

A stylized orange figure with a yellow circle above its head and two yellow circles on its sides, set against a white background. The figure is composed of thick, rounded lines.

# Architettura del software: .NET e gli strumenti

# Lorenzo Barbieri



- Sono un Senior Trainer/Consultant in ObjectWay SpA ([www.objectway.it](http://www.objectway.it)), specializzato in architetture Microsoft .NET, Windows, SQL Server, Visual Studio Team System, Virtual PC/Virtual Server
- Collaboro con UGIdotNET, INETA, Team System Rocks!, Windowserver.it e sono tra i soci fondatori di GUISA
  - [lorenzo.barbieri@objectway.it](mailto:lorenzo.barbieri@objectway.it)
  - [www.geniodelmale.info](http://www.geniodelmale.info)

**Visual Studio 2005 è disponibile: scegli il prodotto più giusto per te**  
[www.microsoft.it/msdn/vs2005/](http://www.microsoft.it/msdn/vs2005/)

- **Visual Studio 2005 Team Edition**

- Visual Studio Team Edition (for Architects, Developers o Testers) con MSDN Premium
- Visual Studio Team Suite con MSDN Premium
- Visual Studio Team Foundation Server

- **Strumenti professionali**

- Visual Studio 2005 Professional
- Visual Studio 2005 Professional con MSDN Professional
- Visual Studio 2005 Professional con MSDN Premium
- Visual Studio 2005 Tools for Microsoft Office System

- **Strumenti di base**

- Visual Studio 2005 Standard
- Visual Studio 2005 Express Edition

- **Altri strumenti**

- Visual SourceSafe 2005
- VisualFox Pro 9.0

**Licenze individuali:  
1 sviluppatore = 1 licenza**

**Dove acquistare:**

[www.microsoft.it/msdn/rivenditori/](http://www.microsoft.it/msdn/rivenditori/)

**Per informazioni:**

[itamsdn@microsoft.com](mailto:itamsdn@microsoft.com)

Prima di partire...

# LA SICUREZZA



# Tool per la analizzare e progettare applicazioni sicure

- Threat Analysis & Modeling (Microsoft)
  - <http://blogs.msdn.com/threatmodeling/>
  - Vi invito a vedere la sessione di Raf al Workshop Architecture Days:  
[http://www.guisa.org/files/folders/architecture\\_days\\_2006/entry247.aspx](http://www.guisa.org/files/folders/architecture_days_2006/entry247.aspx)
- Bisogna pensare SUBITO dall'inizio alla sicurezza della soluzione, non può essere AGGIUNTA in seguito.

Ripasso

# UML

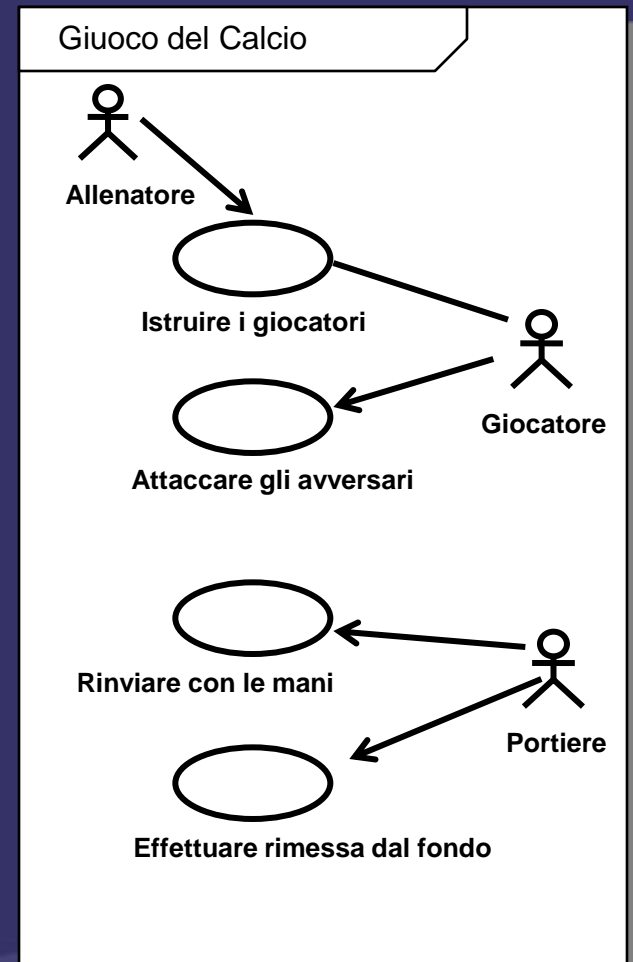


# Piccolo ripasso... UML

- Nei webcast precedenti abbiamo parlato spesso di UML, utilizzabile come linguaggio di analisi e modellazione indipendente dalla piattaforma.

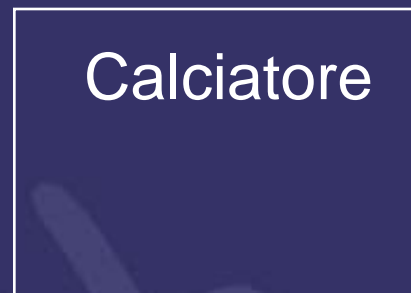
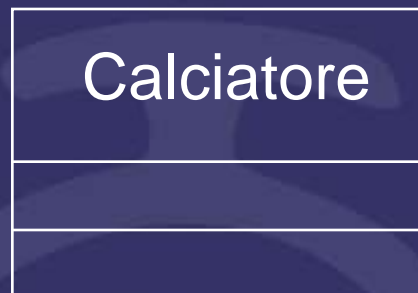
# Use-Case Model

- Non esiste solo il Diagramma "grafico" degli Use Case
- Use Case Model Survey
  - Lista di diagrammi, attori e breve descrizione
- Use Case Specification (una per ogni Use Case)
  - Descrizione
  - Flussi (principale, secondari, eccezione)
  - Pre/Post Condizioni
  - Requisiti supplementari dello UC



# UML: Diagramma delle Classi

- Una classe è composta da tre sezioni
  - Nome della classe
  - Struttura (attributi)
  - Comportamento (operazioni)
- Struttura e Comportamento sono opzionali, soprattutto in fase di analisi



# UML: Stereotipi

- Uno stereotipo è un elemento di modellizzazione che estende la semantica del metamodello
- Ogni classe può avere al massimo uno stereotipo
- Gli stereotipi comuni per le classi sono:

– Attore



– Classe Entity

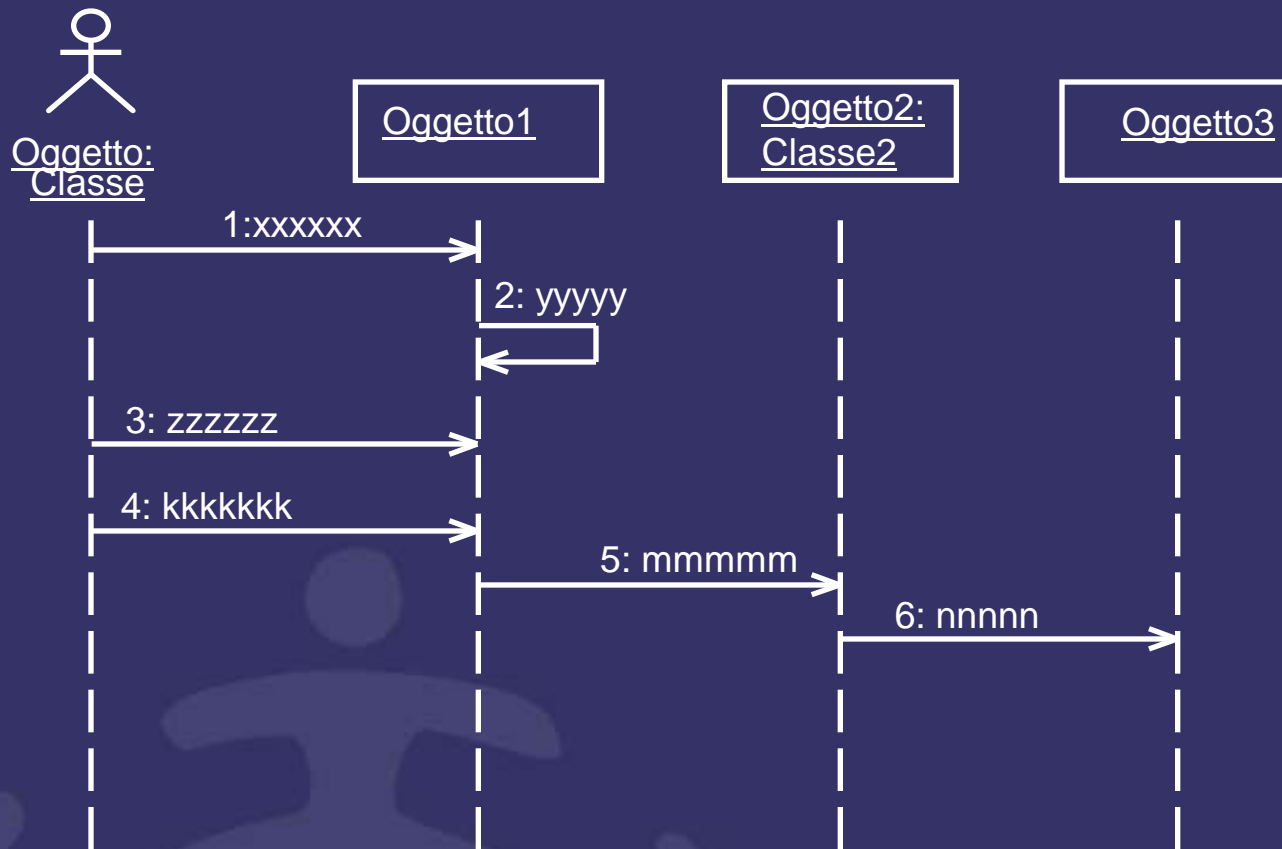
– Classe Boundary

– Classe Control



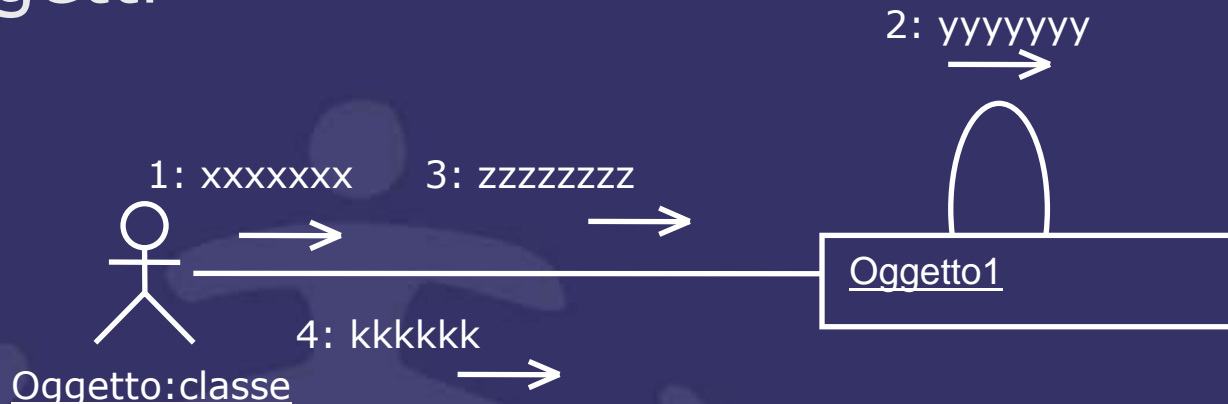
<code>&lt;&lt;Stereotipo&gt;&gt;</code> nome della classe

# UML: Diagramma di Sequenza



# UML: Diagramma di Comunicazione

- Equivale al Diagramma di Collaborazione di UML1.x
- E' un modo alternativo per rappresentare i messaggi scambiati da un insieme di oggetti



# Un ulteriore modello: Modello di Implementazione

- Nel Modello di Implementazione sono rappresentati
  - Processi
  - Componenti
  - Sottosistemi logici
  - Deployment
- **Bisogna valutare attentamente se farlo**
  - In UML, utile per modellare, può diventare complesso e poco utile in pratica
  - Con altri strumenti (Team System for Software Architects ad esempio) più legati all'implementazione specifica

Panoramica di alcuni tool UML

# VEDIAMO ORA ALCUNI STRUMENTI



# Per maggiori informazioni...

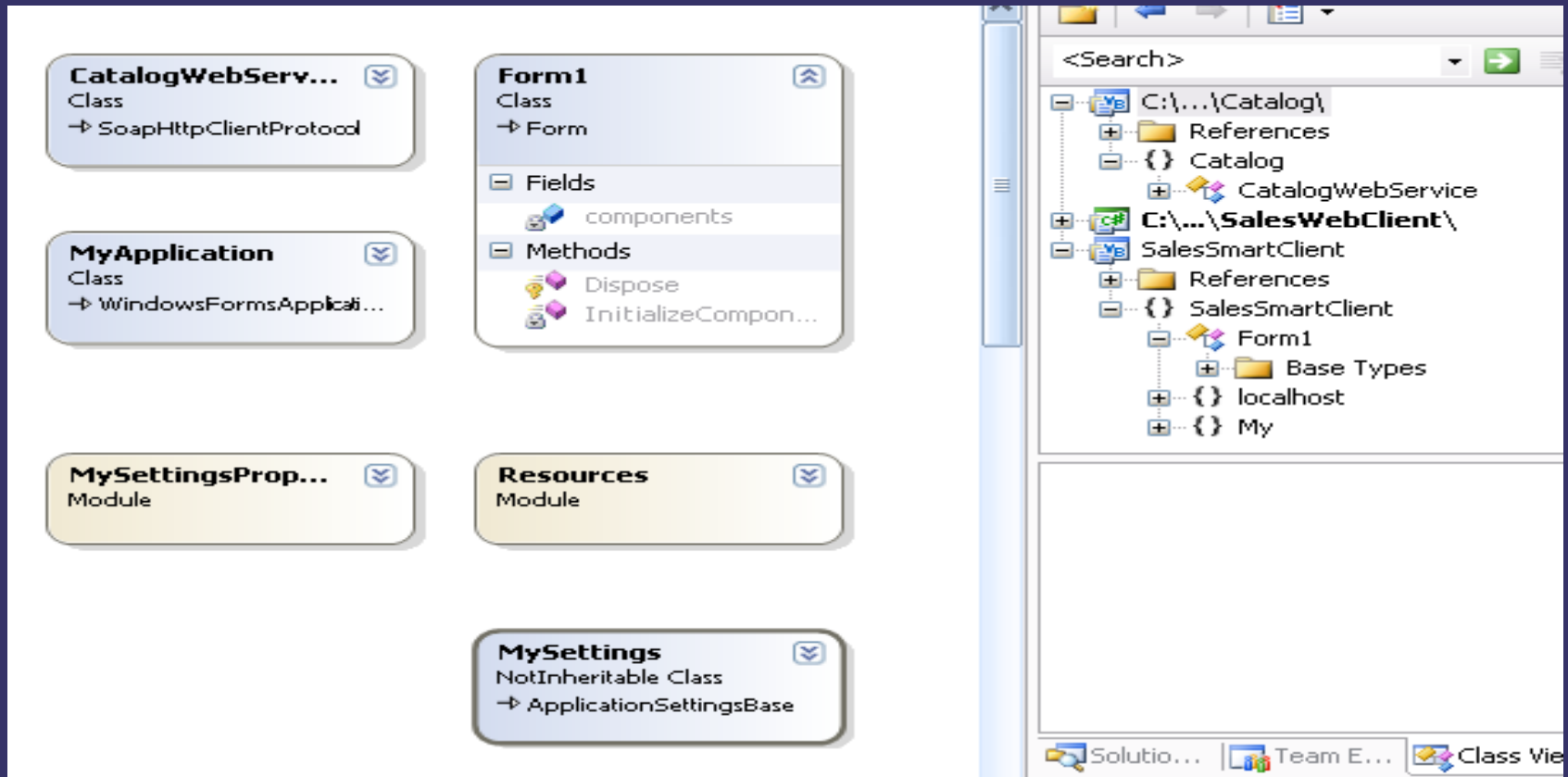
- ...su UML e sugli strumenti disponibili vi rimando alla presentazione di Riccardo al Workshop Architecture Days:
  - [http://www.guisa.org/files/folders/architecture\\_days\\_2006/entry237.aspx](http://www.guisa.org/files/folders/architecture_days_2006/entry237.aspx)

Visual Studio 2005 (dalla Standard in su)

# CLASS DESIGNER



# Class Designer di Visual Studio



# Attenzione... Non è un tool di disegno!

- Il class designer, è un tool bidirezionale, che mappa completamente le classi sottostanti e le loro proprietà, campi, eventi, metodi, etc... etc...
- Cambiando il codice cambia automaticamente anche il disegno e viceversa

# Attenzione... Non è un tool di disegno!

- Non è indicato per scenari di “design prototipale”, soprattutto su codice di produzione per testare eventuali modifiche, in quanto le modifiche verrebbero immediatamente apportate anche al codice!!!

# Class Designer e UML

- I diagrammi fatti con il Class Designer non sono diagrammi UML
  - Sono molto simili, ma più specifici, ad esempio contengono tutti i tipi del CLR, in VB è possibile definire moduli, etc...
- Non esiste un tool di migrazione da UML al Class Designer
  - E' possibile solo una migrazione "logica" ridisegnando le classi

# Alcune note sul Class Designer

- Il Class Designer non supporta l'import/export in formato XMI
  - Ogni tool ha la sua variante... In pratica è un finto standard
- Il Class Designer non supporta (per la V1) altri linguaggi oltre a VB, C# e J#
  - Il supporto per C++ è previsto per la prossima versione

# Tipi generici

- Il Class Designer non supporta la creazione di tipi generici, ma supporta la loro visualizzazione

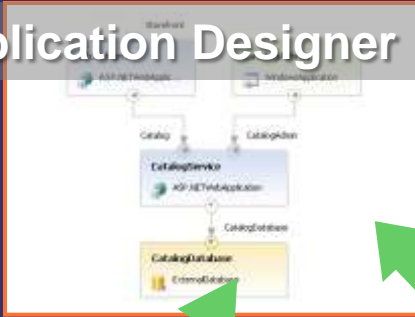
Visual Studio Team System

# DISTRIBUTED SYSTEMS DESIGNERS

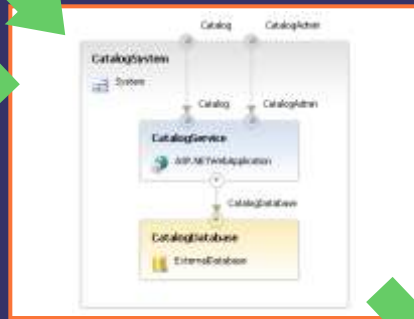


# Distributed Systems Designers

Application Designer



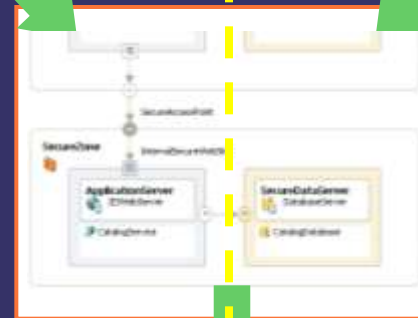
System Designer



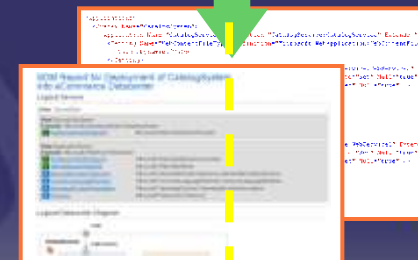
Logical Datacenter Designer



Class Designer, Codice



Deployment Designer



Deployment Report

**MA QUESTI TOOL NON  
SONO UML...**



# Serve veramente UML?

- Molto spesso di UML si usano solo pochi e specifici diagrammi:
  - Use Case, Sequence e Collaboration diagram
  - Diagrammi delle classi
- Per quanto riguarda Use Case, Sequence e Collaboration, si può usare Visio, Word e tutti gli strumenti che si sono sempre usati.

# Diagrammi delle classi

- I diagrammi delle classi sono molto simili a quelli ottenibili dal Class Designer di VS2005 (presente dalla Standard in su)
  - La differenza è che il Class Designer di VS2005 mappa 1:1 tutti i concetti di .NET (attributi, etc...) senza richiedere strane estensioni, permettendo un round-trip completo codice<>modello
  - I diagrammi delle classi UML sono generici, e vanno estesi per mappare i concetti specifici

# E per gli altri diagrammi?

- Per gli altri diagrammi la situazione è mista, perchè quelli UML sono troppo generici per produrre codice, e quelli di VSTS non sono standard.
- Nel dubbio bisogna scegliere quelli che risolvono al meglio il proprio problema:
  - Diagrammi UML se richiesti (bandi di gara, etc...)
  - Diagrammi VSTS se servono
  - Entrambi se ci aiutano

# Attenzione al BDUF!

- Attenzione che se i modelli non servono in pratica, rischiano di restare carta straccia.
- Molto spesso si tende a fare il Big Design UpFront, disegnando modelli su modelli che non verranno però più mantenuti.
- Il vantaggio dei designer di VSTS è di essere completamente bidirezionali

# La visione di Microsoft per il Modeling

**Our vision is to change the way developers perceive the value of modeling.** To shift their perception that modeling is a marginally useful activity that precedes real development, to recognize that modeling is an important mainstream development task and not an activity primarily focused on documentation.

**Visual Studio 2005 Team System Modeling Strategy and FAQ**

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnvs05/html/vstsmode.asp>

# Ruolo dei modelli

- Un modello rappresenta un set di astrazioni che supporta gli sviluppatori in compiti ben definiti
- I modelli possono essere usati per aggregare informazioni da varie fonti, e possono aiutare nel verificare la consistenza del sistema
- I modelli possono essere implementati attraverso un processo di "compilazione", dove il codice, i file di configurazione e tutti gli altri artefatti possono essere generati automaticamente

# Raccomandazioni:

- Microsoft raccomanda di usare UML (o simili) per le seguenti attività:
  - Sketching, White boarding, Documentazione, Disegni e altri diagrammi che non generano direttamente codice
- Raccomanda i DSD per:
  - Astrazioni precise dalle quali può essere generato il codice, Disegni e altri diagrammi che servono per generare codice o altri diagrammi in maniera automatica

# Raccomandazioni:

- Nessuno dei tool precedenti per descrivere in maniera dettagliata la logica dei programmi (almeno per qualche anno ancora... 😊)
- Fonte:
  - <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnvs05/html/vstsmode.asp>

# ALTRI TOOL PER ARCHITETTI



# Tool per la gestione del ciclo di vita del software

- Team Foundation Server permette di gestire completamente il ciclo di vita del software, dall'idea, all'analisi, progettazione, gestione dei cambiamenti, etc...
- Per maggiori informazioni:
  - <http://www.microsoft.com/italy/msdn/risorse/msdn/team/path/default.aspx>

# Tool per la gestione dei requisiti

- Borland CaliberRM permette di gestire i requisiti dalla cattura all'implementazione gestendone il ciclo di vita, la tracciabilità, etc...
  - Dispone di moduli aggiuntivi per stimare tempi e costi, Function Points, etc...
- Esiste una versione integrabile in Visual Studio Team System e Team Foundation Server

Ricordatevi di compilare il modulo di Feedback

**DOMANDE?**

