Этап анализа при разработке программного обеспечения является основным этапом при использовании жёстких методологий разработки. На данном этапе полученные на этапе выявления требований функциональные требования к программной системе представляются в виде отношений между классами предметной области. Создается аналитическая модель предметной области путем выявления классов и отношений между ними. При создании аналитической модели крайне важно ограничиться лишь теми классами, которые являются частью словаря предметной области. Это поможет сделать её кратким, простым и понятным описанием поведения системы [книга uml 2].

Классы анализа должны представлять собой четкую абстракцию предметной области и должны проецироваться на реальные бизнес-понятия предметной области.

Данная программная система состоит из четырех логических модулей, каждый из которых содержит классы, соответствующие определенным логическим аспектам поведения системы: это классы, содержащие в себе информацию о заведениях города, необходимые для оформления заказов; классы, содержащие информацию о блюдах, предлагаемых в меню данных заведений; классы, описывающие заказ, который клиенты могут оформить в существующих заведениях; классы, описывающие работников заведений, в обязанности которых входит обработка создающихся в приложении заказов. Общая схема диаграммы классов анализа представлена в приложении ?.

На рис. 1 представлена часть диаграммы классов, описывающая классы, хранящие информацию о заведениях системы.

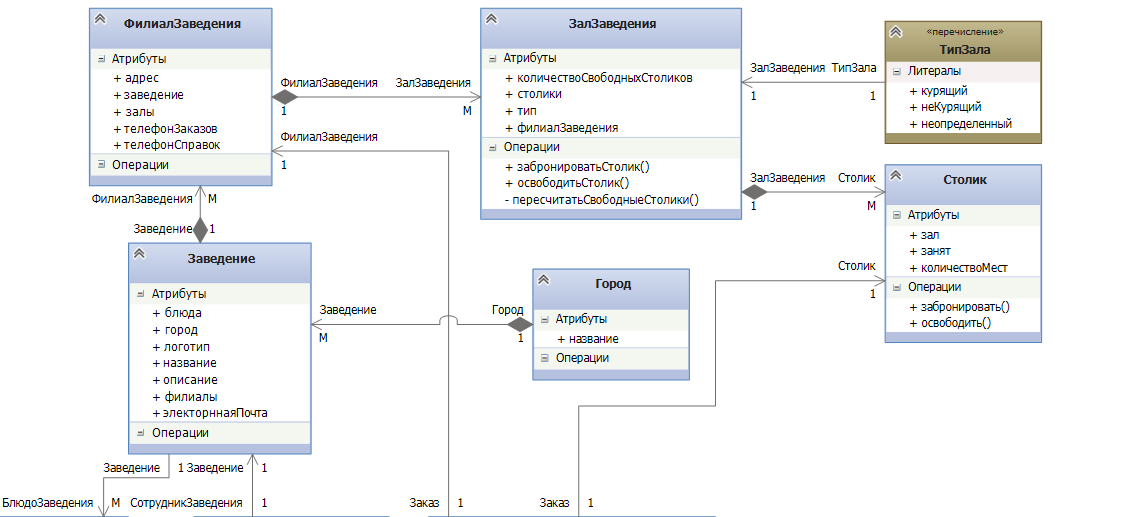


Рис. 1. Классы заведения на диаграмме классов анализа

Как видно на рис. 1, диаграммы классов анализа, описывающие поведения заведений состоят из пяти классов и одной сущности типа перечисление.

Класс «Город» связан отношением композиции с классом «Заведение» с кратностью связи 1: М. Это означает, что класс «Город» может содержать в себе множество объектов класса «Заведение». При этом их связь является жесткой: при исчезновении объекта класса «Город» исчезают все объекты класса «Заведение».

Класс «Заведение» содержит в себе базовую информацию о заведении города. Он связан отношением композиции с классом «ФилиалЗаведения» с кратностью 1: М. Также он связан отношением ассоциации с классом «СотрудникЗаведения» с кратностью 1: 1. Это означает, что у одного заведения может быть только один сотрудник, отвечающий за обработку поступающих этому заведению заказов. Также класс «Заведение» связан отношением ассоциации с кратностью 1: М с классом «БлюдоЗаведения» (рис. 2), который необходим для разрешения связи М: М между классами «Заведение» и «Блюдо».

Класс «ФилиалЗаведения» связан с помощью отношения композиции с классом «ЗалЗаведения» с кратностью 1: М. Также класс «Заказ» связан с классом «ФилиалЗаведения», т.к. заказ доставки и самовывоза требует указания в объекте класса «Заказ» адреса, по которому клиент может получить готовый заказ.

Класс «ЗалЗаведения» связан с помощью отношения композиции с классом «Столик» с кратностью 1: М. Вся иерархия отношений композиции между классами, описывающими заведения представляет собой иерархическую структуру заведений города для данной программной системы. Также отношением ассоциации перечисление «ТипЗала» связано с классом «ЗалЗаведения». Данное перечисление необходимо для указания при заказе столика предпочитаемого клиентом типа зала заведения. Класс «ЗалЗаведения» имеет методы, с помощью которых реализуется поведение, необходимое для бронирования столика в заведении: забронировать столик, освободить столик, а также метод, необходимый для пересчета хранящегося в классе параметра количества свободных столиков в данном зале.

Класс «Столик» необходим для хранения информации о столике заведения. Он предоставляет информацию о том, занят ли он, или нет, необходимую для определения только свободных столиков, при выборе клиентом столика во время оформления заказа на бронирование места в заведении. Также класс «Заказ» (рис. 3.) связан отношением ассоциации с классом «Столик», храня тем самым информацию о столике заведения, который указан в каждом заказе на бронирование в заведении. Класс «Столик» имеет 2 метода, необходимых для осуществления операции бронирования и освобождения столика.

На рис 2. Представлена часть диаграммы классов, содержащая классы, описывающие блюда, представленные в меню различных заведений города.

Как видно на рис. 2, классы, описывающие блюда заведения состоят из двух классов-сущностей и перечисления, описывающего категорию блюда.

Класс «Блюдо» содержит в себе информацию о блюде, а перечисление «КатегорияБлюда» связано с данным классом с помощью отношения ассоциации.

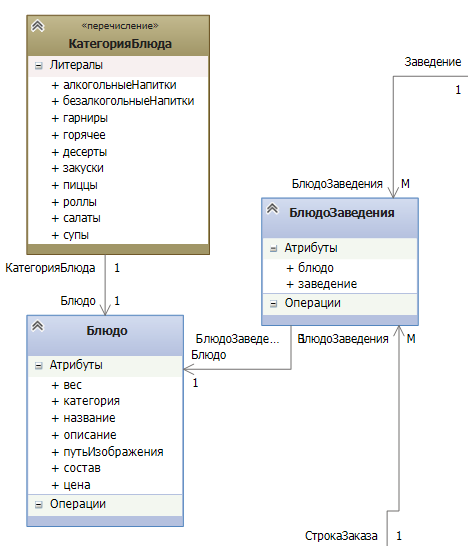


Рис. 2. Классы блюд на диаграмме классов анализа

Класс «БлюдоЗаведения» необходим, как уже говорилось выше, для разрешения связи М: М между классами «Блюдо» и «Заведение». Также класс «СтрокаЗаказа», являющийся разрешением связи М: М Классов «БлюдоЗаведения» и «КорзинаЗаказов» (рис. 3) связан с классом «БлюдоЗаведения» отношением ассоциации с кратностью 1: М.

На рис. 3 представлены классы системы, описывающие заказы и работников заведений, обрабатывающих поступающие заказы.

Класс «СтрокаКорзиныЗаказов» как уже говорилось выше является классом, разрешающим отношение М: М между классами «КорзинаЗаказов» и «БлюдоЗаведения». Данный класс содержит информацию о блюде, добавленном в корзину и количестве порций данного блюда. Класс «КорзинаЗаказов» содержит в себе экземпляры класса «СтрокаКорзиныЗказов», поэтому они связаны между собой отношением композиции с кратностью связи 1: М.

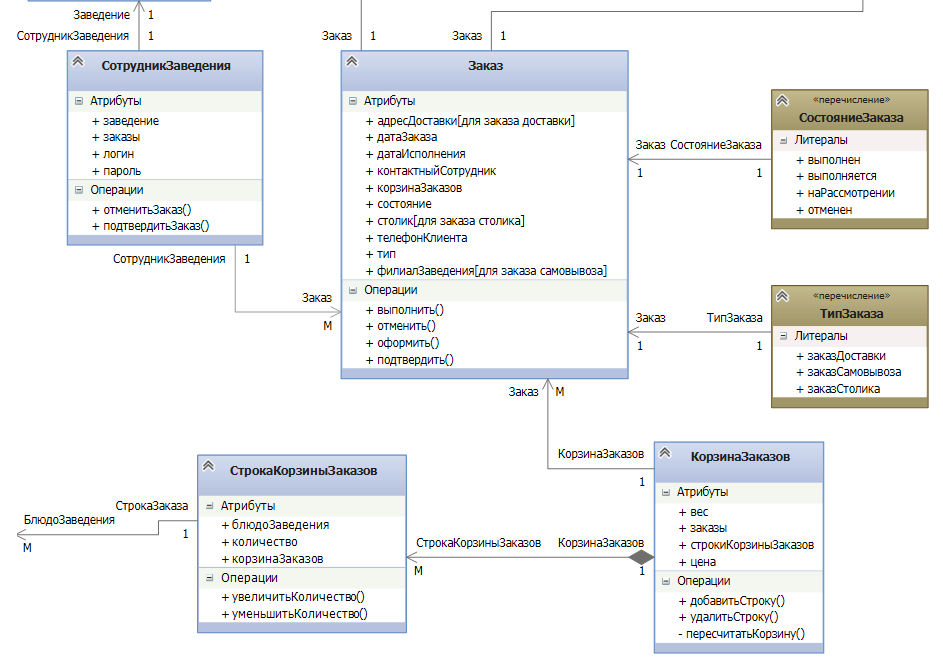


Рис. 3. Классы заказов и сотрудников заведений на диаграмме классов анализа

Класс «СтрокаКорзиныЗаказов» содержит методы, увеличивающие или уменьшающие количество порций блюда в строке корзины.

Класс «КорзинаЗаказов» содержит в себе информацию о строках добавленных в корзину и общих параметрах корзины: общая цена всех блюд в корзине и общий вес блюд в корзине. Данный класс связан отношением ассоциации с классом «Заказ» с кратностью 1: М. Также, класс содержит методы по созданию строк корзины заказов при добавлении нового блюда в корзину и метод пересчета общих параметров корзины после добавления или удаления блюд из корзины.

Класс «Заказ» является основным классом программной системы, содержащим в себе параметры заказа, данные клиента и связи с объектами системы, необходимыми для формирования и отслеживания состояния заказа. Перечисления «СостояниеЗаказа» и «ТипЗаказа» связаны отношением ассоциации. Данный класс реализует свое поведение с помощью четырех методов, изменяющих его состояние: оформить, подтвердить, отменить и выполнить. Каждый из этих методов меняет состояние заказа.

Класс «СотрудникЗаведения» представляет собой пользователя, имеющего доступ к заказам, оформленным на определенное заведение. Он может отменять и подтверждать заказы, что осуществляется с помощью методов класса. Данный класс связан отношением ассоциации кратностью 1: 1 с классом «Заведение» и отношением ассоциации с классом «Заказ» с кратностью 1: М.

Представленные на диаграмме анализа классы главным образом предназначены для хранения информации о тех или иных объектах и являются классами-сущностями. Немногочисленные методы данных классов просты и не требуют дополнительных описаний.

Классом, изменяющимся динамически в данной системе является только класс «Заказ», экземпляр которого изменят свое состояние в процессе своего существования в данной системе.

Динамическое изменения состояния класса описывается с помощью диаграммы конечных автоматов. Данные диаграммы могут применяться как для описания жизненного цикла класса анализа, так и для описания поведения проектного класса. Построение диаграммы конечных автоматов применяется обычно на конечных этапах анализа или проектирования, что позволяет максимально подробно уточнить поведение того или иного класса [uml 2 474-475].

На рис. 4 представлена диаграмма конечных автоматов для класса «Заказ» данной системы.

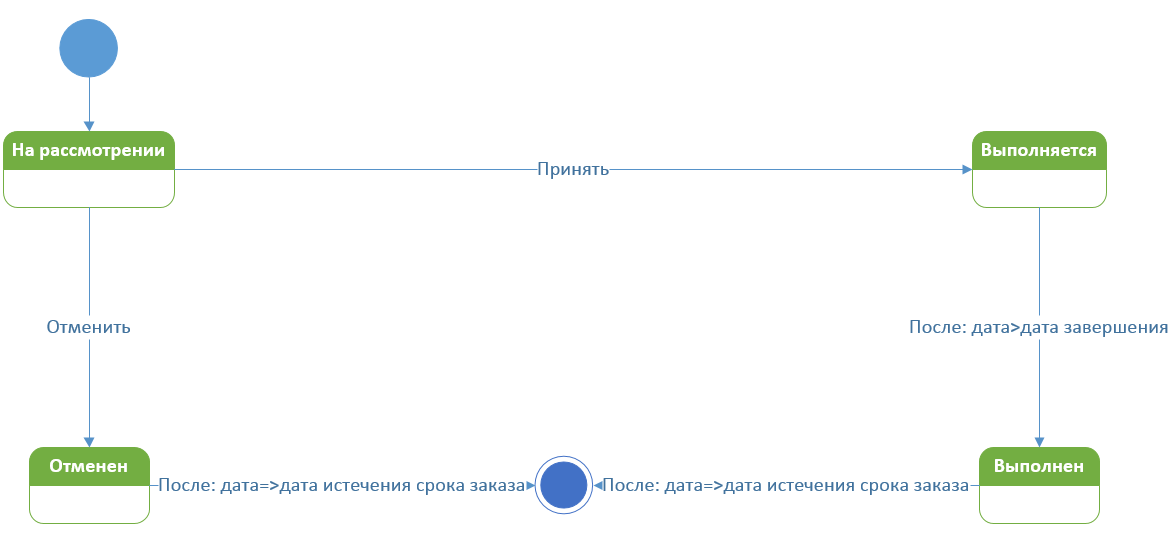


Рис. 4. Диаграмма конечных автоматов класса «Заказ»

Как видно на рис. 4, класс «Заказ» может находиться в 4-х состояниях: на рассмотрении, выполняется, отменен и выполнен.

При оформлении заказа происходит создание экземпляра класса «Заказ», вызывается функция оформления заказа, которая автоматически переводит его в состояние на рассмотрении.

Из состояния заказа «на рассмотрении» перевести в любое другое состояние класс может экземпляр класса «РаботникЗаведения», который переводит его либо в состояние «Выполняется», либо в состояние «Отменен». Это зависит от того, смог ли клиент подтвердить то, что он сделал заказ в данном заведении. Это может быть осуществлено с помощью оставленного клиентом номера мобильного телефона, однако это никак не регламентируется в разрабатываемой системе.

Из состояния «выполняется» заказ может быть переведен в состояние «Выполнен» автоматически, после того, как текущая дата будет больше даты исполнения заказа. При этом, если заказ является заказом столика, то соответствующий столик теряет бронь. После этого его можно будет забронировать снова, оформив заказ бронирования столика.

Из состояний «Отменен» и «Выполнен», в которые каждый из сформированных заказов рано или поздно придет, заказ может быть только удален. Это происходит после того, как текущая дата станет не меньше даты истечения срока заказа, которая определяется автоматически, исходя из даты исполнения заказа, прибавляя 30 дней к значению этой даты.

Проделанные на данном этапе разработки программной системы действия подготовили основу для перехода к проектированию программной системы, ориентируясь на конкретные языки программирования, платформы и средства разработки. Находясь практически в самом начале цикла разработки программной системы, данный этап является одним из самых важных для жестких методологий разработки, потому что детальная проработка данного этапа позволяет снизить повышающуюся на дальнейших этапах жизненного цикла продукта стоимость вносимых изменений в готовую программную систему.