



SCORE/Integration Suite and Microsoft .NET

MA 24010.02

Contents

- SCORE/Integration Suite 3**
- Microsoft .NET and C# 3**
- Adaptive Frameworks 4**
 - Frames 5
 - C# Proxies..... 6
 - Web Services 6
 - Compound Operations / Packaging 7
 - Interface Versioning..... 7
 - Persistence and Session Management 8
 - Transactions 8
- Features..... 8**
 - Data Conversion..... 8
 - ANGIE Generator Technology 9
 - Development Environment 9
 - Further Information..... 10
- Literature.....10**

This document is addressed to project managers, software engineers and readers with an interest in technology, who are involved with the special tasks of application integration (Enterprise Application Integration), component-based software development in general and Microsoft .NET technology in particular.

You will find more information on the technologies and concepts of the DELTA SOFTWARE TECHNOLOGY GROUP on our website at <http://www.D-S-T-G.com>

SCORE/Integration Suite

SCORE/Integration Suite is the powerful solution for the integration of standard and individual software in heterogeneous environments. *SCORE/Integration Suite* is based on the methodical approach of component technology. One particular focal point of the further development of the product is the integration of existing application systems into new component architectures - such as Microsoft .NET.

With *SCORE/Integration Suite* it is easy to use new technologies within the environment of existing ones: Generators transform existing applications into components (Componentising), without the applications themselves having to be modified. Afterwards, these components also can be integrated into environments that do not provide a real component architecture, such as OLTP systems.

SCORE/Integration Suite realises these concepts with Generated Adaptive Frameworks. No prefabricated adapters are used; rather the concept is based on frameworks that are mechanically adapted to the respective applications and environments in an option- and parameter-controlled manner, in other words, by generators.

The integration solutions implemented with *SCORE/Integration Suite* guarantee portability – as opposed to special adapters or self-written connections – and thus the independence of the integrated application from the operating system in use, TP monitors and middleware products, including the *SCORE/Integration Suite*.

SCORE/Integration Suite from DELTA SOFTWARE TECHNOLOGY provides optimum integration in object-oriented component environments.

Microsoft .NET and C#

Note: All statements about .NET in this document are based on the Beta 1 version of Microsoft's .NET Framework. Therefore, there might be some changes before the final version will be released.

With its .NET Framework, Microsoft provides a new platform for the development of Windows and Internet applications. An important focus is the development of components, which should simplify the integration of locally executed applications with the Internet (see also [1]).

Particularly suitable for this are Web Services. The Web Services of .NET are distributed components that can be addressed with various protocols via a network and the Internet. The communication between clients and the Web Services is usually executed via SOAP (**S**imple **O**bject **A**ccess **P**rotocol, see also: [2]) using HTTP.

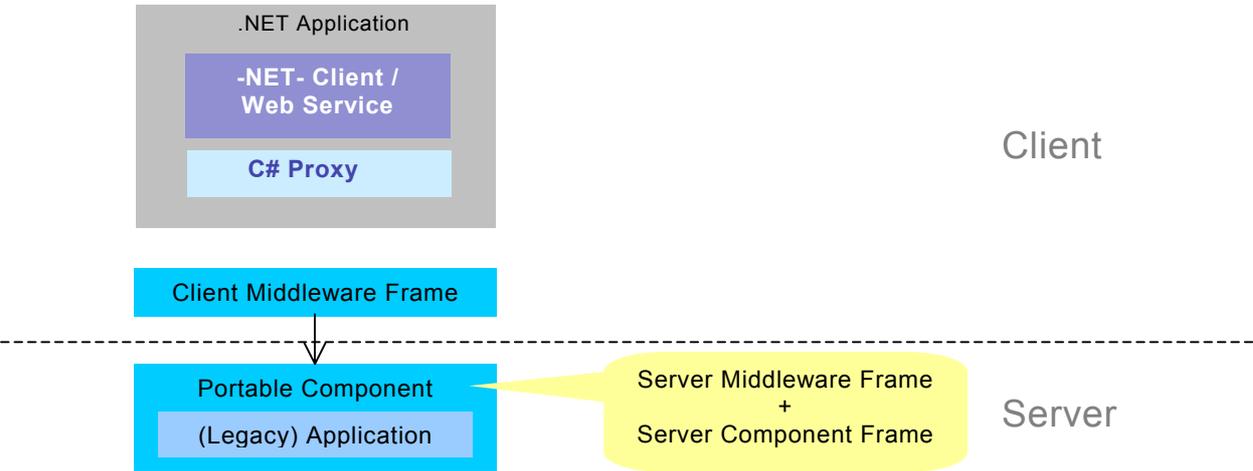
A new programming language is being introduced together with the .NET Framework: C# - a further development of C++ - with the aim of creating a component-oriented language.

In a .NET environment all .NET compatible languages are based on one **Common Language Runtime system (CLR)**. This enables a smooth exchange of data between modules that have been created in different languages. Another feature is that e.g. classes in Visual Basic.NET can be derived from C# classes directly or they can call each other.

SCORE/Integration Suite for .NET will use C# as language for the generated source code.

Adaptive Frameworks

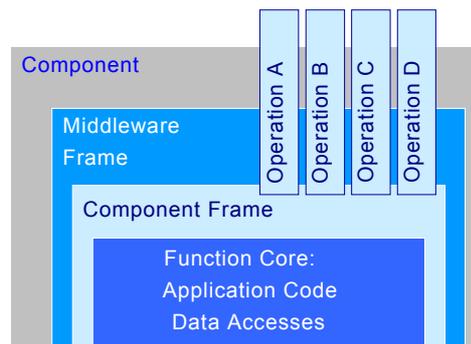
Within the scope of *SCORE/Integration Suite*, the definitions of the components and their interfaces are stored centrally in a Component Repository. The definition is processed on a logical and a declarative level – independent of the subsequent implementation in a concrete environment. The maintenance of these specifications is supported by an intelligent, interactive tool: the *Component Manager*. Both, the external views of a component – these are the interfaces with their operations – and the internal view, in other words, the interface to the function core are defined; they are complemented with the mapping between external and internal view. Various modules are generated from the component definition; these form the intermediate layers between the client components, server components and the respective middleware. In the following these modules will be described as frames. Together with the server application, they form an integrable server component. In other words, the frames implement the connection to the middleware in use and, as well, the mapping rules between the interfaces and the internal function interface of the application that is to be integrated.



Frames

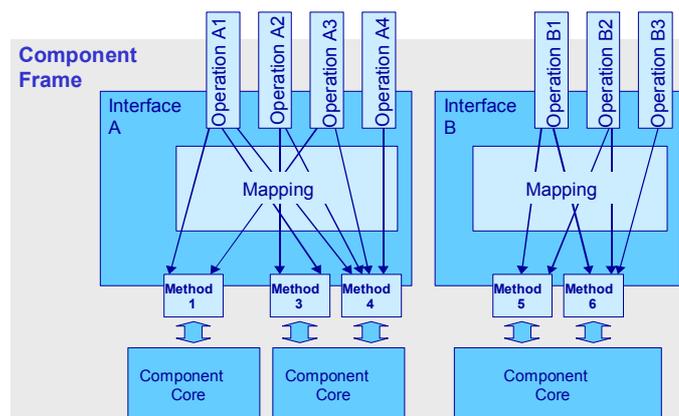
On server side, the generated frames consist of the component frame and the middleware frame.

- In the component frame, the interfaces with the mapping of the operations onto the function core are implemented.
- The middleware frame contains all middleware calls – corresponding to the concept of Isolation Layers. It provides the connection between the .NET-application and the component frame.



Middleware frames are available for the server side and, if required, for the client side as well.

All frames are created in a customised way in order to suit the respective middleware and system platform (operating system, TP monitor). Together, they form so-called generation targets in terms of the *SCORE/Integration Suite*.



The server component frame is generated in the language that corresponds to the server application, for example C/C++ or COBOL 85. Among other things, this module implements the mapping between the interfaces that are available to the outside and the function core of the server application. It executes the calls of the server application and thus ensures the integration of the server application on a logical level.

By means of the generated component frame the application to be integrated becomes a *portable component*. The portability is a relevant characteristic of the component and is also supported and amplified by the generation targets that are available for *SCORE/Integration Suite*.

C# Proxies

SCORE/Integration Suite for .NET generates special proxy classes from the interface definitions in the component repository. These classes enable complete object-oriented access to the server components. They behave as if they were actually written in a .NET language themselves.

The proxies convert the method calls of the .NET-clients into the respective server component calls. Therefore, it is not obvious for a .NET-programmer that a component is being called up which has been written in a different language, or is located on a remote computer.

Web Services

C# proxies can also be generated as complete Web Services. Thus, existing components can be made available via the Internet directly.

The following modules are generated:

- Required C# classes
- WSDL file (**Web Service Description Language**): this file contains a description of the Web Service and the interfaces available. Using this file enables client applications to access the Web Services. The WSDL format is not .NET-specific and has been specified by Microsoft in co-operation with IBM.

Compound Operations / Packaging

By providing the possibility to define so-called Compound Operations, SCORE/Integration Suite supports this optimisation within the scope of the layer architecture.

Compound operations are specified in the component definition. However, they are strictly separated from the modelling of the application logic. The coding that is necessary for Compound Operations is being generated. The separation of technical optimisation and object logic enables the optimisation to be adapted while retaining the logical application structure.

There are two types of Compound Operations: Request Packaging coalesces several operation calls into one; Response Packaging means collecting several responses into one message.



In order to minimise the number of server calls the compound operations are constructed automatically in the generated .NET proxies before executing the server call.

Interface Versioning

A special task in application integration is interface versioning. Particularly when a large number of clients can only be adapted in a time-shifted manner.

If it is only about a new version of the internal interface of the server application, i.e. the modifications are restricted to the internal interface, it is sufficient to adapt the component definition and regenerate the server component frame.

On the other hand, a new version can also require modifications in the (external) interfaces. For this purpose SCORE/Integration Suite provides a versioning concept on operation level: The respective component provides different versions of individual operations simultaneously.

The versioning is processed invisibly to the application programmer. The mechanism is implemented via the generators on the client and server side.

Persistence and Session Management

Occasionally, it is necessary to store certain components or parts of components over several calls. One particular requirement is the ability to access certain data areas even over transaction borders. *SCORE/Integration Suite* supports these requirements with a flexible Workspace Management. Individual data items can be given the attribute "stateful" in order to keep them in the workspace. The necessary calls are generated into the server side frames. The application programmer only is responsible for the specification within the Component Manager.

In addition, *SCORE/Integration Suite* supports logical co-operation connections (Logical Sessions) between client and server components. In combination with the Workspace Management this also allows processing statuses to be stored over transaction bounds. In that end, the co-operation connection that is valid for the client is identified, again without the application programmer having to take care of it. In detail, the possibility is provided of defining global and server-specific workspaces and linking them with sessions if required.

Transactions

Microsoft's .NET framework uses the services of COM+ and DTC (Distributed Transaction Coordinator) for transaction control.

SCORE/Integration Suite supports the mechanisms of COM+ and, if required, forwards transaction terminations to the TP monitor on the server side. Optionally, the server components can be informed of the imminent termination of the transaction before the actual end via a special call and can thus execute "clearing up operations".

A special transaction class is available for applications that do not require a client side transaction control. The transactions can be controlled on the server side with the usage of this class.

Features

Data Conversion

Not all data formats known on the server side are covered by .NET data types. Therefore, *SCORE/Integration Suite* supports the connection of these different worlds with automatic conversions. These conversions also remain hidden from the application developer. Data always is provided in the respectively required format.

ANGIE Generator Technology

The generators of *SCORE/Integration Suite* are based on the latest *ANGIE* generator technology. With it, code fragments are treated as objects that can be stored in a repository. This means that they can be processed and evaluated in several different generation stages. In addition, the frame technology used in *ANGIE* supports the definition of design patterns.

ANGIE is a powerful and robust technology that was especially developed with the objective of long-term maintainability and portability. It thus represents a solid technological basis for *SCORE/Integration Suite*.

Development Environment

SCORE/Integration Suite provides a comfortable interactive development environment for the optimum solution of integration tasks in application development.

DELTA SOFTWARE TECHNOLOGY also supports the development of portable server components with its *SCORE/Development Suite*.

You can integrate your heterogeneous development environments with the development middleware *Scout²*. This is an open environment, which allows the integration of any type of tool and which can itself be integrated into existing environments.

Further Information

Please do not hesitate to contact us if you like to have more information about the possibilities and services of SCORE/Integration Suite and the other products of DELTA SOFTWARE TECHNOLOGY.

DELTA SOFTWARE TECHNOLOGY
Eichenweg 16
57392 Schmallenberg
Germany

Literature

- [1] Microsoft .NET:
<http://msdn.microsoft.com/net>
- [2] Delta Software Technology GmbH: SCORE/Integration Suite and SOAP
(MA 24012)

© 2001-2002 Delta Software Technology GmbH

www.Delta-Software-Technology.com