УДК 681.58.011.56
А.С. Гристан, Д.В. Джулгаков, В.Г. Сухоребрый

Использование реляционной СУБД для разработки АСУ

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

Рассмотрен опыт применения реляционной СУБД Interbase для разработки крупных АСУ ТП. Обсуждены достоинства и недостатки такого подхода. Предложен метод преодоления основного недостатка – больших затрат времени на загрузку данных из БД.
Ключевые слова: реляционная СУБД, АСУ, среда разработки программного обеспечения контроллеров, данные, САПР.

Цель работы

Речь пойдет об АСУ крупных промышленных объектов. Конкретным примером такого объекта может служить АСУ атомной электростанции (АЭС). Система управления АЭС обычно включает в себя несколько сотен шкафов управления, множество рабочих станций разных уровней и другое оборудование. Все это объединяется рядом локальных сетей передачи данных.

Объемы данных, используемые АСУ АЭС при ее разработке (проектные данные) и эксплуатации (рабочие данные), достигают значительных размеров. Причем некоторые виды данных используют как при разработке, так и при эксплуатации АСУ. Поэтому возникает желание обеспечить доступ и хранение проектных и рабочих данных на единой унифицированной основе. Такой основой может быть коммерческая СУБД.

Использование коммерческой СУБД для хранения рабочих данных не является новостью, по крайней мере, для крупных промышленных АСУ. Однако для хранения проектных данных производители САПР, как правило, применяют базы данных собственной разработки [4,5,6].

Цель этой работы – показать, какими преимуществами и недостатками обладает использование коммерческой реляционной СУБД для хранения проектных данных и каким образом свести недостатки к минимуму.

Как это делается в среде СИНТАР 2007?

СИНТАР 2007 [1] – это среда разработки программного обеспечения контроллеров (*softlogic*), ориентированная на крупномасштабные АСУ ТП.

Базы данных СИНТАР 2007 реализованы на основе коммерческой СУБД Interbase [2]. Эта система имеет клиент-серверную архитектуру и поддерживает SQL. Она широко распространена в мире, надежна и обладает хорошими скоростными характеристиками. Кроме того, существует аналог (Firebird), распространяемый на бесплатной основе.

В базах данных содержатся все проектные данные разработки:

1 варианты программно-аппаратных конфигураций:
   - структура аппаратуры и свойства ее элементов (схем, узлов, устройств);
   - линии связи между элементами аппаратуры;
   - локальные сети и их свойства;
   - размещение сигналов на линиях связи;
   - размещение программ на узлах сети;
   - задачи и их свойства;
2 описания сигналов:
   - пользовательские типы сигналов и ограничения;

- свойства конкретных сигналов;
- правила автогенерации подчиненных сигналов;
- варианты пользовательских классификаций сигналов;

3 исходные программы на языках СИНТАР 2007:

- пользовательские типы данных;
- пользовательские классы;
- объекты, в том числе переменные и сигналы;
- вершины и дуги FBD-схем;
- текстовые подпрограммы.

Указанные разновидности данных связаны между собой: в программах сигналы выступают в роли глобальных переменных, в конфигурации узлы локальной сети могут содержать программы, а по связям между узлами передаются сигналы. Логично было бы поместить все проектные данные в одну базу. Тем не менее пришлось отказаться от подобного решения. Причины децентрализации:

1) поскольку крупные АСУ создаются, как правило, несколькими группами или даже организациями разработчиков, необходимо обеспечить разделение ответственности как при разработке, так и при сопровождении системы;
2) использование общего сервера при совместной разработке АСУ несколькими организациями дорого и ненадежно.

Пример структуры баз данных показан на рис.1.

Согласно естественному разделению крупной АСУ на стойки или шкафы управления базы данных сигналов и программ также разделены по стойкам. Кроме того, конфигурации, программы и описания сигналов размещены в разных базах данных.

Преимущества

Применение коммерческой СУБД в разработке всех проектных данных позволяет использовать многочисленные стандартные механизмы и инструменты, которыми оснащены современные СУБД и сопутствующие программные продукты:

1. С помощью простого интерфейса, предоставляемого САПР, пользователь описывает структуру применяемых типов сигналов, которая средствами СУБД преобразуется в необходимые метаданные: таблицы, поля, ограничения и зависимости.
2. Сервер СУБД имеет встроенные механизмы контроля различных ограничений, как заданных пользователем (например, различные отношения между значениями полей), так и создаваемых САПР (уникальность имен, каскадное удаление объектов и т.п.).
3. С помощью SQL-запросов пользователь может генерировать фрагменты программ и проектную документацию. С этой целью предусмотрен встроенный язык скриптов, а также подключен язык php. Документацию можно также генерировать с помощью коммерческих средств генерации отчетов.
4. В прикладной области могут существовать зависимости между сигналами, трактуемые в терминах реляционной модели как отношение «главный-подчиненный». Например, аналоговый входной сигнал может сопровождаться целой «свитой» связанных сигналов, принимаемых непосредственно с аппаратуры или вырабатываемых программно. Генерация описаний подчиненных сигналов происходит автоматически с помощью механизмов сервера базы данных. Необходимую для этого информацию пользователь сообщает САПР в виде определенного набора правил [3].
5. Использование таких средств сервера, как триггеры и хранимые процедуры, позволило легко организовать фиксацию изменений любых проектных данных и формирование автоматического журнала коррекций, а также автоматически генерировать данные на основе пользовательского интерфейса.
6. Средства сервера позволяют организовать автоматическую поддержку целостности данных в рамках одной базы.

Недостатки

1) Большие затраты времени при загрузке данных из БД в систему внутренних объектов.
2) Необходимость специальных средств синхронизации между отдельными базами данных, использующими общие объекты.

Второй недостаток связан с необходимостью децентрализации баз данных крупной АСУ, о чем упоминалось выше. Он преодолевается достаточно просто: БД хранит только ссылки на используемые внешние данные, доступ к ним осуществляется непосредственно по месту хранения.

На устранении первого недостатка остановимся более подробно.

Сокращение времени загрузки

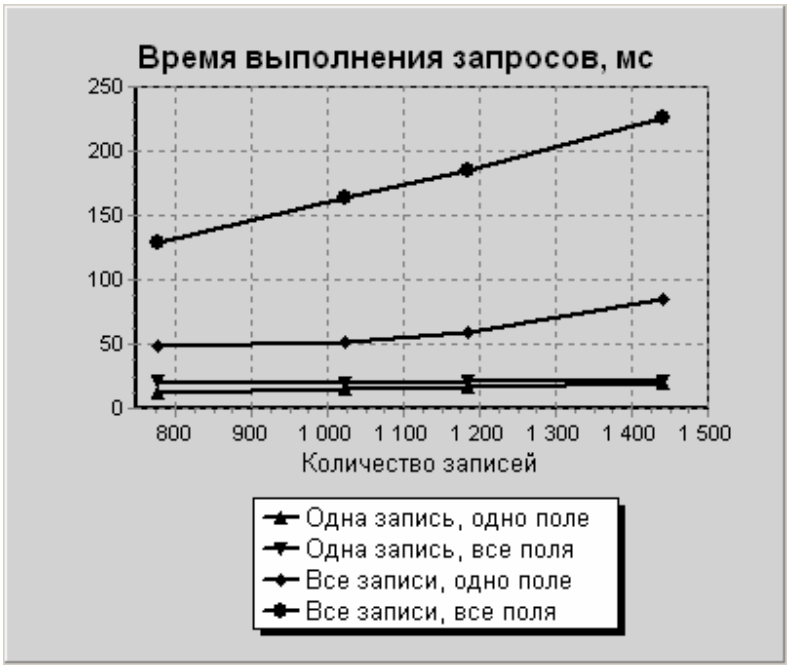
Метод сокращения времени загрузки продемонстрируем на конкретном

элементе САПР – генераторе исходного Паскаль/C++ кода из FBD-схем. Этот генератор работает в два этапа:

1) построение внутренних объектов путем загрузки данных из БД;
2) генерация исходного кода на заданном языке.

На первом этапе для загрузки каждого объекта выполнялся один или несколько SQL-запросов к базам данных. Рис. 2 дает представление об объеме этой работы, особенно если учесть, что количество объектов большинства из показанных классов для средней по величине программы шкафа управления достигает нескольких тысяч.

Для анализа ситуации была получена статистика различных по объему SQL-запросов к одной из таблиц четырех баз данных программ (см. рис. 3).



Эти данные получены на ПК с тактовой частотой процессора 2.4 ГГц, памятью 750 Мб и жестким диском 80 Гб. Исследовались запросы четырех типов (без сортировки и с сортировкой по первичному ключу):

1) выборка одного поля одной записи;
2) выборка всех полей одной записи;
3) выборка одного поля всех записей;

4) выборка всех полей всех записей.

Каждый запрос выполнялся по четыре - шесть раз с осреднением результатов.

Выяснилось, что наличие сортировки практически не влияет на время выполнения запросов: сортировка несколько увеличивает время только в четвертом варианте запроса.

Видно, что выборка одной записи практически не зависит от количества записей в таблице, поскольку для поиска использовался существующий индекс. Отметим также, что времена выборки одного поля и всех полей одной записи практически совпадают.

Наиболее интересными представляются времена выполнения запросов 2 и 4. Естественно, выборка всех записей занимает больше времени и практически линейно зависит от количества записей в таблице. Тем не менее, обращает на себя внимание тот факт, что время выборки одной записи не намного меньше, чем время выборки всех записей: например, для таблицы размером 777 записей время выборки одной записи составляет 21 мс, а время выборки всех записей – 129 мс. Для таблицы размером 1441 запись – 21 и 226 мс, соответственно. Среднее по четырем исследуемым таблицам время выборки одной записи для запроса четвертого типа составляет 0.159 мс, причем, с увеличением числа записей в таблице оно уменьшается.

Это свидетельствует о значительных накладных расходах сервера СУБД на обработку одного запроса.

Из приведенных данных следует однозначный вывод: для минимизации времени загрузки информации из БД следует минимизировать количество запросов. Используемый метод пообъектных запросов вызывает неоправданные затраты времени загрузки.

Новый метод заключается в следующем:

1) загрузить каждую таблиц БД в таблицу в памяти (MRT – memory resident table) одним запросом;
2) построить объекты, выбирая необходимые данные из MRT.

Объявление MRT на языке Delphi выглядит примерно так:

```
TMObject = class
      TableName: string;   //Совпадает с именем таблицы БД
      ObjectID, OwnerID, TypeID, ... : array of integer;  //Целые поля
      ObjectName, VarValue, Info, ... : array of string;  //Строковые поля
      Count: integer;      //Фактическое количество записей
      constructor Create;
      destructor Destroy; override;
      procedure Load;      //Загрузка из БД одним запросом
      procedure IncCount;  //Приращение счетчика записей и длины массивов
end;
```

Поля таблицы БД в MRT представлены одноименными динамическими

массивами соответствующих типов. Конструктор создает эти массивы, а процедура Load заполняет их результатами единственного SQL-запроса, выбирающего все поля всех записей с сортировкой по первичному ключу. При добавлении записи в MRT используется процедура IncCount, которая увеличивает Count на 1 и, при необходимости, наращивает длину динамических массивов на некоторую постоянную добавку. При построении объектов генератора для поиска в отсортированном поле-массиве используется дихотомия, а в не отсортированном – простой перебор.

В результате общее время работы генератора для программы средних размеров (около 2000 вершин FBD) уменьшилось с 92 до 4 с.

Выводы

Непопулярное, на первый взгляд, решение использовать реляционную СУБД для хранения проектных данных при разработке АСУ на самом деле оказалось достаточно привлекательным. Оно имеет целый ряд преимуществ перед использованием БД собственной разработки, которые заметно упрощают создание и использование САПР, а также расширяют ее возможности. Что касается недостатков – здесь, как мы надеемся, показаны пути их устранения.

Список литературы

1. Сухоребрый В.Г. СИНТАР-3 и СИНТАР 2007: сравнительный анализ систем разработки контроллерного ПО / В.Г. Сухоребрый, А.С. Гристан, Д.В. Джулгаков // Открытые информационные и компьютерные интегрированные технологии: сб. науч. тр. Нац. аэрокосм. ун-та «ХАИ». – Вып. 38. - Х., 2008. - С. 170 - 176.
2. А.Н. Ковязин. Мир InterBase. Архитектура, администрирование и разработка приложений баз данных в InterBase/ А.Н. Ковязин, С.М. Востриков.// Firebird/Yaffil – М.: КУДИЦ-ОБРАЗ, 2002. – 2-е изд., доп.- 496 с.
3. Сухоребрый В.Г. Базы данных сигналов в среде СИНТАР 2007 / В.Г. Сухоребрый, А.С. Гристан, Д.В. Джулгаков // Открытые информационные и компьютерные интегрированные технологии: сб. науч. тр. Нац. аэрокосм. ун-та «ХАИ». – Вып. 39. - Х., 2008. - С. 215 - 223.
4. TRACE MODE 6 SOFTLOGIC: программирование контроллеров. http://www.adastra.ru/products/dev/softlogic/
5. ISaGRAF - программирование контроллеров. http://tornado.nsk.ru/catalog/isagraf.shtm
6. Многоплатформенная распределенная SCADA/Softlogic S3 //http://s3.com.ua/

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

Поступила в редакцию 15.06.09.

Використання реляційної СУБД для розроблення АСУ

Розлянуто досвід використання реляційної СУБД Interbase для розроблення великих АСУ ТП. Обговорено переваги та недоліки такого підходу. Запропановано метод усунення основного недоліку – великих витрат часу на завантаження даних із БД.

Ключові слова: реляційна СУБД, АСУ, середовище розроблння програмного забезпечення контролерів, дані, САПР***.***

Using relational DBMS for industrial control development

The experience of using Interbase relational DBMS for the large-scale industrial control development is considered. The approach merits and demerits is discussed. A method to overcome the main demerit – big time costs for the loading information from a database – is proposed.

Keyword: Relational DBMS, industrial control development, softlogic, data, scada.