

РАЗРАБОТКА РАСПРЕДЕЛЕННОЙ ЭКОЛОГИЧЕСКОЙ ГЕОИНФОРМАЦИОННОЙ СИСТЕМЫ

А. Н. Маренков, А. В. Козлов

Воронежский государственный университет

В статье рассматривается задача разработки распределенной геоинформационной системы, ориентированной на работу с экологическими данными. Дается оригинальное решение поставленной задачи, позволяющее менять системы управления базами данных, представления данных, интерфейсы клиента непосредственно во время эксплуатации программы, прозрачно для пользователя. Результатом применения данных концепций является максимально гибкая и расширяемая система, столь необходимая в условиях современного бизнеса.

1. ВВЕДЕНИЕ

Экологическая геоинформационная система (ГИС) характеризуется большим объемом специфической информации. Эта информация включает в себя данные о концентрациях вредных веществ, о распределении уровней звукового давления, об отражающих свойствах препятствий, об источниках выброса и тому подобное. Геоданные, не содержащие информации в экологическом смысле (например, возвышенность без коэффициентов поглощения и отражения шума), не представляют в данных системах никакой ценности.

Экологические же данные, в большинстве своем, являются результатом работы многих разнородных программных комплексов: измерительных программных комплексов, расчетных модулей, экологических справочников, программ расчета предельно допустимого выброса и тому подобное.

Таким образом, экологическая геоинформационная система должна обеспечивать возможность работы с географическими данными и экологическими данными, как с единым набором данных.

2. ЗАДАЧИ РАЗРАБОТКИ ЭКОЛОГИЧЕСКОЙ ГИС

Рассмотрим задачу разработки экологической ГИС. Требуется обеспечить возможность работы нескольких отделов экологического проектирования, разобщенных территориально, над одним проектом, а так же возможность одновременной работы над проектом внутри одного отдела. Кроме этих задач следует не забыть

о задачах экологических геоинформационных системах в целом: возможность задания топологических данных, обеспечение связи экологических и топологических данных, отображение результатов расчета.

Необходимо еще раз отметить, что специфические особенности этого проекта заключаются в многопользовательской направленности разрабатываемой системы. Все существующие системы на данный момент однопользовательские. Кроме этого, авторами статьи предлагается оригинальная концепция организации системы, которая будет описана ниже. Данная концепция позволяет использовать данные из различных источников (экологических программ и ГИС систем), объединяя их «на лету», выдавая пользователям и синхронизируя результаты их работы с данными.

На разработку данной системы накладывались следующие ограничения:

- отделы должны иметь возможность одновременной работы над одним набором данных (в данном месте под набором данных подразумевается совокупность топографической и экологической информации);

- отделы должны иметь возможность автономной работы, независимо от других отделов организации;

- экологические данные необходимо получать непосредственно из сторонних экологических систем (например: для задач расчета шумового давления данные должны быть получены из программы анализа шума);

- топологические данные могут быть получены из файлов данных сторонних ГИС систем (например: ArcView, MapInfo, AutoCAD).

Из первого пункта следует необходимость создания сервера, который будет организовыв-

вать синхронизацию данных и контролировать процесс работы пользователей с данными. Из второго пункта следует, что создание единого центрального сервера невозможно. В противном случае, все отделы зависели бы от центрального сервера, и в случае нарушения его функционирования ни о какой автономной работе речи быть не может. Таким образом, каждый отдел должен быть снабжен локальным сервером. Для того чтобы эти серверы обладали одними и теми же данными, они должны синхронизироваться друг с другом. То есть, система разбивается на группу подсистем, каждая из которых будет иметь свой сервер.

В основу подсистемы (части системы в рамках одного подразделения организации) мы положим трехуровневую модель построения клиент-серверных приложений. Подробнее об этой модели можно узнать в [4]. На рис. 1 представлена схема трехуровневой модели в общем виде.



Рис. 1. Трехуровневая модель

Программные системы, результатами работы которых являются экологические данные, хранят их в базах данных. Примерами таких данных могут являться таблицы источников выброса, таблицы выбрасываемых веществ, таблицы расчетных точек и расчетных площадок.

Именно эти базы данных мы будем подразумевать, при обозначении «БД» на схеме. Наша система будет получать эти данные из базы при помощи сервера приложения, и помещать их обратно, после редактирования.

Сервер приложений, в трехуровневой модели выполняющий роль посредника между клиентами и базой данных, в нашей системе будет отвечать за объединение топологических и экологических данных, отправку их клиентам, синхронизацию результатов работы клиентов и синхронизацию с другими серверами.

Под абстрактными клиентами, изображенными на рис. 1, в нашей системе понимаются клиентские части системы, обладающие графическим интерфейсом и наделенные функциями ввода и отрисовки топоосновы, редактирования экологических данных, отображение результатов расчета.

Мы расширим трехуровневую модель руководствуясь принципами, приведенными в [2], добавив в нее интерфейс взаимодействия с файлами топоосновы и интерфейс взаимодействия серверов между собой. На рис. 2 представлен проект подсистемы.

3. АЛГОРИТМ СИНХРОНИЗАЦИИ ДАННЫХ В ЛОКАЛЬНОЙ СЕТИ

Основными алгоритмами работы сервера являются алгоритмы синхронизации данных внутри локальной сети и между серверами.

Для решения задачи локальной синхронизации мы предлагаем следующий алгоритм (результат синтеза алгоритмов, предложенных в [3]): пусть файл с геоданными внутри себя некий идентификатор, который будет равен максимальному идентификатору в этом файле. Пусть, также, менять его может только сервер. На стороне клиента реализована такая логика, что вновь создаваемые элементы получают идентификатор не меньший, чем максимальный идентификатор. А удаляемые элементы лишь помечаются на удаление, но физически остаются в файле. Тогда после работы, клиент отправляет файл на сервер, где сервер сравнивает хранимый максимальный идентификатор с идентификаторами всех элементов, и те из них, у которых идентификатор больше хранимого, добавляет в общий файл проекта. После чего удаляет физически, помеченные на удаление элементы.

Изменения в данных также накапливаются на стороне клиента, после чего передаются серверу, вместе с файлом геоданных. Сервер обрабатывает их аналогичным образом, после чего помещает в базу данных.

Так как все данные хранятся в единой базе — это обеспечивает общий и одновременный доступ всех пользователей к этой информации. Основная нагрузка по обеспечению целостности данных, при этом ложится на сервер приложений, который выступает сервером данных для клиентов и, одновременно, элементом системы синхронизации серверов.

Как мы видим, подразумевается, что отдел работает автономно, то есть не происходит никакого обмена информацией с внешней средой. Каким же образом тогда организуется работа нескольких отделов над одним проектом?

Очевидно, что каждый отдел должен иметь свою копию проекта, что, несмотря на некоторую избыточность, добавляет устойчивости системе.



Рис. 2. Проект подсистемы

Однако в определенные моменты эти данные должны быть синхронизированы. Эта задача так же возлагается на сервер системы. Для решения этой задачи, сервер фиксирует все изменения данных, проходящих через него (эта возможность вытекает из упомянутой выше особенности модели). После чего данные изменения должны быть доведены до всех остальных серверов.

Серверы не могут обмениваться попарно этой информацией, так как для n серверов этот алгоритм даст $n(n-1)/2$ соединений, что является также неоптимальным. Поэтому будет логичным

выделить на период синхронизации некий сервер синхронизации, роль которого может выполнять любой из серверов. Его задача будет собрать синхронизационную информацию со всех серверов, объединить ее и рассылать обратно каждому серверу пакеты обновлений. Таким образом, мы получаем всего $(n-1)$ соединений. Проект системы в целом представлен на рис. 3.

В общем случае, сервер синхронизации (синхросервер) может выбираться каждый раз для конкретной синхронизации. Наделение любого сервера возможностью проводить син-

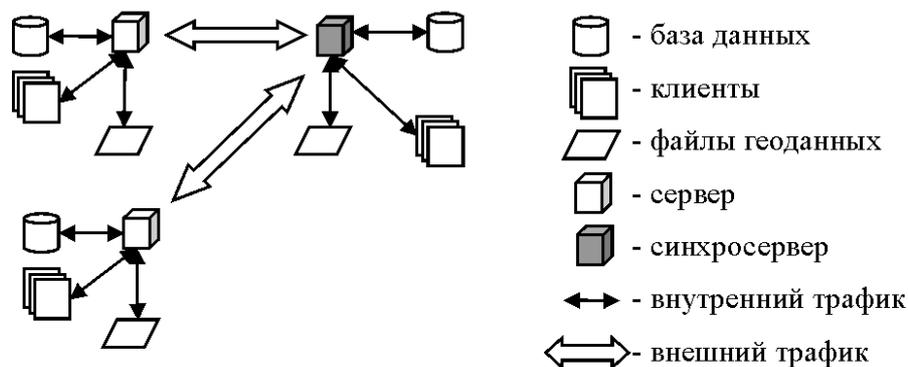


Рис. 3. Проект системы

хронизацию добавляет системе надежности и устойчивости.

4. ОСОБЕННОСТИ ПРОЕКТИРОВАНИЯ КЛИЕНТСКОЙ ЧАСТИ ПРОЕКТА РАСПРЕДЕЛЕННОЙ ЭКОЛОГИЧЕСКОЙ ГЕОИНФОРМАЦИОННОЙ СИСТЕМЫ

Второй существенной частью системы является клиент. Эта та часть, которая непосредственно взаимодействует с пользователем. Согласно постановке задачи, она должна реализовывать возможность отображения топоосновы, редактирования топоосновы, отображения результатов расчета и взаимодействия с сервером.

В начале работы клиент получает пакет данных от сервера. Этот пакет данных клиент разбивает на топологические данные и привязанные экологические данные. После этого клиент заполняет внутренние структуры графического представления топоосновы. Для привязанной информации создаются локальные таблицы, которые заполняются данными из пакета.

Как же должен быть спроектирован клиент, какие модули должны в нем содержаться, чтобы он мог выполнять поставленные задачи? Мы предлагаем следующую концепцию: графические данные и экологические данные разделяются. Графические данные удобно представлять в каком-то упорядоченном виде. Для этого мы введем слои — некие контейнеры для топологических данных. Эти слои могут быть слоями-группами и слоями с топоданными. Слои-группы содержат другие слои-группы или слои с топоданными (аналог папок в операционной системе). К каждому слою с топоданными может быть привязана таблица с дополнительными экологическими данными. Каждый слой также обладает настройками отображения содержимого.

Также клиентская часть содержит модуль печати, который отвечает за вывод результатов работы на печать, или в отдельный графический файл (BMP, JPG и т.п.). В отличие от существующих систем, мы предлагаем реализовать модуль печати в виде панелей: панель карты, панель легенды, панель контрольного отрезка и т.п. Размещая эти панели на листе, пользователь может получить именно тот макет печати, который ему необходим.

Необходимой составляющей клиентской части является модуль графических преобразований топоосновы. Он содержит в себе основные операции, для редактирования топоосновы:

создание, перемещение, вращение, копирование и прочие операции с элементами топоосновы.

При редактировании элементов топоосновы, или записей привязанных таблиц, происходит пометка этих элементов. При отправке результатов работы клиентской части на сервер, отправляются только измененные данные. Далее эти данные проходят обработку на сервере, и сведения об изменениях заносятся в лог-файл сервера.

Для получения гибкого, настраиваемого интерфейса клиентского приложения можно использовать технологию динамического создания всего интерфейса на основе анализа конфигурационных файлов [3]. Эти файлы клиент может при запуске запрашивать с сервера, что обеспечит актуальность клиентского приложения, при каких-либо изменениях в структуре данных без переустановки приложения на всех машинах у пользователей. Так же пользователь получит возможность настраивать интерфейс под себя в процессе работы приложения, а по его завершении клиент создаст файл с пользовательскими настройками и сохранит его на сервере.

5. ЗАКЛЮЧЕНИЕ

Таким образом, задача разработки распределенной экологической геоинформационной системы стоит на стыке трех задач: разработка ГИС системы, создание экологической системы и проектирование распределенной системы. Объем статьи не позволяет более детально описать процесс разработки этой системы. Однако необходимо отметить, что данные концепции распределенной экологической геоинформационной системы были реализованы на практике и показали высокую эффективность и надежность. Они применяются для создания систем для экологических проектных организаций, имеющих большую территориальную разрозненность своих отделений.

ЛИТЕРАТУРА

1. Коннолли Т., Бегг К. Базы данных. Проектирование, реализация и сопровождение. Теория и практика — М.: Вильямс, 2003. — 1440 с.
2. Уилсон С., Мэйплс Б., Лэндгрейв Т. Принципы проектирования и разработки программного обеспечения. — М.: Русская редакция, 2002, — 736 с.
3. Бакнел М. Фундаментальные алгоритмы и структуры данных в Delphi. — М.: ДиаСофтЮП, 2003. — 506 с.

*Статья принята к опубликованию
25 декабря 2006 г.*