УДК 608.01

Д. Д. Куликов, С. А. Чертков

СИСТЕМА ПОИСКА СРЕДСТВ ТЕХНОЛОГИЧЕСКОГО НАЗНАЧЕНИЯ КАК WEB-СЛУЖБА

Представлен способ реализации системы поиска технологического назначения как web-службы. Рассмотрены положительные и отрицательные стороны этого подхода. Приведены преимущества такого подхода перед подходом с использованием непосредственного доступа к серверу баз данных.

Ключевые слова: *автоматизированные системы, технологическая подготовка производства, проектирование технологических процессов, web-служба, клиент-серверная архитектура.*

Производство в настоящее время представляет собой очень сложный процесс и практически ни одно предприятие не в состоянии охватить его целиком, сохранив при этом рентабельность. Подобному положению дел есть множество причин, но основными являются широкий спектр используемых компонентов, а также малые сроки, выделенные на технологическую подготовку производства, иными словами, производство новых изделий требует широкой

номенклатуры новых деталей, а у предприятия просто нет времени на широкомасштабное техническое переоснащение и подготовку линий.

В поисках решения перечисленных проблем предприятиям зачастую оказывается значительно проще, быстрее и дешевле заказать бóльшую часть компонентов у стороннего производителя. Данное явление называется „аутсорсинг“ и в настоящее время встречается практически повсеместно.

Однако любое решение имеет не только положительные, но и отрицательные стороны. В случае аутсорсинга следствием снижения затрат на выпуск широкой номенклатуры компонентов и деталей для готовых изделий является усложнение проектирования и технологической подготовки производства. Иначе говоря, аутсорсинг позволяет решать проблему производства широкой номенклатуры деталей путем территориального разнесения производства, но тем самым порождает проблему интеграции этих разнесенных систем в единый производственный процесс.

Для решения этой проблемы часто используется виртуальное предприятие, т.е. совокупность организаций, функционирующих в области производства сложных технических устройств, имеющих единое информационное пространство, в частности — единую базу данных технологической информации с возможностью удаленного доступа.

Характеристика информации в цикле ТПП. Технологическая подготовка производства (ТПП) — этап, на котором виртуальное предприятие остро нуждается в едином информационном пространстве. Именно на этом этапе требуется полная информация о ресурсах, необходимых для производства изделий.

Принцип организации единого информационного пространства следующий: технологии не должны возникать и храниться отдельно от конструкторской информации. При его соблюдении, т.е. хранении информации в единой базе данных, сведения о составе изделия и ресурсах представляют собой ту платформу, которая позволяет оперативно выполнять любые сводные расчеты и решать вопросы производственного планирования и управления. Основной информацией, накапливаемой и используемой на этапе ТПП, являются данные о доступных ресурсах, выпускаемых деталях и разработанных технологических процессах [1].

Информация о ресурсах — это списки имеющегося оборудования и оснастки. Она представляет собой множество каталогов, каждый из которых содержит относительно небольшое количество записей. Эта информация, как правило, хорошо формализована и ее хранение и поиск не вызывают проблем [2]. Информация о деталях — списки выпускаемых деталей с их параметрами. Каждый элемент, как правило, связан с соответствующими технологическими процессами и ресурсами. Такие данные значительно сложнее поддаются формализации, поскольку обладают большим количеством совершенно различных параметров.

Групповые и типовые технологические процессы — самая трудноформализуемая часть, которая содержит информацию, необходимую для производства выбранной детали или для разработки процесса ее получения на основе типового. В этом случае имеет место хранение процессов в базе знаний.

Наличие формализованной информации позволяет существенно увеличить производительность труда технологов и сократить время технологической подготовки.

Организация удаленного хранилища. Поскольку большинство предприятий, как уже говорилось, активно используют аутсорсинг и внедряют концепцию „виртуального предприятия“, то указанная технологическая информация должна быть доступна всем участникам кооперации. Оптимальным выходом при этом является организация хранилища технологической информации с удаленным доступом через сеть Интернет.

Может показаться, что никаких сложностей в решении данной задачи нет: все современные СУБД являются клиент-серверными, следовательно, изначально предоставляют возможность удаленного доступа. Однако на практике такой подход совершенно неприменим.

Очевидной проблемой является то, что не обязательно все информационные архивы предприятия хранятся в одной базе данных или даже в базах данных под управлением одинаковых СУБД, т.е. сразу возникает проблема объединения данных и выбора единственного физического хранилища — это очень большой труд, к тому же перечеркивающий все предыдущие наработки предприятия, поскольку приложения, ориентированные на работу со старыми хранилищами (например — СУБД других производителей), скорее всего, не смогут функционировать с новыми.

Вторая проблема — предоставление низкоуровневого доступа непосредственно к серверу баз данных через Интернет — неоправданно большой риск. И дело здесь не в опасности, исходящей от хакеров — взлому потенциально подвержена любая система, а в возрастающей сложности клиентского программного обеспечения. Контроль за корректностью и целостностью данных, добавляемых в базу, целиком возлагается на клиентское ПО, после чего оно дублируется, рассылается по всем клиентам, и в дальнейшем устранение ошибок, добавление новых возможностей и вообще внесение каких-либо изменений становится чрезвычайно дорогостоящим.

Третья проблема связана с безопасностью — для подобных систем необходимы жесткий контроль вводимых данных, четкое распределение полномочий пользователей, в некоторых случаях организация шифрованных каналов связи для исключения возможности перехвата передаваемых данных и прочие меры по повышению защищенности данных.

Четвертая проблема связана с масштабируемостью — при превышении определенного уровня нагрузки один сервер, даже очень мощный, не сможет обеспечивать удовлетворительное время отклика для всех клиентов. Именно поэтому в настоящее время во всех проектах, связанных с предоставлением данных большой аудитории, используются распределенные хранилища.

Конечно, есть множество других проблем, их спектр меняется в зависимости от объема и характера информации, количества пользователей и т.п., но эти представляются наиболее очевидными и общими, а значит, с ними столкнется любая организация, пытающаяся организовать единое информационное пространство.

Подводя итог вышесказанному, можно заключить, что прямой доступ к СУБД хотя и возможен, однако является крупным источником проблем и ни в коем случае не может быть рекомендован. Для более эффективной и безопасной работы необходима прослойка между клиентом, получающим доступ через коммуникационную среду, и СУБД, выполняющей непосредственно функции хранения данных. Эта прослойка будет скрывать от пользователя низкоуровневый интерфейс СУБД и вообще структуру хранилища, предоставляя пользователю простой высокоуровневый интерфейс. Пользовательское приложение в таком случае будет выполнять только функцию получения данных с сервера и конечную обработку — например, визуализацию или предоставление полученных данных в какой-либо программный пакет в качестве исходных данных.

Если присмотреться к полученной структуре приложения, без труда можно увидеть проверенную и хорошо зарекомендовавшую себя „трехуровневую архитектуру“, достоинствами которой являются масштабируемость, конфигурируемость, безопасность, высокая надежность, низкие требования к скорости канала (сети) между терминалами и сервером приложений, низкие требования к производительности и техническим характеристикам терминалов. Конечно, у любого решения есть и недостатки. В случае трехуровневой архитектуры „платой“ за полученные преимущества будет более высокая сложность создания приложения — необходимость реализации прослойки с бизнес-логикой, а также существенные требования к производительности серверных систем. Однако даже для небольшого предприятия расходы на достаточно производительный сервер составят очень небольшую сумму в сравне-

нии с другими затратами и будут быстро компенсированы снижением затрат на технологическую подготовку производства.

Компоненты трехуровневой архитектуры связываются между собой с помощью стандартных интерфейсов. Как правило, связь бизнес-логики с сервисом базы данных сводится к передаче СУБД текстовых строк с SQL-запросами и получению ответных данных.

В случае связи между сервером приложений и терминалом — конечным клиентским приложением — возможно множество вариантов. Существует множество средств для реализации систем с удаленным доступом, однако в случае работы через Интернет наиболее предпочтительным является создание web-сервиса.

Построение web-сервисов ориентируется на использование сервис-ориентированной архитектуры (SOA) и универсального языка разметки — XML. Такой подход позволит не только создавать клиентские приложения для абсолютно разных платформ, поскольку обмен идет в платформонезависимом текстовом формате, но и позволит с легкостью включать сервис в состав каких-либо распределенных приложений.

Web-сервисы представляют собой приложения с созданным по стандарту WSDL (на основе XML) программным интерфейсом. Они используют готовый протокол взаимодействия высокого уровня, реализованный практически для всех платформ, — HTTP, а это означает, что оператор может использовать любую операционную систему без каких-либо дополнительных затрат на адаптацию. Кроме того, применение протокола HTTP позволяет воспользоваться всеми средствами, уже имеющимися для этого протокола, такими как, например, кеширование и балансировка нагрузки.

Высокоуровневый интерфейс сервиса выглядит как набор методов — функций, принимающих параметры и возвращающих значение. Использование такого интерфейса позволяет скрыть от пользователя детали реализации, т.е. сегодня можно хранить данные в MS SQL Server, а завтра — перейти на предоставление интерфейса целой группе различных СУБД, находящихся на разных серверах, при этом внешний пользователь не заметит изменений и ему не потребуется вносить какие-либо изменения в клиентское ПО.

Реализация удаленного хранилища. Выбрав общую архитектуру и оптимальную технологию, приступим к проектированию собственно системы, состоящей из модуля хранения информации — СУБД, модуля доступа к информации — web-сервиса и модуля представления информации — клиентского ПО.

Проектирования СУБД не требуется, поскольку обычно используется система сторонних разработчиков, однако необходимо определить структуру базы данных. Технологическая информация является жестко формализованной, и ее хранение не представляет трудностей. Для каждого типа хранимой информации выделяется набор признаков, которые становятся столбцами таблицы. Таблица полученной структуры создается в базе данных и заполняется информацией. Сведения о вновь добавленной таблице заносятся в реестр таблиц и словарь.

Реестр представляет собой таблицу в базе данных, содержащую уникальный идентификатор таблицы, ее название, сведения о положении ее в дереве каталогов и прочую служебную информацию. Словарь играет важную роль в организации единого пространства имен, так как присваивает каждому параметру уникальный идентификатор и содержит всю информацию об этом параметре: название, тип, возможное поле значений и другую информацию, что позволяет обеспечивать целостность информации в базе данных.

Сервис может быть реализован практически на любом языке и под любой web-сервер — главное, чтобы были соблюдены определенные стандарты. Многие фирмы-поставщики средств разработки ПО поставляют инструменты для упрощения создания web-сервисов. В настоящее время де-факто стандартом для предприятий является операционная система Windows, поэтому зачастую целесообразно использовать технологию .NET, чтобы не ставить

предприятие, пожелавшее установить подобный сервис, перед необходимостью серьезных изменений существующей структуры корпоративной сети.

Среда .NET обеспечивает автоматическое создание SOAP-оберток для передаваемых данных, создает WSDL-описание и скрывает другие сложности реализации стандартного web-сервиса от разработчика.

Сервис предоставляет три группы функций для работы с базой данных: навигация по каталогам, работа с информацией, поиск. Навигация по каталогам — это представление всей информации в базе данных в древовидной форме, где узлы — это либо каталоги с набором подкаталогов (например, подкаталоги „сверла“ и „резцы“ в каталоге „режущий инструмент“), либо уже с необходимыми таблицами (например, таблица „Сверло ГОСТ 4010-77“ в каталоге „Сверла“). Работа с информацией заключается в выдаче клиенту информации по уникальным идентификаторам таблиц, удалении, добавлении и изменении информации. И, наконец, собственно то, ради чего и создавалась система — поиск информации, т.е. выдача записей, удовлетворяющих определенным критериям. Поиск производится с помощью собственного языка описания запросов, язык использует данные словаря и имеет более высокий уровень, чем SQL. Предполагается, что клиент не создает запросы самостоятельно, а вводит данные в нужные поля формы поиска, после чего запрос формируется программой.

При рассмотрении уровня представления данных необходимо отметить следующее: он не входит в состав сервиса и конкретная его реализация зависит именно от клиента. В качестве приложения, удовлетворяющего нуждам большинства потребителей, можно обратить внимание на AJAX-клиент, целиком выполняющийся в web-браузере на стороне клиента. Такой подход, при разумной реализации, позволяет снизить интенсивность обмена информацией и разгрузить сервер. Однако возможны и другие варианты, такие как создание представления полностью на стороне сервера, например, с помощью ASP, или реализация клиентского приложения для используемой клиентом PDM/PLM-системы.

Заключение. Итак, рассмотрев процесс проектирования и реализации web-сервиса для хранения технологической информации, уточним, какие преимущества получают поставщик и пользователь такого приложения.

1. Доступность: web-сервис доступен по всему миру. Нестандартные протоколы и приложения могут быть недоступны в некоторых сегментах сети в силу различных причин, однако доступ к ресурсам по протоколу HTTP открыт практически везде.

2. Универсальность: работа с текстовыми строками позволяет не заботиться о различиях представления бинарных данных на разных платформах, а использование открытых стандартов позволяет web-сервисам органично встраиваться в любое приложение.

3. Независимость: поставщик услуги может изменять, расширять и перенастраивать систему хранения данных, а клиент может использовать различные клиентские приложения. Единственное, что должно оставаться неизменным — интерфейс сервиса, при этом он может расширяться с помощью добавления новых методов без потери совместимости со старыми приложениями.

4. Безопасность: для обмена данными с сервисом можно использовать стандартные механизмы авторизации и организации шифрованных туннелей, разработанные для протокола HTTP и имеющиеся практически во всех web-серверах и клиентах.

Конечно, за все эти преимущества приходится платить, и зачастую в прямом смысле этого слова — объем данных при передаче в тестовом виде значительно превышает их двоичное представление, а следовательно, возрастает сетевой трафик, который нужно оплачивать. Кроме того, больший объем информации, естественно, дольше передается. Однако в условиях современных скоростей передачи данных и тарифов на объем трафика эти затраты будут незначительными по сравнению с полученной выгодой.

СПИСОК ЛИТЕРАТУРЫ

1. *Митрофанов С. П., Куликов Д. Д., Миляев О. Н., Падун Б. С.* Технологическая подготовка гибких производственных систем. М.: Машиностроение, 1987.

2. Справочник технолога-машиностроителя / Под ред. *А. М. Дальского, А. Г. Косилова, Р. К. Мещерякова*. М.: Машиностроение, 2001.

Сведения об авторах

Дмитрий Дмитриевич Куликов — д-р техн. наук, профессор; Санкт-Петербургский государственный университет информационных технологий, механики и оптики, кафедра технологии приборостроения; E-mail: ddkulikov@rambler.ru

Сергей Владимирович Чертков — НИИ „Вектор“, Санкт-Петербург; инженер
E-mail: chertkov.s.a@gmail.com

Рекомендована кафедрой
технологии приборостроения

Поступила в редакцию
14.12.09 г.