1

УДК 007.51

СИСТЕМА ВЕДЕНИЯ ОТРАЖЕННЫХ МОДЕЛЕЙ

Гриневич И.Е.[1], Жевжик С.Е.[2], Зеленева И.Я.[1]
[1]Донецкий национальный технический университет, Украина;
[2]Укрзалізниця ГП ПКТБ АСУ ЖТ, г. Донецк, Украина.

В статье рассмотрена задача разработки специализированной ORM-системы, позволяющей работать с данными по технологиям функционирования железнодорожных объектов. Приведены некоторые существующие аналоги разработанной системы. Рассмотрена базовая структура типизированной модели, используемой на предприятии для записи данных о работе железнодорожных объектов. Предложены методы реализации процессов пользовательской сериализации и десериализации с учетом особых требований, предъявленных к процессам передачи данных.

Введение

В настоящее время повсеместно используются автоматизированные системы управления. Не является исключением и государственное предприятие «Укрзалізниця», которое для реализации сложных технологических процессов использует собственное специализированное программное обеспечение, позволяющее оптимизировать работу с учетом современных реалий. Это программное обеспечение состоит из множества частей, немаловажную роль в которых играет обеспечение рабочих мест и серверов приложений всеми необходимыми данными, для чего и предназначена система ведения отраженных моделей, рассмотренная в данной работе. Для уменьшения выполняемой операторами вручную рутинной работы требуется увеличение количества армов, но для обеспечения их работы не подходят стандартные средства. Ввиду этого возникает потребность в разработке специализированной системы, позволяющей работать с данными по технологиям функционирования железнодорожных объектов.

Система ведения отраженных моделей представляет собой специализированное средство для объектно-реляционного отображения баз данных.

Объектно-реляционное отображение (англ. *Object-relational mapping, ORM*) – технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования, создавая «виртуальную объектную базу данных». Существуют как коммерческие, так и свободные реализации этой технологии [1].

Существующие аналоги

ADO.NET Entity Framework позволяет разработчикам создавать приложения для доступа к данным, работающие с концептуальной моделью приложения, а не напрямую с реляционной схемой хранения. Цель состоит в уменьшении объема кода и снижении затрат на сопровождение приложений, ориентированных на обработку данных [2]. Приложения Entity Framework предоставляют следующие преимущества.

- приложения могут работать концептуальной моделью в терминах предметной области — в том числе с наследуемыми типами, сложными элементами и связями;
- приложения освобождаются от жестких зависимостей от конкретного ядра системы управления базами данных или схемы хранения;
- сопоставления между концептуальной моделью и схемой, специфичной для конкретного хранилища, могут меняться без изменения кода приложения;
- разработчики имеют возможность работать с согласованной моделью объектов приложения, которая может быть сопоставлена с различными схемами хранения, которые, возможно, реализованы в различных системах управления данными;

– несколько концептуальных моделей могут быть сопоставлены с единой схемой хранения.

NHibernate – ORM-решение для платформы Microsoft .NET портированное с Java. Это бесплатная библиотека с открытым кодом, распространяется под лицензией GNU Lesser General Public License.

NHibernate позволяет отображать объекты бизнес-логики на реляционную базу данных. По заданному XML-описанию сущностей и связей NHibernate автоматически создает SQL-запросы для загрузки и сохранения объектов [3].

DataObjects.Net – это библиотека, обеспечивающая хранение обычных .Net-объектов в реляционной базе данных.

Использование библиотеки позволяет существенно сократить время разработки приложений, работающих с базами данных — DataObjects.Net берет на себя практически все функции, связанные с взаимодействием с сервером, выполняя их прозрачно (т.е. не требуя написания кода, который обеспечивает их выполнение) для разработчика. Она автоматически обновляет схему БД, поддерживает все типы отношений, запросы, полнотекстовый поиск, многоязычность, версии, управление правами доступа к объектам, .Net Remoting и многое другое.

Использование DataObjects.Net делает все данные автоматически совместимыми с Microsoft SQL Server, MDSE 2000, Microsoft Access, Oracle, Firebird, MaxDB без дополнительного кода [4].

Базовая структура модели, используемая на предприятии

Государственное предприятие «Укрзалізниця» имеет в своём распоряжении сложную базу данных автоматизированной системы управлению грузовыми перевозками, которая работает под управлением СУБД Oracle. Данные по работе железнодорожных объектов записываются в определённые модели конкретного типа объекта. При этом для четкого контроля последовательность ввода данных в автоматизированной системе управления грузовыми перевозками используется событийная модель. Каждая модель имеет типизированную структуру (рис. 1).

Элементы типизированных моделей системы ведения отраженных моделей для связи между собой используют ссылки, что позволяет выполнять быстрый переход от одного элемента к другому.

Структура типизированных моделей в комплексе автоматизированной системы управления грузовыми перевозками полностью генерируется специализированными средствами. Системы ведения отраженных моделей должна в основном копировать эту структуру, при этом некоторые неиспользуемые поля или таблицы могут отсутствовать для уменьшения объемов используемой памяти на серверах приложений системы ведения отраженных моделей и автоматизированных рабочих местах.

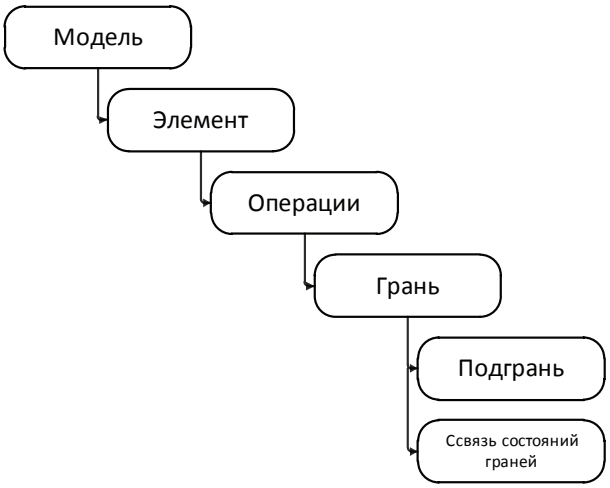

Рисунок 1. Базовая структура модели в комплексе автоматизированной системы управления грузовыми перевозками

Контейнер моделей

Назначение контейнера моеделей – объектное представление моделей автоматизированной системы управления грузовыми перевозками (или любых других моделей системы ведения отраженных моделей), которое реализует функциональность размещения данных в оперативной памяти, обмена данными с внешним программным кодом, сохранение данных на жёсткий жиск и восстановление с жёсткого диска.

Контейнер моделей позволяет сохранять в себе любое количество оперативных моделей, поэтому на этапе выполнения контейнер моделей взаимодействует с компонентами, которые реализуют предсталение моделей в оперативной памяти (библиотечными модельными классами-наследниками для конкретных моделей автоматизированной системы управления грузовыми перевозками).

Контейнер моделей разрабатывается как плагин, который подключается к управлению вызовами, потоками и компонентами.

Внутри себя контейнер моделей содержит коллекцию моделей системы ведения отраженных моделей. Практически всю технологическую функциональность контейнер моделей реализует с помощью этого внутреннего объекта. Можно сказать, что контейнер моделей – обёртка для коллекции моделей, которая превращает последнюю из просто класса в модуль, который подключается к комплексу управления вызовами, процессами и компонентами.

Т.к. контейнер моделей (вычитанные данные) хранятся в оперативной памяти, возможно ускорение работы с содержащимися в нем данными.

Должны использоваться объектно-ориентированные принципы, что позволяет уменьшить объёмы программного кода и формировать определённые базовые структуры и функции объектов.

Однако для большого количества данных необходимы серверы с большим объемом оперативной памяти.

Контейнер моделей должен реализовывать такие функции, как:

- создание и наполнение коллекции типовых моделей;
- обеспечение сохранения и загрузки данных типовых моделей из файлов, которые сохраняются на жёсткий диск в виде контейнеров данных;
- обеспечение обновления коллекции моделей с помощью вставки контейнеров данных с модификациями.

Структура системы ведения отраженных моделей

Комплекс автоматизированной системы управления грузовыми перевозками используется для обеспечения данными автоматизированных рабочих мест. При этом, если каждое автоматизированное рабочее место будет запрашивать информацию непосредственно из базы данных, получится большая нагрузка на сеть и на саму базу данных. В случае расширения функционала, работы с моделями, сбор каких-либо статистических данных и других действий может привести к загруженности автоматизированной системы управления грузовыми перевозками, а следовательно, к увеличению длительности отклика (латентности) комплекса. Следовательно, основными целями системы ведения отраженных моделей являются разгрузка комплекса автоматизированной системы управления грузовыми перевозками, уменьшение сетевого потока данных и обеспечение работы автоматизированных рабочих мест.

Приведенные в разделе 1 аналоги не позволяют выполнить оптимальные настройки по всем приведённым проблемам. Также они не позволяют выполнить оптимальный перенос данных. Для того, чтобы разгрузить автоматизированную систему управления грузовыми перевозками должны быть разработаны специальные сервера приложений, которые выпоняют вычитку информации и её отображение, а также обслуживание автоматизированных рабочих мест (рис. 2). Сервера приложений системы ведения отраженных моделей должны позволять выполнение масштабирования, т.е. один сервер приложений может вычитывать данные из другого, тем самым уменьшая нагрузку на базу

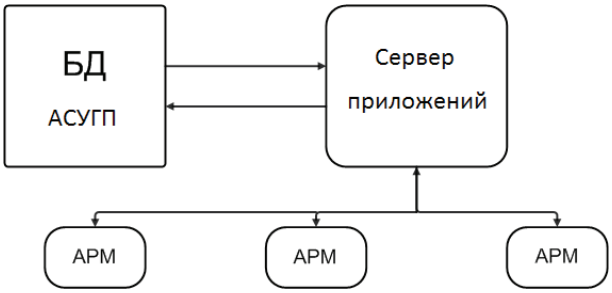

Рисунок 2. Обобщённая структура системы ведения отраженных моделей

данных автоматической системы управления грузовыми перевозками.

В настоящее время в Украине многие автоматизированные рабочие места находятся на станциях, и скорость передачи информации является низкой с вероятными разрывами соединения. Поэтому, количество передаваемых данных должно быть минимальным, что требует отбора минимального количества необходимой информации и дальнейшего её сжатия при передаче.

Структура типизированных моделей в комплексе автоматизированной системы управления грузовыми перевозками полностью генерируется специализированными средствами. Система ведения отраженных моделей должна в основном копировать эту структуру, при этом некоторые неиспользуемые поля или таблицы могут отсутствовать для уменьшения объемов используемой памяти на серверах приложений системы ведения отраженных моделей и автоматизированных рабочих мест.

Процесс генерации

В процессе генерации должно вычитываться описание, на основе которого будет генерироваться код моделей системы ведения отраженных моделей. Считанное описание может быть урезано или дополнено моделями, которые отсутствуют в базе данных. Следующий шаг – на основе описания выполняетсяся генерация программного кода. После этого программный код собирается в библиотеки, которые могут быть использованы разработчиками прикладного кода.

В серверы приложений данные поступают при выполнении запросов, которые также были сгенерированы. Полученная информация формирует контейнеры данных, которые хранятся в оперативной памяти.

Процесс сериализации и десериализации

Объектом разработки авторов является программа, обеспечивающая выполнение процессов сериализации и десериализации.

В настоящее время скорость передачи данных между некоторыми участками критически мала или может периодически обрываться. Поэтому к передаче данных предъявляются особые требования:

- объем передаваемых данных должен быть предельно малым;
- при записи/восстановлении данных должно использоваться минимальное количество памяти;
- скорость передачи данных должна регулироваться (например, между участками с малой пропускной способностью скорость необходимо уменьшать, чтобы избежать потерь данных).

В среде .NET большинство стандартных объектов можно преобразовывать в массив байт. Данный объект называется сериализацией. Обратный ему процесс называется десериализацией. Однако при стандартной сериализации в большинстве случаев записывается лишняя (служебная) информация, что может значительно увеличить размер передаваемых данных. Для уменьшения этих данных был разработан программный проект с поддержкой процессов пользовательской сериализации/

1

десериализации. Кроме того, реализована возможность сжатия архиватором. Также необходимо учитывать, что в большинстве случаев нет необходимости передавать на автоматизированное рабочее место все поля данных. Поэтому информация передается с помощью специальных схем, которые определяют состав передаваемых данных.

Разработанный проект также учитывает особые требования, которые предъявляются к процессу десериализации. Нежелательными являются скачки, т.е. резкие изменения объема выделяемой оперативной памяти, т.к. большая её часть используется для хранения объектов системы ведения отраженных моделей. Для решения этой проблемы в проекте используется последовательная вычитка данных из потока. Итерационное выполнение вычитки данных позволяет существенно сократить объём выделяемой оперативной памяти.

Выводы

Разработанная специализированная система ведения отраженных моделей максимально соответствует всем предъяваленным требованиям, и при этом позволяет получать все необходимые данные. В настоящий момент проект находится в стадии корреляции с предыдущими разработками с целью сохранения данных.

Литература

[1] Object-relational mapping. Материал из Википедии – свободной энциклопедии. Электронный ресурс. Режим доступа: http://ru.wikipedia.org/wiki/ORM

[2] ADO.NET Entity Framework. Электронный ресурс. Режим доступа: http://msdn.microsoft.com/ru-ru/library/bb399572.aspx

[3] NHibernate. Материал из Википедии – свободной энциклопедии. Электронный ресурс. Режим доступа: http://ru.wikipedia.org/wiki/NHibernate

[4] DataObjects.Net. Электронный ресурс. Режим доступа: http://mescontrol.ru/features/do.html