**TEAM ASSIGNMENT 3**

**Huy Son.**

**Version 0.1**

1. **Purpose of document.**
2. **Part 1. Introduction.**
   1. **Introduction of .NET framework.**

The **.NET Framework** (pronounced *dot net*) is a [software framework](http://en.wikipedia.org/wiki/Software_framework) developed by [Microsoft](http://en.wikipedia.org/wiki/Microsoft) that runs primarily on [Microsoft Windows](http://en.wikipedia.org/wiki/Microsoft_Windows).

*Software framework is a reusable set of libraries or classes for a software system (or subsystem).*

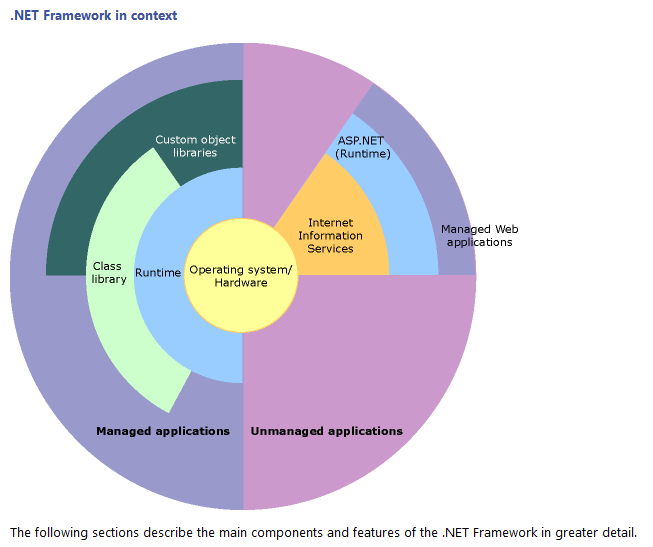
It includes a large [library](http://en.wikipedia.org/wiki/Base_Class_Library) and provides language interoperability (each language can use code written in other languages) across several [programming languages](http://en.wikipedia.org/wiki/Programming_language).

Programs written for the .NET Framework execute in a [software](http://en.wikipedia.org/wiki/Software) environment (as contrasted to [hardware](http://en.wikipedia.org/wiki/Hardware) environment), known as the [Common Language Runtime](http://en.wikipedia.org/wiki/Common_Language_Runtime) (CLR), an [application virtual machine](http://en.wikipedia.org/wiki/Process_virtual_machine) that provides important services such as security, [memory management](http://en.wikipedia.org/wiki/Memory_management), and [exception handling](http://en.wikipedia.org/wiki/Exception_handling). The class library and the CLR together constitute the .NET Framework.

The .NET Framework's [Base Class Library](http://en.wikipedia.org/wiki/Base_Class_Library) provides [user interface](http://en.wikipedia.org/wiki/User_interface), [data access](http://en.wikipedia.org/wiki/Data_access), [database connectivity](http://en.wikipedia.org/wiki/Database_connection), [cryptography](http://en.wikipedia.org/wiki/Cryptography), [web application](http://en.wikipedia.org/wiki/Web_application) development, numeric [algorithms](http://en.wikipedia.org/wiki/Algorithm), and [network communications](http://en.wikipedia.org/wiki/Computer_networking). Programmers produce software by combining their own [source code](http://en.wikipedia.org/wiki/Source_code) with the .NET Framework and other libraries. The .NET Framework is intended to be used by most new applications created for the Windows platform. Microsoft also produces a popular [integrated development environment](http://en.wikipedia.org/wiki/Integrated_development_environment) largely for .NET software called [**Visual Studio**](http://en.wikipedia.org/wiki/Microsoft_Visual_Studio).

As you would expect from an object-oriented class library, the .NET Framework types enable you to accomplish a range of common programming tasks, including tasks such as string management, data collection, database connectivity, and file access. In addition to these common tasks, the class library includes types that support a variety of specialized development scenarios. For example, you can use the .NET Framework to develop the following types of applications and services:

Console applications, Windows GUI application, Windows Presentation Foundation (WPF) applications, ASP.NET applications, web services, Windows service and Workflow-enabled applications using Windows Workflow Foundation (WF).



.NET framework support two type of user.

**For developer.**

The .NET Framework provides the following services for application developers:

**Memory management**. In many programming languages, programmers are responsible for allocating and releasing memory and for handling object lifetimes. In .NET Framework applications, the CLR provides these services on behalf of the application.

**A common type system.** In traditional programming languages, basic types are defined by the compiler, which complicates cross-language interoperability. In the .NET Framework, basic types are defined by the .NET Framework type system and are common to all languages that target the .NET Framework.

**An extensive class library.** Instead of having to write vast amounts of code to handle common low-level programming operations, programmers can use a readily accessible library of types and their members from the .NET Framework Class Library.

**Development frameworks and technologies.** The .NET Framework includes libraries for specific areas of application development, such as ASP.NET for web applications, ADO.NET for data access, and Windows Communication Foundation for service-oriented applications.

**Language interoperability**. Language compilers that target the .NET Framework emit an intermediate code named Common Intermediate Language (CIL), which, in turn, is compiled at run time by the common language runtime. With this feature, routines written in one language are accessible to other languages, and programmers can focus on creating applications in their preferred language or languages.

**Version compatibility.** With rare exceptions, applications that are developed by using a particular version of the .NET Framework can run without modification on a later version.

**Side-by-side execution.** The .NET Framework helps resolve version conflicts by allowing multiple versions of the common language runtime to exist on the same computer. This means that multiple versions of applications can also coexist, and that an application can run on the version of the .NET Framework with which it was built.

**Multi-targeting**. By targeting the .NET Framework Portable Class Library, developers can create assemblies that work on multiple .NET Framework platforms, such as the .NET Framework, Silverlight, Windows Phone 7, or Xbox 360.

You can choose any programming language that supports the .NET Framework to create your application. Because the .NET Framework provides language independence and interoperability, you can interact with other .NET Framework applications and components regardless of the language with which they were developed.

**And for the others user.**

History.

Microsoft started the development on the .NET Framework in the late 1990s originally under the name of Next Generation Windows Services (NGWS). By late 2000 the first beta versions of .NET 1.0 were released.

Version 3.0 of the .NET Framework is included with [Windows Server 2008](http://en.wikipedia.org/wiki/Windows_Server_2008) and [Windows Vista](http://en.wikipedia.org/wiki/Windows_Vista). Version 3.5 is included with [Windows 7](http://en.wikipedia.org/wiki/Windows_7), and can also be installed on [Windows XP](http://en.wikipedia.org/wiki/Windows_XP) and the [Windows Server 2003](http://en.wikipedia.org/wiki/Windows_Server_2003) family of operating systems. On 12 April 2010, .NET Framework 4 was released alongside [Visual Studio 2010](http://en.wikipedia.org/wiki/Visual_Studio_2010).

The .NET Framework family also includes two versions for [mobile](http://en.wikipedia.org/wiki/Mobile_computing) or [embedded](http://en.wikipedia.org/wiki/Embedded_system) device use. A reduced version of the framework, the [.NET Compact Framework](http://en.wikipedia.org/wiki/.NET_Compact_Framework), is available on [Windows CE](http://en.wikipedia.org/wiki/Microsoft_Windows_CE) platforms, including [Windows Mobile](http://en.wikipedia.org/wiki/Windows_Mobile) devices such as [smart phones](http://en.wikipedia.org/wiki/Smartphones). Additionally, the [.NET Micro Framework](http://en.wikipedia.org/wiki/.NET_Micro_Framework) is targeted at severely resource-constrained devices.

* 1. **Structure of .NET framework.**

the next section will introduce .NET framework architecture.

### First, Common Language Infrastructure (CLI).

### CLI is an open [specification](http://en.wikipedia.org/wiki/Specification) developed by [Microsoft](http://en.wikipedia.org/wiki/Microsoft) and standardized by ISO and ECMA that describes the executable code and runtime environment that form the core of the Microsoft [.NET Framework](http://en.wikipedia.org/wiki/.NET_Framework) and the [free and open source](http://en.wikipedia.org/wiki/Free_and_open_source_software) implementations [Mono](http://en.wikipedia.org/wiki/Mono_(software)) and [Portable.NET](http://en.wikipedia.org/wiki/Portable.NET). The specification defines an environment that allows multiple high-level languages to be used on different computer platforms without being rewritten for specific architectures.

### 

### *Above picture present utility of CLI.*

### The purpose of the Common Language Infrastructure (CL) is to provide a language-neutral platform for application development and execution, including functions for [Exception handling](http://en.wikipedia.org/wiki/Exception_handling), [Garbage Collection](http://en.wikipedia.org/wiki/Garbage_collection_(computer_science)), security, and interoperability. By implementing the core aspects of the .NET Framework within the scope of the CL, this functionality will not be tied to a single language but will be available across the many languages supported by the framework. Microsoft's implementation of the CLI is called the [Common Language Runtime](http://en.wikipedia.org/wiki/Common_Language_Runtime), or CL.

***.NET assembly****is an****assembly****is a*[*compiled*](http://en.wikipedia.org/wiki/Compiler)*code library used for deployment, versioning, and security. There are two types: process assemblies (*[*EXE*](http://en.wikipedia.org/wiki/EXE)*) and library assemblies (*[*DLL*](http://en.wikipedia.org/wiki/Dynamic-link_library)*). A process assembly represents a process that will use*[*classes*](http://en.wikipedia.org/wiki/Class_(computer_science))*defined in library assemblies. .NET assemblies contain code in*[*CIL*](http://en.wikipedia.org/wiki/Common_Intermediate_Language)*, which is usually generated from a*[*CLI language*](http://en.wikipedia.org/wiki/List_of_CLI_languages)*, and then compiled into*[*machine language*](http://en.wikipedia.org/wiki/Machine_language)*at*[*run time*](http://en.wikipedia.org/wiki/Run_time_(program_lifecycle_phase))*by the*[*CLR*](http://en.wikipedia.org/wiki/Common_Language_Runtime)[*just-in-time compiler*](http://en.wikipedia.org/wiki/Just-in-time_compiler)*.*

### Memory management. The .NET Framework CL frees the developer from the burden of managing memory (allocating and freeing up when done); it handles memory management itself by detecting when memory can be safely freed. Memory is allocated to instantiations of .NET types (objects) from the managed heap, a pool of memory managed by the CL. As long as there exists a reference to an object, which might be either a direct reference to an object or via a [graph](http://en.wikipedia.org/wiki/Graph_(data_structure)) of objects, the object is considered to be in use. When there is no reference to an object, and it cannot be reached or used, it becomes garbage, eligible for collection. NET Framework includes a [garbage collector](http://en.wikipedia.org/wiki/Garbage_collection_(computer_science)) which runs periodically, on a separate [thread](http://en.wikipedia.org/wiki/Thread_(computing)) from the application's thread, that enumerates all the unusable objects and reclaims the memory allocated to them.

### The .NET [Garbage Collector](http://en.wikipedia.org/wiki/Garbage_collection_(computer_science)) (GC) is a non-deterministic, compacting, [mark-and-sweep](http://en.wikipedia.org/wiki/Garbage_collection_(computer_science)#Copying_vs._mark-and-sweep_vs._mark-and-don.27t-sweep) garbage collector. The GC runs only when a certain amount of memory has been used or there is enough pressure for memory on the system. Since it is not guaranteed when the conditions to reclaim memory are reached, the GC runs are non-deterministic.

### The GC used by .NET Framework is actually [*generational*](http://en.wikipedia.org/wiki/Garbage_collection_(computer_science)#Generational_GC_.28ephemeral_GC.29). Objects are assigned a *generation*; newly created objects belong to *Generation 0*. The objects that survive a garbage collection are tagged as *Generation 1*, and the Generation 1 objects that survive another collection are *Generation 2* objects. The .NET Framework uses up to Generation 2 objects. Higher generation objects are garbage collected less frequently than lower generation objects. This helps increase the efficiency of garbage collection, as older objects tend to have a larger lifetime than newer objects. Thus, by removing older (and thus more likely to survive a collection) objects from the scope of a collection run, fewer objects need to be checked and compacted.

* 1. **Comparison between .NET framework architect and Java architect**.

*Java EE is Oracle's enterprise java computing platform. The platform provides an API and runtime environment for developing and running enterprise software, including network and web services, and other large-scale, multi-tiered, scalable, reliable, and secure network applications*.

*.NET is Microsoft's platform for XML Web services. XML Web services allow applications to communicate and share data over the Internet, regardless of operating system or programming language*

|  |  |  |
| --- | --- | --- |
| **Component** | The Common Language Runtime (CLR), an application virtual machine that provides important services such as security, memory management, and exception handling. The class library and the CLR together constitute the .NET Framework.  Base Class Library provides user interface, data access, database connectivity, cryptography, web application development, numeric algorithms, and network communications. Programmers produce software by combining their own source code with the .NET Framework and other libraries. | A J2EE component is a self-contained functional software unit that is assembled into a J2EE application with its related classes and files and communicates with other components. The J2EE specification defines the following J2EE components:   * Application clients and applets are client components. Java Servlet and Java Server Pages technology components are web components. * Enterprise JavaBeans components (enterprise beans) are business components. * Resource adapter components provided by * EIS and tool vendors |

|  |  |  |
| --- | --- | --- |
| **Architecture** | Designs support multiple difference programming language. Currently, 30 languages support the .net framework. Runs primarily on Microsoft Windows.  The .NET CLR contains an Intermediate Language (IL) engine. Code and objects written in a language can be compiled into the IL runtime, once an IL compiler is developed for the language  CRL implement algorithm continuous distribution  Compiling works faster | Though other language’s code can be converted run under JVM they don’t acquire true. Cross language capability. Run in multiple platforms.  Compiling source code into the Java "byte code.", as long as platforms have a Java Virtual Machine to execute byte code.  JVM implement algorithm non -continuous distribution |

1. **Part 2. Description of standard.**
2. **Part 3. Analysis of Properties, Tradeoffs, and Applicability**