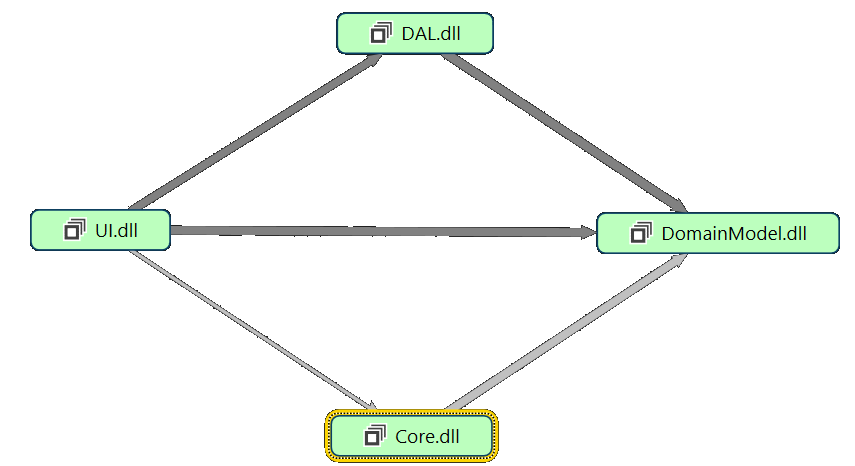
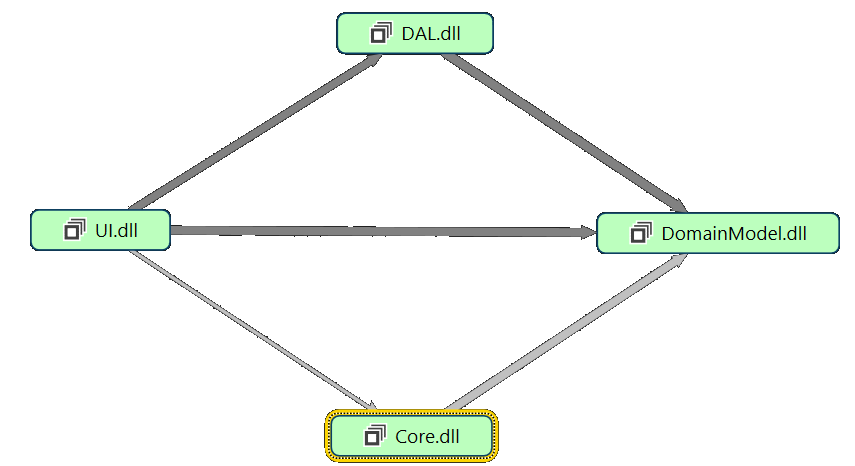
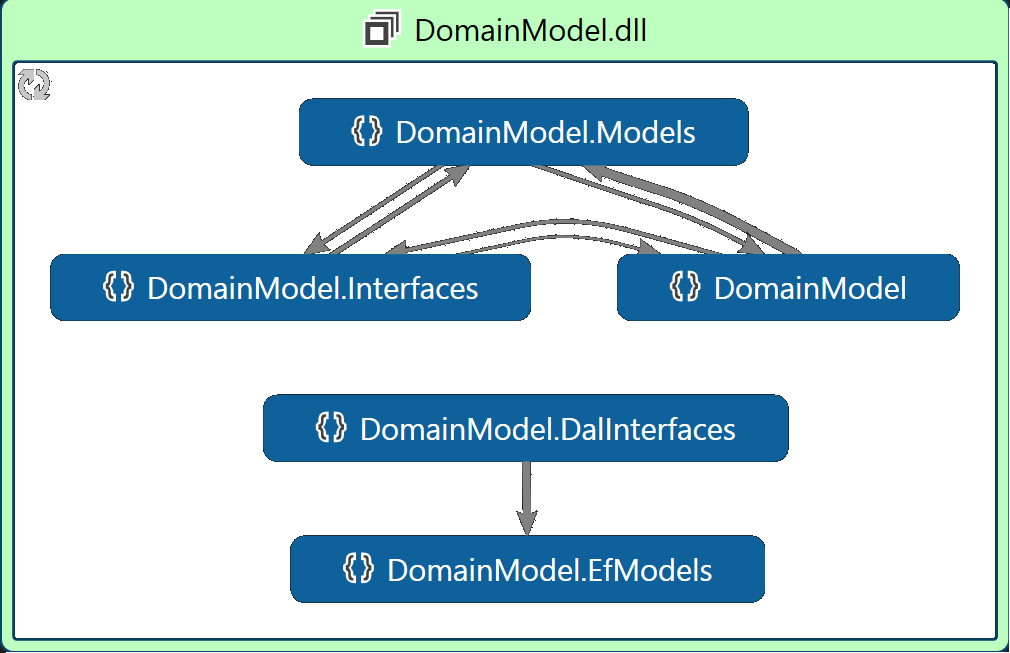
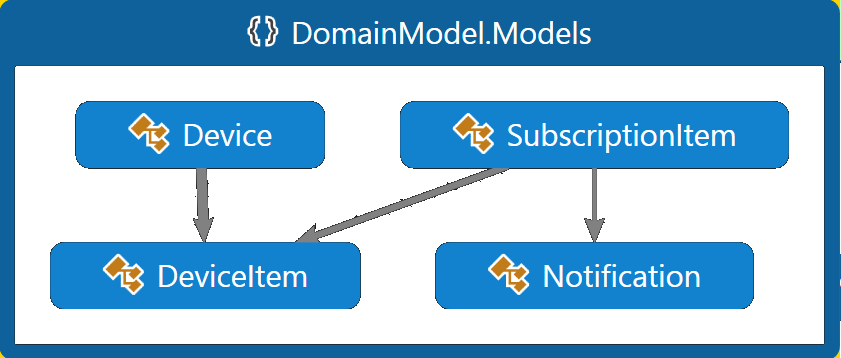
Проект можно разделить на 4 модуля:

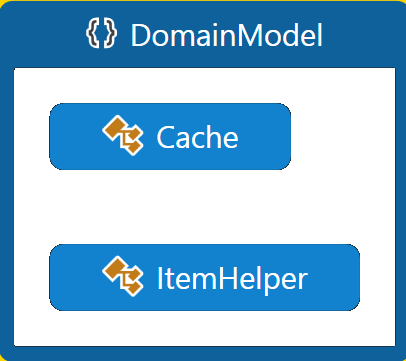
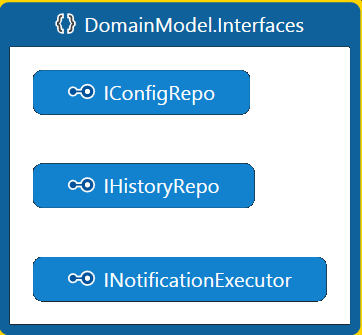
1. DomainModel – модуль содержащий в себе объекты предметной области.
2. DAL – Модуль доступа к БД. Инкапсулирует в себе репозитории для работы с базой данных. Работает с объектами предметной области, описанными в модуле DomainModel
3. Core – ядро системы. Инкапсулирует в себе базовые алгоритмы работы системы
4. UI – модуль отображения. Содержит в себе Web формы, отражающие интерфейс пользователя

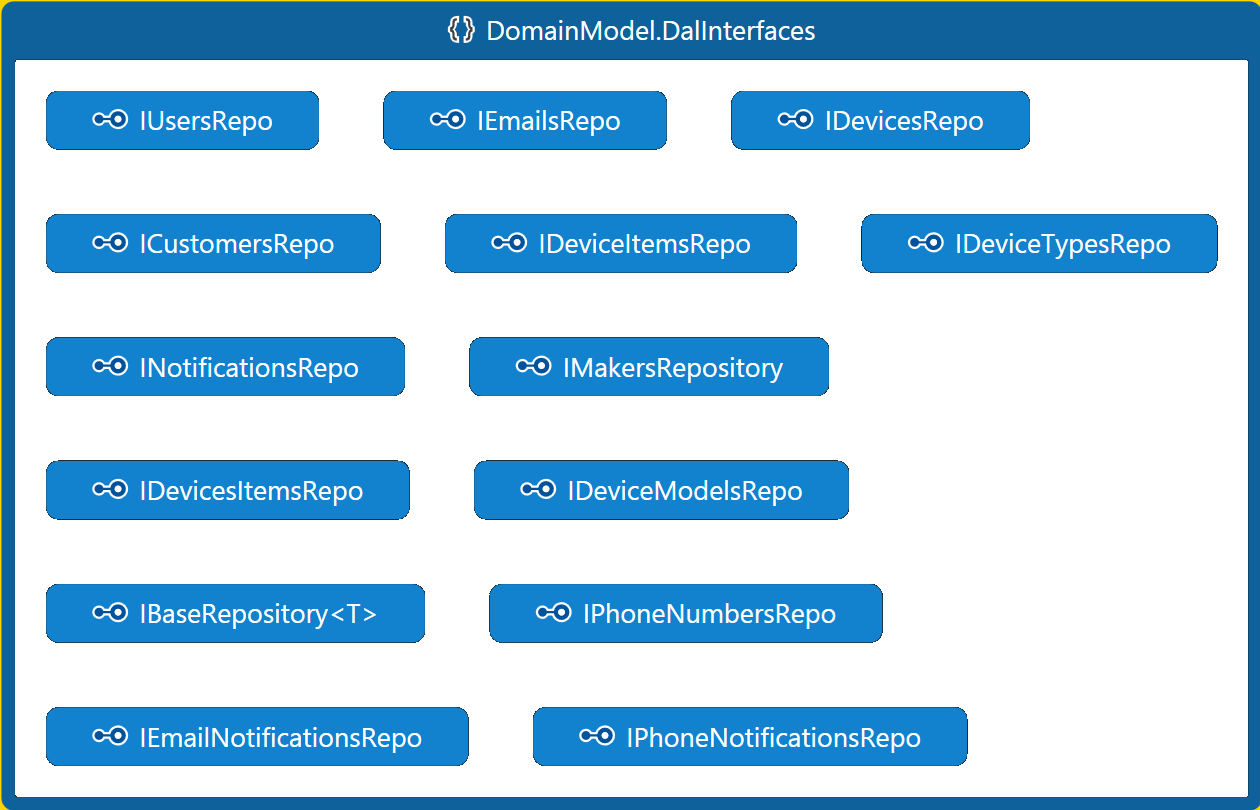
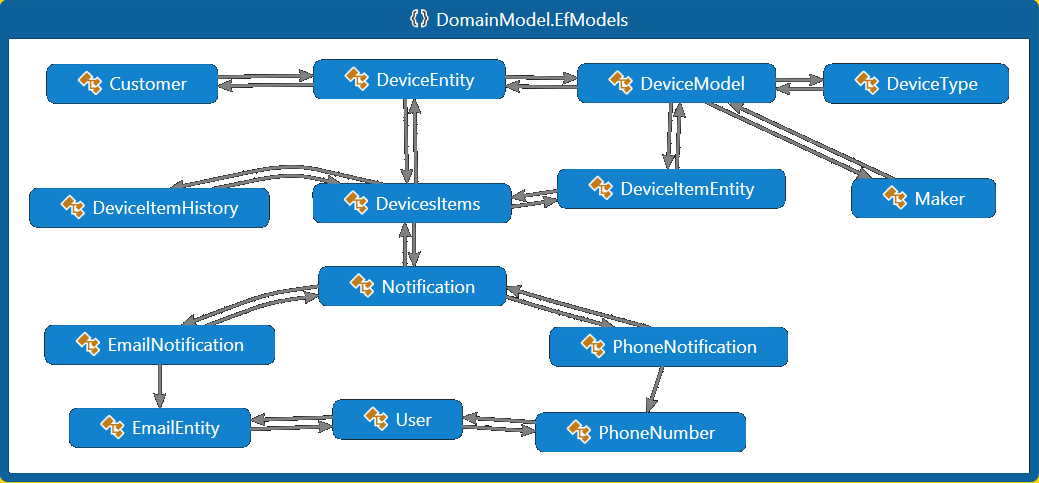


1. DomainModel – данный модуль можно разделить на совокупность следующих элементов:
   1. Cache – базовый класс алгоритма работы системы. Инкапсулирует в себе основные потоки обновления значений, уведомления клиентов о изменении значений
   2. ItemHelper – вспомогательный класс.
   3. Models – содержит в себе модель для работы ядра системы
      1. Device – устройство. Инкапсулирует в себе данные SNPM устройства
      2. DeviceItem – параметр устройства. Инкапсулирует в себе параметр
      3. SubscriptionItem – элемент подписки. Инкапсулирует в себе настройки подписки на уведомление об обновлении параметра устройства
      4. Notification – информация о уведомлении. DTO объект для передачи информации о обновленном параметре
   4. Interfaces – содержит в себе описание интерфейсов по работе с моделью предметной области
      1. IConfigRepo – интерфейс репозитория для чтения начальной конфигурации системы
      2. IHistoryRepo – интерфейс репозитория для записи исторических значений
      3. INotificationExecutor – интерфейс класса для выполнения уведомлений
   5. EfModels – cодержит в себе модель, которая используются в модуле UI для бизнес логики, а так же в модуле DAL для записи в БД.
      1. Maker – производитель устройства
      2. DeviceType – тип устройства
      3. DeviceModel – модель устройства
      4. DeviceItemEntity – параметр модели устройства
      5. Customer – клиент, которому принадлежит конкретное устройство
      6. DeviceEntity – конкретное устройство
      7. DevicesItems – параметры конктретного устройства
      8. DeviceItemHistory – история изменения параметра конкретного устройства
      9. Notification – уведомление пользователя о изменении параметра конкретного устройства
      10. EmailNotification – уведомление электронной почтой
      11. PhoneNotification – уведомление СМС сообщением
      12. EmailEntity – информация о адресе почты
      13. PhoneNumber – информация о номере телефона для смс оповещения
      14. User – пользователь уведомлений системы
   6. DalInterfaces - содержит интерфейсы репозиториев модели. Для каждой модели есть соответствующий интерфейс

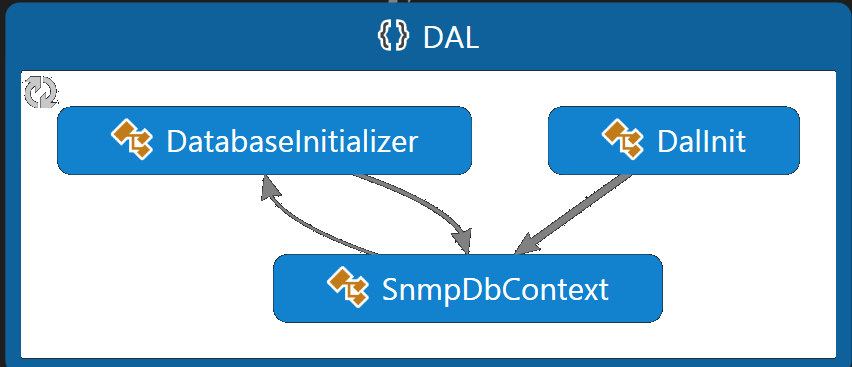
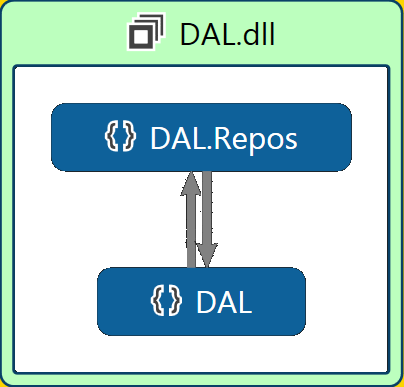


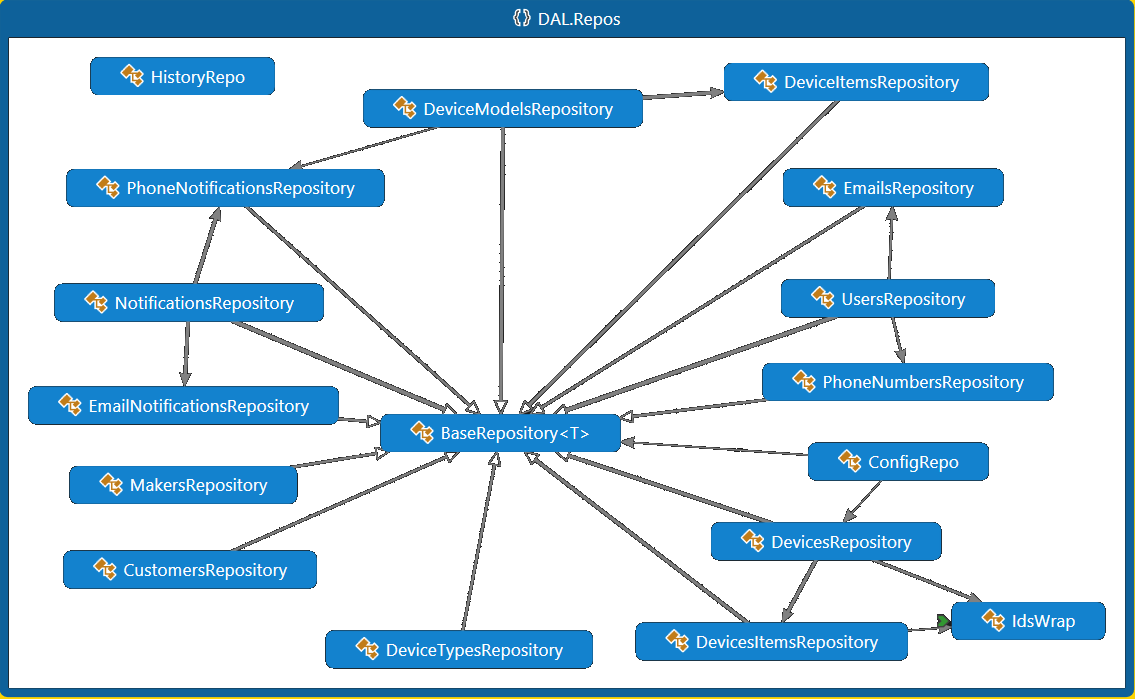




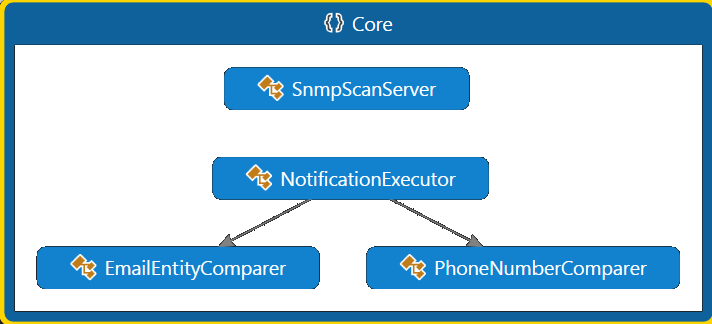


1. DAL – Модуль содержащий в себе реализацию интерфейсов репозиториев описанных в DomainModel.DalInterfaces. Репозиторий – это класс обеспечивающий CRUD операции с таблицей БД. Данный модуль можно разделить на совокупность следующих классов:
   1. SnmpDbContext – EntityFramework (EF) контекс для работой с БД
   2. DatabaseInitializer – класс для создания БД
   3. DalInit – класс для инициализации DAL. Связывает интерфейсы репозиториев с конкретными классами (используется IOC контейнер StructureMap), настраивает маппинг EF сущностей с POCO объектами (используется AutoMapper)
   4. Repos – содержит в себе реализации конкретных репозиториев и вспомогательные классы. Для каждого интерфейса описанного в DomainModel.DalInterfaces есть собственная реализация

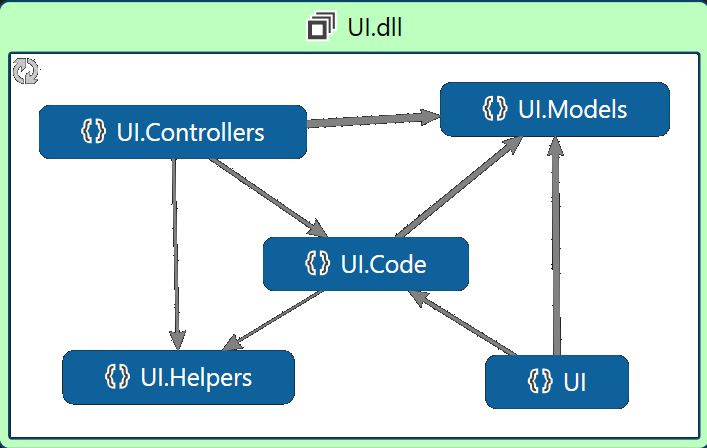


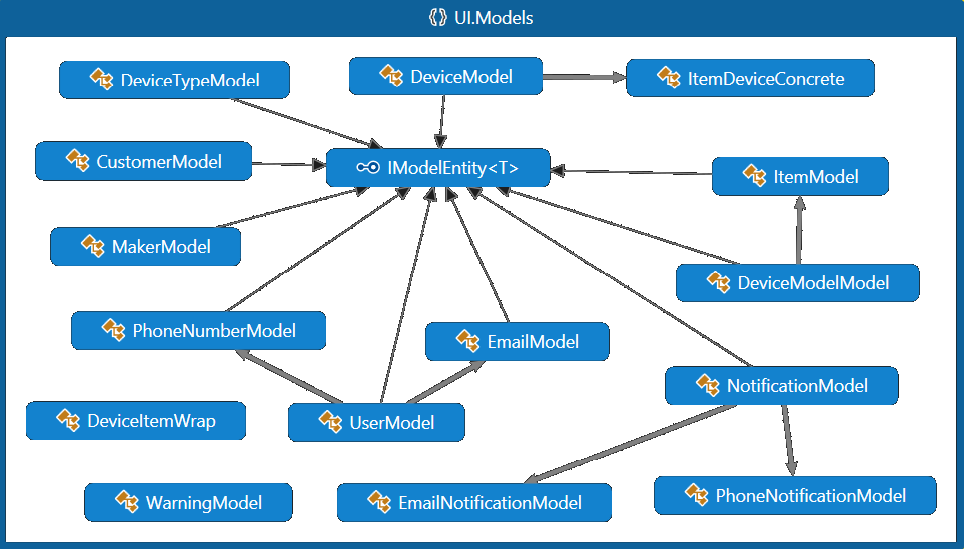


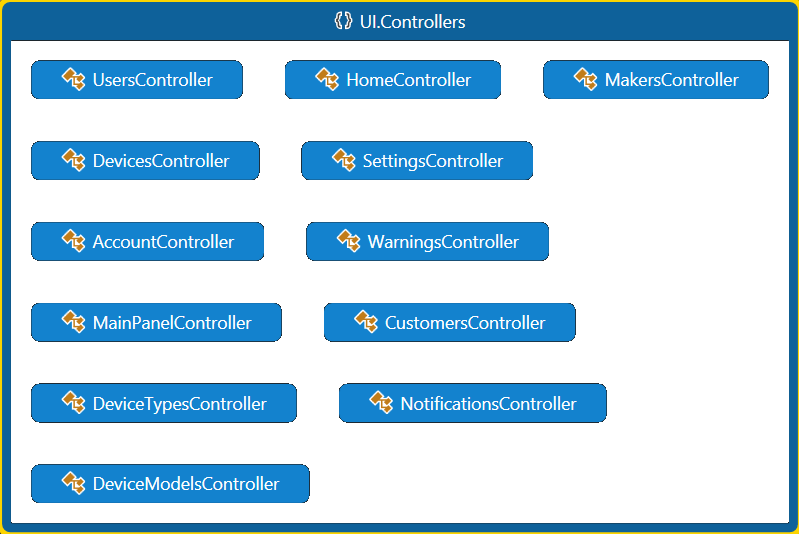
1. Core – ядро системы.
   1. SnmpScanServer – класс инкапсулирующий в себе алгоритм опроса устройств, уведомления пользователя о изменении величины параметров согласно заданной конфигурации
   2. NotificationExecutor – класс инкапсулирующий в себе логику уведомления пользователей
   3. EmailEntityComparer – вспомогательный класс
   4. PhoneNumberComparer – вспомогательный класс

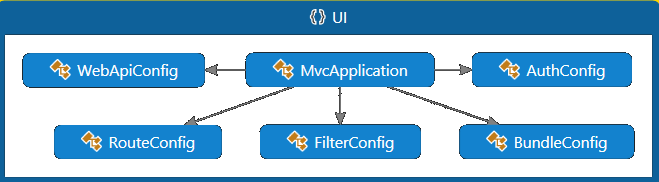


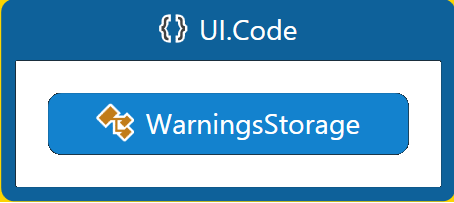
1. UI – модуль содержит в себе Web формы и вспомогательные классы. Используется паттерн MVC
   1. Models – содержит в себе модель для отображения пользователю. Связана с моделью в DomainModel.EfModels по идентификаторам.
   2. Controllers – контроллеры (обработчики) действий пользователя. Для каждого элемента модели есть свой контроллер
   3. UI – классы для конфигурирования приложения
   4. Code – содержит классы инкапсулирующие логику Web форм
   5. Helpers – вспомогательные классы.

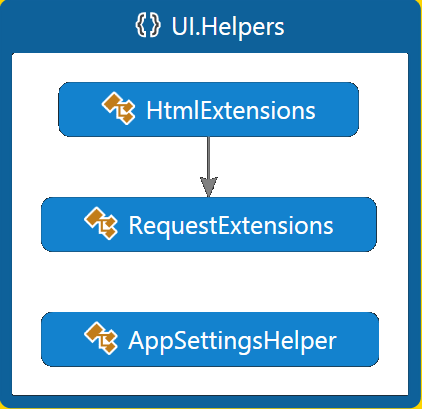












Общий алгоритм работы системы:

При старте система проверяет наличие БД. Если БД нет, то она создается заново и наполняется начальными тестовыми значениями. Далее инициализируются репозитории и маппинг сущностей. После этого создается статический экземпляр класса SnmpScanServer (ядро системы). SnmpScanServer читает конфигурацию системы, подключается к заданным устройствам и начинает в отдельном потоке раз в секунду читать заданные параметры. При этом учитывается сконфигурированные dt и dv каждого параметра. Если прошло времени > dt и за это время значение параметра изменилось больше чем dv, то значение сохраняется в историческую БД. Другой поток параллельно раз в секунду анализирует изменение значений для сконфигурированных подписок. Для каждой подписки учитывается сконфигурированный dt и dv. Если прошло времени > dt и за это время значение параметра изменилось больше чем dv, то происходит уведомление пользователя. С каждой подпиской связаны уведомления по СМС или электронной почте. Если за один проход было обнаружено несколько уведомлений, то они группируются по адресу почты и номеру телефона. Далее происходит групповое уведомление клиентов, т.е. каждый клиент получит СМС или письмо с всеми параметрами за текущий проход.

Интерфейс пользователя позволяет конфигурировать систему в полном объеме. Создавать пользователей, типы и модели устройств, клиентов и их конкретные устройства. Конфигурировать частоту обновления параметров, а также величину изменения значения для записи в историческую БД. Настраивать уведомления пользователей через смс и электронную почту.