

кафедра информационно-измерительных систем (ИС)



дисциплина: «Языки программирования»

Лекция 5: «Концепция и возможности подхода .NET»

1. Среда .NET
 - 1.1. .NET как концепция
 - 1.2. .NET как вычислительная модель
 - 1.3 .NET как технологическая платформа
 - 1.4. .NET как инструментальное средство
2. Основной компонент среды .NET – .NET Framework
3. Среда исполнения общего языка (CLR)

1. Среда .NET

Согласно определению корпорации Microsoft:

Среда .NET – это решение Microsoft для web-служб или следующее поколение программного обеспечения, которое призвано объединить миры информации, устройств и людей унифицированным индивидуализированным способом.

Рассмотрим подробнее среду .NET – с четырех позиций:

- как концепцию,
- как вычислительную модель,
- как технологическую платформу,
- как инструмент проектирования программного кода

1.1. .NET как концепция

Среда .NET **как новая концепция** к проектированию и реализации программного обеспечения решает следующие **4 задачи**:

1. Идеология проектирования и реализации программного обеспечения
2. Модель эффективной поддержки жизненного цикла прикладных систем
3. Унифицированная, интегрированная технологическая платформа
4. Современный, удобный в использовании, безопасный инструментарий для создания, размещения и поддержки программного обеспечения

При презентации корпорацией Microsoft среды .NET было не совсем понятно, что это: новая среда выполнения программного кода, новая операционная система или новый язык программирования...

Среда .NET – это **название концепции** корпорации Microsoft для распределенной среды разработки, созданной на базе новых технологий Microsoft.

Комплекс этих технологий был назван средой **.NET Framework** и предназначен для разработки web-служб.

Однако: среда .NET – это не только создание распределенных систем, основанных на web службах. Назначение ее шире...

1.1. .NET как концепция

Корпорацией-разработчиком Microsoft сформулированы следующие важные *предварительные* аспекты видения *идеологии .NET*:

1. Легкость развертывания приложений в глобальной среде Интернет;
2. Экономичная разработка программного обеспечения;
3. «Бесшовная», гибкая интеграция программных продуктов и аппаратных ресурсов;
4. Предоставление программного обеспечения как сервиса;
5. Новый уровень безопасности и удобства использования.

Все перечисленные аспекты видения .NET удалось реализовать на качественно новом уровне, обеспечив существенное продвижение вперед в направлении гибкости интеграции с программно-аппаратными ресурсами, *безопасности* и удобстве использования кода, а также снижении затрат на производство программного обеспечения.

1.2. .NET как вычислительная модель

Идеология .NET реализовалась в практические вопросы проектирования программного обеспечения по следующим направлениям:

1. **Компонентный подход как развитие объектно-ориентированной модели** (корпорация Microsoft предложила новаторский *компонентно-ориентированный* подход, являющийся развитием *объектно-ориентированного подхода* □ стандартизируется хранение и повторное использование компонент приложения в условиях распределенной сетевой среды вычислений)
2. **Универсальная система типизации: «всякая сущность есть объект»; унификация данных и метаданных** (благодаря разработке усовершенствованной обобщенной системы типизации CTS – Common Type System □ «всякая сущность является объектом *гетерогенной* программной среды»)
3. **Строго иерархическая организация кода, пространств имен и классов** (что позволяет стандартизировать, унифицировать и повторно использовать компоненты программного кода)
4. **Универсальный интерфейс .NET Framework** (обеспечивает интегрированное проектирование и реализацию компонент программных приложений, разработанных согласно *различным* подходам к программированию)
5. **Высокая вариативность экземпляров реализации, в частности, на основе веб-сервисов** (благодаря универсальному интерфейсу стало возможным ускоренное создание приложений для пользователей Интернет)

1.3. .NET как технологическая платформа

.NET как технологическая платформа обеспечивает:

- 1. Многоязыковая поддержка (десятки языков программирования (ЯП))**
(одновременная поддержка проектирования и реализации ПО с использованием *различных* ЯП: от самых ранних (COBOL, FORTRAN) и заканчивая современными (Delphi, C#, VBasic))
- 2. Использование технологии веб-сервисов для обеспечения интероперабельности и масштабируемости в глобальной сетевой среде**
(*масштабируемость* - возможность плавного роста времени ответа программы на запрос с ростом числа одновременно работающих пользователей; в случае web сервисов масштабируемость реализуется посредством распределения вычислительных ресурсов между сервером, где выполняется прикладная программа и, как правило, хранятся данные и компьютером пользователя. *Интероперабельность* – возможность интегрированной обработки данных, поступающих от **разнородных** прикладных программ; в этом случае данные называются гетерогенными.)
- 3. Унификация доступа к библиотекам API-интерфейса независимо от языка и программной модели** (стала возможной благодаря *интероперабельности*)
- 4. Соответствие современным технологическим стандартам** (технология .NET не только всю используется, но и официально признана – стандарт ECMA (European Computer Manufactures Association))

1.4. .NET как инструментальное средство

.NET как инструментальное средство проектирования и реализации программного кода обеспечивает:

- 1. Поддержка многоязыковой среды CLR** (*англ.*, Common Language Runtime) (Эта возможность появилась благодаря универсальному межъязыковому интерфейсу CLI (*англ.*, Common Language Infrastructure), или, который поддерживает разработку программных компонент на различных языках программирования.)
- 2. Возможность создавать компоненты проекта в единой среде на наиболее подходящем языке программирования** (теперь программисты могут разрабатывать/дорабатывать ПО на наиболее подходящем ЯП с учетом: 1) характера задачи: например, символьная обработка прозрачнее и с меньшими трудозатратами реализуема на языке функционального программирования, а формализация структуры предметной области – на объектно-ориентированном языке; 2) опыта работы программистов в команде разработчиков; 3) ЯП, на котором изначально создавалось программное приложение)
- 3. Доступность всех средств .NET для каждого из широкого спектра языков программирования**
- 4. Сервисные возможности для разработчиков, (отладка, анализ кода, ...) одинаковы для всех ЯП** (□ программистам нет необходимости заново постигать особенности среды разработки, если необходимо перейти с одного ЯП на другой ЯП)

1.4. .NET как инструментальное средство

.NET как инструментальное средство проектирования и реализации программного кода обеспечивает (*продолжение*):

- 5.** Возможность облегченной самостоятельной разработки транслятора для любого языка программирования (Microsoft – VB, C#, ... другие – APL, COBOL, Eiffel, Fortran, Haskell, SML, Perl, Python, Scheme, Smalltalk, ...) (т.е. разработка транслятора не вызывает трудностей даже у программистов, не имеющих профессионального опыта в области разработки компиляторов)

1.4. .NET как инструментальное средство

.NET как инструментальное средство проектирования и реализации программного кода обеспечивает (*продолжение*):

- 5.** Возможность облегченной самостоятельной разработки транслятора для любого языка программирования (Microsoft – VB, C#, ... другие – APL, COBOL, Eiffel, Fortran, Haskell, SML, Perl, Python, Scheme, Smalltalk, ...) (т.е. разработка транслятора не вызывает трудностей даже у программистов, не имеющих профессионального опыта в области разработки компиляторов)

2. Основной компонент среды .NET – .NET Framework

Платформа .NET Framework – это технический элемент, позволяющий разработчикам создавать *программные приложения (коммерческие!)*

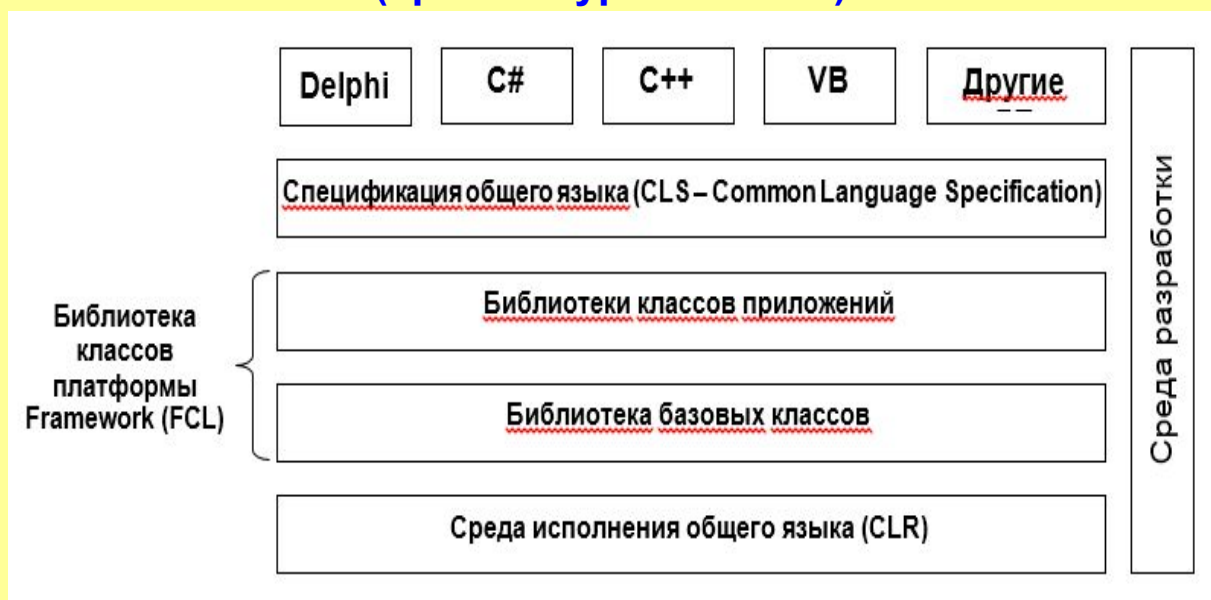
Разработка и выполнение приложений .NET состоит из следующих этапов:

1. Создание программы на любом подходящем языке .NET
2. Компиляция кода в модуль на промежуточном языке (IL)
3. Объединение управляемых модулей (manage module) и создание сборки (assembly)
4. Развертывание или установка сборки на предназначенную платформу
5. Обращение к среде CLR, которая загружает, компилирует, выполняет созданный код и управляет им

Чтобы проделать все эти этапы, необходимы следующие компоненты (см. [архитектурную схему](#)):

2. Основной компонент среды .NET – .NET Framework

Платформа .NET Framework (архитектурная схема):



1. Языки программирования (отображены в верхнем ряду)
2. Спецификация общего языка CLS (*англ.*, Common Language Specification) – набор правил (обеспечивающий взаимную интеграцию между разными языками) для средств разработки на языках .NET, которым удовлетворяет объектно-ориентированная библиотека классов .NET Framework – FCL (*англ.*, Framework Class Library). FCL подобна библиотеке VCL Delphi, библиотеке MFC Visual C++ и др.
3. Среда исполнения общего языка CLR (*англ.*, Common Language)
4. Среда разработки, например, **Delphi for .NET** или Visual Studio .NET

3. Среда исполнения общего языка (CLR)

Среда исполнения общего языка (CLR) - ключевой компонент платформы .NET Framework, обеспечивающая выполнение приложений .NET, а также предоставляет службы, необходимые этим приложениям во время выполнения.

При обращении к ней приложения .NET среда CLR сама выполняет все необходимые операции:

- **КОМПИЛЯЦИЯ** (в ходе компиляции программа на .NET-совместимом языке программирования) трансформируется в соответствии с заранее заданной обобщенной спецификацией языка **Common Type System (CTS)**. Система типов CTS; полностью описывает все типы данных, поддерживаемые средой выполнения, определяет их взаимосвязи и хранит их отображения в систему типов .NET.
- **распределение памяти;**
- **типами данных;**
- **межъязыковым взаимодействием;**
- **управление кодом.**

Среда CLR работает с промежуточным представлением исходного кода, написанного на любом совместимом с платформой .NET языке, в частности: Delphi или C#. Подобный код на *промежуточном языке* MSIL (Microsoft Intermediate Language) или просто *промежуточном языке* IL (Intermediate Language), содержится в файле называемом **управляемым модулем**.

3. Среда исполнения общего языка (CLR)

Управляемые модули – создаются компиляторами языка .NET.

Управляемый модуль состоит из **четырёх частей**:

1. **Заголовок PE** (Portable Executable – *англ.*, переносимый, исполняемый) – стандартный заголовок PE файла Windows.
2. **Заголовок CLR** – содержит информацию, специфическую для среды CLR и предназначенную для использования в ней.
3. **Метаданные** – таблицы, описывающие типы данных, членов модуля и ссылок на другие модули.
4. **Управляемый код** – код, создаваемый компиляторами ЯП, совместимых с платформой .NET, и находится в формате общего языка; этот язык называется также промежуточным языком Microsoft – MSIL.

Управляемые модули не могут быть запущены автономно, хотя и являются переносимыми исполняемыми файлами – сначала их следует включить в **сборку**, а затем передать на выполнение среде CLR.

Сборки – это пакет (развернутый блок) программных приложений .NET и наборов функций, предназначенных для совместного использования другими приложениями .NET.