



**Model based multi-tier &
multi-platform application
generation**

Javier Hernandez

CARE Technologies S.A.

<http://www.care-t.com>

Partida Madrigueres, 44.

03700 Denia, Alicante, Spain.

jhernandez@care-t.com

EXECUTIVE SUMMARY

As the **Object Management Group** [10] states in its **Model Driven Architecture** (MDA) [11] initiative, a key success factor in software development should be separation of business and application logic from implementation platform. Existing platforms evolve quickly and new ones appear, so we need to separate our application definition from the implementation details. This will allow us to easily migrate from one platform/technology to another, especially if tools exist that support this process.

OlivaNova Model Execution products [9] were conceived with these ideas in mind, even before the MDA initiative. Applications are specified with object-oriented formal models, which are used to generate, in an automatic manner, the full application in the latest multi-tier platforms. Starting from a conceptual model, we can quickly see the same application running in a Java 2 Enterprise Edition (J2EE) [3] application server or as a rich desktop client using Microsoft COM+ [7] components.

This paper is intended for

- system developers,
- system analysts,
- system designers,
- application development managers,
- IT managers,
- IT consultants.

TRADEMARKS

OlivaNova is a trademark of CARE Technologies S.A.
OMG is a trademark of the Object Management Group
CORBA is a Registered Trademark of the Object Management Group
MDA is a trademark of the Object Management Group
UML is a trademark of the Object Management Group
Windows is a trademark of Microsoft Corp.
Java and J2EE are trademark of Sun Microsystems, Inc.

While the information in this publication is believed to be accurate, CARE Technologies makes no warranty of any kind to this material including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. CARE Technologies shall not be liable for errors contained herein, or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

COPYRIGHT NOTICE

No part of this publication may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, photocopying, recording or otherwise, without prior written consent of CARE Technologies. No third-party intellectual property right liability is assumed with respect to the use of the information contained herein. CARE Technologies assumes no responsibility for errors or omissions contained in this white paper. This publication and features described herein are subject to change without notice.

Copyright © 1999-2003 CARE Technologies. All rights reserved.

All products or services mentioned in this white paper are covered by the trademarks, service marks, or product names as designated by the companies that market those products.

Table of Contents

1	Introduction	5
2	Overview of Client/Server Software Architectures.....	5
3	Platforms for building n-tier applications	6
4	Building n-tier applications with Oliva Nova Products.....	7
4.1	The Persistence Tier	8
4.2	The Middle Tier	8
4.3	The Client Tier	9
5	Conclusions.....	9

1 Introduction

In the last two decades as hardware technology has evolved, the software that runs on this hardware has evolved as well. New technological advances have meant new software architectures and new products supporting these architectures. But migrating existing applications has been a difficult and a highly resource-consuming task. One of the reasons has been that application semantics have been specified on paper as well as being in the source code.

Now is the time to use newly developed tools that allow us to express, store and process application specifications for the purpose of building them. Therefore in the future, when new products and new architectures appear, we will only need a migration tool to create new versions of our applications.

Oliva Nova Model Execution products are a real example of these ideas: with **Oliva Nova Modeling Software** you can express your business database-based application by using a powerful object-oriented formal language with a Unified Modeling Language (UML)-like [13] graphical notation. The specification is stored in an eXtensible Mark-up Language (XML) [14] file that is used by **OlivaNova Transformation Engines** to generate full applications for diverse platforms. Up to now, with OlivaNova you can move from one n-tier platform to another. In the future, when new architectures or new enhancements to current products will emerge, new transformation engines will allow you to migrate to these without much hassle.

2 Overview of Client/Server Software Architectures

Well, at the beginning it was ... the *mainframe*. This was not really a client/server architecture, a central host with many terminals. It was difficult to build Graphical User Interfaces (GUI) or access multiple, disperse databases. With the advent of PC networks, *file sharing* architectures appeared where the server provided files from a shared location and made them available to the desktop environment. With this architecture, Graphical User Interfaces became popular. However, this architecture can only satisfy a very limited number of simultaneous users.

To solve file sharing architecture limitations, the *client/server* architecture appeared, where instead of a file server we have a database server. The architecture provides a query-response paradigm instead of file transfer. Clients are heavy GUI applications placed in the

user's desktop environment. The management of the process is split between these applications and the database server environment. This architecture is a good solution when the number of clients on a Local Area Network are limited to about a hundred. Significant problems occurring with this approach are client distribution and the upgrading of client software. The evolution of client-server architectures lead us to *3-tier* and *n-tier* architectures.

In the 3-tier architecture a middle tier was added between the client (Graphical User Interface), and the database server. This new tier can be implemented in several ways, such as transaction processing monitors, message servers or application servers. The result is improved performance for groups with a large number of users (in their thousands) and improved flexibility. With the enhancement of web technologies, a new kind of light client is used: the browser. As a result, new tiers such as the web server and engines that provide dynamic web content are added. Advantages of this architecture include an easy deployment and updating of the clients.

3 Platforms for building n-tier applications

N-tier architectures are the most versatile ones and there are a wide variety of products on the market that support the different layers. Here is a quick summary:

In the persistence tier, Relational databases and SQL were and are still today, the most predominantly used. We can choose between many products that offer high quality and performance. Object-Oriented Databases [8] are trying to get on the market but with little success up to now. Therefore, the component that is almost always present in a business application is a relational database.

In the middle tier we can choose between numerous technologies, many of them are based on server component models. The best known are: CORBA [2], COM+ [7] and EJB [3]. CORBA is the most complete and powerful model but with a high level of complexity that has not lead to its acceptance. COM+ is a Microsoft extension of their ActiveX model for server components and are easy to develop and use. EJB is the SUN Java™ extension for server components. These are far richer than COM+ but not as complex as CORBA.

For the client part, GUI desktop clients have been mostly used. These have been built in many different programming languages, but Rapid Application Development environments, like Microsoft's Visual Basic, have been the most popular and successful tools on the market. In the last few years, a new approach has been taken: the web client. To support

this possibility, several extensions to web servers have been created for adding dynamic web content. The most popular are ASP [6], Cold Fusion [5], PHP [12] and JSP [4]. ASP is well suited to work with COM+ server components. JSP can work seamlessly with EJB (as both are part of the same platform Java 2 Enterprise Edition). Cold Fusion is a very flexible environment and can interoperate with a wide variety of server components like COM+ and EJB.

4 Building n-tier applications with Oliva Nova Products

The **OlivaNova Modeling Software** captures application logic, rules, and the user interface. This is done using an Object-Oriented formal language that is for the most part generated from UML-like graphical diagrams. The Modeler uses class diagrams, state diagrams and also introduces new ways to capture functional and GUI requirements. All these components are incorporated in a model that is then transferred in XML format to the **OlivaNova Transformation Engines**.

The **OlivaNova Transformation Engines** take the XML models and act as model compilers to create all the artifacts needed in the different tiers: the database scripts, uncompiled server and client code and dynamic web pages. We support the main Relational Databases as well as the following architectures: Microsoft COM+, J2EE and .NET (in development). These are implemented using Java, Macromedia Cold Fusion, Microsoft Visual Basic and C# (in development).

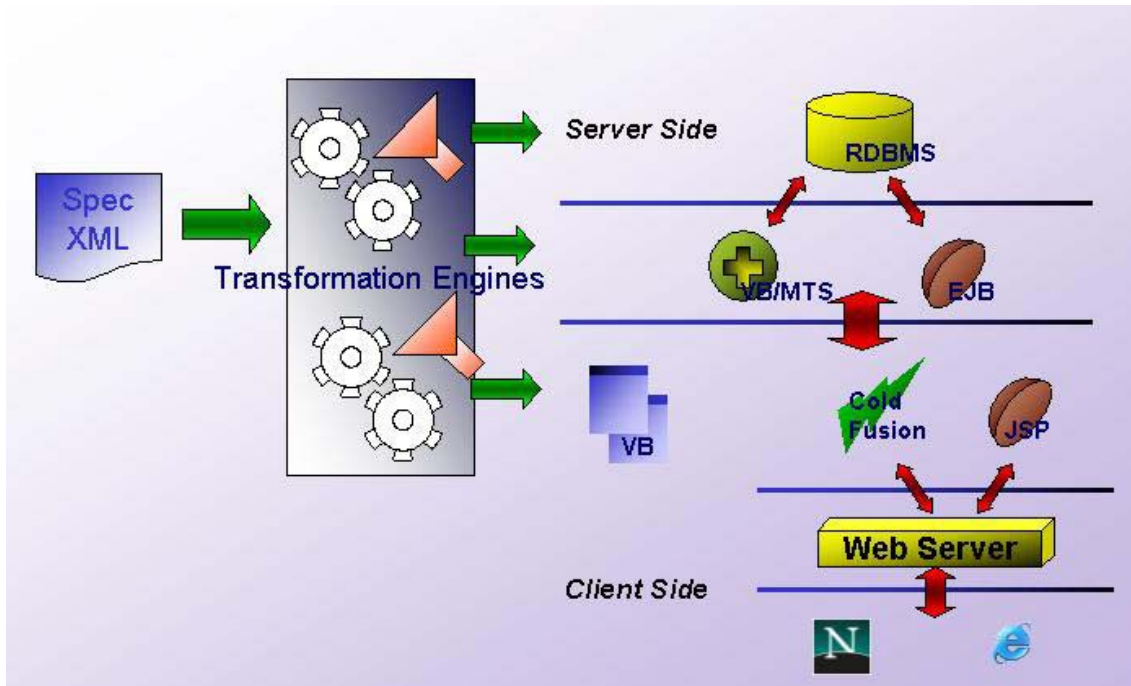


Figure 1 N-Tiers support by OlivaNova Model Execution Technology

4.1 The Persistence Tier

The **OlivaNova Server Transformation Engines** generates the Data Definition Language (DDL) scripts for creating the relational database that the modeled application requires. The created databases follow well-known object-relational mappings and also imposes constraints that were captured in the model.

Database specific scripts for the main Relational Databases such as Oracle, MS SQL Server and DB2 are provided.

4.2 The Middle Tier

For Windows-based systems, **OlivaNova Server Transformation Engines** generate complete Visual Basic projects containing COM+ components. The components implement and expose an Application Programming Interface (API) with all of the functionality required by the client and as it was expressed in the model. These components can be executed via DCOM or through Web Services calls.

For systems with Java™ and J2EE support, **OlivaNova Server Transformation Engines** generates the Java source code, XML descriptors (standard and specific for the most popular application servers) and scripts to build all of the EJB modules. Those can then be assembled in an enterprise application that can include one or several web clients. Components can be invoked through Java Remote Method Invocation calls or using a HTTP/XML based protocol.

The functionality and API are equivalent for EJB and COM+ generated components as both implement the same model specification. It is possible to exchange the middle tier without affecting the persistence or the client tier.

4.3 The Client Tier

OlivaNova Client Transformation Engines can generate rich interactive desktop clients. The artifacts produced are full Microsoft Visual Basic (VB) projects which are then compiled to obtain the clients. The code is based on ActiveX components that allow easy customization. An important characteristic is that clients can operate transparently with both COM+ or EJB server generated components.

For light web clients, **OlivaNova Client Transformation Engines** generate Cold Fusion and JSP pages. Here, the interaction is more limited than in the VB client due to Dynamic HTML limitations, but in general both clients allow the same functionality. As with VB, the Cold Fusion client can work transparently with COM+ or EJB server generated components. The JSP client was designed to be the natural client part of EJB components in a J2EE application, and uses Java RMI to interact with them. The generated artifacts include scripts to build the full J2EE application.

5 Conclusions

The software industry has reached a stage where it is necessary to think about a Model Driven Architecture. The specification is incorporated in a conceptual model that is non-implementation specific, thereby isolating your model from the technology. Automatic generation tools then allow you to quickly generate implementations for different platforms and architectures. **OlivaNova Model Execution products** are a clear cut example. The **OlivaNova Modeling Software** enables capturing application semantics and the **OlivaNova Transformation Engines** generates the implementation for current or legacy technologies. Any new technology can be utilised by the creation of a corresponding transformation engine. These will then ease the migration or enhancement of existing applications. An example of this is the current development of a .Net and PDA client transformation engine that would give the developer access to these technologies.

6 References

- [1] Carnegie Mellon - Software Engineering Institute. <http://www.sei.cmu.edu/str/descriptions/clientserver.html>
- [2] CORBA Common Object Request Broker Architecture. <http://www.corba.org/>
- [3] Java™ 2 Enterprise Edition. <http://java.sun.com/j2ee/>
- [4] Java Server Pages. <http://java.sun.com/products/jsp/>
- [5] Macromedia Cold Fusion. <http://www.macromedia.com/software/coldfusion/>
- [6] Microsoft Active Server Pages. <http://msdn.microsoft.com/asp>
- [7] Microsoft COM+. <http://www.microsoft.com/com/tech/COMPlus.asp>
- [8] ODMG Object Data Management Group. <http://www.odmg.org/>
- [9] OlivaNova Model Execution. <http://www.care-t.com/>
- [10] OMG Object Management Group. <http://www.omg.org/>
- [11] OMG Model Driven Architecture. <http://www.omg.org/mda/>
- [12] PHP: Hypertext Pre-processor. <http://www.php.net/>
- [13] UML Unified Modeling Language. <http://www.uml.org/>
- [14] XML Extensible Mark-up Language. <http://www.w3c.org/XML/>

CONTACT INFO

Spain:

Mr. Emilio Iborra
CEO
CARE Technologies S.A.
care-technologies@care-t.es
Tel. +34 96 643 5555
Fax. +34 96 643 5554



USA:

Mr. Chris Lema
VP Technology
Sosy Inc.
180 Montgomery Street
Suite 828
San Francisco, CA 94104
information@sosyinc.com
Tel. +1 415 362 4333
Fax. +1 415 362 4703

Germany:

Joachim Fischer
Prokurist
CARE Technologies GmbH
Wagmüllerstrasse 18-20
80538 München
info@care-t.de
www.care-t.de
Tel. +49 89 21923774
Fax. +49 89 21923756

CARE Technologies white papers can be downloaded at <http://www.care-t.com>.