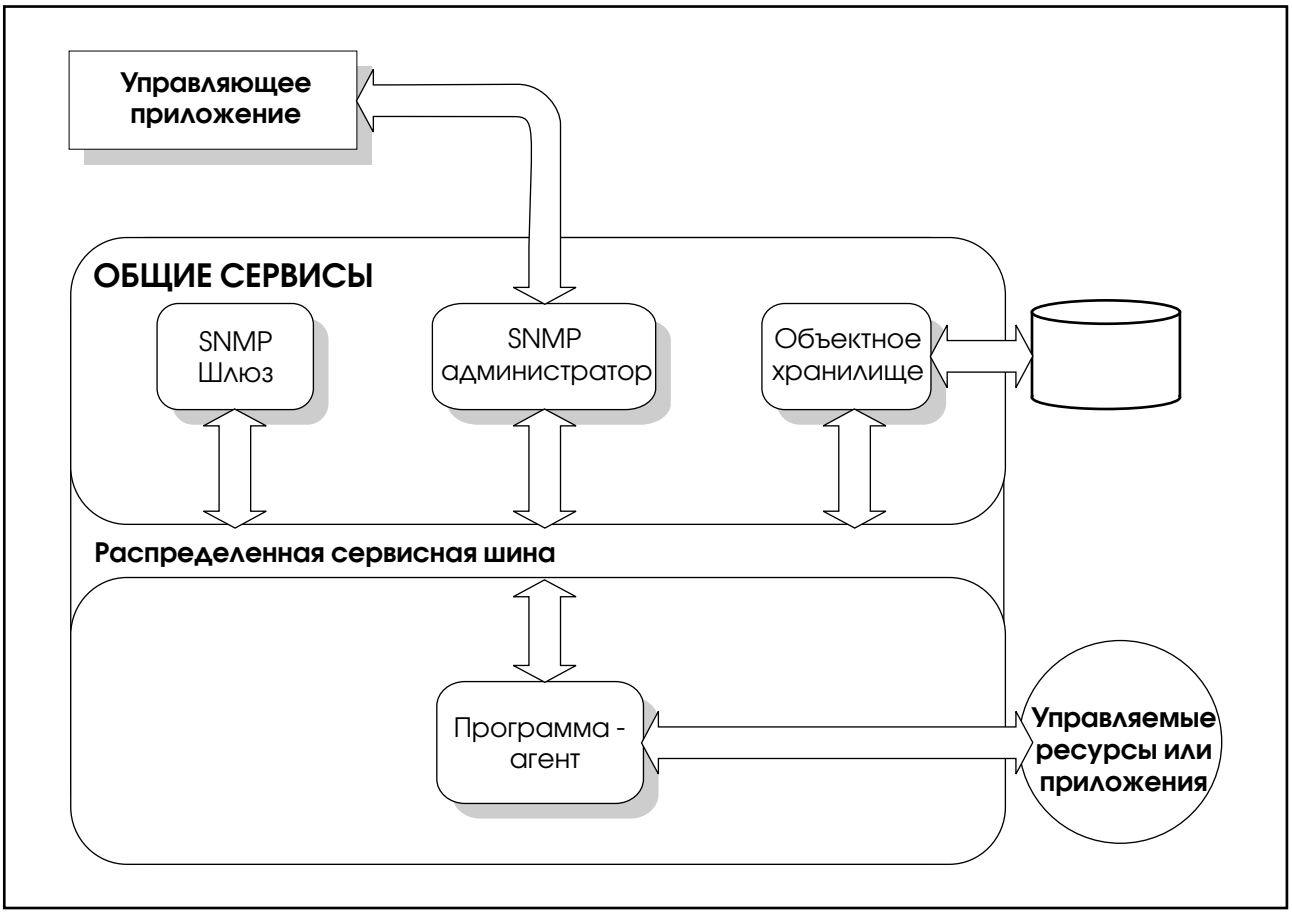

Рис. 17. Архитектура сгенерированного агента.

6.6. Некоторые сервисы управления в Unicenter TNG

Пользователю системы управления, то есть администратору ИС, важна не столько внутренняя архитектура, сколько предоставляемая функциональность. Ее мы и рассмотрим в данном разделе.

Будут описаны следующие сервисы управления:

1. управление безопасностью;
2. управление загрузкой;
3. управление проблемными ситуациями;
4. управление событиями.

Это далеко не полный перечень сервисов, которые предоставляет Unicenter TNG, однако мы попытались выделить самые важные и удачно реализованные.

6.7. Управление безопасностью

Под управлением безопасностью мы, расширяя принятую в Unicenter TNG трактовку, будем понимать два вида действий:

1. разграничение доступа;
2. обеспечение высокой доступности данных.

6.7.1. Разграничение доступа

Разграничение доступа реализуется менеджером безопасности (Security Manager), который осуществляет идентификацию/аутентификацию пользователей, контроль доступа к ресурсам и протоколирование неудачных попыток доступа. Можно считать, что менеджер безопасности встраивается в ядро операционных систем контролируемых узлов сети, перехватывает соответствующие обращения и осуществляет свои проверки перед проверками, выполняемыми ОС, так что он предоставляет еще один защитный рубеж, не отменяя, а дополняя защиту, реализуемую средствами ОС.

Практически при любом подходе к разграничению доступа основными понятиями являются субъект (пользователь) и объект (защищаемый ресурс). В этом смысле Unicenter TNG не является исключением. И пользователи, и ресурсы могут объединяться в группы, в том числе вложенные. Разумеется, без возможности иерархической организации управляемых сущностей о масштабируемости не было бы и речи.

Наиболее интересными особенностями менеджера безопасности в Unicenter TNG нам представляются две:

1. наличие гибких средств поддержки политики безопасности;
2. возможность проведения единой политики безопасности в рамках информационной системы, независимо от используемых на узлах сети операционных систем и от расположения в сети субъектов и объектов (впрочем, возможность задать разную политику для разных узлов сети также остается).

В принципе, обе особенности являются следствием того факта, что некоторое представление политики безопасности хранится в централизованной базе данных. С логической точки зрения можно считать, что при каждой попытке доступа выполняется просмотр правил, сохраненных в базе, в результате которого выясняется наличие у пользователя необходимых прав. Если таких прав нет, доступ все же может быть предоставлен, но с выдачей предупреждения пользователю и с фик-

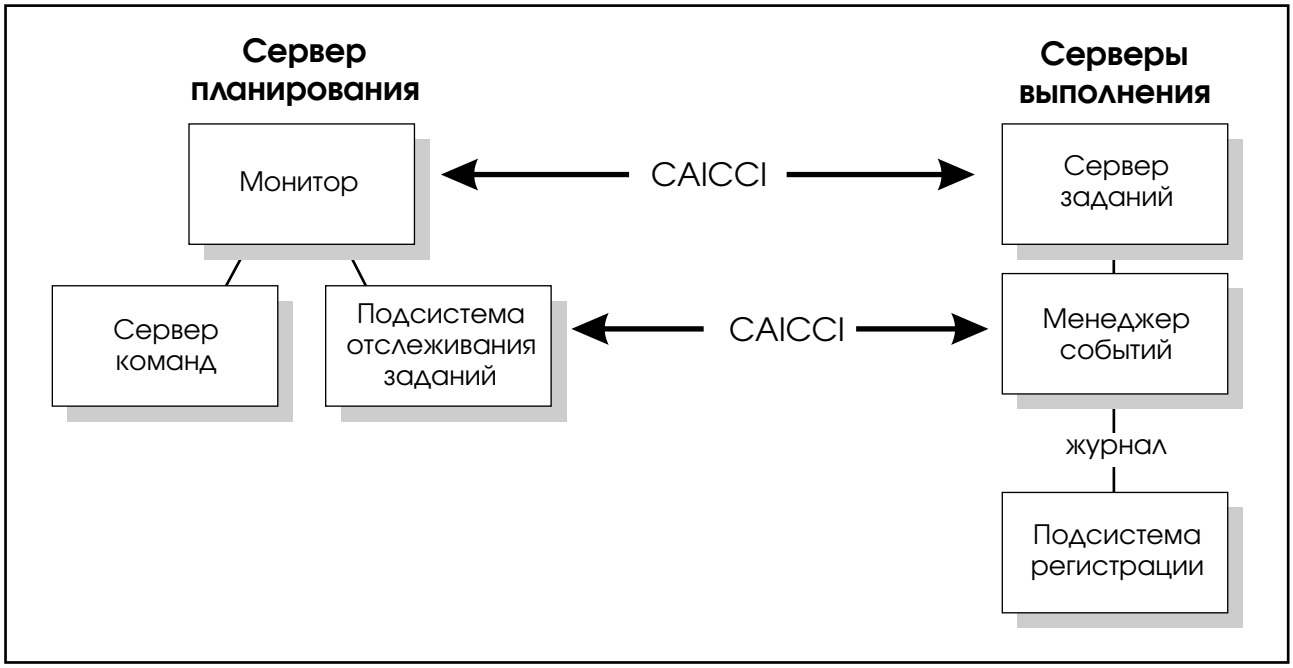

Рис. 18. Архитектура подсистемы управления прохождением заданий в Unicenter TNG.

сацией этой попытки в регистрационном журнале. Разумеется, Unicenter TNG допускает задание и более жесткой реакции на попытки неавторизованного доступа вплоть до завершения сеанса работы пользователя и блокирования последующих запросов на вход в систему.

Хранение параметров безопасности в базе данных дает администраторам еще одно важное преимущество — возможность выполнения разнообразных запросов. Можно получить список ресурсов, доступных данному пользователю, список пользователей, имеющих доступ к данному ресурсу, и т.п.

Отметим, что в плане контроля доступа Unicenter TNG предлагает немного больше, чем это делают привычные операционные системы. Введены дополнительные режимы доступа (например, создание, удаление), поддерживается расщепление прав суперпользователя, обеспечивается дополнительный контроль за программами, переустанавливающими действующий идентификатор пользователя/группы. В результате разграничение доступа становится более простым и логичным, теснее увязанным с бизнес-потребностями (а не особенностями той или иной ОС).

6.7.2. Обеспечение высокой доступности данных

Обеспечение высокой доступности данных реализуется в Unicenter TNG подсистемой автоматического управления хранением данных, выполняющей резервное копирование данных, их архивирование (резервное копирование с удалением оригинала), а также автоматическое отслеживание перемещения данных между основными и резервными носителями.

Конечно же, операции резервного копирования/архивирования могут планироваться на определенное время или инициироваться агентами, следящими за состоянием «своих» систем. Разумеется, информация о расположении файлов хранится в базе данных. Конечно же, резервное копирование может сопровождаться сжатием данных. Разумеется, резервные носители защищены от случайного стирания. Вообще, перечень подобных «конечно же» и «разумеется» может быть продолжен практически неограниченно, поскольку Unicenter TNG выполнен на достаточно высоком техническом уровне, что делает поведение системы управления естественным, простым для понимания.

6.8. Управление загрузкой

Под управлением загрузкой мы (опять-таки несколько отступая от терминологии Unicenter TNG) будем понимать два вида действий:

1. управление прохождением заданий;
2. контроль производительности.

На наш взгляд, эти действия взаимосвязаны. Выяснив, когда и на каких компьютерах наибольшее количество свободных ресурсов, можно планировать время и место выполнения заданий. И, наоборот, располагая информацией о заданиях, их ресурсах и желательных сроках выполнения, можно оценить индуцируемую ими нагрузку и необходимую производительность.

6.8.1. Управление прохождением заданий

Управление прохождением заданий (Workload Management) позволяет автоматически запускать пакетные задания и отслеживать ход их выполнения. Этот компонент использует следующие основные понятия:

1. задание — работа, которая должна быть выполнена;
2. набор заданий — группа заданий, имеющих однородные параметры;
3. станция — система, на которой будет выполняться задание;
4. календарь — набор дней и часов, когда задание может выполняться;
5. ресурсы — совокупность объектов, которые должны быть в наличии, чтобы выполнение задания было возможным.

Разумеется, помимо календарных сроков Unicenter TNG учитывает зависимости между заданиями (которые должны быть предварительно определены). Он также анализирует коды завершения заданий, чтобы сделать вывод об успехе или неуспехе их выполнения. Впрочем, история автоматического управления прохождением заданий насчитывает десятилетия, так что едва ли есть смысл подробно объяснять «как ходят, как сдают».

Особенностью управления прохождением заданий в Unicenter TNG является то, что реализовывать его приходится в распределенной, разнородной среде клиент/сервер. Обычно в информационной системе, где установлен Unicenter TNG, выделяется сервер планирования и несколько серверов выполнения (см. рис. 18). Главная задача сервера планирования — подобрать время и место выполнения задания и передать его (задание) серверу выполнения. Помимо этого, сервер планирования получает данные о прохождении заданий от серверов выполнения и осуществляет их централизованное накопление. В результате распределенная обработка успешно сочетается с целостной картиной хода планирования и выполнения заданий.

Планирование выполнения заданий по времени реализовано в Unicenter TNG весьма гибко. Учитывается несколько параметров:

1. время, раньше которого задание не должно запускаться;

2. время, после которого задание запускать уже не нужно;
3. время, к которому задание обязательно должно завершиться;
4. максимальная длительность выполнения задания.

Располагая этими данными, планировщик может оптимизировать загрузку систем, стремясь к наиболее эффективному использованию ресурсов (наиболее ценным из которых является время пользователей).

6.8.2. Контроль производительности

Контроль производительности (Performance Management) — понятие многогранное. Под ним понимается и оценка быстродействия компьютеров, и анализ пропускной способности сетей, и отслеживание числа одновременно поддерживаемых пользователей, и время реакции, и накопление и анализ статистики использования ресурсов. Обычно в распределенной системе соответствующие данные доступны «в принципе», они поставляются точечными средствами управления, но проблема получения целостной картины, как сиюминутной, так и перспективной, остается весьма сложной. Кроме того, непросто (но необходимо) связывать данные о производительности с ходом бизнес-процессов, на поддержку которых направлена работа информационной системы. Именно подобные проблемы показывают, насколько важна система управления корпоративного уровня, такая как Unicenter TNG.

Средства контроля производительности в Unicenter TNG можно подразделить на две категории:

1. выявление неадекватного функционирования компонентов информационной системы и автоматическое реагирование на эти события;
2. анализ тенденций изменения производительности системы и долгосрочное планирование.

Для функционирования обеих категорий средств необходимо выбрать отслеживаемые параметры и допустимые границы для них, выход за которые означает «неадекватность функционирования». Иными словами, в первую очередь выполняется конфигурирование механизмов контроля производительности.

Разумеется, слежением за производительностью компонентов информационной системы заняты соответствующие агенты. Наличие хранилища общих объектов позволяет производить централизованное конфигурирование агентов. Об аномалиях в функционировании сообщается на центральную консоль, графические средства которых дают наглядное представление о «масштабах бедствия».

Набор параметров, за которыми следят агенты Unicenter TNG, достаточно широк. Это использование процессора, оперативной памяти, дисков, средств буферизации и кэширования и многие другие. Строго говоря, данный перечень является системно-зависимым, поскольку, например, ОС Unix и Windows NT отличаются по составу и архитектуре. Unicenter TNG обладает глубокими знаниями об особенностях различных ОС, так что администратор единой системы управления в состоянии улавливать специфические аспекты поведения контролируемых компьютеров и производить тонкие регулировки.

Выбор допустимых границ (пороговых значений) для отслеживаемых параметров является делом довольно сложным. Здесь важно не «завысить» порог (тогда тревога будет поднята слишком поздно), но и не «занизить» его (тогда будет много ложных тревог). Поскольку параметров много, то пытаться выставлять пороги «из опыта», эмпирически, значит постоянно менять регулировки, по многу раз возвращаться к каждому параметру, не имея гарантий успеха. Важным достоинством Unicenter TNG является наличие предопределенных профилей, предлагающих разумные значения границ и позволяющих немедленно начать эффективный контроль производительности.

Постоянный сбор и анализ регистрационной информации, сохранение «исторических» данных позволяют не только анализировать тенденции в поведении информационной системы, но и докапываться до первопричин сиюминутных проблем. Обнаружив выход каких-либо параметров за допустимые границы, администратор может «прокрутить» события назад и определить, когда и какой именно параметр начал вести себя подозрительно. Скорее всего, это позволит обнаружить и устранить «корень зла».

Предполагается, что в следующей версии Unicenter TND (The Next Dimension) обнаружение будет осуществляться с помощью так называемых нейросетевых агентов (Neugents), действубщих на самом деле методами математической статистики. Аналогичная идея уже реализована в системах активного аудита информационной безопасности; есть все основания для ее обобщения на управление в целом.

Анализ тенденций изменения производительности невозможен без серьезной математической базы. Вероятно, наличие такой базы позволило компании Computer Associates создать один из самых продвинутых продуктов — нейросетевого агента, обладающего способностью не просто обнаруживать, но и предсказывать аномалии поведения (см. рис. 19), то есть осуществлять упреждающее управление.

На этом примере можно видеть, что правильный, научный подход к проблемам информацион-

ных технологий дает значительный долговременный эффект и в чисто коммерческом плане, позволяет оказаться на корпус впереди конкурентов.

6.9. Управление событиями и проблемными ситуациями

Управление событиями (точнее, сообщениями о событиях) — это базовый механизм, позволяющий контролировать состояние информационных систем в реальном масштабе времени. Unicenter TNG поддерживает три вида сообщений о событиях:

1. сообщения, соответствующие спецификациям SNMP;
2. сообщения, направляемые демону syslog;
3. сообщения, порожденные с помощью программного интерфейса, предложенного компанией Computer Associates.

Unicenter TNG позволяет классифицировать события и назначать для некоторых из них специальные процедуры обработки. Тем самым реализуется важный принцип автоматического реагирования.

Внутренние коммуникационные возможности Unicenter TNG позволяют перенаправлять сообщения о событиях в зависимости от их характера, консолидируя сообщения там, где они нужны. Интерфейсные средства, доступные из подсистемы управления событиями, достаточно выразительны, чтобы привлечь при необходимости внимание операторов. Наконец, интеллектуальные агенты могут не только генерировать, но и обрабатывать сообщения о событиях, уменьшая тем самым сетевой трафик и распределяя индуцируемую нагрузку.

Одним из видов реакции на событие может быть формирование проблемной ситуации, требующей особого внимания администраторов.

Описание проблемной ситуации включает в себя как компоненты информационной системы, так и происходящие события. Иными словами, ситуация показывает, что и с чем случилось. Имеющееся в Unicenter TNG понятие зоны ответственности дает возможность привлечь к проблеме внимание строго определенного круга лиц, а приоритет задает желательную меру внимания.

В жизненном цикле проблемной ситуации можно выделить три этапа:

1. возбуждение ситуации;
2. отслеживание хода разрешения проблемы;
3. закрытие проблемы и архивирование данных о ней.

Unicenter TNG позволяет автоматизировать весь жизненный цикл, хотя возможность ручного вмешательства (в частности, ручного возбуждения проблемной ситуации), конечно же, остается. Важно отметить, что на этапе отслеживания возможна автоматическая эскалация проблемы, то

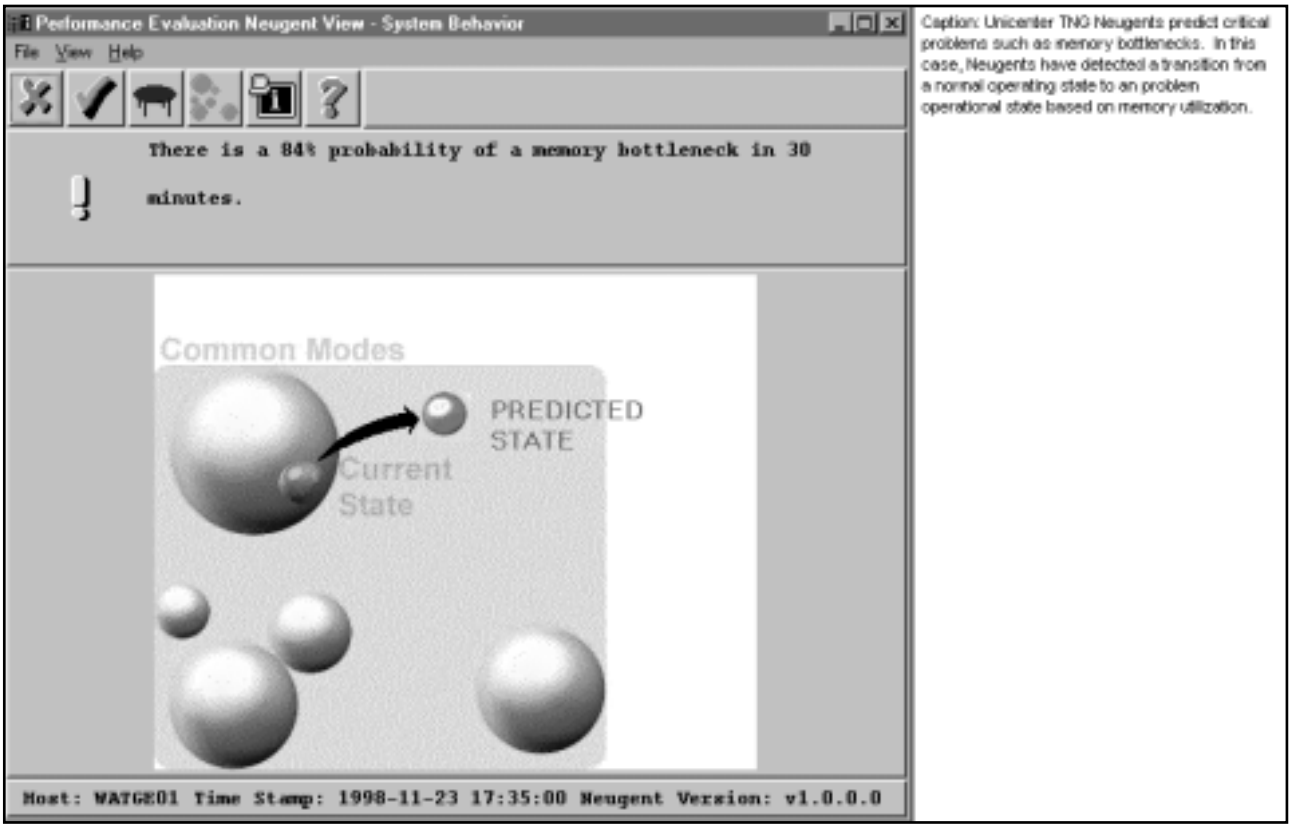

Рис. 19. Предсказание аномалии производительности.

есть расширение круга проинформированных лиц и повышение приоритета. Проблему не удастся просто так «отложить в долгий ящик», поскольку с течением времени Unicenter TNG будет все настойчивее привлекать внимание к ней.

Мы весьма кратко и фрагментарно описали сервисы управления, представленные в Unicenter TNG. Продукт этот интенсивно развивается, как в плане повышения функциональности, так и в плане расширения охвата компонентов информационных систем, хотя уже сейчас он является достаточно мощным и многоплатформным.

7. Заключение

Исторически горизонтальные решения возникли раньше вертикальных и быстро эволюционировали именно потому, что были связаны только с информационными технологиями, то есть с тем, что формализуемо. Вертикальные решения охватывают плохо формализуемую область бизнес-процессов и их информационного сопровождения. Стандарты на этот симбиоз пока не созданы, а самому вертикальному управлению предстоит пройти относительно долгий путь, чтобы достичь подлинной зрелости.

В деле управления взаимодействуют интересы четырех категорий субъектов:

1. пользователи систем управления (то есть организации, информационными системами которых необходимо управлять);
2. разработчики продуктов (аппаратных и программных) информационных технологий;
3. разработчики систем управления;
4. органы, ведающие разработкой стандартов.

Пользователи должны осознать важность проблемы управления, необходимость выделять на решение этой проблемы и материальные, и интеллектуальные ресурсы. К сожалению, современные информационные системы не могут работать «сами собой», а последствия возможных сбоев зачастую оказываются не только неожиданными, но и весьма неприятными.

Разработчики продуктов информационных технологий должны понять (в первую очередь это относится к программистам), что неуправляемые системы являются неполноценными, как автомобиль с мощным двигателем, но без руля. Разного рода зонды, агенты, диагностические утилиты должны входить в обязательный комплект поставки.

Главными задачами разработчиков систем управления, на наш взгляд, являются обеспечение большей модульности и масштабируемости вниз. Не должно быть проблем в сопряжении интегрированных систем с лучшими точечными продуктами, а порог потребляемых ресурсов и сложности использования должен быть понижен.

Наконец, от органов по стандартизации требуется разработка спецификаций, регламентирующих не только нижние уровни, такие как SNMP, но и высокоуровневые, содержательные аспекты, такие как управление определенными классами продуктов.

8. Литература

1. Hailstone R. Ensuring the Health of IT Systems: The Next Dimension of Enterprise Management. — White Paper Number: 1998-5.3, Bloor Research, 1998.
2. Management Framework for Open Systems Interconnection (OSI) for CCITT Applications. Recommendation X.700. — ITU, 1992.
3. Information Technology — Open Systems Interconnection — Systems Management Overview. Recommendation X.701. — ITU, 1992.
4. Information Technology — Open Systems Interconnection — Structure of Management Information: Management Information Model. — ITU, 1992.
5. Harrington D., Presuhn R., Wijnen B. An Architecture for Describing SNMP Management Frameworks. — Internet-Draft, Nov. 1998. http://www.ietf.org/internet-drafts/draft-ietf-snmpv3-arch-02.txt.
6. Bierman A., Greene M., Stewart B., Waldbusser S. Distributed Management Framework. — Internet-Draft, Aug. 1998. http://www.ietf.org/internet-drafts/draft-ietf-disman-framework-02.txt.
7. Daniele M., Wijnen B., Francisco D., Ed. Agent Extensibility (AgentX) Protocol. Version 1. — RFC 2257, 1998. ftp://ftp.isi.edu/in-notes/rfc2257.txt.
8. Desktop Management Specification. Version 2.00. — DMTF, 1996.
9. Common Information Model (CIM). Core Model White Paper. Version 0.91 (Draft). — Desktop Management Task Force, Jul. 1998.
10. Application Management Specification. Version 2.0. A DMTF Common Information Model Based Approach to Application Management. — Tivoli Systems, 1997.
11. Таранов А., Цишевский В. Java в три года. — Jet Info, 1998, 11/12.
12. Java Management API Overview. — Sun Microsystems, 1996.
13. Unicenter TNG. Версия 2.0. Концепции. — Computer Associates International, 1997.

9. Приложение. Возможные критерии оценки систем управления

9.1. Общие положения

Основными показателями, характеризующими системы управления, являются:

1. полнота набора управляемых объектов;
2. спектр и степень детальности отслеживаемых/управляемых характеристик;
3. полнота набора сервисов управления;
4. расширяемость системы управления;
5. настраиваемость системы управления;
6. степень автоматизации функционирования системы;
7. возможность работы в реальном масштабе времени;
8. возможность прогнозирования поведения управляемых объектов;
9. качество средств разграничения доступа к управляющей информации и сервисам управления;
10. степень защищенности управляющего трафика;
11. технологичность системы управления.

Подразумевается, что система управления способна поддерживать распределенные конфигурации.

Ниже каждый из перечисленных показателей рассматривается более детально. Требования для каждого показателя упорядочены по возрастанию, то есть сначала следуют минимальные требования, затем — дополнения к ним.

9.2. Показатели систем управления

9.2.1. Полнота набора управляемых объектов

В число управляемых объектов должны входить:

1. базовые аппаратно-программные конфигурации компьютеров (персональные компьютеры с ОС Windows NT/95/98, серверные платформы с ОС Unix, Windows NT и Netware);
2. базовый набор активного сетевого оборудования (концентраторы, коммутаторы, маршрутизаторы ведущих производителей);
3. базовый набор периферийных устройств (принтеры, сетевые карты, модемы и т.п.);
4. базовое программное обеспечение (реляционные СУБД, Web-сервис и т.п.);
5. прикладные программные системы;
6. некомпьютерные ресурсы (системы безопасности, специфические устройства, пользователи и т.п.);
7. бизнес-процессы и связи между ними и прочими объектами.

9.2.2. Спектр и степень детальности отслеживаемых/управляемых характеристик

К спектру и степени детальности отслеживаемых/управляемых характеристик можно предъявить следующие требования:

1. возможность отслеживания/изменения базового набора характеристик управляемых объектов (тип, исправность, нагрузка и т.п.);
2. возможность задания собственного набора отслеживаемых/управляемых характеристик;
3. возможность управления степенью детализации отслеживаемых характеристик и задания различной детализации для разных управляемых объектов;
4. возможность отслеживания характеристик в реальном масштабе времени.

9.2.3. Полнота набора сервисов управления

В число сервисов управления должны входить:

1. базовый набор сервисов (автообнаружение, управление событиями, управление нагрузкой, управление безопасностью, реагирование на проблемные ситуации);
2. дополнительные сервисы, разработанные третьими фирмами.

9.2.4. Расширяемость/настраиваемость системы управления

Система управления должна быть расширяемой/настраиваемой по:

1. набору управляемых объектов;
2. спектру и степени детализации отслеживаемых/управляемых характеристик;
3. набору и режиму функционирования сервисов управления;
4. способам реагирования на происходящие события;
5. способам представления для пользователей управляющей информации;
6. способам обработки управляющей информации;
7. режиму протоколирования работы системы управления.

9.2.5. Степень автоматизации функционирования системы

Система управления должна допускать автоматизацию следующих действий:

1. реагирование на выход отслеживаемых характеристик за допустимые границы (в том числе реагирование на отказы);
2. реагирование на нарушение информационной безопасности.

9.2.6. Возможность работы в реальном масштабе времени

Система управления должна предоставлять возможность слежения в реальном масштабе времени за:

1. исправностью управляемых объектов;
2. нагрузкой управляемых объектов;
3. событиями, происходящими в выделенном подмножестве управляемых объектов;
4. характеристиками выделенных управляемых объектов.

Система управления должна реагировать в реальном масштабе времени на:

1. отказы в управляемых объектах;
2. нарушения информационной безопасности;
3. выход характеристик выделенных управляемых объектов за допустимые границы.

9.2.7. Возможность прогнозирования поведения управляемых объектов

Система управления должна предоставлять возможность:

1. предсказания отказов на основе данных о предшествовавших сбоях;
2. предсказания выхода отслеживаемых характеристик за допустимые границы на основе ранее накопленных профилей поведения системы;
3. предсказания нарушения информационной безопасности на основе выявления нетипичного и/или злоумышленного поведения пользователей или иных компонентов информационной системы.

9.2.8. Качество средств разграничения доступа к управляющей информации и сервисам управления

Система управления должна разграничивать доступ к:

1. управляющей информационной базе (по управляемым объектам и компонентам MIB);
2. сервисам управления (по отдельным сервисам, по функциям этих сервисов, по управляемым объектам).

Должно поддерживаться:

1. произвольное управление доступом (с выделением владельца ресурса и назначением прав для именованных субъектов);
2. управление доступом на основе системы правил.

Средства аутентификации должны быть устойчивы к пассивному и активному прослушиванию сети.

Должны присутствовать средства контроля целостности управляющей информации, в том числе для распределенных конфигураций.

9.2.9. Степень защищенности управляющего трафика

По отношению к управляющему трафику должны обеспечиваться:

1. контроль аутентичности;
2. контроль целостности отдельных сообщений;
3. конфиденциальность отдельных сообщений;
4. контроль целостности последовательности сообщений.

9.2.10. Технологичность системы управления

Система управления должна быть масштабируемой по:

1. числу управляемых объектов;
2. числу и степени детальности отслеживаемых характеристик.

Система управления должна поддерживать многоуровневую архитектуру менеджер/агент.

Система управления должна поддерживать как централизованное, так и распределенное администрирование с различными способами выделения зон ответственности (территориальный, функциональный и т.п.).

Должны предоставляться средства восстановления после отказов.

Система управления должна обладать высокой доступностью, быть устойчивой к отказам собственных компонентов.

Jet Info
ИНФОРМАЦИОННЫЙ БЮЛЛЕТЕНЬ
Издается с 1995 года
Издатель: компания Джет Инфо Паблишер

Главный редактор: Галатенко В.А. *(galat@jet.msk.su)*
Технический редактор: Антонов А.Н. *(silver@jet.msk.su)*

Россия, 103006, Москва, Краснопролетарская, 6
тел. (095) 972 11 82, 972 13 32
факс (095) 972 07 91
e-mail: JetInfo@jet.msk.su

Подписной индекс по каталогу Роспечати **32555**