Intelligent traceability system of Cabrales cheese using MDA TALISMAN

Vicente García-Díaz¹, Héctor Fernández-Fernández¹, Elías Palacios-González¹, B. Cristina Pelayo G-Bustelo¹, Juan Manuel Cueva Lovelle¹
¹Department of Computer Science, University of Oviedo, Oviedo, Asturias, Spain

Abstract – Thanks to the Model Driven Engineering we can talk of intelligent systems capable of generating other systems from given specifications. In this paper we present a real case of application of Model Driven Engineering (or Model Driven Development) with which you can create complete systems ready to be deployed from models used as input specifications. In our example we show how this concept can be applied to applications for food traceability, focusing on the particular case of Cabrales cheese in which each producer is a different world, as each producer uses a different process of development, in keeping with its preferences and quality of the final product that it is looking for. Were it not for MDD, we would have to tailor each application to each producer of Cabrales cheese but now that work is no longer necessary.

Keywords: Intelligent, Automatic, MDA, MDD, Web Engineering, Traceability.

1 Introduction

Model Driven Development (MDD) is an approach to software development that is gaining more strength each day and that allows to generate the code of applications either automatically or semi-automatically. MDD is a generic term that refers to different paradigms as Software Factories (proposal of Microsoft based on MDD) [10] or primarily to the Model Driven Architecture (MDA) [2], [11], and [12], promoted by the Object Management Group (OMG) [4] that is who is giving further impetus to MDD. OMG has other defined standards that serve as building blocks for the definition of MDA (MOF [3], UML [4], XMI [5], OCL [6] and QVT [7]).

The use of models has several advantages, increases portability, interoperability and reusability of systems. Thanks to the models, we can build a platform based on MDA (we will make a short introduction to our platform MDA TALISMAN in the next section) that can transform models into a certain output platform (.NET, JAVA, PHP, etc.), intelligently, without human intervention. The idea, shown in Figure 1, is to start with some models (all models must be based on a formal meta-model, in such a way as to allow specify all elements of the system and make transformations among different models) of a high level of abstraction (CIM, Computational Independent Model) that pick up the requirements of the system without using a computer language. Later becomes a transformation lowering the level of abstraction to a computer model but independent of the computer platform used (PIM, Platform Independent Model) which represent solutions at design level for the requirements of the CIM. The PIM can be transformed into one or more models dependent on the computing platform used (PSM, Platform Specific Model) that provides specific models of one or more desired technological solutions. The last transformation is to convert the various PSM (their number depends on the number of platforms) to source code, ready to be used or to refine it previously before being used.

Figure 1. Model Driven Architecture

It must be said that although the OMG is making a great effort to promote Model Driven Development, There are concepts that are not clear in its guide [2], which is the main reference about MDA, including the concept of
platform or the degree of automation that the transformation should have. In [8] these concepts can be quite clear.

The remaining article is structured as follows: Section 2 introduces as is the architecture used in MDA TALISMAN. Section 3 describes both what is and why food traceability is needed and their need in Cabrales cheese. Section 4 shows how we apply our MDA TALISMAN to generate a system for food traceability in the particular case of Cabrales cheese, and finally, sections 5 and 6, respectively, are the conclusions and references to other publications.

2 MDA TALISMAN Architecture

The MDA TALISMAN, our proposal based on MDD, use the approach promoted by the OMG, doing separations at different levels of abstraction. The highest level of abstraction, CIM, is transformed to a PIM manually, and since then the process is automatic, the PIM is refined and becomes another PIM, the PIM becomes a PSM, the PSM is refined and becomes another PSM, and finally the PSM is transformed directly into source code. Although we have adapted it to the real case presented in this paper, its basic architecture is shown in the following figure.

![Figure 2. Basic architecture of the MDA TALISMAN](image)

There are three distinct layers, Content, Web Application (we are oriented towards Web applications), and Functionality. Each one of the three layers can be divided into the three views promoted by MDA (CIM, PIM, and PSM):

1. **Content**. This layer defines the data to create the business layer and the main structure of the database

2. **Web Application**. This layer is responsible for creating the structure of the pages viewed by the end-user and the relationships between them. It is also responsible for user profiles, and users who can access the cited pages or portions thereof.

3. **Functionality**. Thanks to this layer it is able to offer Web Services to the outside or may use other Web Services supplied from other URLs.

3 Traceability of the Cabrales Cheese

Now we will discuss which food traceability is and which Cabrales cheese is, to finally connect the two concepts.

3.1 Food traceability

Although many people are unaware of what is food traceability, the reality is that since January 1, 2005 (in accordance with article 18 of the European regulation 178/2002) [9] all food businesses must have a traceability system.

It can be concluded that food traceability is a tool with which to tackle the problems that can give products for the food consumption. It consists of collecting data during all phases of the production process of an article and whether health authorities require it, to have such information.

Information is beginning to store with the raw materials used in manufacturing (origin, quantity, supplier, and any other information that may be of interest). Later will be stored intermediate processes that occur in the manufacturing of articles (e.g. dates or temperatures) and finally, who sold the article before arriving at the hands of the end-user (such as an intermediary or a supermarket). The idea is to have fully determined the history of an article.

The goal is to have stored information and that can be used by health authorities, but it may also be of interest to the producer to store statistics, ensure product quality or determine responsibilities, and of course the ultimate consumer may be interested in know the origin of what they eat. For that reason might be a good choice to provide consumers with mechanisms to be able to get some kind of information on the articles they consume, as this will increase confidence through transparency.

One of the main causes of the introduction of food traceability is that when there is an alert food (one of the most famous cases could be the mad cow disease, although it could be in a much smaller scale) we must locate and withdraw from the chain supply any product that might be affected in some way, from effectively and efficiently, because so far no one can say that things are not well without food traceability systems but neither can anyone prove that the companies are doing well and of course a failure in a food business for a particular item (e.g. chicken meat) can affect the image and thus in the economic aspect of all other companies who market the same or similar articles.
3.2 Production of Cabrales cheese

The Cabrales cheese is a cheese made in Asturias (Spain), especially characteristic of the council of Cabrales but also occurs in some localities of its neighbour council, Peñamellera Alta.

We are talking about a blue cheese made with milk from cows, goats and sheep, where it is not necessary to mix the three types of milk; it can use only one type of milk, a mixture of two types, or a mixture of all three types of milk. For its elaboration, once mixed milk with rennet and optionally with salt will, have to wait some time for at least one hour. When the curd (result of the mixture) has been formed is broken and is left to stand for several hours. Then, we have to use a few moulds that will shape each cheese, and we have to let dry each mould of curd for several days. Once it is dry enough, the moulds are removed, and it is salted on one side and past few days on the other. The maturation takes place in the caves that are in a high mountain known as Picos de Europa. The high humidity and temperature there is conducive to be developed in the cheese moulds of the type penicillium, what gives the cheese the greenish blue colour. When the cheese comes out from the cave, it will be packaged to be sold later. It should be borne in mind that this whole process, from the curd to the packaging, has carried out at least 60 days.

Figure 3. Asturias in Spain and Cabrales in Asturias

The cheese that we are talking about is a cheese with a long tradition in Asturias. For that reason it has been formed a body (Board of the Designation of Origin "Cabrales") [1] which ensures the quality, authenticity and enforcing standards, performing tasks such as inspect cows, sheep or goats in the area, inspect cheese factories and maturation caves in the area, controlling milk and cheese produced, advise and collaborate with the producers of cheese, and promote Cabrales cheese. The Cabrales cheese has Denomination of Origin since 1981 at the “Instituto de Denominaciones de Origen” (INDO) and complies with the European standard (EN-45011) on product certification agencies. So to preserve its Designation of Origin and ensure the quality of cheese produced, is necessary adapt to the current rules and implement a system of traceability for all Cabrales cheese factories.

4 Case Study

The idea of introducing a system of food traceability to each of the cheese factories shows the need to study how the different cheese factories are. The study showed that all cheese factories are small and not all follow the same manufacturing process, each cheese factory has different nuances to the other, so the food traceability system should be adapted to each of the cheese factories independently (neither all the factories perform the same processes of preparation, nor all in the same order). That is why we thought of using MDD to avoid having to change the source code of any of the created applications. We use models tailored to each individual case and our MDA generates a specific food traceability application for each cheese factory, based on the introduced model PIM, or to put it in another way, based on the analysis of the manufacturing process used in each factory. The truth is that our MDA also could generate food applications for other industries: sausages, fish, etc.

4.1 Model

Our model represents the manufacturing process that could take every one of the cheese factories, so it must adapt
to them. To achieve this, we have been created a language called XPDML (eXtensible Process Definition Markup Language), subset of XML (eXtensible Markup Language), divided into seven main sections:

1. **Actions.** This section describes unambiguous manner all actions that will be on development process. One action, as well as its attributes, has to indicate what products will receive as input and what products will appear in the output after running the action. It should be seen as a graph in which a node has entrances and has exits (e.g. milk mixture consists of mixing milk, rennet and salt and as a result we get curd). In addition, to control the various operations (e.g. the range of temperature of the milk that is received at the factory) restrictions can be defined for each. Another interesting thing is the list of hardware devices to intervene in an action (for example when a cheese is packed, it will generate a label with a labeller).

```xml
<item Id="ITEM_COMMERCIALIZED_CHEESE"
     Measurement=""
     Type="SERIAL"
     <properties>...
     <forecast>...
     </forecast>
     <locations>...
     </locations>
     <devices>...
     </devices>
</item>
```

Figure 7. Code fragment – Description of one item

2. **Items (articles).** As important as the actions are the articles, because the actions (nodes) will have articles at the entrance, and articles at the exit (arcs). In the example shown in Figure 7 is showed as is defined a cheese marketed. The products have other sub-definable features such as properties (for example, who customer has bought the cheese or on what date), definition of forecasts for production calculating (e.g. how many kilos of cheese will be produced with x litres of milk), definition of locations (e.g. different drying caves where cheeses can lead), and lastly the hardware devices associated with that article (a scale weighing would be an example). For the above example, it has not been necessary to define forecasts, locations or hardware devices, but it will be necessary for other articles.

```xml
<device Id="DEVICE_BIZERBA_1"
    Connection="Ethernet"
    Protocol="TCP/SERVER"
    IP="192.168.2.2"
    Port="3000"
    Type="Terminal"
    System="BIZERBA" />
```

Figure 8. Code fragment – Description of one device

3. **Hardware devices.** For the system to work properly it is necessary that the server and other hardware interact. Below is listed as a device could be defined through an Ethernet connection to an IP address and a determined port. It also notes that the hardware is a terminal (could be another, such as a labeller).

4. **List of items.** We used different lists of items to give functionality to the system. The code below shows the list of possible designs for a label to be printed, could see that in the application, initially only, there will be two possible designs, design one and design two, respectively stored in the files D0001 and D0002. In addition to this list, there will be very different lists with other information such as customers, suppliers, business data, types of milk, and so on.

```xml
<item Id="ITEM_RAW_MILK"...
     <inputs>...
     <outputs>...
     <constraints>...
     <devices>...
</item>
```

Figure 6. Code fragment – Description of one action

Figure 5. Code fragment – Initial model
5. **Reports.** The reports are very important, because they collect and display necessary information about the system. There are several types of reports, such as lists of facility status, cleaning, product description, temperature control, etc. Below is a description of one report, but the source code is not expanded for reasons of space.

```xml
<reports>
  <report Id="REPORT_CHECK_1"/>
  <report Id="REPORT_CLEAN_1"/>
  <report Id="REPORT_DESCRIPTION_1"/>
  <report Id="REPORT_ELABORATION_1"/>
  <report Id="REPORT_CONTROL_TEMPERATURE_1"/>
  <report Id="REPORT_RECEIPTION_RAW_MILK_1"/>
</reports>
```

Figure 10. Code fragment – Description of one report

6. **Labels.** It is necessary that all factories have a labelling system to label their products before selling them to an intermediary or a final customer. Below is a description of how to define a label for a client and a definite article, indicating diverse information, such as label design, the style of the label, the type of bar code, the initial digits of the bar code, or a series of fields which give descriptions (in the image we use only four descriptions but in reality there are many more).

```xml
<label Client="#1" Product="#1">
  <idModel Value="#2"/>
  <design Value="#2"/>
  <ceilingLabel Value="#1"/>
  <idBarcode Value="#2"/>
  <barcodeDigits Value="012345678901"/>
  <idCost Value="#3"/>
  <costFixedValue Value="#2"/>
  <expirationDate1 Value="#10"/>
  <expirationDate2 Value="#30"/>
  <description1 Value="Weight:"/>
  <description2 Value="Batch:"/>
  <description3 Value="Serial:"/>
  <description4 Value="Cost:"/>
  ...
</label>
```

Figure 11. Code fragment – Description of one label

7. **Traceability points.** Traceability is essential in the implementation generated by the MDA TALISMAN. The snippet below shows an interesting point we want to register. The important thing is that the code indicates the product for which we want to record information and the property that we want to save (for example, we might want to save the date and time when creating a new batch of cheese).

```xml
<traceabilityPoint
Id="DAIE_MIX_MILK"
Item="#ITEM_CREATE_BATCH_CHEESE"
Property="DATE_AND_TIME"
Visible="True"
Attributes="LOW"/>
```

Figure 12. Code fragment – One traceability point

The application will be generated entirely adjusted to each cheese factory and it needs only the model described above, style sheets, images customized for each customer and finally the language files, as a feature of the application is generated which is multilingual and can be adapted to the language you want no more than adding files to the desired language translation, even at runtime. Note that in the above definitions we have used constants (to make the translations), defining them with capital letters and hyphens.

### 4.2 Transformations

Our PIM will be transformed into another PIM with the operation PIMtoPIM implemented internally by MDA TALISMAN. Then, the last PIM becomes a PSM model, generated as well as PIM, automatically by the MDA TALISMAN with the operation PIMtoPSM (initially targeted at the platform ASP.NET 2.0 and SQL Server 2005, although it could be any other platform, thanks to
MDD, due to it makes possible to separate the model and the target platform). The PSM, finally, will be transformed automatically by MDA TALISMAN with the operation PSMtoCODE in the source code ready to be compiled with Visual Studio or any other compatible environment. This source code can be compiled and deployed directly but of course is also open to possible amendments that someone could want to perform before his compilation and deployment.

The central point is the PIM, which represent all the work that can be done in the cheese factory. Inside the PIM the crucial to specify the work of each cheese factory are the actions and the articles that form the graph of production of each cheese factory, making changes since the emergence of the raw materials to marketing the cheese. The hardware devices, lists of items, reports, labels and the traceability points will complement the graph, giving more level of detail.

4.3 Result

As a result of the transformations described previously, we obtain a software solution consisting of seven projects ready to be compiled and deployed:

1. **Web Application.** In this project is all the source code required for the presentation of Web to the users. It includes, among other features, logs, Multilanguage support, authentication by profiles and usernames, and compliance with WCAG AA. With the Web application, it can be configured aspects of the system and see all the information about it. To make the different choices it will be necessary to authenticate on the system, and depending on the user profile, different choices can be performed (see the production map, see the map of the different locations, access to visualize traceability, access to reports, manage labels, review production forecasts, perform actions with articles, managing articles, manage lists of items, or access the administration menu). In the figure below you can see a snapshot with which you can make a report through a check list.

2. **Windows Service.** Given the characteristics of the food traceability it is clear that we must interact with hardware (terminals, scales, printers, bar-code readers, etc.). The Windows Service, which is running continuously on the server, will be the manager of hardware and the communications between all hardware. It is possible performed through the terminals managed by the Windows Service, as well as through the Web application, all actions that the system allows.

3. **Business Layer.** To generate a solution with multiple layers, it has been created with a business layer to manage objects and allow then, their persistence in database.

4. **Business Facade Layer.** Among the business layer and the data access layer is this layer which allows actions such as inserting, updating, deleting, or selecting.

5. **Data Access Layer.** It is the responsible for independent access to the data. Thus, if the database suffers changes, the rest of the logic of the system is not affected.

6. **Unit Testing.** The solution provides a project to make unit testing automatically without the need to create them by hand, thus saving time and effort.

7. **Utilities.** It includes utility classes such as creating bar codes, batches, or the management of label printing, among others.

5 Conclusions and Future Work

Although the idea is to reach this end, we realize that it is still far to create a generic MDA for all kinds of applications. While we continue working with that goal, we...
are satisfied with a smart MDA capable of generating so fully automatic food traceability applications, which is already in operation.

Our MDA TALISMAN begins its process with a PIM (A representation of the manufacturing process of each individual case). Our idea is to investigate further and determine how we might get in the best possible way and automatically from CIM to PIM this would greatly facilitate the task, because people outside the world of computers could generate their own applications for food traceability from their specific requirements independent of computing (CIM).

It is also interesting the idea of being able to dynamically create the graph of the process of manufacturing in the factory, so that someone can change at runtime the actions and products that the system will handle.

Last but not least, note that this work has been done with the contract FUO-EM-120-07 “Software Development for the realization of traceability”. Both University of Oviedo and the company LINK working together.

5 References


[8] Javier Muñoz, Vicente Pelechano. “MDA a Debate”; I Taller sobre Desarrollo de Software Dirigido por Modelos, MDA y Aplicaciones (DSDM), 1-12, Nov 2004


