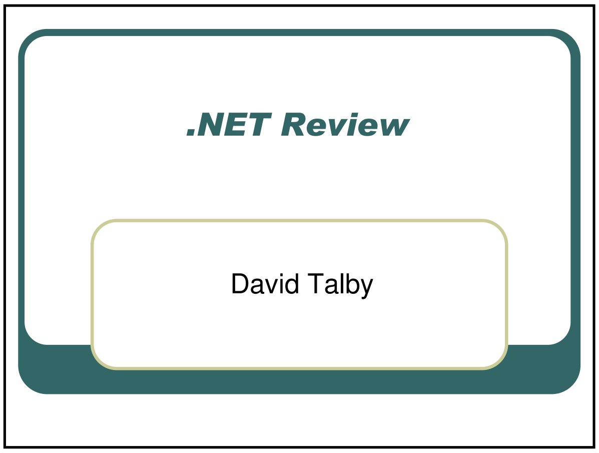
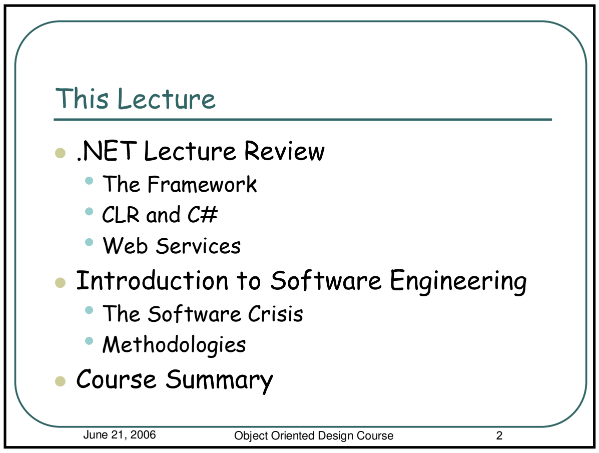


.NET Review

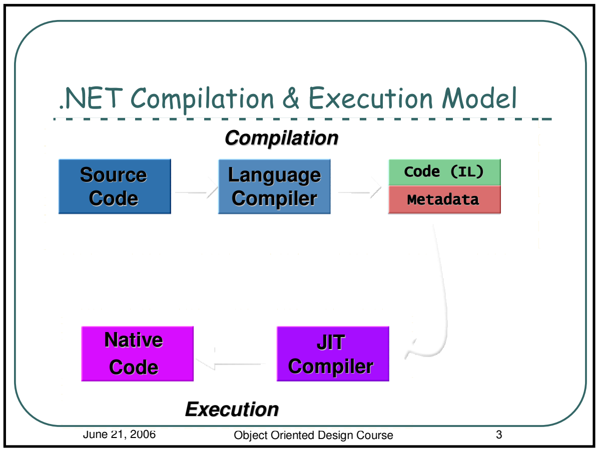
David Talby



This Lecture

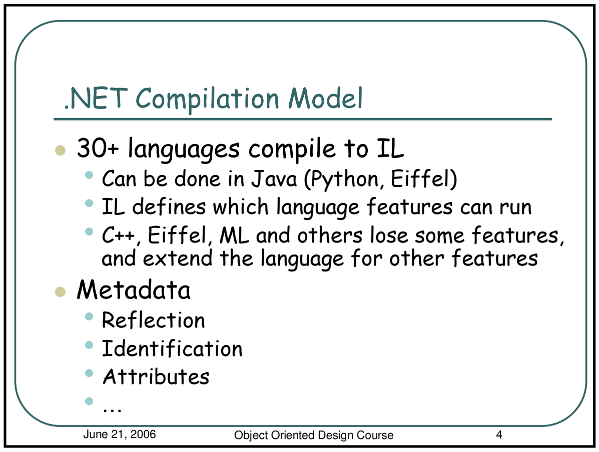
- .NET Lecture Review
  - The Framework
  - CLR and C#
  - Web Services
- Introduction to Software Engineering
  - The Software Crisis
  - Methodologies
- Course Summary

June 21, 2006 Object Oriented Design Course 2



.NET Compilation & Execution Model

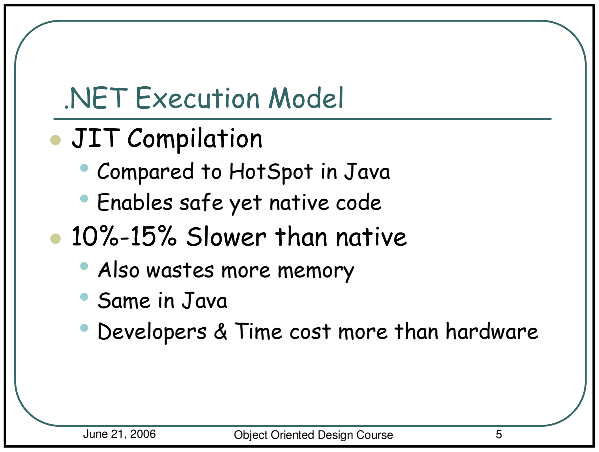
June 21, 2006 Object Oriented Design Course 3



.NET Compilation Model

- 30+ languages compile to IL
  - Can be done in Java (Python, Eiffel)
  - IL defines which language features can run
  - C++, Eiffel, ML and others lose some features, and extend the language for other features
- Metadata
  - Reflection
  - Identification
  - Attributes
  - …

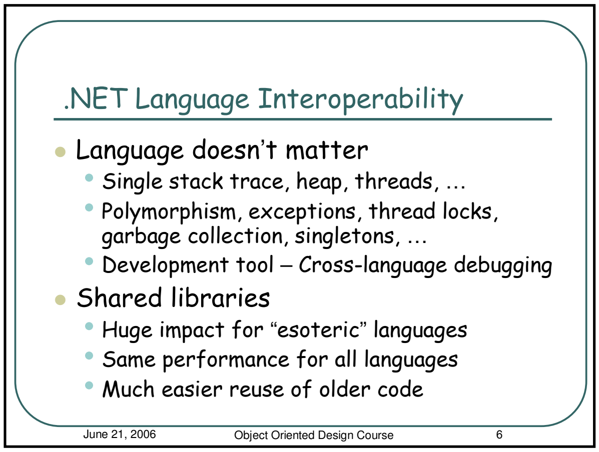
June 21, 2006 Object Oriented Design Course 4



.NET Execution Model

- JIT Compilation
  - Compared to HotSpot in Java
  - Enables safe yet native code
- 10%-15% Slower than native
  - Also wastes more memory
  - Same in Java
  - Developers & Time cost more than hardware

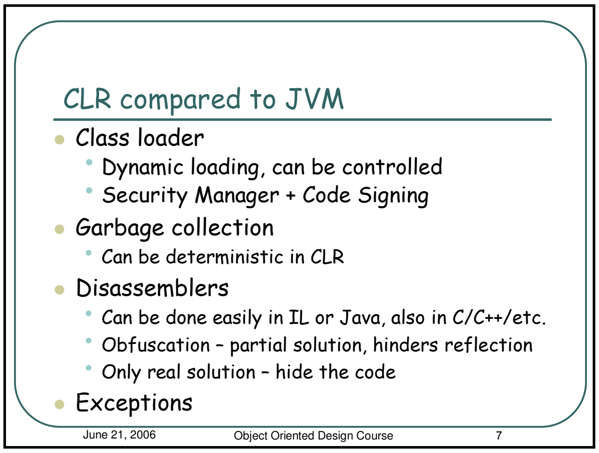
June 21, 2006 Object Oriented Design Course 5



.NET Language Interoperability

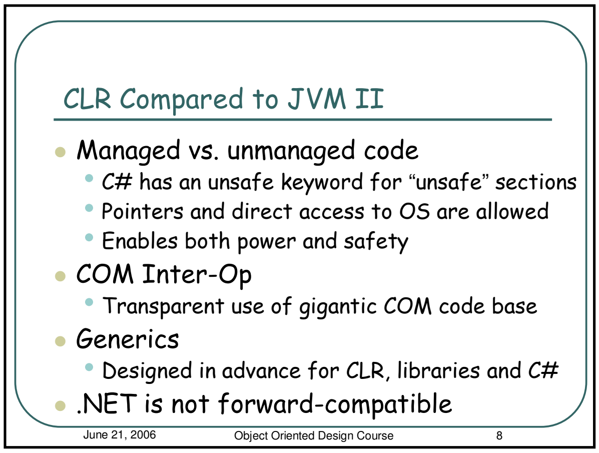
- Language doesn't matter
  - Single stack trace, heap, threads, …
  - Polymorphism, exceptions, thread locks, garbage collection, singletons, …
  - Development tool – Cross-language debugging
- Shared libraries
  - Huge impact for "esoteric" languages
  - Same performance for all languages
  - Much easier reuse of older code

June 21, 2006 Object Oriented Design Course 6



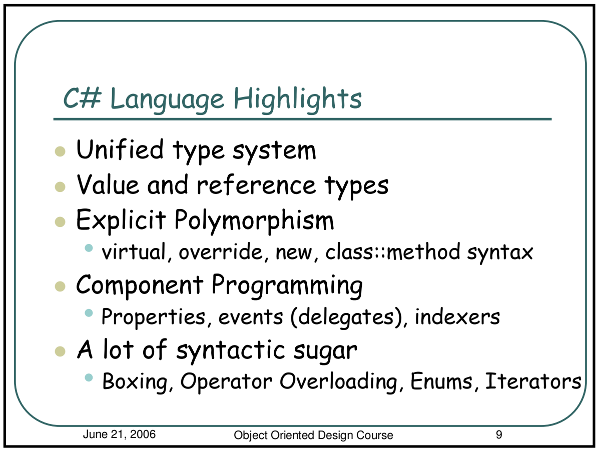
CLR compared to JVM

- Class loader
  - Dynamic loading, can be controlled
  - Security Manager + Code Signing
- Garbage collection
  - Can be deterministic in CLR
- Disassemblers
  - Can be done easily in IL or Java, also in C/C++/etc.
  - Obfuscation – partial solution, hinders reflection
  - Only real solution – hide the code
- Exceptions



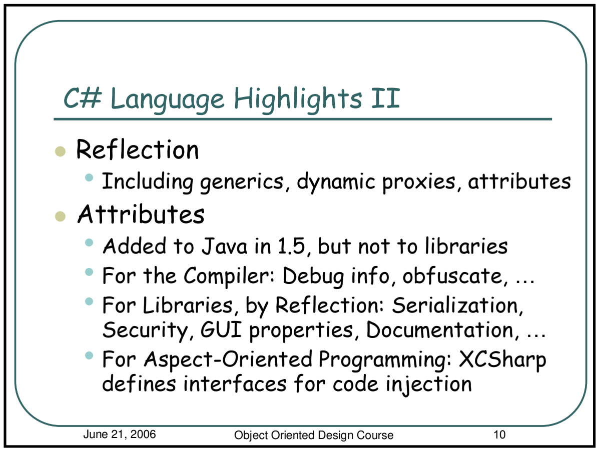
CLR Compared to JVM II

- Managed vs. unmanaged code
  - C# has an unsafe keyword for “unsafe” sections
  - Pointers and direct access to OS are allowed
  - Enables both power and safety
- COM Inter-Op
  - Transparent use of gigantic COM code base
- Generics
  - Designed in advance for CLR, libraries and C#
- .NET is not forward-compatible



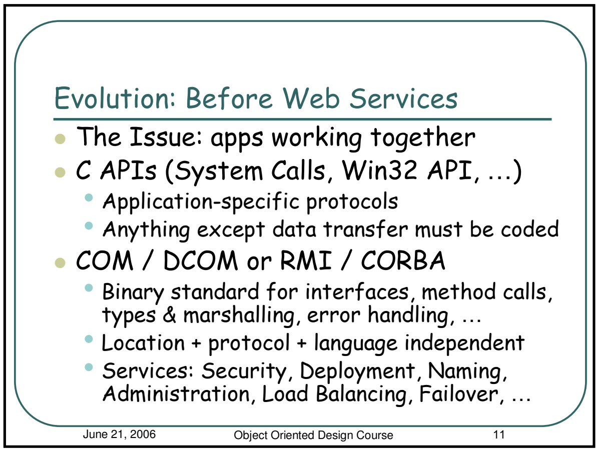
C# Language Highlights

- Unified type system
- Value and reference types
- Explicit Polymorphism
  - virtual, override, new, class::method syntax
- Component Programming
  - Properties, events (delegates), indexers
- A lot of syntactic sugar
  - Boxing, Operator Overloading, Enums, Iterators



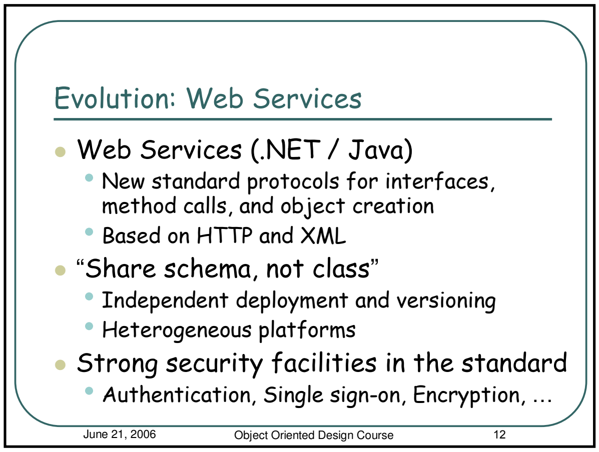
C# Language Highlights II

- Reflection
  - Including generics, dynamic proxies, attributes
- Attributes
  - Added to Java in 1.5, but not to libraries
  - For the Compiler: Debug info, obfuscate, …
  - For Libraries, by Reflection: Serialization, Security, GUI properties, Documentation, …
  - For Aspect-Oriented Programming: XCSharp defines interfaces for code injection



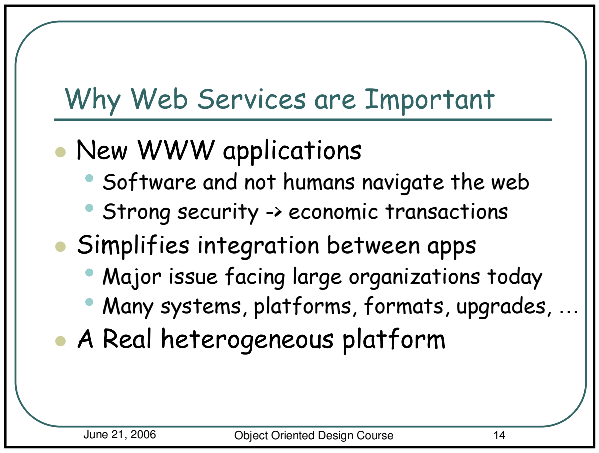
Evolution: Before Web Services

- The Issue: apps working together
- C APIs (System Calls, Win32 API, …)
  - Application-specific protocols
  - Anything except data transfer must be coded
- COM / DCOM or RMI / CORBA
  - Binary standard for interfaces, method calls, types & marshalling, error handling, …
  - Location + protocol + language independent
  - Services: Security, Deployment, Naming, Administration, Load Balancing, Failover, …



Evolution: Web Services

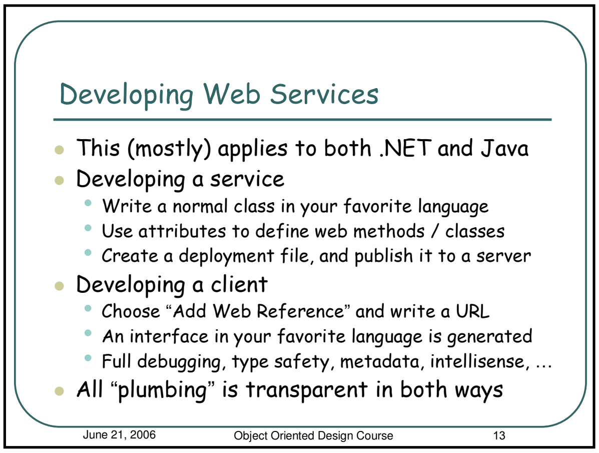
- Web Services (.NET / Java)
  - New standard protocols for interfaces, method calls, and object creation
  - Based on HTTP and XML
- “Share schema, not class”
  - Independent deployment and versioning
  - Heterogeneous platforms
- Strong security facilities in the standard
  - Authentication, Single sign-on, Encryption, …



Why Web Services are Important

- New WWW applications
  - Software and not humans navigate the web
  - Strong security -> economic transactions
- Simplifies integration between apps
  - Major issue facing large organizations today
  - Many systems, platforms, formats, upgrades, …
- A Real heterogeneous platform

June 21, 2006 Object Oriented Design Course 14



Developing Web Services

- This (mostly) applies to both .NET and Java
- Developing a service
  - Write a normal class in your favorite language
  - Use attributes to define web methods / classes
  - Create a deployment file, and publish it to a server
- Developing a client
  - Choose “Add Web Reference” and write a URL
  - An interface in your favorite language is generated
  - Full debugging, type safety, metadata, intellisense, …
- All “plumbing” is transparent in both ways

June 21, 2006 Object Oriented Design Course 13