
Introduction à .NET

Clémentine Nebut & Abdelhak-Djamel Seriai
Université de Montpellier

Plan du cours

- Pourquoi .NET ?
- .NET et les autres solutions disponibles
- .NET : Présentation
- Les trois (ou quatre) C de .NET
- Le Common Language Runtime (CLR)
- Les bibliothèques de base
- Les outils

Hétérogénéité et Interopérabilité

- L'objectif d'interopérabilité
 - Assurer l'interopérabilité des composants d'un système malgré :
 - L'Hétérogénéité des CPUs
 - L'Hétérogénéité des systèmes d'exploitation
 - **L'Hétérogénéité des langages**
 - L'Hétérogénéité des formats de données
 - Etc.
 - Hétérogénéité des langages de programmation
 - Exemples :
 - Les langages n'ont pas la même façon de considérer les types de données
 - Les langages de programmation ne supportent pas les mêmes mécanismes, liés souvent à leurs paradigmes d'appartenance

Hétérogénéité des langages de programmation

- Les langages de programmation n'ont pas la même façon de considérer les types de données
 - Les types de données sont aussi important que les données elles-mêmes : Les types de données sont nécessaires pour interpréter les données
 - Exemple : 0000 0000 0000 0000 0000 0000 0100 1101 est considéré comme un 77, 'M', 1.079E -403 suivant le type associé : entier, caractère ou réel.
 - Les sources du problème :
 - Présence/Absence de certains types
 - Incompatibilité des types : pas la même sémantique

Hétérogénéité des langages de programmation

- Les langages de programmation n'ont pas la même façon de considérer les types de données
- Les langages de programmation ne supportent pas les mêmes mécanismes, liés souvent à leurs paradigmes d'appartenance
 - Exemples :
 - Support à l'héritage multiple ou pas
 - Support de la notion de type signé ou pas
 - Support de la surcharge d'opérateurs ou pas

Plan du cours

- Pourquoi .NET ?
- .Net et les autres solutions disponibles
- .NET : Présentation
- Les trois (ou quatre) C de .NET
- Le Common Language Runtime (CLR)
- Les bibliothèques de base
- Les outils

La solution Java, JVM

- Consiste à utiliser un seul langage et une seule plateforme (Java, JVM)
- La plateforme Java est portée sur différents systèmes d'exploitation offrant ainsi la possibilité de : « écrire une fois, exécuter n'importe où » ou en anglais « Write once, run anywhere »
- Problème : il faut écrire tout en Java !

La solution COM

- Définit un standard binaire pour que les langages qui adhèrent à ce standard utilisent un ensemble de types de données primitifs qui constitue un **dénominateur commun de types**
- Avec COM, les applications sont interopérables même si elles sont codées dans différents langages
- Problème : seuls les langages Windows adhèrent à ce standard + complexité et fragilité de ce standard

La solution .NET

".NET is the Microsoft Web services strategy to connect information, people, systems, and devices through software."

<http://www.microsoft.com/net/basics.msp>

"The .NET Framework is a development and execution environment that allows different programming languages & libraries to work together seamlessly to create Windows-based applications that are easier to build, manage, deploy, and integrate with other networked systems."

<http://www.msdn.microsoft.com/netframework/gettingstarted/default.aspx>

Plan du cours

- Pourquoi .NET ?
- .Net et les autres solutions disponibles
- .NET : Présentation
- Les trois (ou quatre) C de .NET
- Le Common Language Runtime (CLR)
- Les bibliothèques de base
- Les outils

La solution .NET

- Une plateforme multi-langages avec un langage dédié : C#
- Conçu pour faciliter le développement d'applications n-tiers
- Un ensemble de classes de bibliothèque
- Utilisation/création facile d'objets distribués et de services web, P2P (3.5)
- Accès facile aux données (ADO.NET)
- Présentation par pages web dynamiques (ASP.NET)
- La réponse de Microsoft à J2EE ...

Les langages .NET

- Le langage "dédié" : C#
 - proche de java
- Les classiques
 - VB.NET, C++, J#, Eiffel#
- Mais aussi :
 - ADA, Scheme, Python, Fortran, ...
- En tout une vingtaine de langages supportés

J# n'est plus supporté ...

Plan du cours

- Pourquoi .NET ?
- .Net et les autres solutions disponibles
- .NET : Présentation
- Les trois (ou quatre) C de .NET
- Le Common Language Runtime (CLR)
- Les bibliothèques de base
- Les outils

Les trois C de .Net : CTS, CLS et CLR

- CTS : Common Type System
 - Un système de types communs
- CLS : Common Langage Specification
 - La spécification d'un langage commun
- CLR : Common Langage Runtime
 - Moteur d'exécution du langage commun

CTS : La solution pour l'interopérabilité des types

- Est un socle commun de types standards avec la possibilité de définition de types spécifiques par l'extension de ces types
- Tout langage qui cible la plateforme .Net doit réaliser une correspondance (Mapping) entre ses propres types et ceux du CTS
- Tous les langages .Net partagent le même système de type et ils sont ainsi interopérables
 - Par exemple, il est possible de dériver une classe écrite en VB .Net à partir d'une classe écrites en C#
- Les éditeurs de langages de programmation et d'outils .Net ont besoin de connaître le CTS et pas l'utilisateur de .Net

CLS : La solution pour l'interopérabilité des mécanismes supportés par les langages

- Contient la spécification des langages .net
 - Spécifie l'ensemble de caractéristiques minimales qu'un langage doit supporter pour qu'il puisse être intégré à la plateforme .Net et les caractéristiques complètes pour son interopérabilité avec les autres langages .Net
 - Contient entre autres :
 - Le système de types
 - Un ensemble de règles
 - règle 7 : "The underlying type of an enum shall be a built-in CLS integer type."
 - règle 25 : "The accessibility of a property and of its accessors shall be identical."
- Les langages basés sur CLS constituent un sous ensemble de ceux basés sur CTS
- Les éditeurs de langages et d'outils .Net ont besoin de connaître le CLS et pas l'utilisateur de .Net

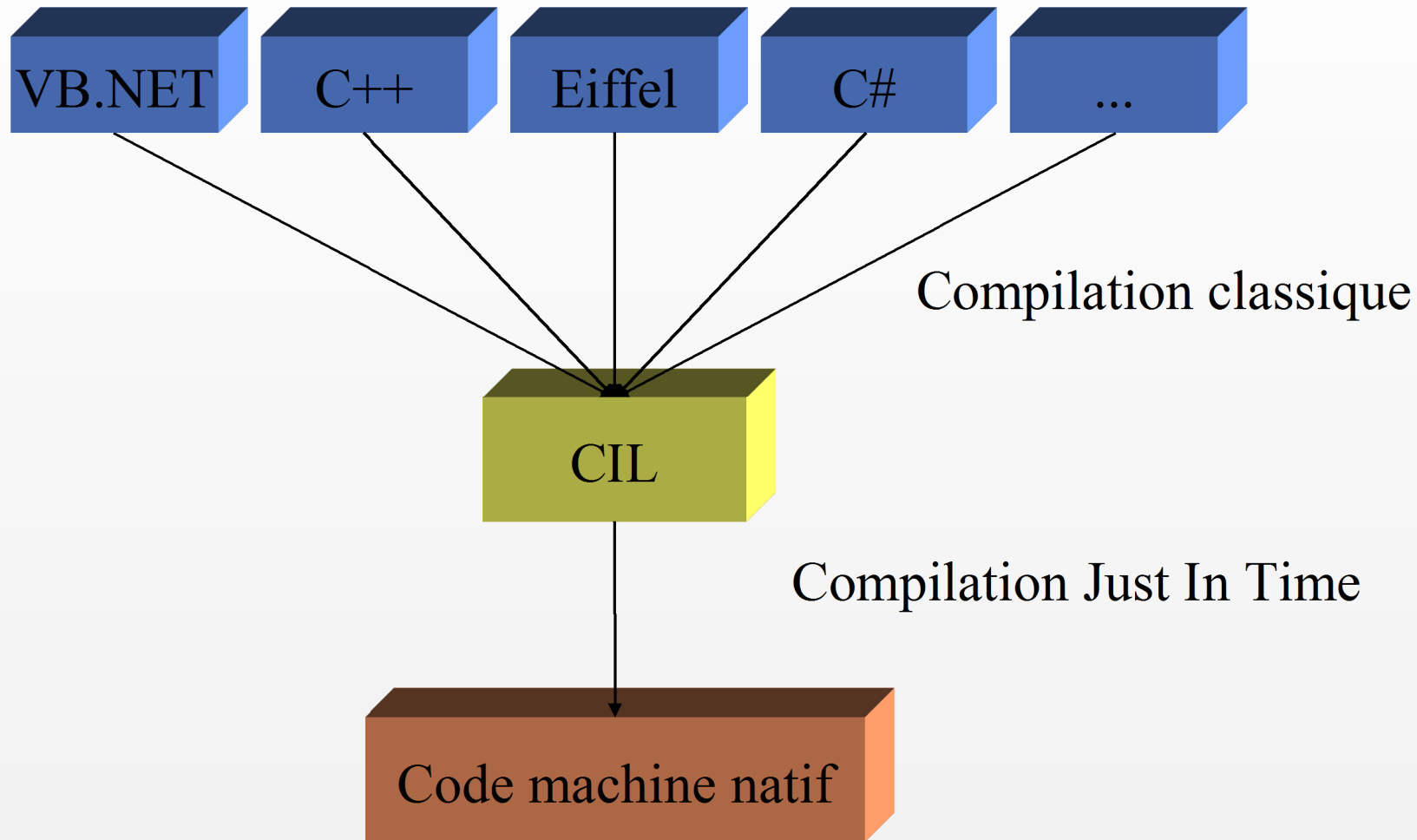
CLR : La solution pour l'exécution des langages interopérables

- Constitue un moteur d'exécution des applications .Net.
 - Responsable de charger et exécuter de manière sûr le code des applications,
 - Fournit un « ramasse miettes »
 - Vérifie la sécurité de ces applications
 - Fournit une bibliothèque de classes
 - E/S, Thread, dialogue box, etc.
 - Etc.

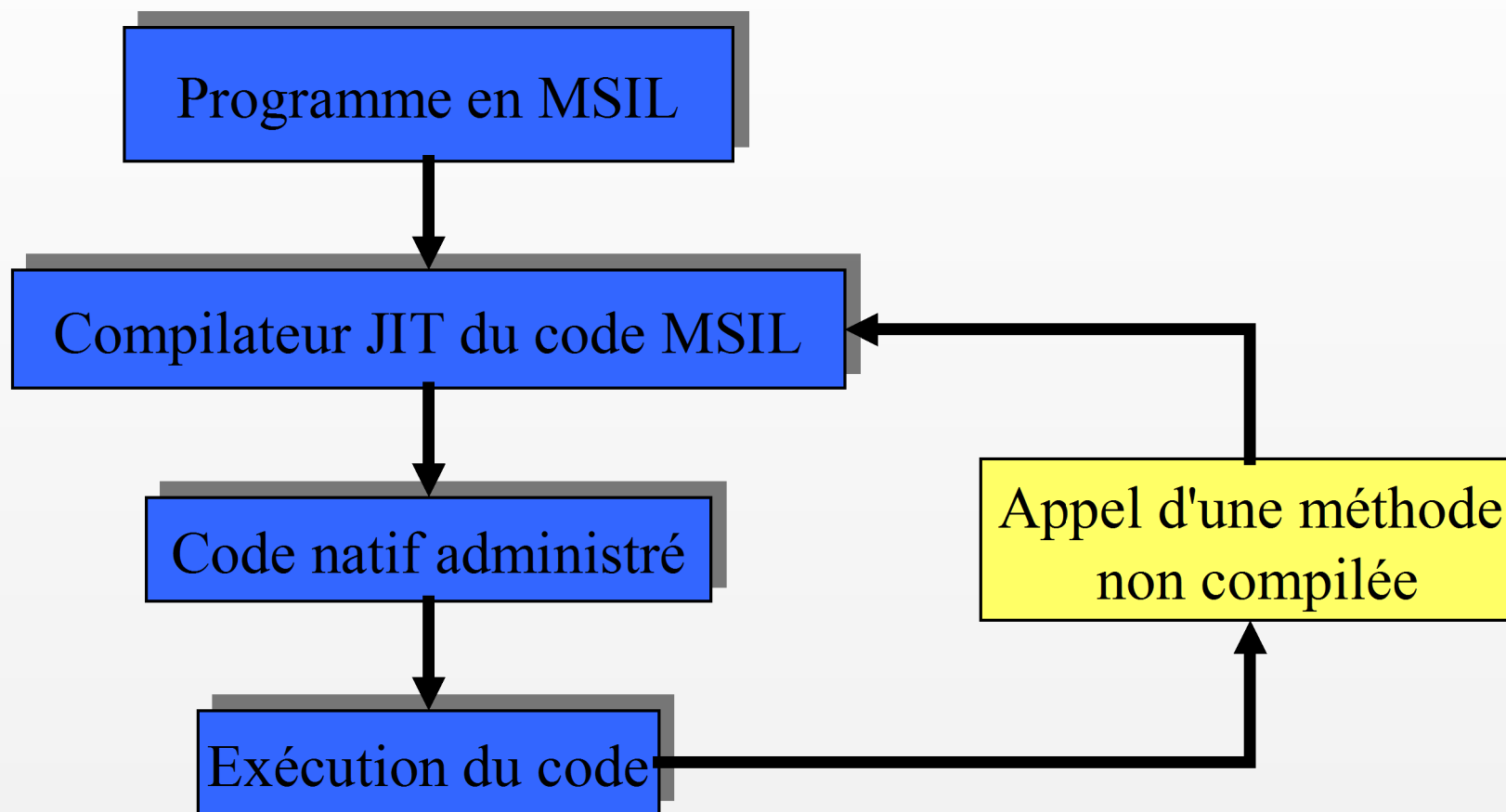
CIL : Le quatrième C (en plus des trois premiers !)

- CIL : Common Intermediate Language
 - Tous les compilateurs des langages .Net génèrent un code indépendant par rapport à la plate-forme « plateforme-neutral code » : le CIL ou IL (plus simplement)
 - Un langage .NET
 - est compilé vers du CIL
 - est conforme à la CLS
 - MSIL = CIL
 - = Microsoft Intermediate Language
 - = Common Intermediate Language
 - ≈ Byte code Java
 - interprété par un moteur d'exécution (machine virtuelle)

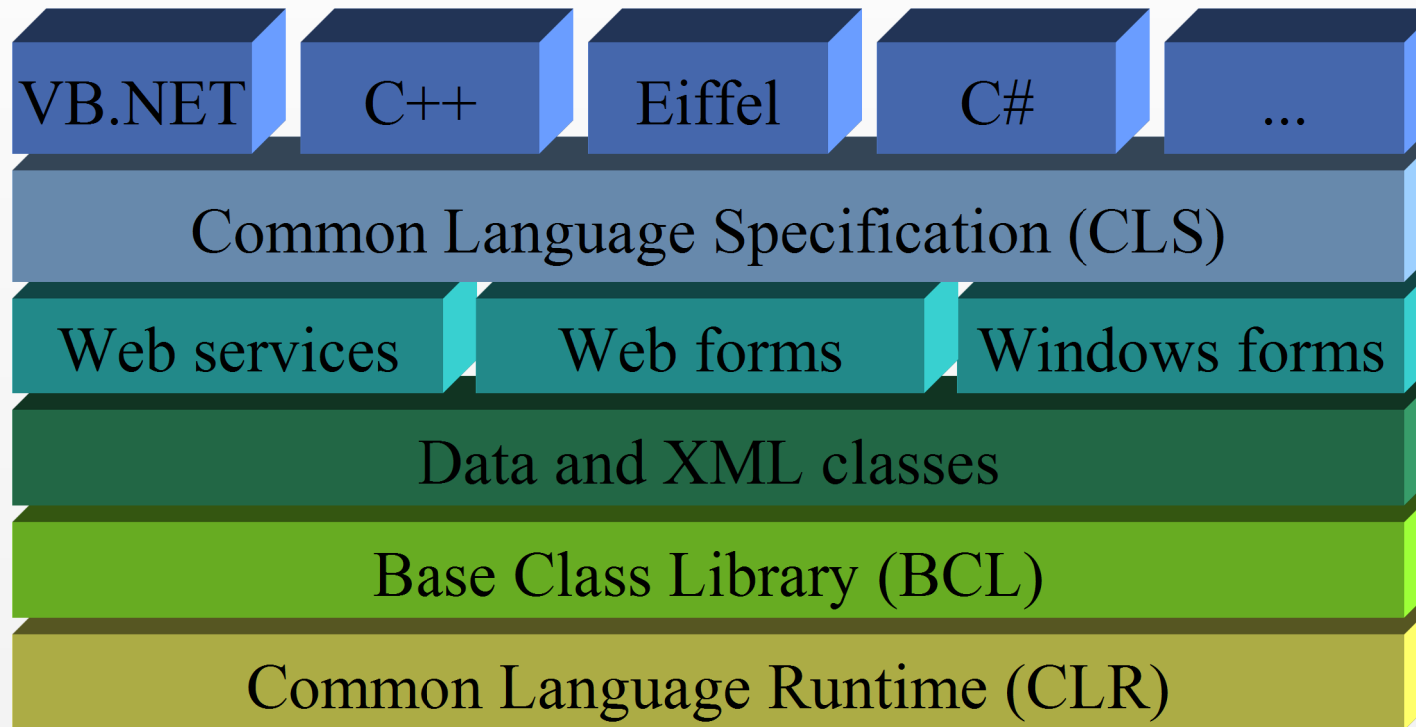
Compilation vers CIL



Compilation Just In Time



L'architecture .NET



Plan du cours

- Pourquoi .NET ?
- .Net et les autres solutions disponibles
- .NET : Présentation
- Les trois (ou quatre) C de .NET
- Le Common Language Runtime (CLR)
- Les bibliothèques de base
- Les outils

Plan du cours

- Pourquoi .NET ?
- .NET et les autres solutions disponibles
- .NET : Présentation
- Les trois (ou quatre) C de .NET
- Le Common Language Runtime (CLR)
- Les bibliothèques de base
- Les outils

Common Language Runtime (CLR)

Prise en charge de la bibliothèque de classes de base

Prise en charge multithread

Marshaleur COM

Vérificateur de type

Gestionnaire d'exceptions

Moteur de sécurité

Moteur de débogage

Compilateur MSIL
en code natif

Gestionnaire
de code

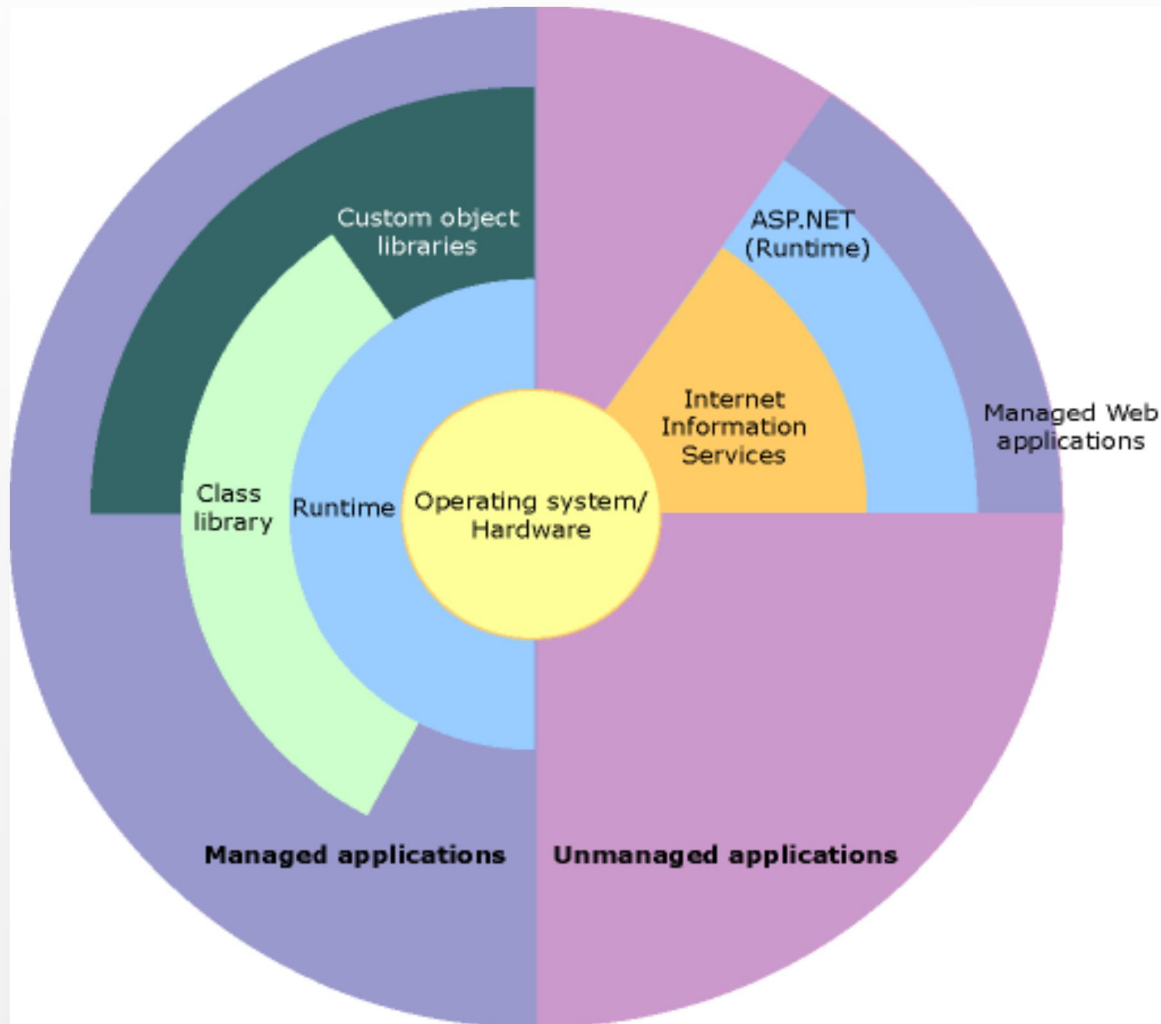
Ramasse
miettes

Chargeur de classes

Managed / unmanaged code

- Le code exécuté par CLR est appelé « code ménagé » (ou managed code en anglais)
- Managed code
 - code écrit pour le CLR
 - code "géré" par le CLR : ramasse-miettes, politique de sécurité, vérification du typage à l'exécution, ...
- Unmanaged code
 - code non géré par le CLR
 - exécution plus rapide

Managed / Unmanaged



Pourquoi .net ? .net versus... Présentation

Les 3 'C'

CLR

BCL

Le statut à part de C++ :

- Il y a le managed C++ et le unmanaged C++ ...
- Le managed C++ est une extension au C++ pour satisfaire les CLS
- Cohabitation possible entre du C++ managed et du C++ unmanaged (spécificité C++)
 - réutilisation de C++ existant
 - performances
- Le managed C++ a évolué en C++/CLI en 2005
 - syntaxe + agréable, meilleure intégration au CLI (Common Language Infrastructure), + proche de C++

CLR : Les principaux composants

- Système de type
- Système de métadonnées
- Système d'exécution

Le système de types

- 2 sous-systèmes :
 - types "valeur" (value types)
 - de base ou construits par le programmeur
 - types "référence" (reference type)
 - objet
 - Tous les objets héritent de la classe object
 - Méthodes : equals, finalize, GetHashCode, GetType, memberwiseClone, ToString
 - interfaces
 - pointeurs

Built-In Types

- Bool
- Char (16-bit unicode)
- Entiers signés et non signés, sur 8, 16, 32 et 64 bits
- Flottants sur 32 et 64 bits
- Object
- String
- Flottants et entiers dépendant de la machine

Systeme d'exécution

- Chargement et vérification de code
- Gestion des exceptions
- Compilation JIT
- Gestion mémoire
- Sécurité

Les métadonnées

- Pour stocker des informations sur les composants
- Les compilateurs stockent des infos sur les types
 - pour l'interopérabilité
 - pour le moteur d'exécution (gestion mémoire par exemple)
- Extension des métadonnées possible

Assemblies et Manifests

- Assembly
 - un manifeste
 - des classes de n'importe quel langage placées dans un espace de nom
 - des méta-données (images, textes, vidéos, ...)
 - créé par Assembly Linker (al.exe)
- Manifeste
 - informations de nom et de version
 - liste des types et l'endroit où ils sont dans l'assembly
 - informations de dépendance

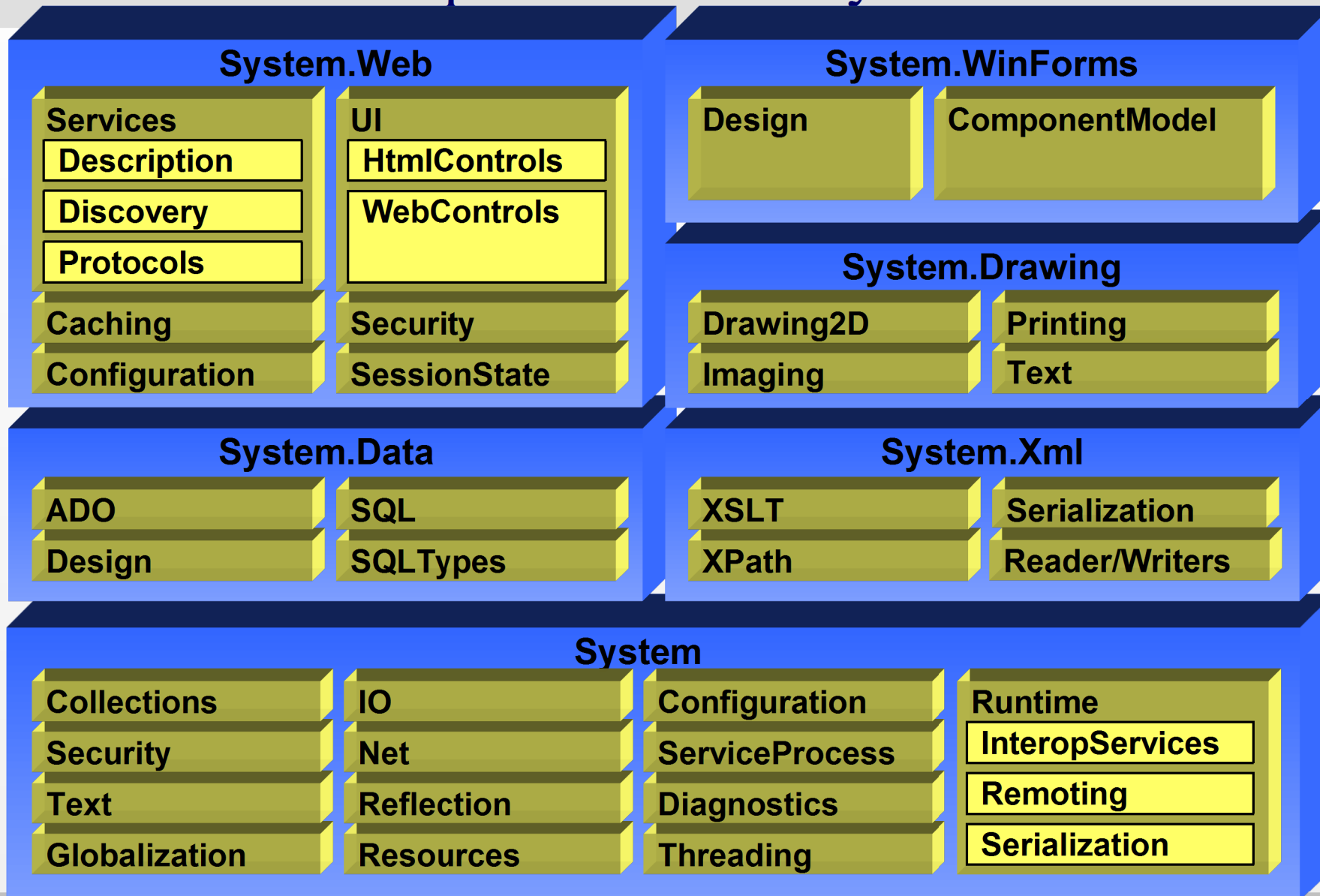
Plan du cours

- Pourquoi .NET ?
- .NET et les autres solutions disponibles
- .NET : Présentation
- Les trois (ou quatre) C de .NET
- Le Common Language Runtime (CLR)
- Les bibliothèques de base
- Les outils

La bibliothèque de classes

- Commune pour tous les langages
- Microsoft.Csharp, VisualBasic, JScript
 - classes pour supporter la compilation et la génération de code dans différents langages
- Microsoft.Vsa
 - pour l'intégration de scripts dans les applications
- Microsoft.Win32
 - classes pour gérer les exceptions systèmes et pour manipuler les registres du système
- System

L'espace de noms System



Les outils

- CLR
 - .net framework (gratuit !) - SDK, runtime
 - Rotor (win, FreeBSD, MacOS X)
 - Mono (Linux)
 - DotGNU (Linux)
- IDE
 - VisualStudio (payant ...)
 - SharpDevelop (gratuit)
 - WebMatrix (gratuit)
 - ...

Conclusion

- .NET et J2EE
 - .NET+COM+ $\cong$ J2SE + J2EE
 - efficacité ?
- Plusieurs langages et un langage dédié (C#)
- Plateforme cible principale : Win
- Une volonté de Microsoft vers des solutions non propriétaires
 - standardisation ECMA/ISO de C#
 - implémentations open source