Тема 24. Автоматизация проектирования БД

Современный этап разработки БД характеризуется широким использованием средств автоматизации проектирования (CASE-средств) - программных систем, поддерживающих построение и сопровождение информационных приложений на всех этапах их жизненного цикла.

Этапы проектирования

Жизненный цикл программного обеспечения (ПО) – непрерывный процесс, начинающийся с момента принятия решения о создании ПО и заканчивающийся при завершении его эксплуатации.

Наиболее распространены следующие модели жизненного цикла ПО: каскадная, с промежуточным контролем, спиральная.

Каскадная модель

Последовательность этапов:

Анализ→ проектирование→ реализация→ внедрение→ сопровождение.

Достоинства модели: простота, возможность планирования сроков завершения этапов и затрат.

Недостатки: несоответствие реальному процессу создания ПО.

Модель с промежуточным контролем

Схема следования этапов представлена на рис. 24.1.

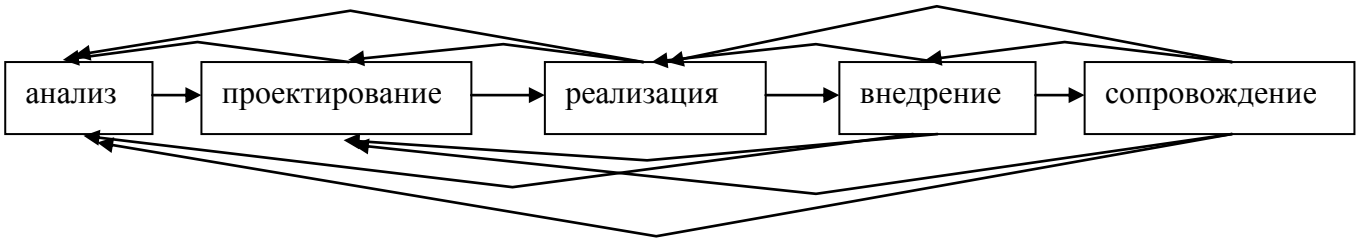


Рис. 24.1. Модель с промежуточным контролем.

После каждого этапа производится проверка достигнутого результата на соответствие требованиям. Если результат не удовлетворяет требованиям, то происходит возврат к одному из предыдущих этапов. Эта модель в большей степени, чем предыдущая соответствует реальному процессу создания ПО,

обеспечивает высокую надежность проекта. Недостаток модели – большие сроки разработки.

Спиральная модель

Спиральная модель (рис. 24.2) позволяет устранить недостатки предыдущих моделей. Большое внимание уделяется начальным этапам – анализу и проектированию. Реализуемость технических решений проверяется с помощью создания прототипов разрабатываемой системы. Неполное завершение работ на очередном этапе не является препятствием для перехода на следующий этап – несделанная работа будет завершена на следующих витках спирали. Одно из достоинств модели – быстрое создание начальной версии системы, которая в дальнейшем заменяется более мощными версиями.

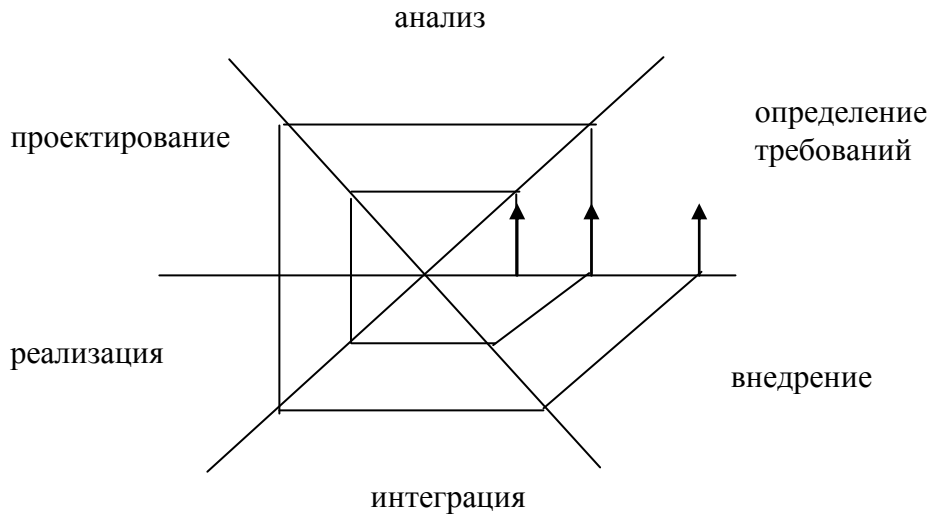


Рис. 24.2. Спиральная модель.

Модели структурного проектирования

Различают модели структурного и объектно-ориентированного проектирования.

Структурный анализ и проектирование системы - рассмотрение ее с общих позиций с последующей детализацией и представлением в виде иерархической структуры. На верхнем уровне обычно представляется функциональное описание системы.

Для графического представления функций системы и отношений между данными используются:

- Диаграммы сущность-связь (ER);
- Диаграммы потоков данных (DFD);
- Метод структурного анализа и проектирования (SADT);
- Иерархические схемы вход-обработка-выход (HIPO);
- Диаграммы Варнье-Орра.

Диаграммы потоков данных

Модель системы строится как иерархия диаграмм потоков данных, описывающих процесс преобразования информации. Диаграммы верхних уровней иерархии, или контекстные диаграммы, определяют основные процессы или подсистемы с внешними входами и выходами. Их декомпозиция выполняется с помощью диаграмм более низкого уровня.

Основными компонентами диаграмм потоков данных являются:

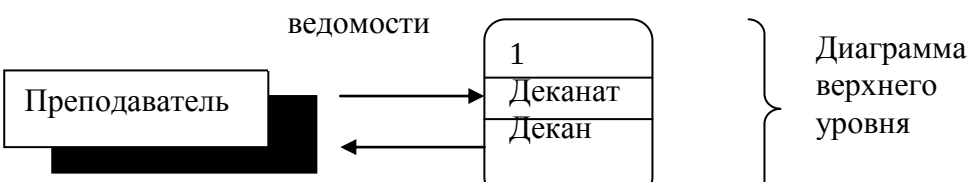
- внешние сущности – источники или потребители информации;
- системы/подсистемы, преобразующие информацию и порождающие новые потоки;
- процессы преобразования входных потоков в выходные;
- накопители данных – абстрактные устройства для хранения информации;
- потоки данных.

Пример. Опишем работу деканата в терминах потоков данных (рис. 24.3).

Внешние сущности (пользователи деканата) – преподаватели и студенты.

Система – сам деканат, подсистем нет.

Процессы преобразования информации: обработка экзаменационных ведомостей, заявлений студентов, разработка учебных планов и т.д.



Объектно-ориентированные модели проектирования

Стандартом в области объектно-ориентированного моделирования является язык UML – унифицированный язык моделирования. UML предназначен для спецификации, визуализации, конструирования и документирования программных и иных систем.

Типы диаграмм UML:

- прецедентов использования;
- классов;
- состояний,
- активности;
- следования;
- сотрудничества;
- компонентов;
- размещения.

Диаграммы состояний, активности, следования и сотрудничества описывают поведение системы. Диаграммы следования и сотрудничества характеризуют взаимодействие объектов. Диаграммы компонентов и размещения описывают физическую реализацию системы.

Диаграммы прецедентов использования описывают функциональность ИС, видимую пользователями системы. Функциональности изображаются в

виде прецедентов использования – типичных взаимодействий пользователя с системой. Прецедент изображается в виде овала.

Диаграммы классов описывают статическую структуру классов. В состав диаграмм входят классы, объекты и взаимоотношения между ними. Класс изображается прямоугольником, в котором указывается имя класса, атрибуты, операции. Объекты изображаются аналогично. Можно указывать дополнительную информацию: интерфейсы, ассоциации и связи между классами, отношения агрегации и обобщения классов.

Диаграммы состояний описывают поведение объектов во времени. Состояние объекта представляется в виде прямоугольника с закругленными углами, содержащего имя состояния и значения атрибутов объекта. Переход от одного состояния к другому происходит при наступлении событий.

Диаграммы активности представляют частный случай диаграмм состояний. Состояния рассматриваются как выполнения операций. Переход в новые состояния происходит при завершении операции. Диаграммы активности задают процедурное, синхронное управление, используются для описания операций классов.

Диаграммы следования определяют временную последовательность передаваемых сообщений и указывают участвующие в передаче объекты. Диаграммы следования двумерны: первое измерение – объекты взаимодействия, второе – время.

Диаграммы сотрудничества описывают взаимодействие объектов системы, направленное на достижение некоторого результата.

Диаграммы компонентов определяют архитектуру системы.

Диаграммы размещения задают конфигурацию компонентов.

Пример. Библиотека. Диаграмма прецедентов использования.

На диаграмме (рис. 24.4) указаны прецеденты использования (администрирование, поиск книги, учет книг, составление отчетов) и пользователи (читатель, администратор). Стрелки показывают связи прецедентов и пользователей.

CASE-системы

CASE-системы принято классифицировать по следующим признакам:

- ориентация на этапы жизненного цикла,
- функциональная полнота,
- тип модели,
- степень независимости от СУБД,
- допустимые платформы.

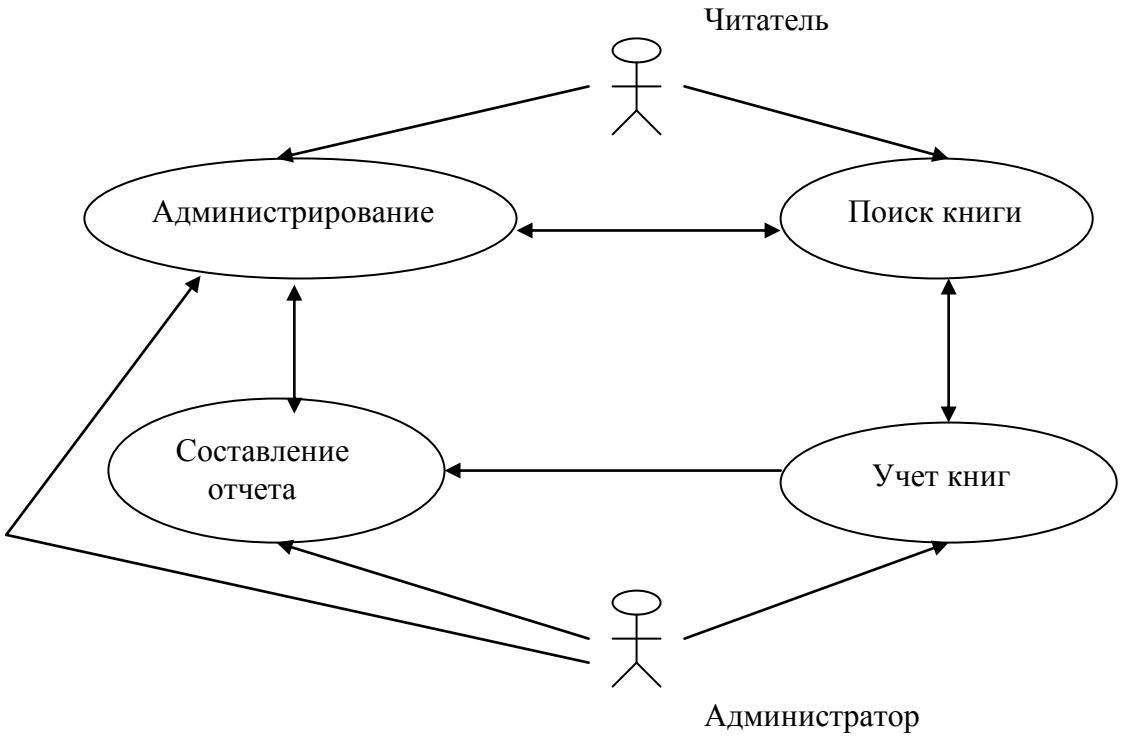


Рис. 24.4. Диаграмма прецедентов использования.

Например, по типу модели CASE-системы разделяются на три разновидности: структурные, объектно-ориентированные и комбинированные. Структурные CASE-системы основаны на методах структурного и модульного программирования, методах структурного анализа и синтеза. Объектно-ориентированные системы используют методы ООП. Комбинированные средства поддерживают как структурные, так и объектно-ориентированные методы.

CASE-система проектирования баз данных ERwin

Система автоматизирует проектирование БД. Имеется два уровня моделирования: логический (ER-диаграммы) и физический (схема БД для СУБД).

На логическом уровне данные представляются так, как они выглядят в реальном мире. Используются понятия сущностей, атрибутов и связей сущностей.

На физическом уровне модель зависит от СУБД.

При переходе от логического к физическому уровню сущности переходят в таблицы, атрибуты - в поля таблиц, связи - в связи таблиц (или таблицы).

Построение модели проходит несколько уровней детализации. Проектирование начинается с уровня сущностей. Далее следует детализация сущностей – уровень атрибутов. Потом устанавливаются связи сущностей типа 1:1, 1:M, M:M. Имеется возможность разделения общей модели на несколько подмоделей (областей).

Готовая логическая модель далее преобразуется в физическую модель конкретной СУБД.