Developing eBusiness solutions with a Model Driven Approach

Piero Fraternali, Stefano Ceri, Massimo Tisi
Politecnico di Milano and Web Models s.r.l.
Piazza Leonardo da Vinci, 32
Milano, Italy
ceri/fraterna/tisi@elet.polimi.it

Emanuele Tosetti
Acer Europe Services s.r.l. (Acer Inc Emea Quarter)
Via Lepetit, 40
20020 Lainate (MI), Italy
emanuele_tosetti@acer-euro.com

Abstract

This paper addresses the problem of developing enterprise-class eBusiness solutions in a more economically viable and time-effective way, by adopting Model Driven Development (MDD), a novel paradigm for the construction of complex software systems based on the high-level modeling of application requirements and on the automatic generation of code. Specifically, we report on an experience of more than five years of collaboration between Acer Inc. (the 4th branded PC vendor worldwide) and Web Models, an Italian startup company and spinoff of Politecnico di Milano, innovator in the market of software tools and methodologies for MDD. The research results clearly demonstrate that MDD can shorten the time to market of complex eBusiness solutions deployed on the web, improve the quality and conformance to requirements of the developed systems, and increase the economic profitability of solutions, by lowering the total cost of ownership and extending the life span of systems. The original contributions of the paper are both technical and organizational. In the former area, an in-depth evaluation of the MDD approach is conducted by applying it to the construction of a wide range of complex web systems, using innovative modeling and code generation techniques; in the latter area, MDD is shown to have the potential of alleviating the gap between the business and IT units, by fostering a better partition of responsibilities. Finally, the economic impact of MDD on software development and management is assessed quantitatively.

Keywords: competitive advantage, model driven development, eBusiness solution deployment, synergies in customer-supplier relationships

Introduction

Highly competitive markets demand fast adaptation of business to changing environmental conditions, an objective made possible by the deployment of core business processes over added-value networks integrating all the actors of a company's ecosystem.

The advent of the Web as a universal platform has initially facilitated the shift towards enterprise integrated applications, by offering a standard means of distributing data and functions. However, the unprecedented speed of evolution typical of the Web and the fierce competition on technologies among the major ICT players challenges the long-term sustainability of IT projects, due to a number of cross-cutting complexity factors:

1. The spectrum of relevant standards and architectures constantly increases, often not guaranteeing interoperability and backward compatibility, which poses IT investments at risk.
2. The learning curve of technologies is slower than their evolution rate, which induces a chronic shortage of skilled people and consequently augments time-to-market.
3. Outsourced or distributed development challenges classical software project management techniques, mixing geographical, human and technology factors.

A solution to the growing complexity of IT projects requires a thorough innovation of the approach to software development, as advocated by the modern research on software engineering, which proposes Model Driven Development (MDD) as a means of improving the governance and end-to-end productivity of software (Warmer 2003). In essence, MDD promotes a novel approach to software development centered around the notions of: *platform independent model* (*PIM*), which is a representation of the system's functionality independent of any specific technological detail, and *model transformation*, which is the process of progressively refining high-level models into lower-level ones, until a model is produced that is executable on a concrete technological platform.

This paper elaborates on a five-years experience of applying MDD to a set of enterprise-scale applications, developed by the EMEA branch of Acer Inc. using WebRatio, an innovative MDD methodology and tool suite built by Web Models, an Italian startup.

The milestones of the collaboration between WebRatio and Acer can be summarized as follows:

1. 2001-2002: The introduction of Model Driven Development methods and tools in the company, as a means of mastering the deep reorganization of the marketing and communication function imposed by a major transition in the company's business model.
2. 2003-2004: The consolidation of the MDD approach as a key success factor in the deployment of mission-critical applications, which led to its diffusion outside the marketing and communication area, e.g., to the distribution channel management, sales, and financial services sectors.
3. 2005-today: The integration of the MDD approach within the design of a global Service Oriented Architecture, with the aim of managing large-scale projects, involving not only the internal business units and systems of Acer, but also the information systems of its distributors and partners.

The contribution of the paper is original under several respects: technological, organizational, and economic.

From the **technical standpoint**, MDD is an innovative paradigm, which is receiving a strong impulse by the Object Management Group (OMG), promoter of the Model Driven Architecture (MDA) (OMG 2006), and by leading IT vendors like IBM, Microsoft and Oracle. However, very limited experience is available in the literature on the practical application of model-driven development in long-term projects, on its adequacy to the development of Web applications, and on the acceptance and profitability of this approach compared to traditional development techniques. In this paper, we discuss the lessons learned in transferring MDD to traditional Web developers, in training them about application modeling and code generation, in measuring their productivity, and in evaluating the quality and conformance to requirements of the produced systems.

From the **organizational viewpoint**, the reported experience shows that MDD has the potential of alleviating the well-known gap between the business units, whose mission is to deploy as quickly as possible solutions to the customers, and the IT department, whose core responsibility is governing the harmonic evolution of the corporate IT infrastructure and application portfolio. We anticipate that the development of mission-critical solutions with MDD has been driven and realized by the commercial business units, and not by the IT department, thanks to the simplification of the development process granted by the model-driven approach; however, this assignment of responsibilities has not produced a fracture in the governance of the corporate IT infrastructure, thanks to the adoption of a Service Oriented Architecture, which permits the seamless integration of autonomously developed software systems, packaged as reusable services, into a coherent technological framework.

From the **economic viewpoint**, the research results clearly demonstrate that building B2C and B2B applications with MDD outperforms traditional development under a number of dimensions: development effort and time to market, total cost of ownership (including corrective ad evolutive maintenance), return of the investment, and developer's productivity. Formal application size estimation procedures have been applied to measure the quantity of software developed and to compare the productivity of model-driven development and traditional development with conventional programming languages, for which international productivity rates are available.

The paper is organized as follows: the first section "Business process adaptation as a key competitive factor" introduces the addressed problem: supporting the business strategies of ACER EMEA with the design, implementation and rapid deployment of Web solutions for accelerating the enterprise workflows and optimizing the interaction with channel operators and customers. The second section "The WebML methodology for Model-Driven Development" describes the principles of MDD and illustrates an original incarnation of this paradigm: the Web Modeling Language (WebML) notation for visually specifying application requirements, and the WebRatio development environment, for editing high-level models and transforming them into the complete implementation code for the Java 2 Enterprise Edition architecture. The third section "Case Studies" considers a subset of the systems developed by Acer EMEA and illustrates their business requirements and the main technical and economical dimensions of their development, usage and evolution. The fourth section "Results and critical considerations" examines the conducted experience and highlights the fundamental lessons learned on the technical viability of MDD, on its impact on the organization and division of responsibilities, and on its economical performance with respect to traditional software development. The fifth section "Related work" positions the described research with respect to the relevant literature. The final section "Conclusions and future work" draws the general conclusions of the work described in the paper and points out the directions of the ongoing and future work.

Business process adaptation as a key competitive factor

Acer is a multinational company, with sales and operations spread over five continents. As many other organizations competing on a global scale the company had to face the web revolution and the globalization challenge. On December 2000, Acer Inc. launched a worldwide reorganization effort that started the separation of the Acer-brand side of the business from the manufacturing side, with a threefold goal: transitioning from a manufacturing company to a marketing and service business; reacting to the changing nature of the global economy, moving from the IT era to the era of the knowledge-based economy; and moving from a technological innovation focus to a user-centric focus.
This transformation had a strong impact on the business processes and market positioning, which triggered a thorough renovation in all sectors and locations of the company. In the European region, Acer EMEA, operating in Europe, Middle East and South Africa, was assigned the objective of providing centralized support to all the national subsidiaries for a number of functions, including product management, communication and channel marketing, information technology, and human resource management. Prior to September 1999, every national subsidiary was autonomous and had its own policies. Local operations were built upon a set of heterogeneous systems (over 30 of them only in the marketing and sales area), autonomously built and managed. Such a situation caused both commercial and managerial difficulties, due to duplicate efforts and misalignment of customer service and communication. To streamline EMEA operations and to comply with the new global business model and corporate identity, Acer launched a large-scale project aimed at integrating the product management, communication and channel marketing functions of the EMEA national subsidiaries.
To support the integration, it was decided that a novel system, code-named *Acer-Euro*, should replace all the existing local applications (Figure 1). Due to the geographical distribution of the involved subsidiaries and the heterogeneity of the IT infrastructures and systems at each national site, it was established to use the Internet as a connectivity network and the Web as a uniform deployment architecture. This choice was supported by a number of technical and business motivations:

- Centralized IT infrastructure: applications could be installed and maintained in one single location, and data could be accessed everywhere by leveraging standard Internet connectivity.
- Flexible mix of centralized and distributed control: content administration could be managed flexibly, by granting content management rights to product and marketing managers of the various countries in a controlled manner (e.g., over a corporate Virtual Private Network).
- Ease of deployment: with browser-based access, new application releases could be easily deployed centrally, without the need of complex client-side installation procedures at each subsidiary.

- Uniform technology for B2E, B2C and B2B solutions: the Web could be exploited as a uniform platform for B2C applications (e.g., online catalogs), extranet applications (e.g., channel marketing solutions), and intranet applications (e.g., content and workflow management systems).

The Acer-Euro project had an impressive time framework, due to the need to synchronize with the worldwide marketing deadlines of the group: only 7 weeks were allocated to produce the new system reflecting the novel corporate identity and business process specification. Such constraint required a completely new approach to system development, conjugating an agile software development process, iterative prototyping, and massive code generation. After a thorough market review, Acer decided to invest in innovation and partnered with Politecnico di Milano, the largest Italian ICT University, which was in the process of spinning-off a company, called Web Models, in the Web development sector, exploiting the results of an internationally patented research on Model Driven Development. As the outcome of such partnership, the Acer-Euro (http://www.acer-euro.com) application went online as scheduled and within budget: it initially consisted of 14 multi-lingual B2C web sites publishing product catalogs, corporate information, news, and service information. The Web front-end applications were backed by a complex network of integrated content management systems, targeted to multiple organization profiles (product managers, marketing managers, country managers, text editors and translators) and completely supporting the content management workflow across the EMEA countries.

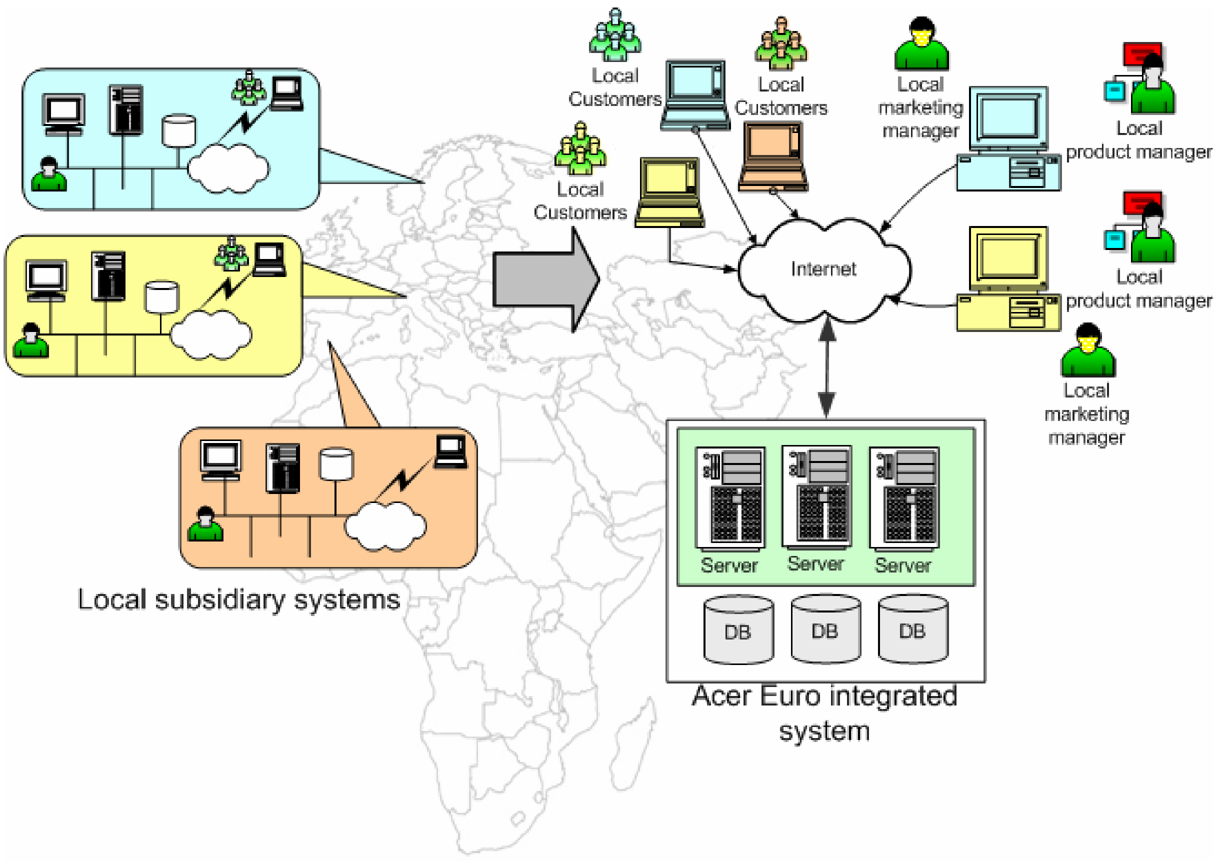


Figure 1: The transition from the local subsidiary systems to the Acer-Euro integrated system

The successful completion of the first version of the Acer-Euro application was only a first milestone of a far-reaching process, which led to a new process for solution development and maintenance. The goals of such reorganization were manifold:

- Accelerating the transition to the user-centric approach requested by new corporate business model, by directly involving the relevant business units in the development of customer solutions.
- Reducing training costs, limiting resource and skill duplication, and preserving the quality of the developed software, while delegating part of the development to the business units.
- Granting the rapid prototyping and quick deployment of new solutions, with the possibility of adjusting the business model during the usage of the applications, without interrupting the customer service.

- Reducing the initial investment and total cost of ownership of new business solutions, improving the return on investment.
- Maximising the reuse of solutions across multiple countries, allowing both global uniformity and local customizations.

To achieve these goals during a period of organizational transition, the top management of Acer EMEA decided to take the risk of radically innovating the approach to application development. The decision was taken to assign the responsibility of mission-critical customer solutions to the marketing and communication unit, and to focus the IT department on the redevelopment of the core administrative systems (financials, billing and accounting) requested by the new corporate governance scheme and legal setting.

Model Driven Development seemed the most adequate methodology for bringing software development to non-IT business units. The idea was to exploit the knowledge about the business processes of the marketing and communication personnel and to promote the formal modelling of business solutions by domain experts, delegating as much as possible of the construction of the implementation code to suitable development tools. In the rest of the paper, we elaborate on the adopted model driven development methodology and on the achieved results.

The WebML methodology for Model-Driven Development

Developing complex, enterprise-scale Web application requires the ability to master a broad spectrum of tasks, jointly performed by a number of persons with different skills. Model Driven Development advocates a well-organized development process, centered on the appropriate high-level concepts, which can be used to describe the features of a software solution using “conceptual models”, i.e., descriptions independent of the technological details of the deployment platform. Platform-independent models can then be progressively transformed, by humans and/or by software tools, into more and more concrete and platform-specific models, until one model is produced that is directly executable on a well-defined technological architecture.

Inputs and Outputs. Figure 2 recalls the fundamental inputs and outputs of the software development process. The most important input is the set of *business requirements* that drive application development. These requirements are mostly non-technical and express the essential goals of building the application, by stating the value that the application is expected to produce to its users and to the organization who builds it. Business requirements identify such aspects as the business actors (human beings or organization’s functions) taking advantage of the application, the processes affected by the application, the boundaries between the application and the pre-existing systems, and the quality that the application must ensure to its users, such as the quality of content, services and interfaces, response times, availability, security, privacy, and so on.

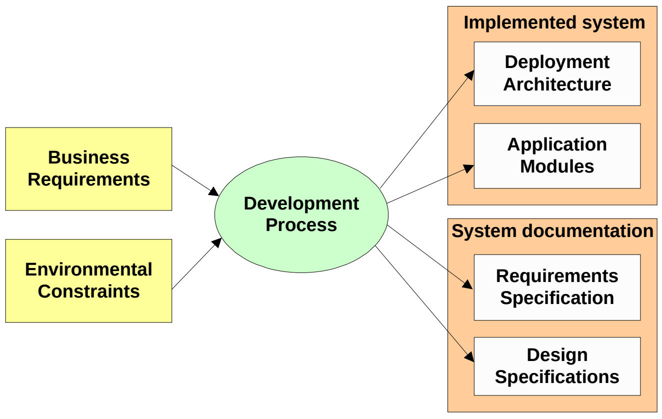


Figure 2: Inputs and outputs of the development process

The second input is the set of *environmental constraints* that affect the construction of the application, which include architectural restrictions, compatibility with existing systems and applications, available technical skills, time and development resources limitations. The deployed application is the result of a careful trade-off between the business requirements and the environmental constraints.

The output of the process is the implemented system, consisting of the deployment architecture, of the application software modules installed on this architecture, and of the system documentation:

Development Roles. The MDD process involves actors with different skills and goals:

- The ***application analyst*** collects the business requirements and turns them into a specification of the application requirements, in terms of served processes and expected functionality.
- The ***data expert*** focuses on the part of the application requirements that deals with the data, and produces the conceptual data model.
- The ***application architect*** focuses on the part of the application requirements dealing with the services to be delivered, and designs the application user's interfaces.
- The ***graphic designer*** conceives the presentation styles, based on the business requirements that deal with the organization's visual identity and communication standards.
- The ***developer and site administrator*** is responsible of implementation, architecture design and tuning, deployment and evolution. In particular, he focuses on the non-functional requirements about performance, availability, security, scalability, and manageability, and is responsible of ensuring the appropriate level of service.

MDD fosters the collaboration between the business expert and the application analyst and the data expert, by exploiting conceptual notations for describing data and processes that can be shared by technical and business professionals.

Development Lifecycle. The MDD process, illustrated in Figure 3, is an iterative and incremental procedure, in which the various phases are repeated and refined until results meet the business requirements.

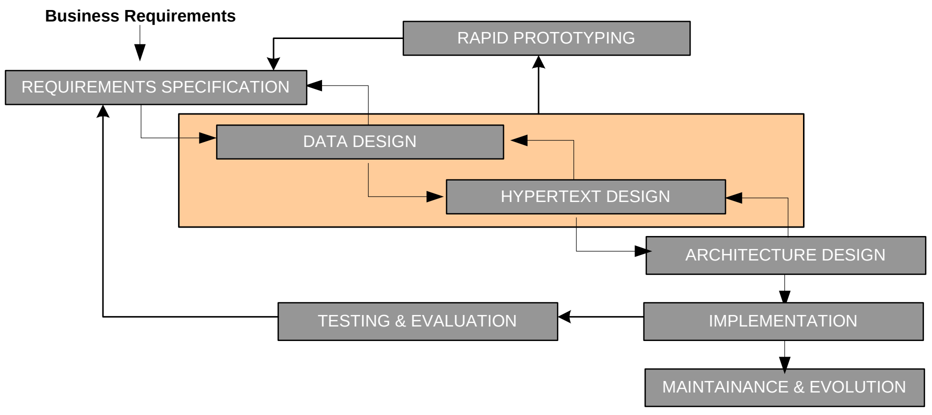


Figure 3: Phases in the model-driven development process

The "upper" tasks of design are those most influenced by the adoption of a conceptual model. In particular, the process of Figure 3 highlights *data design*, in which the fundamental business objects of the application are identified and their relationships are established, and *hypertext design*, a task typical of web application development, in which the hypertext front end of the application is designed. MDD promotes rapid prototyping, that is, the quick transformation of design documents into application blueprints, which can be demonstrated to the business experts to get feedback and detect misunderstandings early in the developments process.
The "lower" phases of the development process are closer to the physical aspects and concern the delivery and evolution of the application.

The Web Modeling language. WebML (Web Modeling Language – http://www.webml.org) is a conceptual model for Web applications conceived at Politecnico di Milano. It is one ingredient of a broader Web development methodology, described in the book "Designing Data Intensive Web Applications" (Ceri et al 2003).
In WebML , the specification of a Web application covers three main perspectives:

- The *content*, that is the information assets managed by the application, their properties and logical correlations.
- The *hypertext*, that is the hyper textual interfaces of the application, consisting of pages for publishing content and of links for navigating and performing actions.

- The *presentation*, that is the graphical rendition of the interface.

WebML adopts a visual approach for specifying the organization of hypertexts, exploiting "native" Web concepts like sites, areas, pages and links, easily understandable also by business and communication experts. However, WebML constructs have a precise meaning (technically speaking, a formal semantics), so that WebML diagrams can be automatically checked for correctness and transformed into application code.

To exemplify WebML, in the sequel we will draw examples from the *Acer-Euro* application.

Content modelling. Typically, a Web application offers an interface for the publication or management of some content. Therefore, design typically starts from describing the structure of the managed content. For this, WebML relies on existing standards like Entity-Relationship (Chen 1976) and UML Class Diagrams (Booch, Jacobson and Rumbaugh 1998), well known to database and object-oriented developers, but also easily mastered by domain experts. For the sake of simplicity, in the sequel we use the Entity-Relationship model. Figure 4 shows a schema representing, with some simplifications, a subset of the Acer-Euro content. The application manages data about *Products*, clustered in *ProductBrands* and *ProductGroups*; products are described by several components (*Logo*, *Award*, *Benefit*, *TechSpec*, and *Configuration*). Other information assets are the *NewsItem* entity, which denotes pieces of news, classified by category (*NewsCategory*), and *LocalLink*, describing classified links to local resources. A piece of news may refer to a product or to an award won by a product, as expressed by the relationships connecting the involved concepts. Finally, Acer-Euro is a multi-country application; therefore, content must be published in different languages. This is represented by the entity *Country* and its relationships to the *NewsCategory, LocalLink*, and *ProductGroup* entities; in this way, selecting a country entails selecting news, suggested links, and product data written in the language suited to that country.

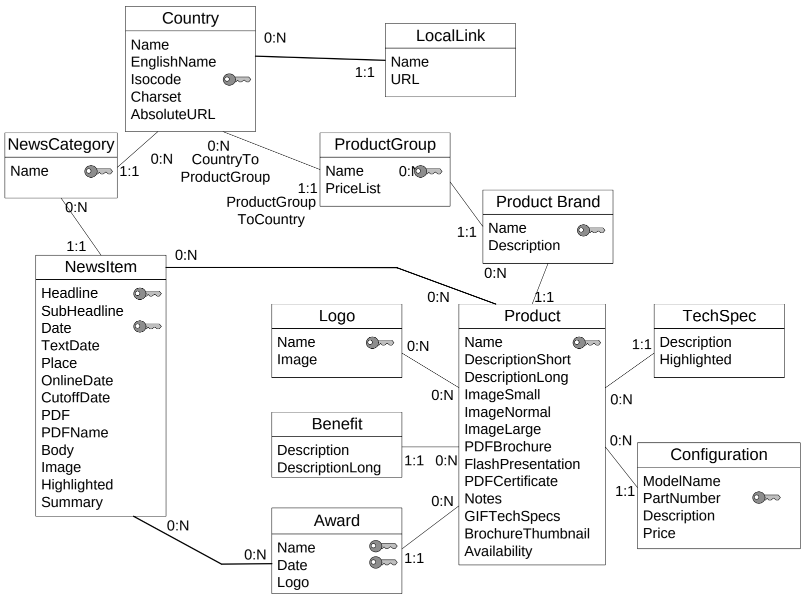


Figure 4: Data model of the Acer-Euro application

Once the properties of the information objects and their relationships are modelled, the hypertexts for publishing and managing content can be specified. WebML provides different constructs that can be combined for describing the hierarchical structure of the hypertext, its pages and their content, and the links for navigating and performing actions.

Hypertext modelling in the large: site views, areas and pages. The general structure of a hypertext is described in terms of *site views*, *areas* and *pages*. A *site view* is a specific hypertext, designed for a particular class of users. For example, Acer-Euro comprises one public site view for the Internet customers and a number of private site views, targeted to different Acer functions, like product managers, marketing managers, administrators, and so on. Site views may exhibit an internal organization, represented with the concept of *area*, defined as a module containing pages or sub-areas. Figure 5 shows the navigation bar of the Acer-Euro public site view, with links pointing to the different areas (*About Acer*, *Products*, *News*, *Service&Support*, *Partners*, *Where to Buy*).

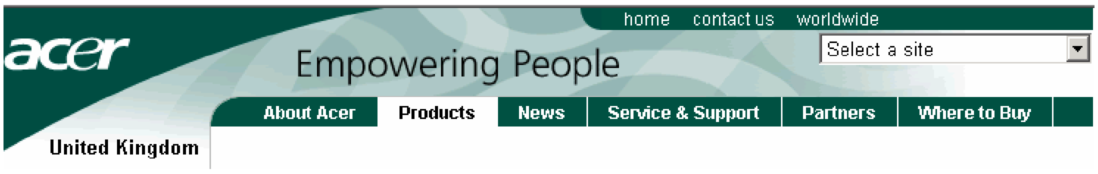


Figure 5: Areas in the Acer-Euro public Web site

The central concept of hypertext specification is the *page*, defined as a container of data and navigational commands. A WebML page is an abstraction of an HTML page or of a page template implemented in JSP, PHP or ASP.NET. Figure 6 shows the WebML notation for representing site views, areas and pages. They are labelled containers, appropriately nested to represent the hierarchical structure of the site. The figure describes the public site view of Acer-Euro: it contains three top-level pages (*Home, Contact us, Worldwide*), and the six areas already mentioned.

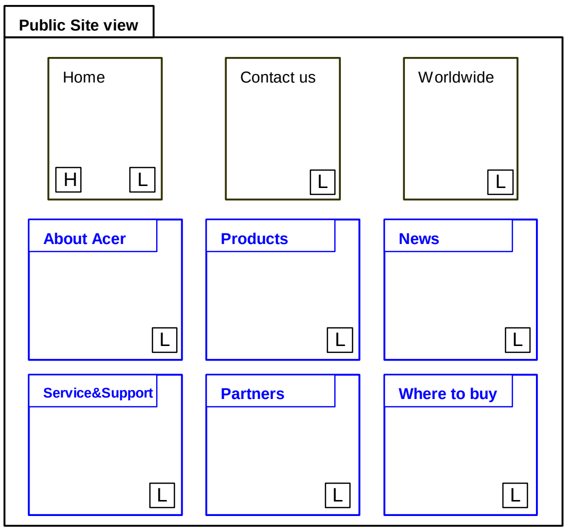
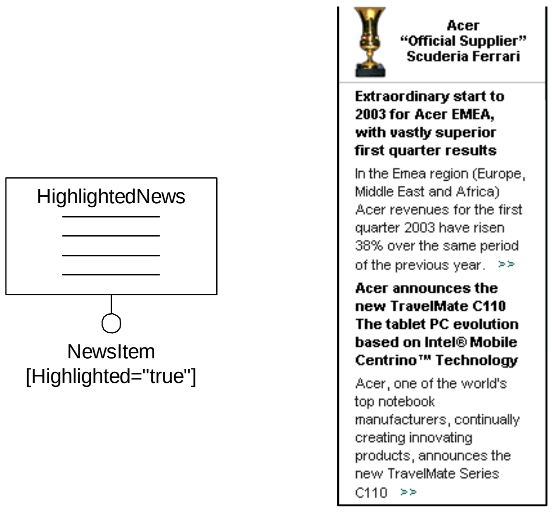


Figure 6: WebML notation for site views, areas and pages

Pages and areas may have special markers, denoting visibility properties:

- One page or area can be marked as *Home* (H) to denote that it is presented by default when the site view is accessed.
- Pages and areas can be marked as *Landmark* (L) to denote that they are “global” and can be accessed from any other page or area of the site. Landmark pages and areas contribute the links that appear in the global navigational bar of the Web site (see Figure 5).

Hypertext modelling in the small: units and links. Once the general organization of the site view is established, the content of each page and the navigational commands must be specified. For this purpose, WebML provides the notions of *content unit* and *link*.
A *content unit* is a component for the publication of information inside a page. Typically, the content of a unit is retrieved from the database, whose schema is expressed by the Entity-Relationship model. The binding between the data schema and the hypertext is represented by the *source entity* and the *selector* of the content unit. The source entity specifies the type of objects published by the content unit, by referencing an entity of the Entity-Relationship diagram; the selector is a filter condition over the instances of the source entity, which determines the actual objects published by the unit. WebML offers six predefined units (data, index, multidata, scroller, multichoice index, and hierarchical index), and allows designers to define their own custom units. Figure 7 shows an index unit, called *HighlightedNews*, from the Home page of the Acer-Euro national Web sites. It is defined over the source entity *NewsItem* and includes the selector condition [*Highlighted* = *“true”*]. The rendition of the unit, shown on the right, displays the news items whose *Highlighted* attribute is true.



Acer
"Official Supplier"
Scuderia Ferrari

Extraordinary start to 2003 for Acer EMEA, with vastly superior first quarter results

In the Emea region (Europe, Middle East and Africa) Acer revenues for the first quarter 2003 have risen 38% over the same period of the previous year. >>

Acer announces the new TravelMate C110 The tablet PC evolution based on Intel® Mobile Centrino™ Technology

Acer, one of the world's top notebook manufacturers, continually creating innovating products, announces the new TravelMate Series C110 >>

Figure 7: A content unit from the Acer-Euro Home page (WebML specification and rendition)

To compute its content, a unit may require the "cooperation" of other units and the interaction of the user. For example, a unit publishing the full text of some piece of news requires in input the identifier of the specific news item to display. This information can be provided by an index unit, publishing a list of news titles, from which the user may choose one. Therefore, each WebML unit exposes an input interface, specifying the parameters that it can accept, and an output interface, defining the parameters that the unit can provide to other units. Making two units interact requires connecting them with a *link*, represented as an oriented arc between a source and a destination unit. The aim of a link is twofold: permitting navigation (possibly displaying a new page, if the destination unit is placed in a different page) and enabling the parameter passing from the source to the destination unit. Figure 8 shows the WebML specification of the Home page of the Acer-Euro national Web sites, which exemplifies several units connected by links.

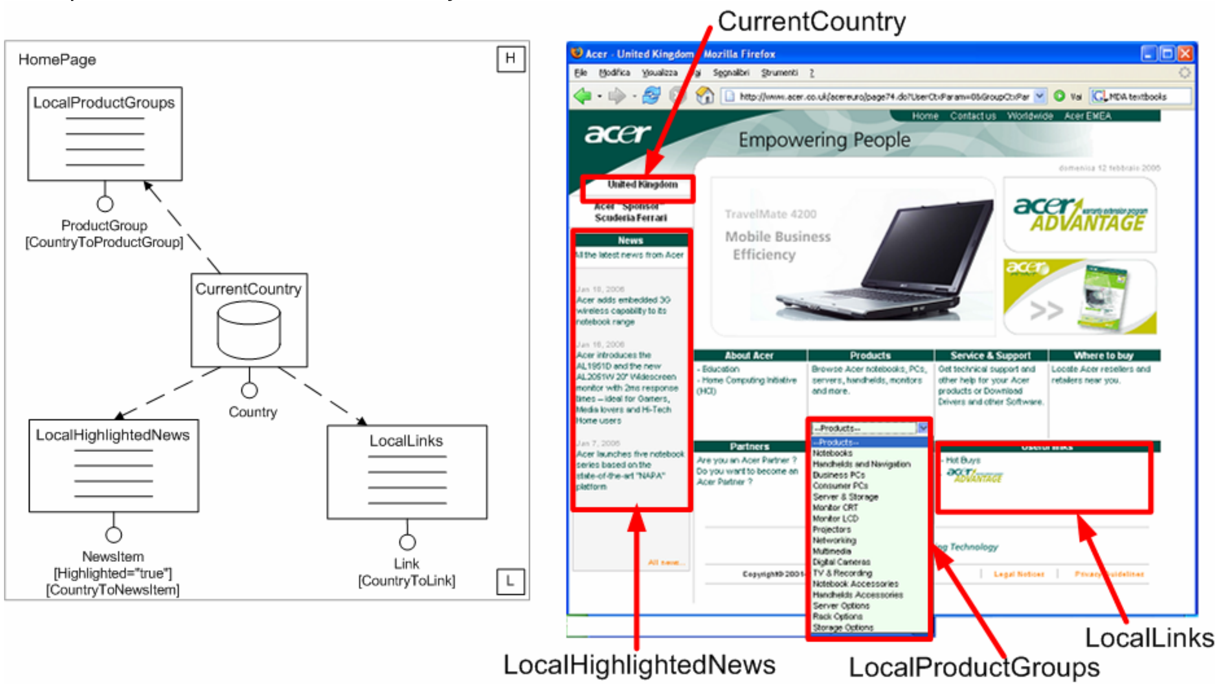


Figure 8: The WebML specification of the Home page of a national site of Acer-Euro (left) and its rendition in HTML (right)

At the centre of the page, the data unit *CurrentCountry* displays the name of the country to which the national Web site refers. The page contains three further indexes: *LocalProductGroups*, *LocalLinks*, and *LocalHighlightedNews*, the content of which depends on the currently selected country. To express such a dependency, these index units have an incoming link from the *CurrentCountry* data unit, which provides the identifier of the selected country, and a suitable selector condition. For example, unit *LocalProductGroups* includes the selector *[CountryToProductGroup]*, which refers to the relationship connecting entity *Country* to entity *ProductGroup* (see Figure 4). The meaning of such a selector is that only the product groups connected to the country passed in input to the index unit are displayed. In a similar manner, all the content units inside Acer-Euro pages have localized content.

Modelling data entry and operations. An important aspect of Web applications is the capability of accepting input from the user, for example to perform keyword-based searches or to create new information objects. WebML provides a unit representing entry forms for inputting data: the *entry unit*. An entry unit contains a set of *fields* whereby the user can input different types of data (texts, numbers, images, files, etc.). WebML also models the execution of arbitrary business logic, by means of *operation units*. An operation unit is a component similar to a *content unit* that, instead of publishing content, describes the execution of an action. An operation unit is placed outside the pages and can be linked to other operations or content units. WebML incorporates some predefined operations for creating, modifying and deleting the instances of entities and relationships, and allows developers to extend this set with their own operations.

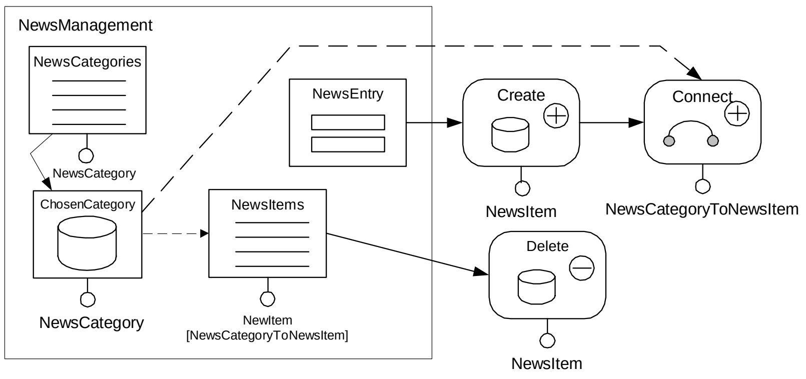
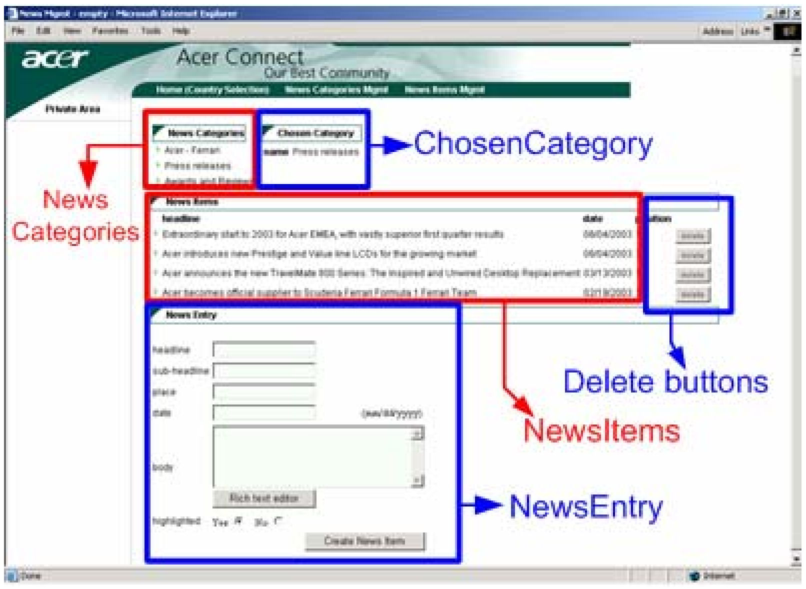


Figure 9: The WebML specification of a page for creating and deleting news

As an example, Figure 9 shows the specification of a page of the Acer-Euro site for marketing managers. The *NewsCategories* index unit displays the set of available categories and lets the user select one category, which is displayed in the *ChosenCategory* data unit. The *NewsItems* index unit shows the set of news items of the current category and its outgoing link allows the user to activate a delete operation, which eliminates the chosen piece of news. The page also comprises an entry unit (*NewsEntry*), whereby the user can enter a piece of news. The outgoing link of the entry unit is rendered as the submit button of an input form (see Figure 10) and triggers the sequence of two operations: a news item is created (operation unit *Create*) and connected to the chosen category (operation unit *Connect*).



News Categories
- Acer - Ferrari
- Press releases
- Awards and Reviews

Chosen Category

name Press releases

News Items

headline	date	action
Extraordinary start to 2003 for Acer EMEA, with vastly superior first quarter results	08/04/2003	delete
Acer introduces new Prestige and Value line LCDs for the growing market	08/04/2003	delete
Acer announces the new TravelMate 800 Series. The inspired and Unwired Desktop Replacement	03/13/2003	delete
Acer becomes official supplier to Scuderia Ferrari Formula 1 Ferrari Team	02/19/2003	delete

News Entry

headline
sub-headline
place
date (mm/dd/yyyy)
body
highlighted Yes [✓] No

Figure 10: The rendition of the NewsManagement page

The WebRatio software development environment

The use of WebML is supported by a tool called *WebRatio* (http://www.webratio.com), which assists the production of the conceptual models and automates the generation of the application code. More precisely, WebRatio focuses on five main aspects:

- *Data design:* it supports the design of data schemas, providing a graphical user interface for drawing and specifying the properties of entities, relationships, attributes, and hierarchies.
- *Hypertext design:* it assists the design of site views, providing functions for drawing and editing the properties of areas, pages, units, and links.
- *Data Mapping:* it permits declaring the data sources to which the conceptual data schema has to be mapped to, and automatically translating Entity-Relationship diagrams into relational databases, and vice versa.
- *Presentation design:* it allows the graphic designer to associate presentation style sheets to pages, and organize layout, by arranging the relative position of content units in the page.
- *Code generation:* it automatically translates site views into running Web applications built on top of the Java2EE platform, relational databases, and Web services.

Figure 11 shows the architecture of WebRatio, comprising an editor for designing Web applications using WebML, shown in Figure 12, a set of customizable code generators, and extensible runtime support libraries.

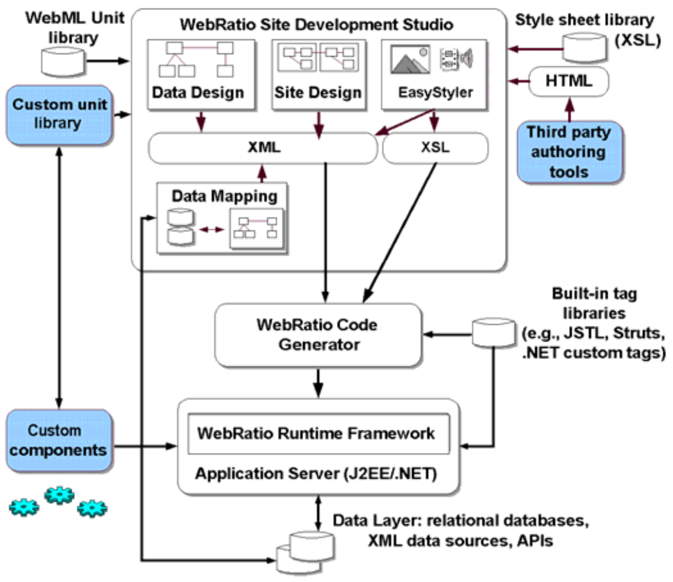
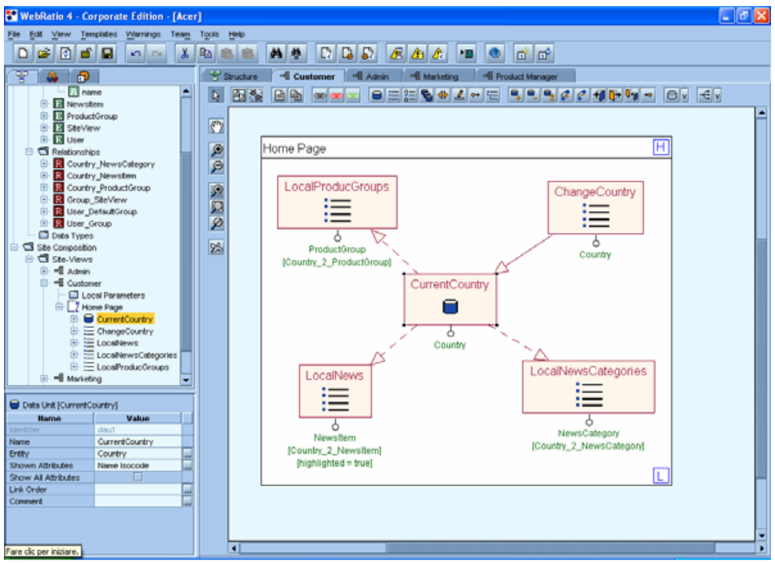

WebML Unit library
WebRatio Site Development Studio
Style sheet library (XSL)
Data Design
Site Design
EasyStyler
HTML
Custom unit library
XML
XSL
Third party authoring tools
Data Mapping
Built-in tag libraries (e.g., JSTL, Struts, .NET custom tags)
WebRatio Code Generator
Custom components
WebRatio Runtime Framework
Application Server (J2EE/.NET)
Data Layer: relational databases, XML data sources, APIs

Figure 11: Architecture of WebRatio


WebRatio 4 - Corporate Edition - [Acer]
File Edit View Templates Warnings Team Tools Help
Structure
Customer
Admin
Marketing
Product Manager
Home Page
LocalProducGroups
ChangeCountry
CurrentCountry
LocalNews
LocalNewsCategories
ProductGroup [Country_2_ProductGroup]
Country
NewsItem [Country_2_NewsItem] [highlighted = true]
NewsCategory [Country_2_NewsCategory]
Fare clic per iniziare.

Data Unit [CurrentCountry]

Name	Value
Identifier	dau1
Name	CurrentCountry
Entity	Country
Shown Attributes	Name Isocode
Show All Attributes	
Link Order	
Comment	

Figure 12: Graphical User Interface of WebRatio

Case Studies

In this Section we overview the most significant realized projects, highlighting their functional and non-functional requirements, their dimensional parameters, and the key aspects of their development, deployment, evolution, and economic evaluation.

Acer-Euro and Acer Connect

The first version of the Acer-Euro application (called Acer-Euro 1.0) aimed at establishing a software infrastructure for managing and Web-deploying the marketing and communication content of an initial group of 14 countries out of the 31 European Acer national subsidiaries. The content of Acer-Euro 1.0 included the following main areas:

1. About Acer
2. Products
3. News
4. Service & Support
5. Partner Area
6. Where to buy

The Acer-Euro 1.0 system supported two main functions:

1. CONTENT PUBLISHING: comprising the architecture, tools and processes to make content about the Acer European web sites available on the Web to the users of the target countries.
2. CONTENT MANAGEMENT: comprising the architecture, tools and processes needed to gather, store, update and distribute to the destination countries the content related to the Acer European web sites.

Acer Euro 1.0 supported a seven-steps distributed workflow, illustrated in Figure 13, involving Local and Central Product Managers (LPMs and CPMs), Central Marketing Managers (CMMs), the central IT department, and the Internet Service Providers (ISPs) of the affected countries. Content updates were collected daily at the national subsidiaries and forwarded to the Central Product Manager's office, in charge of their insertion into the Content Management System. Twice per week, content updates were subject to a formal approval process and consolidated into a new publishable "master version". Static versions of the national web sites (called "local copies") were then produced from the master copy and uploaded into the hosts of the national Internet Service Providers for publishing. In this way, Acer could completely renovate the content and workflow of the marketing and communication functions, while reusing the existing national Internet infrastructures and service contracts. Version 2.0 of Acer-Euro was deployed after the installation of a new high-bandwidth corporate Virtual Private Network, which substituted the local ISPs and further streamlined the workflow, by eliminating the need of centralized content update insertions (Phase 2) and local copies creation (Phase 5 and 6).

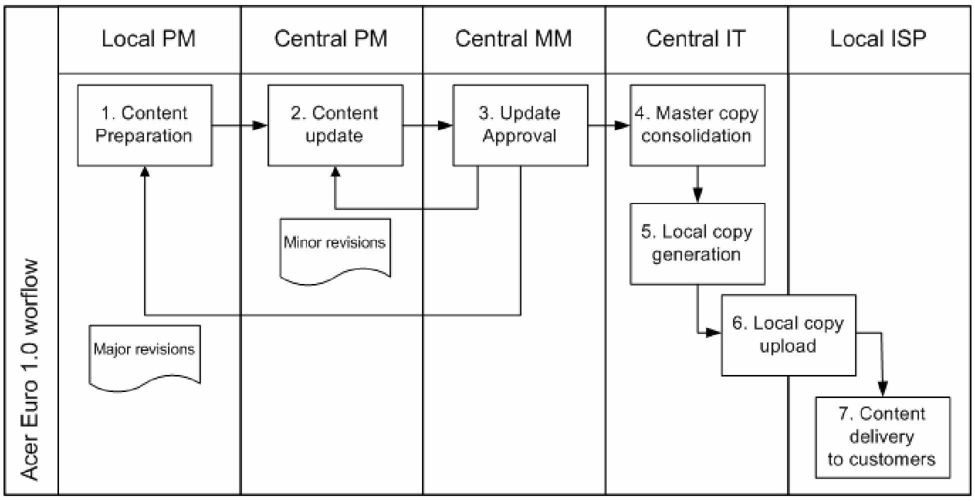


Figure 13: Acer-Euro 1.0 workflow

Figure 14 shows the schedule and milestones of the Acer-Euro 1.0 projects. Only seven week elapsed from the approval of the new site map and visual identity to the publishing of the 14 national web sites and to the delivery of the CMS to Acer employees. In this period, two distinct prototypes were formally approved by the management: prototype 1, with 50% of functionality, was delivered at the end of week

2; prototype 2, with 90% of functionality, at week 5. Overall, 9 prototypes were constructed in 6 weeks: 2 formal, 7 for internal assessment.

Mile stone	Week 1 22/01 - 26/01	Week 2 29/01 - 02/02	Week 3 05/02 - 09/02	Week 4 12/02 - 16/02	Week 5 19/02 - 23/02	Week 6 26/02 - 02/03	Week 7 05/03 - 09/03
M0	🟧						
M1		🟩 ◆Prototype 1					
M2			🟧 ◆Prototype 2				
M3					🟩		
M4					🟧		
M5					🟩		
M6					🟧		
M7	🟩						
M8	🟧						
M9			🟩				
M10					🟧		
M11						🟩	
M12						🟧	
M13							🟩

M0: agreement of site map and Visual Identity	M7: information on data and traffic across countries
M1: prototype 1.0 (50% of features) + initial CMS	M8: initial stress test and tuning
M2: approval of prototype V.1 + change list	M9: definition of data and application clustering policies
M3: prototype 2.0 (90% of feature) + revised CMS	M10: final network configuration and country clustering
M4: approval of prototype 2.0	M11: database and template installation
M5: localized static texts and images	M12: content upload
M6: localized dynamic database content	M13: final stress test and tuning, publishing of the 14 sites + CMS

Figure 14: The schedule and milestones of the Acer-Euro 1.0 project

The development team consisted of four persons: one business expert and one junior developer from Acer, and one analyst and one Java developer from Politecnico di Milano.

Class	Dimension	Value
Time & effort	Number of elapsed workdays	49
	Number of development staff-months (analysts and developers)	6 staff-months (6 weeks x 4 persons)
	Total number of prototypes	9
	Average elapsed man days between consecutive prototypes	5,4
	Average number of development man days per prototype	15,5
Size	Number of localized B2C web sites	14
	Number of localized CMS applications	4 (Admin, News, Product, Other content),
	Number of supported languages	12 for B2C Web sites, 5 for CMS
	Number of data entry masks	39
	Number of automatically generated database tables	46
	Number of automatically generated database views	82
	Number of automatically generated database queries	279 for data extraction, 89 for data update
	Number of automatically generated JSP page templates	48
	Number of automatically generated or reused Java classes	250
	Number of automatically generated Java lines of code	12500 Non commented lines of code
Degree of automation	Number of manually written SQL statements	17 (SQL constraints)
	Percentage of automatically generated SQL code	96%
	Number of manually written/adapted Java classes /JSP templates	10% JSP templates manually adapted
	Percentage of automatically generated Java and JSP code	90% JSP templates, 100% Java classes
Cost and ROI	Total cost of software development of first version	75.000 €
	HW, SW licenses, and connectivity cost of first version	70.000 € (db server license)
	Return on investment of first version	12-15 months
	Average effort of extension to one additional country	0,5 staff-months
	Average cost of extension to one additional country	7.500 €
	Average ROI of extension to one additional country	2 months
Productivity	Number of function points	177 (B2C web site) + 612 (CMS) = 789
	Average number of function points delivered per staff-month	131,5

Table 1: Main dimensional and economic parameters of the Acer-Euro project

As Table 1 clearly shows the most relevant aspect of the development of Acer-Euro 1.0 is the time-to-market with respect to the complexity of the application: only six week of development plus one week of testing were sufficient for analyzing, designing, implementing, verifying, documenting, and deploying a set of mid-size, functionally complex, multi-lingual Web applications. Such result has to be ascribed to:

1. the high degree of automation brought to the process by the use of the model-driven approach: more than 90% of the application and database code were synthesized automatically by the WebRatio development environment from the WebML models of the applications, without the need of manually intervening on the produced code.
2. the overall productivity of the development process: the productivity value is obtained by counting the number of Function Points (FPs) of the project and dividing this value by the number of staff-months employed in the development. The result is a productivity rate of 131,5 FP / staff month, which is 30% greater than the maximum value expected for traditional programming languages in the Software Productivity Research Tables (SPR 2005).This latter result is a consequence of the former: high automation implies a substantial reduction of the manually written repetitive code and a high reuse of design patterns.

Another critical success factor has been the velocity in focusing the requirements, thanks to the rapid production of realistic prototypes. At the end of week 2 the top management could already evaluate an advanced prototype, which incorporated 50% of the requested functionality, and this initial round of requirement validation proved essential to the delivery of a compliant solution in such a limited time. With respect to traditional prototyping, which exploits a simplified architecture, WebRatio generates code directly for the actual delivery platform; in this way, stress test and architecture tuning could start already at week 1 on the very first prototype, greatly improving the parallelism of work and further reducing time to market.
From the economic standpoint, the investment in the first version was very limited, around 10k€ per national web site. Software license cost were abated, thanks to the possibility of generating code for open-source and even freeware platforms. Connectivity costs were also cut, due to the possibility of reusing the network infrastructure already in place. This permitted Acer to spread the costs of building the new EMEA network infrastructure over a longer period of time, achieving a better amortization of the investments in infrastructure.

The benefits of MDD manifested non only in the development of the first version, but were even more sensible in the maintenance and evolution phase. Figure 15 shows the timeline of the additional releases and spinoff projects of Acer-Euro. Four major releases of Acer-Euro were delivered between 2001 and 2006, and the number of applications grew from the initial 5 to 13 intranet and Internet applications, serving more corporate roles and supporting more sophisticated workflow rules.

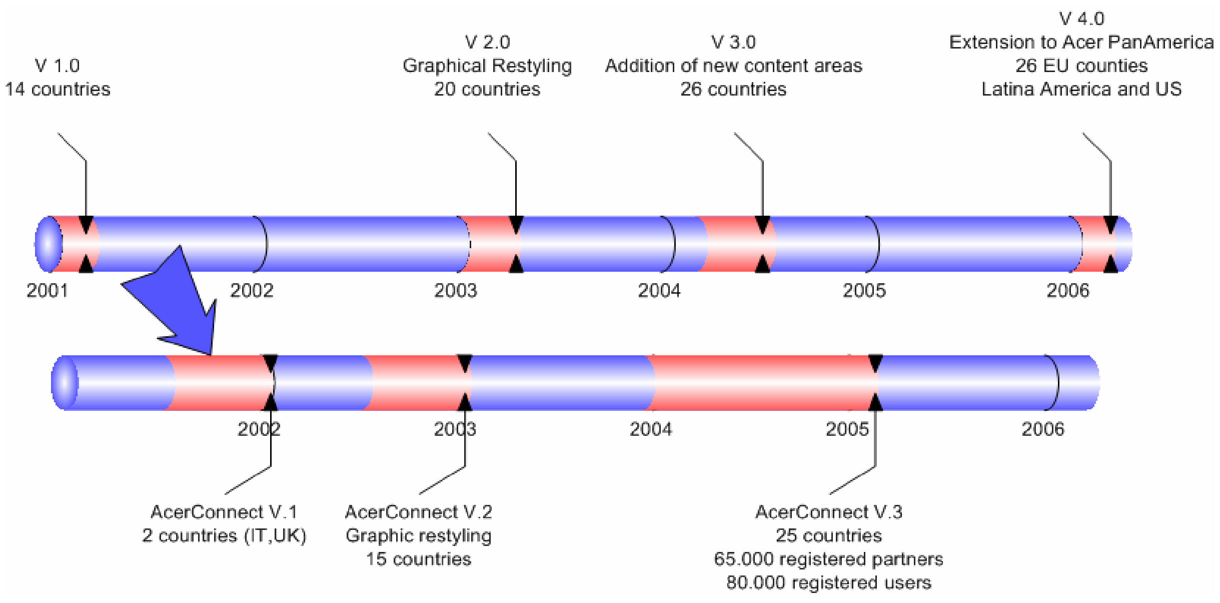


Figure 15: the evolution of the Acer-Euro Project in five years

At the end of 2005, Acer Euro was rolled out in 26 European counties and extended also to the Acer PanAmerica subsidiaries, including Latin America and the U.S. As early as June 2001, an extension of the Acer Euro platform was scheduled, to address the delivery and management of content for the channel operators (Acer partners). This spinoff project, called Acer Connect, is a multi-actor extranet application targeted to Acer partners, characterized by the following features:

1. The segmentation of the users accessing the site into a hierarchy of groups corresponding to both Acer's and partners' business functions.
2. The definition of different access privileges and information visibility levels to groups.
3. The provision of an Acer European administration role, able to dynamically perform via the Web all administrative and monitoring tasks.
4. The provision of an arbitrary number of nation-based and partner-based administration roles, with responsibility of local content creation and publishing, and local user administration.
5. A number of group-tailored Web applications (e.g., sales, marketing) targeting content to corporate-specific or partner-specific user communities.
6. The management of administrative and business functions in multiple languages, flexibly set by administrators and users.
7. A security model storing group and individual access rights into a centrally managed database, to enforce global control over a largely distributed application.
8. Content personalization based on group-specific or user-specific characteristics, for ensuring one-to-one relationships with partners.
9. Advanced communication and monitoring functions for the effective tracking of partners' activity and of Acer's quality of services.

The first version of Acer Connect was deployed in Italy and UK in December 2001, after only seven elapsed months of development and with an effort of 24 man months. Today, Acer Connect is rolled out in 25 countries and hosts 65.000 registered partners, delivering content and services to a community of over 80.000 users. Acer Connect and Acer-Euro share part of the marketing and communication content, and therefore the former project was realized as an evolution of the latter; starting from the data model of Acer-Euro, the specific functions of Acer Connect were added and the new applications were modeled and automatically generated. The model-driven approach greatly reduced the complexity of integration, because the high-level models of the two systems were an effective tool for reasoning about the functionality to reuse and develop.
Besides Acer Connect, several other projects were spun-off, to exploit the customer and partner communities gathered around these two portals. Figure 16 overviews the delivered B2C project, which collectively totalize over 10.800.000 visits per month.

As a remark on the long-term sustainability of MDD, we note that, despite their complexity and multi-national reach, both Acer-Euro and Acer Connect are maintained and evolved by one junior developer each, working on the project at part time. In total, only 5 junior developers, allocated to the projects at part time, maintain all the mission-critical Web applications depicted in Figure 16.

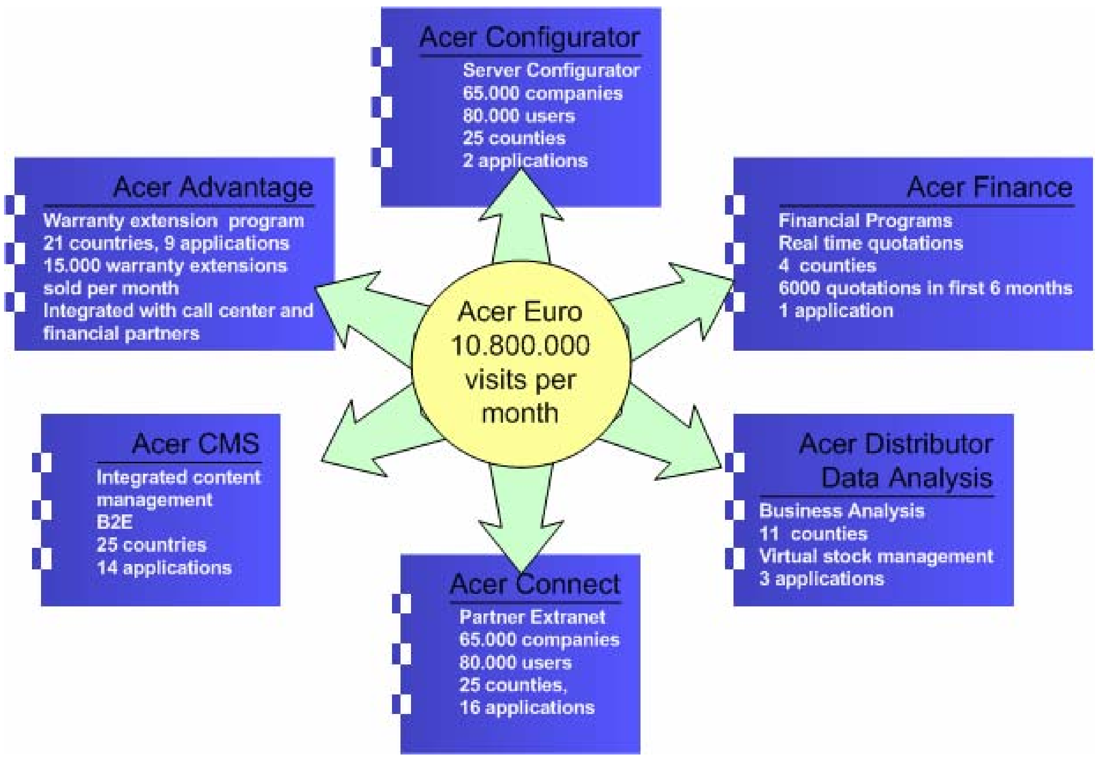


Figure 16: Overview of the main applications developed in Acer with the MDD approach

Acer Advantage

As a second case study, we describe in more detail the Acer Advantage project (http://www.aceradvantage.com), a satellite application of Acer-Euro supporting a warranty extension program associated with Acer products such as notebooks, personal computers, and so on. The business goal of the project was to offer online Acer customers a three year warranty extension with flexible implementation options such as: service at certified Acer Repair Centers or on customer site, with diversified response times, quick repair using the AcerAdvantage customer privileged program, and temporary spare part replacement. A key factor in the success of the program has been the flexibility in activating and managing the contracts: customers who purchase the program with an Acer product can simply activate their contract online, or refer to their country's Call Center, which is integrated with the web application. The back-end system manages the complete workflow from customer sign-in to contract definition and delivery: at the end of the process Acer sends the customer a confirmation of his warranty extension activation. Warranty extensions, personalized services and call centers support an integrated "Customer Service" strategy that Acer is successfully implementing on the market to consolidate its customer relations and develop a unique European level CRM. The initial version of the AcerAdvantage service, including the customer application front-end and the back-end system supporting the contract definition workflows, has been developed in less than three months. The return on the investment and marginality of the project have been extremely successful, as illustrated in Figure 17.

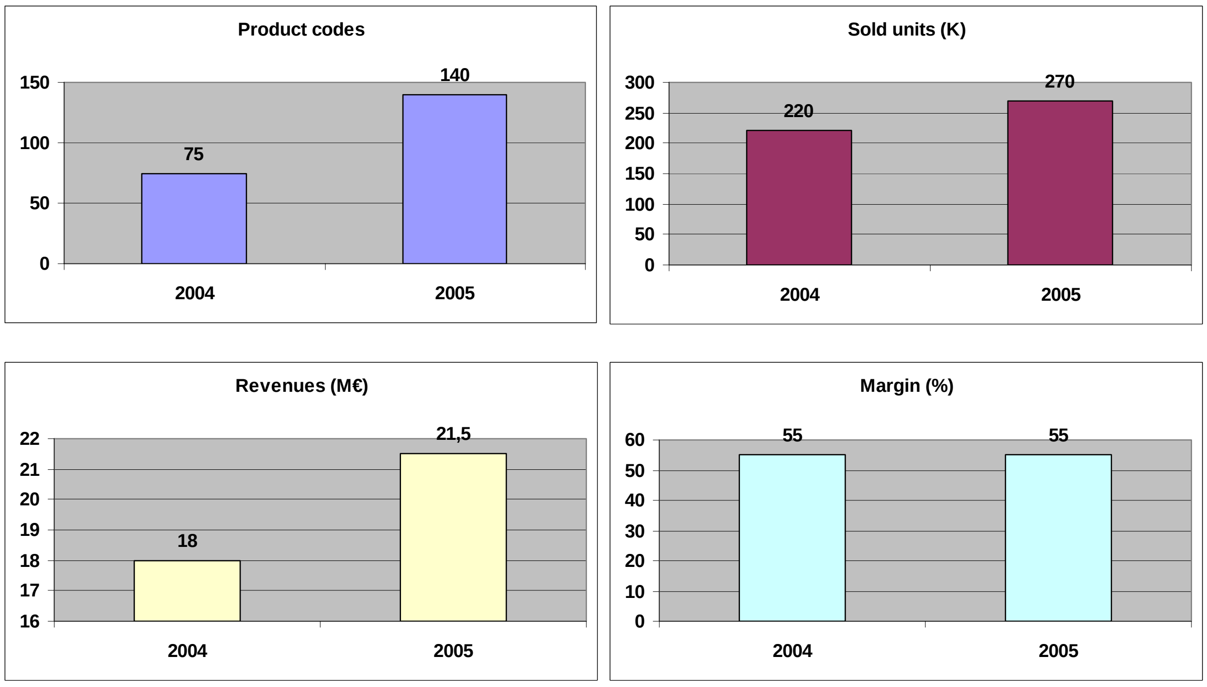


Figure 17: The sales and revenue stream of the Acer Advantage project

Results and critical considerations

In this section, we summarize the main lessons learned in the application of the MDD principles to web development.

The impact of MDD on the software development process

The major effect of MDD is to shift the focus of development from implementation to requirement analysis. Almost 80% of the delivery effort concentrates in the phases of data design, hypertext design and prototyping. This means that more development time is spent with the application stakeholders, to refine design models and evaluate prototypes. The result is a better quality of the delivered applications and a higher rate of acceptance, because design errors and requirements misinterpretations are eliminated as early as possible in the development process. MDD also benefits the more technical tasks of testing, maintenance, and evolution, because reasoning on the properties of a developed system is far more effective at the conceptual level than at the physical level of the source code.

Reorganizing software development responsibilities with MDD

MDD lowers the technical barriers for developing complex Web applications, allowing a more flexible distribution of responsibilities between the IT department and the business units. In a phase of transition, when business goals are rapidly evolving and quick adaptation to changing environmental conditions is a critical success factor, the possibility of developing, monitoring and adjusting mission critical systems, especially in the B2C area, directly within the marketing and communication division greatly improves efficiency. For example, Acer has been able to seize important business opportunities and increase the sales margin on services by quickly deploying a number of systems to its customers and dealers offering value-added services in the financial sector. Business requirement identification, technical specifications, development and evolution have taken place within the business unit responsible of the project, which could jointly master the technical, communication, and business aspects, eliminating any idle time in the decision process and consequently reducing the time to market.
The solutions were developed with an open and standard architecture, in which the integration of heterogeneous systems takes place by means of data and function exchanges realized using web services. In this way, a well-defined architectural protocol can be established to integrate systems autonomously developed in different departments and business units, avoiding the duplication of software functions and data.

Evaluating the economic impact of MDD

Last but not least, MDD has proven an economically profitable and sustainable way of developing Web systems. The peak productivity rates experienced in the Acer projects have reached five times the number of delivered function points per staff-month of a traditional programming language like Java, and three times the productivity rates of office application development languages and tools like Microsoft Excel or IBM Lotus Notes (SPR 2005). This productivity level greatly reduces the front-end investment in software development and allows the short-term return on investment, which helps the managerial decision process about high risk projects like B2C solutions.
On the negative side of MDD, the initial training costs must be considered. MDD requires non-technical knowledge on the modeling of software solutions, which must be acquired with a mix of conventional and on-the-job training. Acer estimates that it took developers from 4 to 6 months to become fully productive with MDD, WebML, and WebRatio. However, as Figure 18 shows, the initial investment in human capital required by MDD pays off in the mid term. The number of applications developed and maintained per unit of development personnel increases with the developers' expertise and exceeds ten fully operational, complex and distributed Web applications per developer.

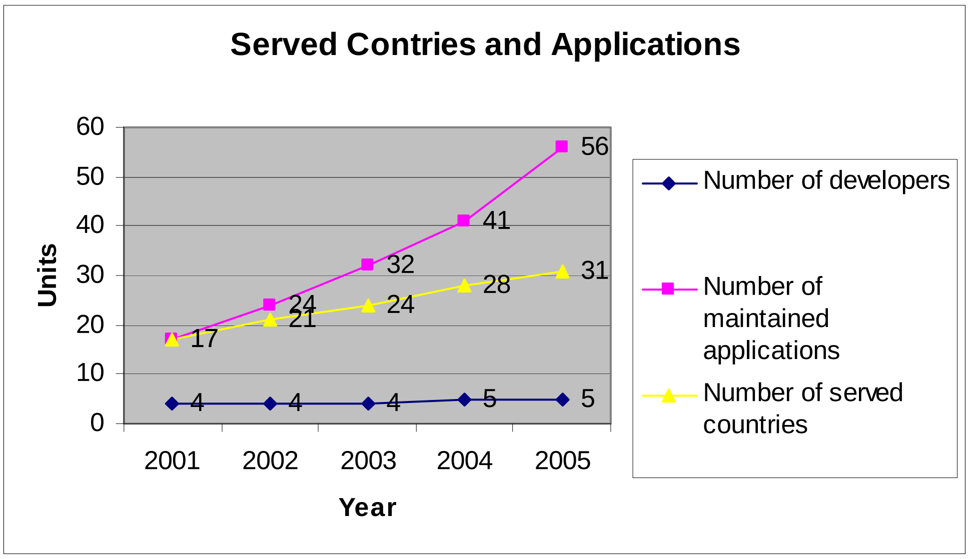


Figure 18: Evolution of manpower versus number of maintained applications and served countries

Final considerations

The Acer experience has demonstrated the feasibility of MDD and the efficiency it introduces into the development lifecycle, largely anticipating the current debate on the Model Driven Architecture.
However, the benefits of a model-based approach to the construction of software systems must not only carry over the initial development phase, but also extend to the maintenance and evolution steps, which account for a substantial fraction (over 60%) of the total lifecycle cost of a solution.
After more than five years of applying MDD with WebRatio, Acer has gained sufficient experience to draw some general conclusions. Today, the use of WebRatio has spread from the Acer-Euro project to most of the Web-based B2C, B2E, and B2B platforms of Acer EMEA and has been exported from Europe also to Acer PanAmerica. The developed solutions cover all the most critical sectors of Acer's business and have given tangible benefits over the years: 1) the distributor data analysis system allows the centralized management of distributor's stock data across multiple countries; 2) the financial system targeted to dealers supports the online quotation process; 3) the sophisticated Web-based server configuration application rolled out in 12 countries totalizes over 3000 configurations and 22.000 potential user activations per year; 4) the Acer Connect business partners Extranet reaches more than 65.000 companies in 25 countries and has dramatically cut the cost and effort of targeting content and services to partners; 5) Acer-Euro application and its extensions are now serving 34 countries with 10.800.000 million visitors per month and have become a fundamental competitive factor of the deployment of Acer business in the EMEA and PanAmerica regions. The abovementioned portfolio of solutions has been deployed, and is continuously being maintained, by an internal team consisting of five developers only. With a traditional development methodology and using conventional programming-oriented tools, the company estimates that the construction of the deployed systems would have required at least three times the resources that have been invested. None of the developed systems has been retired or has become obsolete; new requirements have been smoothly incorporated into the running versions and rolled out by iteratively extending the deployed systems.

In conclusion, MDD appears to be a powerful tool for renovating businesses and taking advantage of the advent of low-cost distributed network infrastructures. However, the transition requires innovation not only in the business strategies but also in the IT departments.

Related work

WebML builds on several previous proposals for hypermedia and Web design notations, including HDM, HDM-lite, RMM, OOHDM, and Araneus. The design principles, notations, and development procedures are described at large in (Ceri et al 2003); HDM (Garzotto et al 2003) pioneered the model-driven design of hypermedia applications and influenced several subsequent proposals like HDM-lite (Fraternali and Paolini 1998), a Web-specific version of HDM, RMM (Isakowitz et al 1995), Strudel (Fernandez et al 1998), and OOHDM (Rossi et al 1999). All these methods offer powerful built-in navigation constructs, as opposed to WebML, which exploits simple, yet orthogonal, composition and navigation primitives. Araneus is a recent proposal for Web design and reverse-engineering, in which the data structure is described by means of the Entity Relationship Model and navigation is specified using the Navigation Conceptual Model (NCM) (Atzeni et al 1998). Conceptual modelling is followed by logical design, using the relational model for the structural part, and the Araneus Data Model (ADM) for the navigation aspects. Araneus also includes predefined navigation primitives and does not model presentation at an abstract level. The relevance of using conceptual modelling techniques in the context of the Web was addressed in a work by Rossi et al, which describes an evolution of OOHDM (Object-Oriented Hypermedia Design Method), a methodology for designing hypertexts that shares with WebML the vision of orthogonal design dimensions (Rossi et al 1999); specifically, OOHDM addresses the conceptual modelling, navigation design, interface design, and implementation. Navigational contexts in OOHDM provide a rich repertoire of fixed navigation options. Given that WEB applications are, after all, software artefacts, it is not surprising that several proposals exist for using UML for their specification (Booch et al 1998). In particular, (Conallen 1999) shows how the architecture of Web applications can be modelled in UML. Web pages are represented as UML classifiers, distinguishing among their "server side aspects" (i.e., their relationship with middle tiers, databases, and other resources) and their "client side aspects" (i.e., their relationships with browsers, Java applets, ActiveX controls and so on). This distinction of roles and the use of stereotypes enables a quite effective representation of Web applications using standard UML, however the author does not show how to use the proposed notations for model transformation and code generation. In the tool

market, a few tool vendors have started extending their development environments with application modelling and code generation capabilities. The well-known BEA Workshop tool allows developers to specify Web applications and Web services, but with a limited high-level view of the application because it concentrates on the implementation tasks (BEA 2004). Another popular tool is ArcStyler, which adopts UML and predefined architectural patterns and model transformations (called cartridges) for representing Web applications and generating their code for the J2EE and .NET platforms (Interactive Objects). From a methodological perspective, the WebML approach is close to conceptual methodologies like W2000 (Baresi et al. 2000) and OO-H (Gomez et al. 2001]), which are based on the Unified Modeling Language.
The influence of MDD on the development of mission-critical business applications is still a subject under investigation. The OMG Web site lists a number of industrial cases, where the MDD approach has been exploited in the construction of industrial systems (OMG 2006). The published data are mostly qualitative, but they confirm the increase of productivity that we have found during the development of the Acer Systems.

Conclusions and future work

In this paper we have reported on a multiyear experience in applying Model Driven Development to the construction of mission-critical eBusiness solutions, jointly conducted by Acer, the 4th branded PC vendor worldwide, and Web Models, a spinoff company of Politecnico di Milano, the largest Italian IT school. The focus of the experience is on exploiting innovation in the software development process to support the business transformation requested by the Web revolution and the globalization of competition. The key idea is to shorten the time to market and total cost of ownership of mission-critical B2C and B2B solutions by exploiting the Web as a standard delivery platform and by leveraging novel-generation development methodologies based on the high-level modeling of requirements and on the automatic generation of code. We have reviewed a number of projects conducted in Acer since 2001, which demonstrate that MDD can help the business units to take a proactive role in the conception, design and deployment of customers solutions, without incurring in high training and development costs, duplicated efforts with the IT department and jeopardazing the corporate IT infrastructure. The key ingredients for such a performance are: a suitable high-level model for capturing business requirements and application design, which can be communicated to non-technical people, rapidly transferred to junior analysts and developers; a set of software tools for automatically transforming the models into source code and managing all the collateral software development tasks (testing, documentation, maintenance, quality and performance verification) starting from the application models.
Our future work will concentrate on improving and extending the quantitative assessment of the benefits of Model Driven Development in the Web application sector. We are finalizing a novel software tool for automatically performing the analysis of software projects conducted with the WebML methodology, which will support the measurement of different project parameters related to size, effort and cost, and the construction of a statistic baseline for productivity and return on investment data. We plan to perform a broad review within the customer base of Web Models, in order to verify to what degree the benefits occurred in Acer take place in different industries (e.g., in manufacturing companies), with different software development processes (e.g., outsourced development) and in different markets (e.g., business, software and technology integrators).

References

Josè Allouche (ed) (1995) *Technology Management and Corporate Strategies: A Tricontinental Perspective*, North-Holland

Paolo Atzeni, Giansalvatore Mecca, and Paolo Merialdo (1998), Design and maintenance of data-intensive Web sites. In Proceedings of the International Conference on Extending Database Technologies. Lecture Notes in Computer Science. Valencia, Spain. Springer, 436–450.

BEA (2004), BEA Workshop Product Family.
[available at http://www.bea.com/content/products/workshop/].

Grady Booch, Ivar Jacobson, James Rumbaugh (1998), *The Unified Modeling Language User Guide*, Addison-Wesley

Peter Chen (1976), The Entity-Relationship Model - Toward a Unified View of Data, *ACM Transactions on Database Systems* Vol. 1 (1), 9-36

Stefano Ceri, Piero Fraternali, et al (2003), *Designing Data-Intensive Web Applications*, The Morgan Kaufmann Series in Data Management Systems

Jim Conallen (1999) Modeling Web Application Architectures with UML, *Communications of the ACM*, 42:10, pp. 63-70.

David Ford et al., *Managing Business Relationships*, second edition (2003) J. Wiley & Sons

Mary Fernandez, Dana Florescu, et al (1998), Catching the boat with Strudel: Experiences with a Web-site management system. In *Proceedings of the 24th ACM SIGMOD International Conference on the Management of Data*, Seattle, WA

Dana Florescu, Alon Levy, Dan Suciu and Kalhed Yagoub (1999), Optimization of run-time management of data intensive Web-sites, In *Proceedings of the 25th International Conference on Very Large Databases*, Edinburgh, Scotland. Morgan Kauffman, 627–638.

Piero Fraternali, Paolo Paolini (1998), A Conceptual Model and a Tool Environment for Developing More Scalable, Dynamic, and Customizable Web Applications, *Proc. EDBT 1998*, pp. 421-435.

Piero Fraternali (1999), Tools and approaches for developing data-intensive Web applications: A Survey, *ACM Computing Survey* 31, 3, 227–263.

Franca Garzotto, Paolo Paolini, Daniel. Schwabe (1993), HDM - A Model-Based Approach to Hypertext Application Design. *ACM TOIS* 11, (1): 1-26

Jaime Gomez, Christina Cachero and Oscar Pastor (2001), Conceptual modeling of device-independent Web applications, *IEEE MultiMedia* 8, 2, 26–39.

Interactive Objects, ArcStyler v.5, [available at http://www.interactive-objects.com/}

Tomas Isakowitz, Edward Stohr, P. Balasubramanian, (1995), RMM: A Methodology for Structured Hypermedia Design. *CACM* 38(8): 34-44.

Chuck Martin *Digital Estate*, *Strategies for Competing and Thriving in a Networked World*, (1998) McGraw-Hill Education

Object Management Group (OMG), Model Driven Architecture (MDA), [available at http://www.omg.org/mda]

Gustavo Rossi, Daniel Schwabe, Fernando Lyardet (1999), Web Application Models are More than Conceptual Models, in *Proc. Int. Workshop on the World Wide Web and Conceptual Modeling*, Springer-Verlag, LNCS 1727, pp.239-252

C. Sauer, Philip W. Yetton**,** *Steps to the Future: Fresh Thinking on the Management of IT-Based Organizational Transformation***,** Wiley 1997.

Software Productivity Research (SPR), SPR Programming language Table – Version PLT2005a, (2005), [available at http://www.spr.com]

Jos Warmer, Wim Bast, Diane Pinkley, Mario Herrera, Anneke Kleppe, (2003) *MDA Explained: The Model Driven Architecture: Practice and Promise*, Addison Wesley