

VISUAL UNIT PROGRAMMING WITH CAPEOPENSTUDIO.NET

Maurizio Fermeglia, Marco Carone, Luca De Bernardi and Marco Parenzan - MOSE Lab, Università degli Studi di Trieste

maurizio.fermeglia@mose.units.it; marco.carone@yahoo.it; luca.debernardi@gmail.com;
marco.parenzan@libero.it

.NET introduced in 2001 a revolutionary breakthrough in programming techniques. It gave to Visual Basic 6.0 community a new broad development platform with the same user interface but a full new object oriented set of tools.

In a long search for developer productivity, the Molecular Simulation Laboratory (MOSE) found out that object-oriented domain models and code generation are useful for minimizing infrastructural, repetitive coding in the field of process engineering and particularly in Cape Open generation code. Indeed, MOSE designed developed in 2007 the capeopentoolkit.net, a developing tool that hides all the calls to COM API thus simplifying the writing of the equations for Cape Open Modules.

Visual development environments are available since Visual Basic 1.0 (1990!), allowing the definition of user interface code in terms of Forms. Visual Studio.NET (2001) allowed the definition of Forms in terms of .NET code generation: the Visual Studio form designer is a “visual language” to generate the code that defines forms. Visual Studio.NET 2005 have introduced a great new feature: the ability to customize and develop new kinds of visual designers, with a technology broadly known as Domain Specific Languages (DSLs). Visual Studio.NET 2008 have finally released that ability and the license to customize and redistribute the entire Visual Studio.NET IDE as a vertical tool for specific needs (no royalty or other fees are due to Microsoft).

Recently, we have developed Visual Cape Unit, a DSL for the definition of CAPE-OPEN units in a visual form into Visual Studio.NET, based on the original API defined into capeopentoolkit.net. In this work we will present capeopenstudio.net, a vertical Visual Studio .NET distribution freely available, specific for CAPE-OPEN development, and including natively Visual Cape Unit. Details on the implementation and examples of use will be presented.