ЛАБОРАТОРНАЯ РАБОТА №2

Курс «Технологии разработки программного обеспечения»

Тема: Проектирование к макетирование программного продукта.

Цель: Научиться проектировать простейшие системы и составлять документацию по проектированию программного продукта.

Вариант 3

«Информационная система кинотеатра»

Оглавление

[ХАРАКТЕРИСТИКА ИНФОРМАЦИОННОЙ СИСТЕМЫ 3](#_Toc87655880)

# ХАРАКТЕРИСТИКА ИНФОРМАЦИОННОЙ СИСТЕМЫ

Описание атрибутов сущностей информационной систесы

|  |  |  |
| --- | --- | --- |
| № | Сущность | Атрибуты |
| 1 | Фильмы | Код\_Фильма, Название фильма, Часы |
| 2 | Зал | Код\_Зала, Зал, Код\_Уровня, Статус |
| 3 | Стоимость | Код\_Смотрящего, Код\_Фильма, Стоимость |
| 4 | Уровень | Код\_Уровня, Уровень, Статусность |
| 5 | Смотрящий | Код\_Смотрящего, Номер билета, Фамилия, Код\_Уровня, |

Связь между сущностями

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Родительская сущность | | Дочерняя сущность | | Тип связи |
| Название | Атрибут | Название | Атрибут |
| Фильмы | Код\_Фильма | Оценки | Код\_Фильма | один - ко -многим |
| Смотрящий | Код\_Смотрящего | Стоимость | Код\_Смотрящего | один - ко -многим |
| Зал | Код\_Зала | Смотрящий | Код\_Зала | один - ко -многим |
| Уровень | Код\_Уровня | Зал | Код\_Уровня | один - ко -многим |

Описание сущности «Смотрящий»

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Поле | Тип | Размер | Длина, байт | Свойства | | |
| Маска ввода | Индексированное поле | Обязательное поле |
| 1 | Код\_Смотрящего | Счетчик | Длинное целое | 4 | - | Да | Да |
| 2 | Номер билетп | Короткий текст | Длинное целое | 6 | 000000 | Нет | Нет |
| 3 | Фамилия | Короткий текст | Длинное целое | 30 | - | Нет | Нет |
| 9 | Код\_Уровня | Числовой | Длинное целое | 4 | SELECT [Уровень].[Код\_Уровня], [Уровень].[Уровень] FROM Уровень; | | |
| Всего: | | | | 44 |  |  |  |

Описание сущности «Уровень»

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Поле | Тип | Размер | Длина, байт | Свойства | | |
| Маска ввода | Индексированное поле | Обязательное поле |
| 1 | Код\_Уровня | Счетчик | Длинное целое | 4 | - | Да | Да |
| 2 | Уровень | Короткий текст | Длинное целое | 30 | - | Нет | Нет |
| 3 | Статусность | Короткий текст | Длинное целое | 30 | - | Нет | Нет |
| Всего: | | | | 64 |  |  |  |

Описание сущности «Стоимость»

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Поле | Тип | Размер | Длина, байт | Свойства | | |
| Маска ввода | Индексированное поле | Обязательное поле |
| 1 | Код\_Смотрящего | Числовой | Длинное целое | 4 | SELECT [Смотрящий].[Код\_Смотрящего], [Смотрящий].[Номер смотрящего], [Ученики].[Фамилия], FROM Смотрящий; | | |
| 2 | Код\_Фильма | Числовой | Длинное целое | 4 | SELECT [Фильмы].[Код\_Фильма], [Фильмы].[Название фильма] FROM Фильмы; | | |
| 3 | Стоимость | Числовой | Байт | 4 | 0 | Нет | Нет |
| Всего: | | | | 12 |  |  |  |

Описание сущности «Зал»

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Поле | Тип | Размер | Длина, байт | Свойства | | |
| Маска ввода | Индексированное поле | Обязательное поле |
| 1 | Код\_Зала | Счетчик | Длинное целое | 4 | - | Да | Да |
| 2 | Зал | Короткий текст | Длинное целое | 30 | 0\-? | Нет | Нет |
| 3 | Код\_Уровня | Числовой | Длинное целое | 4 | SELECT [Уровень].[Код\_Уровня],] FROM Зал ORDER BY | | |
| 4 | Статус | Числовой | Длинное целое | 4 | Нет | Нет | Нет |
| Всего: | | | | 42 |  |  |  |

Описание сущности «Фильмы»

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Поле | Тип | Размер | Длина, байт | Свойства | | |
| Маска ввода | Индексированное поле | Обязательное поле |
| 1 | Код\_Фильма | Счетчик | Длинное целое | 4 | - | Да | Да |
| 2 | Название Фильма | Короткий текст | Длинное целое | 30 | - | Нет | Нет |
| 3 | Часы | Числовой | Длинное целое | 4 | 000 | Нет | Нет |
| Всего: | | | | 38 |  |  |  |

Контрольные вопросы

1. В чем отличие предварительного проектирования от детального?

В процессе внешнего проектирования структурируются цели проекта, определяется взаимодействие с пользователем, но не учитываются многие детали. Во время детального внешнего проектирования определяются внешние спецификации каждой функции для пользовательского проекта (таблица решений).

1. Перечислите и кратко опишите архитектурные системные паттерны

Репозиторий - Все совместно используемые подсистемами данные хранятся в центральной базе данных, доступной всем подсистемам. Репозиторий является пассивным элементом, а управление им возложено на подсистемы.

Клиент/сервер - Данные и процессы системы распределены между несколькими процессорами. Паттерн имеет три основных компонента: набор автономных серверов, (предоставляют сервисы другим подсистемам), набор подсистем - клиентов (которые вызывают сервисы, предоставляемые серверами) и сеть (служит для доступа клиентов к сервисам). Клиенты должны знать имена серверов и сервисов, в то время как серверам не надо знать имена клиентов и их количество. Клиенты получают доступ к сервисам, предоставляемым серверами посредством удаленного вызова процедур.

Обьектно - ориентированный, Модель предметной области (Domain Model), модуль таблицы (Data Mapper) - Бизнес - логика крайне сложна, имеется множество правил и условий, оговаривающих различные варианты поведения системы. Система представляется состоящей из совокупности связанных между собой обьектов. Объекты представляют сервисы (методы) другим объектам и создаются во время исполнения программы на основе определения классов обьектов. Объекты скрывают информацию о представлении состояний и, следовательно, ограничивают к ним доступ.

Многоуровневая система (Layers) или абстрактная машина - В соответствии с паттерном "Многоуровневая система" структурные элементы системы организуются в отдельные уровни со взаимосвязанными обязанностями таким образом, чтобы на нижнем уровне располагались низкоуровневые службы и службы общего назначения, а на более высоких - объекты уровня логики приложения. При этом взаимодействие и связывание уровней происходит сверху вниз. Связывания обьектов снизу-вверх следует избегать.

Потоки данных (конвейер или фильтр) - Система состоит из функциональных модулей, которые получают на входе данные и преобразуют их некоторым образом в выходные данные (конвейерный подход). Каждый шаг обработки данных реализован в виде преобразования. Преобразования могут выполняться последовательно или параллельно, обработка данных может быть пакетной (пакетный последовательный паттерн) или поэлементной.

1. Перечислите и кратко опишите паттерны управления.

Вызов - возврат (сценарий транзакции - частный случай). Вызов программных процедур осуществляется "сверху - вниз", то есть управление начинается на вершине иерархии процедур и через вызовы передается на нижние уровни иерархии.

Диспетчер. Один системный компонент назначается диспетчером и управляет запуском и завершением других процессов системы и координирует эти процессы. Процессы могут протекать параллельно.

Передача сообщений - В рамках данного паттерна событие представляет собой передачу сообщения всем подсистемам. Любая подсистема, которая обрабатывает данное событие, отвечает на него.

Управляемый прерываниями - При использовании данного паттерна внешние прерывания регистрируются обработчиком прерываний, а обрабатываются другим системным компонентом.

1. Что такое «связность модуля»?

Перечислите и кратко опишите типы связности модуля с указанием значения силы связности для каждого типа. Связность модуля – внутренняя характеристика модуля, характеризующая меру прочности соединения функциональных и информационных объектов внутри одного модуля. Связность модуля характеризует степень его «плотности», степень зависимости его частей и направленности на решение определенной задачи. Чем выше связность модуля, тем меньше «ручек управления» на модуле и тем они проще. При проектировании модулей нужно стремиться к высокой связности, ибо чем выше связность, тем лучше спроектирован модуль.

Существует 7 типов связности:

- Функциональная связность - 10 (сильная)

- Последовательная связность - 9

- Информационная связность - 7

- Процедурная связность - 5

- Временная связность - 3

- Логическая связность - 1

- Связность по совпадению – 0 (слабая)

1. Что такое «сцепление модулей»? Перечислите и кратко опишите типы сцепления модулей.

Сцепление модулей - это мера относительной независимости модулей. Слабое сцепление определяет высок уров независимости модулей.

Модули являются полностью независимыми, если каждый из них не содержит о другом никакой информации. Чем больше информации о другом модуле в них используется, тем менее они независимы и тем более сцеплены.

Независимое сцепление возможно, если модули не вызывают друг друга и не обрабатывают одну и ту же информацию.

Модули сцеплены по данным, если они имеют общие простые элементы данных, которые передаются от одного модуля к другому как параметры.

Модули сцеплены по образцу, если в качестве параметров используются структуры данных.

Модули сцеплены по общей области, если они разделяют одну и ту же глобальную структуру данных.

Модули сцеплены по управлению, если какой-либо из них управляет решениями внутри другого с помощью передачи флагов, переключателей и т.д. Модуль сцеплен по внешним ссылкам, если у него есть доступ к данным другого модуля через внешнюю точку входа.

Модули сцеплены по кодам, если коды их команд объединены друг с другом, используют общий участок памяти.

1. Опишите следующие фундаментальные паттерны проектирования: делегирование, неизменяемый объект, интерфейс, MVC.

В разработке ПО, паттерн делегирования— это способ, которым объект внешне выражает некоторое поведение, но в реальности передаёт ответственность за выполнение этого поведения связанному объекту. Шаблон делегирования является фундаментальной абстракцией, которая поддерживает композицию, примеси и аспекты.

MVC шаблон разделяет работу веб-приложения на три отдельные функциональные роли: модель данных, пользовательский интерфейс и управляющую логику. Таким образом, изменения, вносимые в один из компонентов, оказывают минимально возможное воздействие на другие компоненты

Неизменяемый объект — в объектно-ориентированном программировании объект, который не может быть изменён после своего создания. Объект может быть неизменяемым как полностью, так и частично. Например, применение директивы const к какому-либо члену класса в C++ делает объект частично неизменяемым. В некоторых случаях объект считается неизменяемым с точки зрения пользователя класса, даже если изменяются его внутренние поля. Как правило, неизменяемый объект получает все внутренние значения во время инициализации, либо значения устанавливаются в несколько этапов, но до того, как объект будет использован.

В информатике, шаблон интерфейса не является особым шаблоном среди шаблонов проектирования. Он является общим методом для структурирования компьютерных программ для того, чтобы их было проще понять. В общем, интерфейс — это класс, который обеспечивает программисту простой или более программно-специфический способ доступа к другим классам. Интерфейс может содержать набор объектов и обеспечивать простую, высокоуровневую функциональность для программиста; он может обеспечивать более чистый или более специфический способ использования сложных классов; он может использоваться в качестве «клея» между двумя различными API; и для многих других целей.