База данных калибровок (проект)

И.Б.Логашенко

27 декабря 2001

1 Логическая структура базы данных калибровок

Калибровкой считается массив вещественных (double) коэффициентов. База данных калибровок предназначена для организации централизованного хранения и унифицированного доступа к калибровкам для всех систем. Калибровка может иметь как предопределенную размерность (количество коэффициентов), так и переменную размерность.

Для идентификации *типа калибровок* используется 3 идентификатора: имя системы, имя алгоритма калибровки и имя набора данных. Все калибровочные массивы принадлежащие одному типу имеют один и тот же формат и смысл. Пример типа калибровки: калибровка калориметра по космике — система "CSI", алгоритм "COSMIC", имя данных "ECOEFF".

Внутри одного и того же типа калибровок могут существовать различные *версии*, идентифицируемые названием. Основное назначение версии состоит в возможности одновременного хранения калибровок, имеющих один и тот же смысл и предназначенных для анализа одних и тех же данных, но полученных либо с использованием разных данных, либо с помощью модифицированного алгоритма. В подобной организации БД существует неоднозначность: вместо определения новой версии можно определить новый алгоритм калибровки, и наоборот. Общая рекомендация следующая: если два набора калибровок взаимозаменяемы с точки зрения программы-клиента, то эти два набора являются двумя версиями одного и того же типа калибровки. Для любого типа калибровки существует версия, используемая по умолчанию.

Отдельная калибровка внутри одного типа/версии идентифицируется по времени создания, началу и концу интервала времени, для которого калибровка *действительна* (должна использоваться). Внутри одного типа/версии калибровки для каждого момента времени действительно не более одной калибровки, т.е. не допускается наличия нескольких калибровок одного и того же типа/версии, действительных для одного и того же момента времени. Такое ограничение существенно упрощает логику поиска калибровки в БД (не с точки зрения запроса к БД, а с точки зрения интерпретации результатов программой-клиентом). Возможна ситуация, когда

для каких то моментов времени не существует действительной калибровки определенного типа/версии.

Возможно разбиение одного калибровочного массива на одинаковые части, идентифицируемого целым числом. Такое разбиение удобно в случае, когда система состоит из нескольких идентичных частей, которые могут калиброваться независимо друг от друга (например, торцы торцевого калориметра). В этом случае уникальный калибровочный массив дополнительно идентифицируется номером части.

2 Физическая структура базы данных

В качестве сервера БД предполагается использовать СУБД PostgreSQL. На сервере определяются две таблицы: ClbrInfo и ClbrData. В первой таблице хранится описание всех определенных типов калибровок и их версий. Во второй таблице хранятся собственно калибровки.

Приблизительная структура таблицы ClbrInfo:

Колонка	Содержание
Id	Уникальный ID типа/версии калибровки
System	имя системы
Algorithm	имя алгоритма калибровки
Name	имя данных
Version	имя версии
Size	размер (если 0, то размер калибровки переменный)
Parts	количество частей (если 0, то калибровка не разбивается на части)
Comment	текстовое описание данного типа/версии калибровки
Parameters	параметры данного типа/версии калибровки

Приблизительная структура таблицы ClbrData:

Колонка	Содержание
Id	Уникальный ID калибровки
Type	ID типа/версии калибровки (из таблицы ClrbInfo)
Time	время создания калибровки
BeginTime	начало интервала времени действительности
EndTime	конец интервала времени действительности
Comment	текстовая информация о конкретной калибровке
Parameters	параметры калибровки
Part	номер части калибровки
Data	собственно калибровочный массив (array)

Весь административный доступ к базе данных (фактически, к таблице ClbrInfo) предполагается осуществлять с помощью Web-интерфейса. Доступ к самим калибровкам предполагается осуществлять с помощью C++ API и Web-интерфейса.

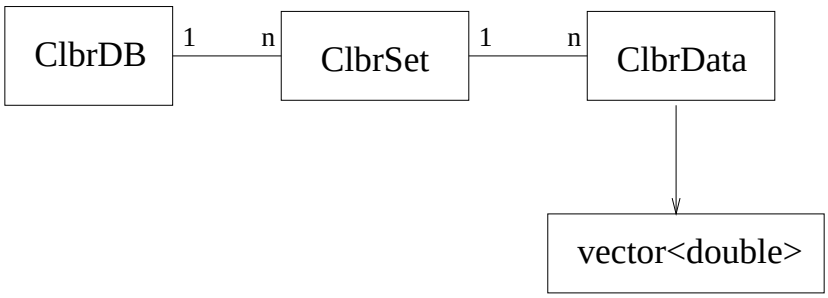

Рис. 1: Диаграмма классов API к базе данных калибровок

3 C++ API к базе данных

Доступ к БД калибровок осуществляется с помощью 3 классов: ClbrDB, ClbrSet и ClbrData (рисунок 1). Все описания классов, встречающиеся далее, являются очень приблизительными и будут уточняться по мере разработки.

Класс ClbrDB предназначен для осуществления связи с БД и операций над БД в целом. Связь с БД открывается при создании объекта класса ClbrDB и закрывается при его удалении. Типичные методы объекта класса ClbrDB: получить списки систем, алгоритмов и т.п., начать и закончить транзакцию, создать объект класса ClbrSet, представляющий отдельный тип/версию калибровки. Пример интерфейса класса ClbrDB:

```
class ClbrDB {
  ClbrDB(const char* username, const char* password = 0,
         const char *host=0, const char* port=0);
  ~ClbrDB();
  list<string> getSystemList();
  list<string> getAlgorithmList(string system);
  list<string> getNameList(string system, string algorithm);
  list<string> getVersionList(string system, string algorithm,
                              string name);
  ClbrSet* getSet(string system, string type, string data,
                  string version=0);
}
```

Класс ClbrSet предназначен для работы с определенным типом/версией калибровки. Создание объекта класса ClbrSet является методом объекта класса ClbrDB. При одновременной работе с несколькими типами/версиями калибровок создаются несколько объектов класса ClbrSet с помощью одного и того объекта класса ClbrDB (при необходимости возможно использование нескольких объектов класса ClbrDB). Типичные методы объекта класса ClbrSet: выбрать отдельную калибровку, либо список калибровок данного типа/версии, удовлетворяющего условию, создать “пустую” калибровку,

получить описание либо параметры данного типа/версии калибровки, и т.п.
Пример интерфейса класса ClbrSet:

```
class ClbrSet {
  list<ClbrData> select(string request, int part_from=0, int part_to=0);
  ClbrData* selectUnique(string request, int part=0);
  list<ClbrData> selectValid(string time, int part_from=0, int part_to=0);
  ClbrData* selectValidUnique(string time, int part=0);
  ClbrData* makeDefault(int part=0);
  int getSize();
  int getNumberOfParts();
  string getComment();
  ...
}
```

Класс ClbrData предназначен для работы с отдельной калибровкой. Объекты класса ClbrData создаются объектом класса ClbrSet либо при чтении данных из БД, либо при создании "пустой" калибровки. Последние используются для пополнения БД новыми калибровками. Типичные методы объекта класса ClbrData: методы доступа к собственно калибровочному массиву (наследованные от vector«double»), чтение/изменение параметров калибровки (диапазон действительности, комментарий и т.п.), занесение/замена/удаление калибровки из БД. При операциях с базой данных может осуществляться автоматическое изменение интервалов действительности соседних калибровок так, чтобы не возникало их перекрытие. Пример интерфейса класса ClbrData:

```
class ClbrData: vector<double> {
  /* The following pair of get/set methods defined for all
     parameters of the calibration, such as Comment, Time,
     BeginTime, EndTime, Parameters */
  string getComment();
  void setComment(string comment);
  ...
  void delete();  // Delete calibration from DB
  void replace(); // Replace existing calibration in DB
  void insert();  // Insert new calibration to DB
}
```

4 Что желательно добавить

1. В API следует добавить определение исключений
2. Следует обдумать необходимость и возможность определения не только double, но и integer, boolean и др. калибровок

3. Необходимо разработать унифицированный доступ к параметрам калибровок, типов калибровок и т.п. (поле Parameters). В настоящее время предполагается хранить параметры в формате XML.

4. Возможно следует определить права доступа к типам калибровки, чтобы пользователи не могли изменять “чужую” калибровку.

5 Пример программы

```
// Связаться с БД
ClbrDB *db = new ClbrDB("online");

// Создать объект для работы с калибровкой калориметра по космике
// Использовать версию по умолчанию
ClbrSet *cosmic = db->getSet("CSI","COSMIC","ECOEFF");

// Найти калибровку, действительную на 12:00 12-Дек-01
ClbrData *ecoeff = cosmic->selectValidUnique("12-Dec-01 12:00:00");

// Распечатать значения калибровочных коэффициентов
for( int i=0; i<ecoeff->size(); i++ ) {
  printf("%f\n",(*ecoeff)[i]);
}

// В массиве array лежат калибровочные коэффициенты, полученные в
// результате калибровки калориметра по космике. Требуется занести
// новую калибровку в базу данных.

// Создать ‘‘пустую’’ калибровку
ClbrData *newclbr = cosmic->makeDefault();

// Заполнить калибровку новыми значениями из массива array
for( int i=0; i<newclbr->size(); i++ ) {
  (*newclbr)[i] = array[i];
}

// Определить параметры калибровки
newclbr->setComment("During calibration we were watching TV");
newclbr->setTime(newclbr->TimeNow()); // Time = current time from DB
newclbr->setStartTime(newclbr->getTime()); // StartTime = Time
newclbr->setEndTime(ClbrDB::InfinityTime); // EndTime = +infinity

// Занести калибровку в БД
newclbr->insert();
```