

Construction of a Domain Metamodel using EMF for Semiautomatic Generation of Web Applications

Lady Viviana Garay González

Universidad Distrital Francisco José de Caldas, Bogotá, Colombia

Abstract

The construction of web applications is one of the most explored working tools in software development, so it requires new or existing methodologies that allow software development to continue to evolve, providing the possibility to continue building web applications. This article presents a preliminary analysis of the use of MDE for semi-automatic generation of web applications based on the process of building a domain metamodel, that is, the basis of all the artifacts that are present in an application from the context to be developed, which forces to acquire a certain level of abstraction to achieve the generation of the domain metamodel for the context of Web applications, in MDE the transformation chains are those that allow the models can be transformed to obtain the final product. The initial proposal is the creation of a context metamodel that includes concepts related to the specific domain entities to be resolved, which would abstract the different elements that can have a solution for the domain. The purpose of this paper is to carry out the analysis as a starting point for constructing a metamodel in a domain-specific language using Model Driven Engineering (MDE) and implementation in Eclipse Modeling Framework (EMF). For the proposed case study, three academic enrollment web applications are considered that maintain the same business logic for different organizations, the solution of the case is to create a domain metamodel that allows generating the three tools, each with its particular characteristics.

Keywords

Domain, Eclipse Modeling Framework, Metamodel, Transformations Chains, Model Driven Engineering

1. Introduction

There are components such as frameworks, development environments, and plugins available for the implementation and execution of web applications, which fulfill a particular function and are related to achieving the requirements established by the users who request their construction. When an application is developed, it can share similar functionality with another system, even though its development is carried out independently to maintain the development process planned for its implementation [1, 2].


Web applications offer complete and easy-to-use functionalities allowing users to connect with each other and organizations with all those involved in the business, which allows establishing the foundations of the digital future [3, 4]. The methodologies and strategies for software development have evolved and given the possibility of building applications on the Internet with characteristics that satisfy the needs of the user who requires them. The implementation will be

ICAIW 2022: Workshops at the 5th International Conference on Applied Informatics 2022, October 27–29, 2022, Arequipa, Peru

✉ ladyviviana.garay@gmail.com (L. V. Garay González)



© 2022 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

carried out with EMF (Eclipse Modeling Framework) since it represents the core of the Eclipse platform for model-driven development, it is a very useful framework for the development of metamodels, through an abstract syntax, and it also allows automatically generating classes of implementation in Java code for the elements of the metamodels.

MDE (Model-Driven Engineering) and MDD (Model-Driven Development) offer the possibility of representing requirements through models applied to the domain worked on [5]; one of its branches is the model transformation chains, this is an alternative to explore its application in the web application generation domain and study if it is possible to perform semi-automatic code generation as the base of software that must be completed with developments individuals to fulfill the acceptance criteria of the requesting user.

Model Driven Engineering is a proposal that has been worked on for several years, the use of models as a fundamental axis throughout the life cycle of a software project reduces development time and effort. The purpose of this work is to build a prototype in Eclipse the source code base of web applications by creating a metamodel for the representation of a specific domain model.

The work was done taking into account the following work structure context, evolution, web applications, case study through a real context, and a first approach to the construction of the domain metamodel for the specific case study. This document will show the implementation of the Ecore metamodel developed and organized in the following way: 2. Context, 3. Evolution, 4. Web applications, 5. Case study, and 7. Conclusions.

2. Context

By modeling, we can abstract an image or model of an object that represents some aspect of reality. The meaning of modeling [6] "In the widest sense, it is the profitable use of something instead of something else for some cognitive purpose. We are allowed to use something that is simpler, safer, or cheaper instead of reality for some purpose. A model represents reality for the given purpose; the model is an abstraction of reality in the sense that it cannot represent all aspects of reality. This allows us to deal with the world in a simplified way, avoiding the complexity, the danger, and the irreversibility of reality".

For this reason, models are considered important because they allow different levels of abstraction to be represented, particularly for the emphasis of software engineering, transformation chains are important as a strategy for software development projects, chains allow models to be transformed until the desired product is obtained.

- **Transformation chains:** That is, models are generated from the most abstract to the most concrete through transformation steps and refinements until reaching the final code by applying the last transformation. In general, it can be said that a transformation definition consists of a collection of rules, which are specifications of the ways in which a model (or part of it) can be used to create another model (or part of it) [7].
- **Refinement:** It is a semantically correct transformation that captures the essential relationship between the specification (i.e., the abstract model) and the implementation (i.e., the code). Refinement is usually verified by showing that the concrete system simulates the abstract system. [8].

In this case, MDE models are important to perform the transformation chains, for example, IBM Rational Software Architect Designer is a tool defined by MDE, IBM Rational Software Architect Designer¹ (RSAD, and previously called RSA) is a complete design tool, modeling, and development for end-to-end software delivery.

It uses the Unified Modeling Language (UML) to design enterprise Java applications and web services. Rational Software Architect Designer is based on the Eclipse open-source software framework and can be extended with various Eclipse plugins. You can also enhance functionality by tailoring it to your specific requirements with separately purchased Rational extensions.

In different contexts related to the development of software products, it is possible to see that, most of the functional requirements that are represented around the same context, respond to the same set of characteristics [9]. In general, software developers reuse code generated in previous projects that solves certain requirements to improve the development time of a particular product. This code reuse is an immediate solution that does not necessarily solve the problem in a new project due to the specific situations in the new project. These situations in almost all cases have differences from the application made in previous projects; however, this type of solution requires a large amount of time and effort on the part of the development team. In this way, a better alternative is to build metamodels that allow the abstraction of the general characteristics of a particular context to carry out transformation processes, it would be enough to make a design for a new metamodel based on the context, which would apply different transformations necessary to provide results in the source code [10].

For instance, Virtual laboratories (VL) are a computer representation of traditional laboratories and allow experiments, research, and academic and scientific practices, giving the sensation of their real existence; supporting and promoting student learning, and increasing the options for experiments available in educational institutions. Despite the pedagogical advantages that LV have, it has been identified that some of these tools do not comply with the standardization that allows their interoperability, reuse, and accessibility. This causes, among other things, the creation of non-reusable applications and the investment in time and cost that this entails. As a possible solution to these problems, the use of model-driven development (MDD) is adequate, since the models allow to increase the level of abstraction and reuse; as a software specification and design tool, they allow to simplify tasks, analysis, detection of problems in the early stages of development and independence from platforms, technologies, or implementation languages.

MDD and its Model-Driven Architecture (MDA) are accepted approaches to software development, which propose the use of models in all its phases. Model transformation is the foundation of MDA, starting with a Platform-Independent Model (PIM) and aiming to achieve Platform-Specific Models (PSM).

Figure 1 presents the representation by means of a conceptual map with the context of the related components in the whole software development cycle. In the concept, the map is highlighted in blue with the components directly related to the path that must be taken into account using model transformation chains to reach the final result that is the basis of the source code of a web page.

¹<https://www.ibm.com/products/rational-software-architect-designer>

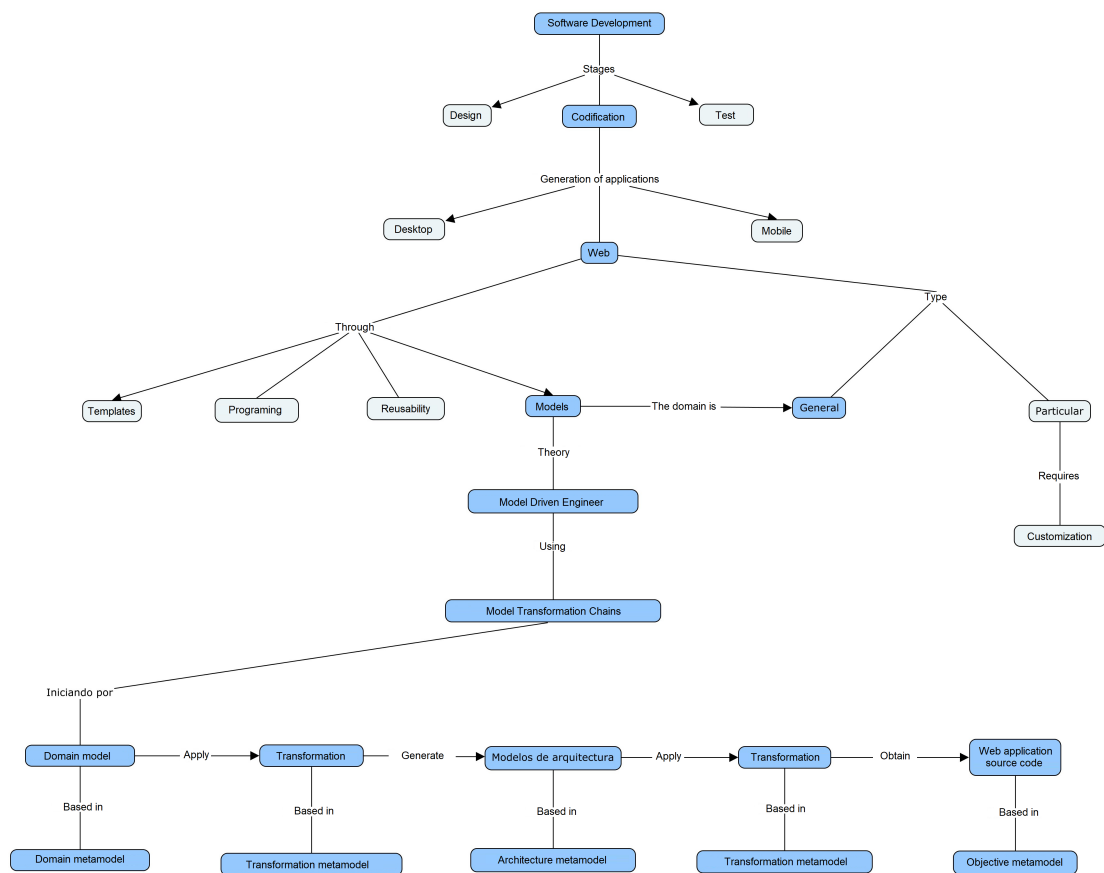


Figure 1: Conceptual map for the construction of a web app from a model transformation chain.

3. Evolution

In recent years, the methodologies and strategies with which the possibility of building applications with particular characteristics have evolved.

Some of the most relevant ideas that can be included and have an impact on the development of content and applications running on the Internet are:

- Use of special glasses in which a layer of virtual reality would be superimposed on physical reality.
- We will be able to dialogue naturally and online with an intelligent virtual agent. Through it, banking or electronic commerce operations can be carried out.
- An Internet that, together with tactile devices, will offer a complete sensory reality and allow almost real virtual experiences thanks to 3D.
- The internet will be integrated into the vehicles.
- Internet in which there will be neural implants with direct access to the Internet that

will improve “higher brain functions such as memory, learning speed and intelligence in general”.

Model Driven Engineering provides the possibility of using model transformations that allow any reality to be represented and displayed as desired in a final product, regardless of the existing technological advances. Model Driven Engineering will be able to represent, implement and launch a web application that meets the requirements given by the domain on which it is working.

The academic and industrial community will be able to make use of model transformation chains that not only allow the export of a web application that fulfills the established acceptance criteria but will also be able to adjust the necessary components of the model for its maintenance at the pace of technological progress, which favors the development cycle according to the methodology used in the construction and the developer’s productivity.

4. Web Applications

In software engineering, web applications are those tools that users can use by accessing a web server through the Internet or an intranet through a browser. In other words, it is a software application that is coded in a language supported by web browsers that the browser is trusted to execute.

A Web application is provided by a Web server and used by users connecting from anywhere via Web clients (browsers). The architecture of a website has three main components:

- A web server
- A network connection
- One or more clients

The Web server distributes pages of formatted information to clients that request them. The requests are made through a network connection, and for this, the HTTP protocol is used. Once this request is requested through the HTTP protocol and received by the Web server, it locates the Web page in its file system and sends it back to the browser that requested it.

Figure 2 shows the components required for the generation of the domain metamodel for the context of Web applications. It presents an initial prototype for a model that allows the abstraction of the minimum objects required to obtain the proposed metamodel in the context of a web application. Figure 2 shows the components required for the generation of the domain metamodel for the context of Web applications.

5. Case study

Building web applications is one of the most explored fields of work in software development. The industry has components such as frameworks, development environments, and complements for its implementation and execution; each of these with specific purposes and is related to the purpose of the application to provide the expected result. Currently, web applications

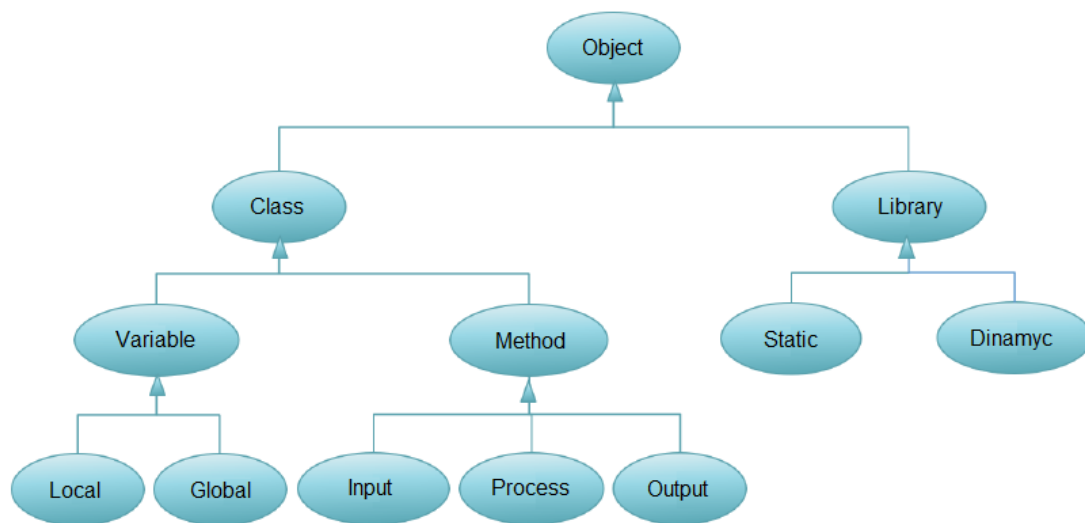


Figure 2: Elements of the domain metamodel for web application

are developed with very similar objectives, however, their construction must be carried out independently, because each one must maintain its structure and/or standardization that will be evident at the time of generation and visualization of the web applications of project executable.

The different alternatives offered by technology can compromise and force the developer to create code with the use of certain components or execution environments of the final products, which in our case are web applications.

For example, consider that 3 academic registration web applications are generated that retain the same business logic as a base, however, one is built to be used in a public university, the second in a private university, and the last in secondary schools. For its development, each work team will have to build an application that satisfies the acceptance criteria of its respective requirement, allowing the student registration process. The solution of the case is to create a domain metamodel that allows generating the three tools, each one with its particular characteristics and with different styles, allowing to create the source code base for the development of the web page, creating components such as User, Login, Student, Course, Subjects, Notes and the generation of each page corresponding to the objects required by each work team. The three cases must generate the necessary code to manage the base objectives that are the same for all cases and subsequently customize to solve the specific requirements of the development; which means that the code generation is being carried out three times for the same purposes, where it was possible to use a single development and invest effort focusing on the particularity of each case, further minimizing the total time of software construction.

MDE offers the possibility of representing software requirements extracted from reality and displaying them by means of models with which the business logic applied to the domain being worked on is evidenced. For the specific domain of web application generation, it should be clarified that code generation is semiautomatic, the resulting program is the base of an application to which a series of customization is applied according to a requirement in a specific

context. The problem to pose is focused on the duplicate creation of commands that perform the same functions under the same contexts, the only thing that must be done is to relate and represent a domain with which the base or initial code is extracted for the general field worked and simply focus in giving the added value that characterizes the specific field.

With the clarity that the representation of a situation or a requirement of the real world is expressed through models, MDE offers the possibility of performing transformations of models that logically must be linked to a metamodel. In our case, more than one transformation is required to obtain the proposed result, which must be carried out sequentially and that is why they are called Model Transformation Chains.

Currently, society has integrated with the new trends in technology, establishing a total relationship. Every day millions of people seek to solve many problems through the Internet, finding different solutions provided by Web applications.

Web applications are functional tools, coded in a language that can be executed in a browser. For the construction of web applications, it is necessary to have specific knowledge regarding the programming language and each of its components and to guarantee compatibility with all available browsers and software that must be considered in the construction of web applications.

It is possible to find many advantages that the standardized development of web applications has, such as saving development time, compatibility would not be a problem if the browsers are up to date and with the necessary components for their execution, the availability of a web page from any device with a browser and above all the power of these applications to process data securely.

6. Domain Metamodel for the representation of Web applications

For the development of the transformation chain software-based, the process begins with the creation of a domain metamodel MDE. The initial proposal is the creation of a domain metamodel that includes concepts related to the specific domain entities to be resolved, which would abstract the different elements that can have a solution for the domain.

This paper presents the process of building a domain metamodel that is the basis of the context to be developed in the final project, which is the semiautomatic generation of Web applications. For the domain model, the construction of a metamodel defines the abstract syntax of the language that is developed, that is, the basis of all the artifacts that are present in an application.

For the construction of the domain metamodel, EMF (Eclipse Modeling Framework)² is used, which is a modeling framework and a code generation resource to build tools and other applications based on a structured data model. Starting from a model specification described in XMI, EMF provides tools and runtime support for producing a set of Java classes for the model, along with a set of adapter classes that enable command-based viewing and editing of the model and a basic editor.

- The EMF metamodel consists of two parts: the Ecore and genmodel description files.

²<http://www.eclipse.org/modeling/emf/>

- The XML Ecore file contains the information about the defined classes.
- The Ecore file allows for defining Classes, Attributes, References, and DataTypes.

6.1. Ecore Metamodel

Ecore is the framework proposed by the Eclipse community that is a simplified version of MOF. Ecore metamodels are stored in XML files with the extension `.ecore`. Under this framework, models can be created and modified from an Ecore metamodel, there is also support for managing model persistence through XML serialization and an API for object reflection.

Using Ecore as a foundational metamodel allows a modeler to take advantage of the entire EMF ecosystem and tools, to the extent that it is reasonably easy to map application-level models back to Ecore [12]. This does not mean that it is good practice for applications to take advantage of Ecore directly as their metamodel, but they could consider defining their own metamodels based on Ecore.

Figure 3 is a general representation of the Ecore components and how they are related. The main components of Ecore are the following:

- EPackage: it is a component that allows organizing classes and data types.

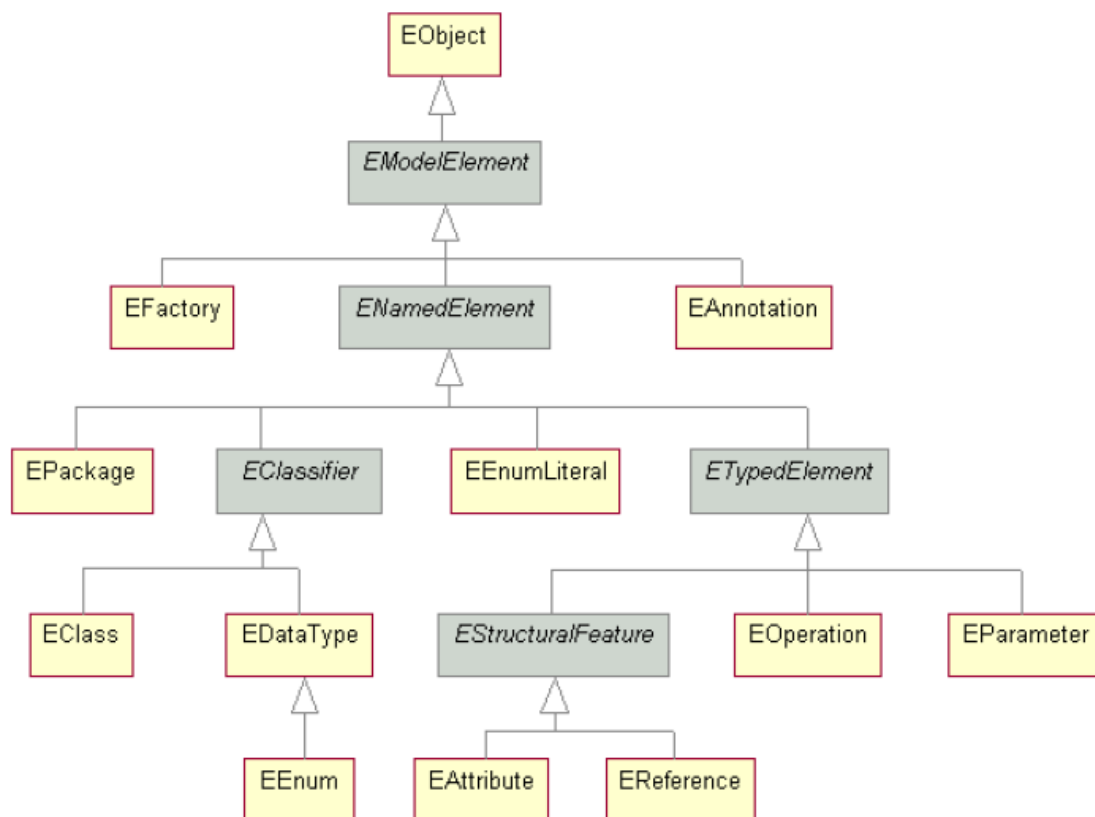


Figure 3: Ecore Components [11]

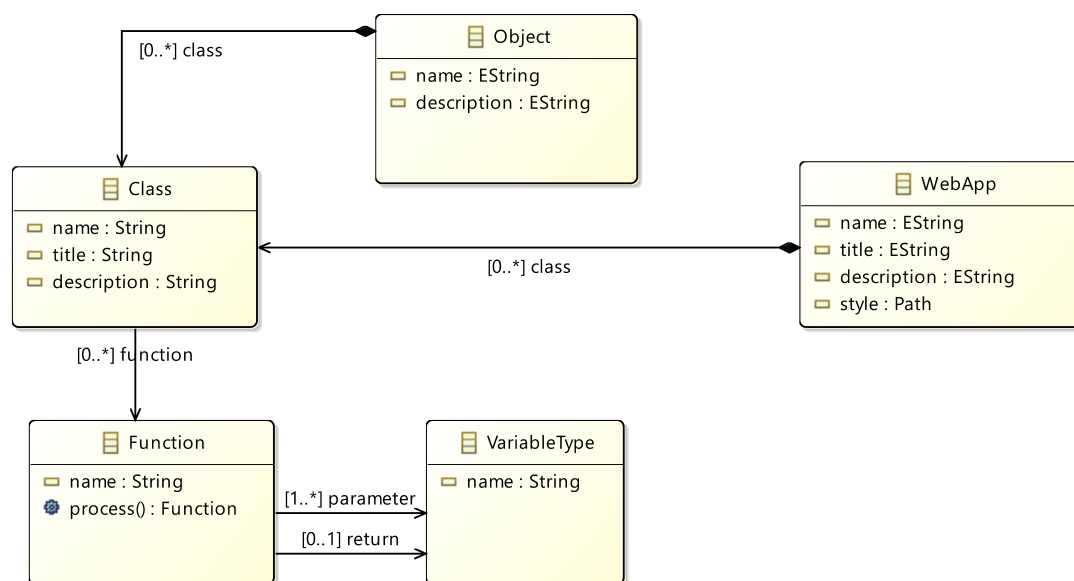


Figure 4: EMF App Web Metamodel

- EClass: allows defining concepts in the metamodel
- EReference: allows defining association between concepts
- EAttribute: allows defining properties of the concepts
- EDataType – defines a type for an attribute such as EString, EInt, EDouble, etc.

Figure 4 represents the metamodel created using the Eclipse Modeling Tool, in which we perform the following tasks: the creation of an EMF modeling project and Java code generation, for the metamodel editor. This metamodel contains the following EClass information:

- Class – EClass – Object – WebApp System – Etype required
- Class – EClass – Class – Page type – Etype required
- Class – EClass – WebApp – Object Page show – Etype required
- Class – EClass – Function – Process method – Etype required
- Class – EClass – VariableType – Parameter-Return – Etype required

The entities are generated from the Ecore model, this process is iterative, that is, if elements of the model are changed, added, or removed, one can regenerate the entities again. To generate the entities, the generator model must first be created. This model allows setting various properties before generating the Java code that allows manipulating the entities and creating a simple editor for the model. Also, can be indicated that the generated code is created as an Eclipse Plugin project with a series of folders and Java packages. In the project explorer, the previously mentioned projects must be created as shown in Figure 5

Finally, in the src folder of the EMF project, the metamodel code is generated, taking into account that the base package must be renamed to keep proper control of the sources generated by the metamodel created.

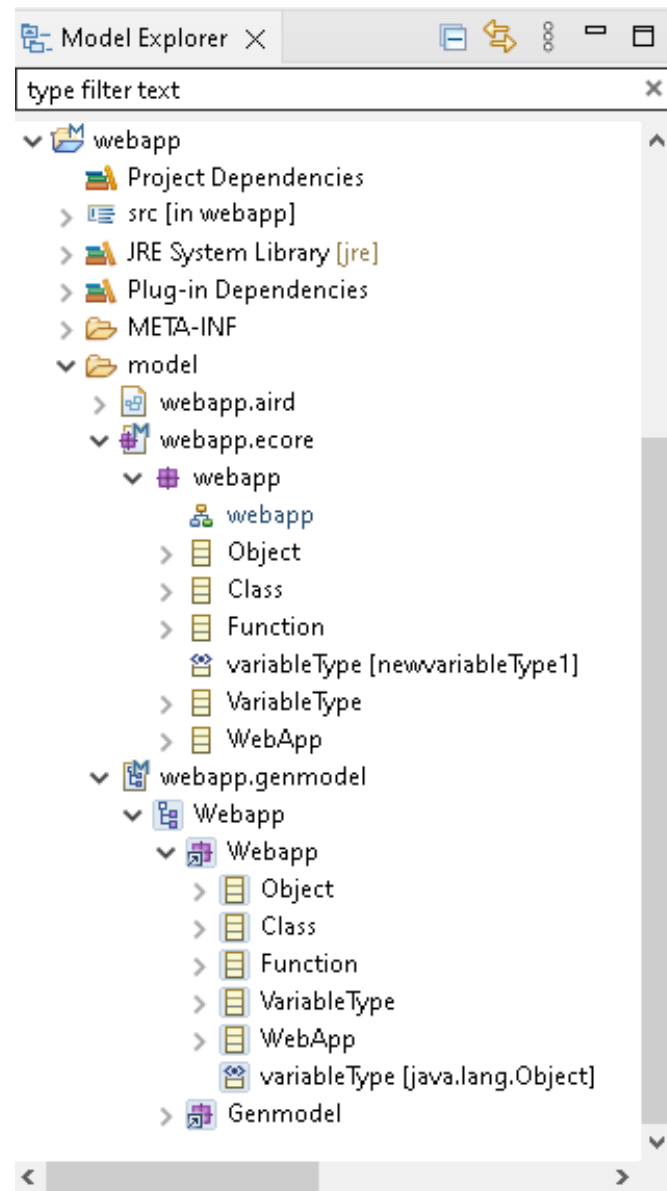


Figure 5: Project Elements

7. Conclusions

Software development has had a very rapid advance in recent years, where all kinds of technologies are involved to obtain quality and a result that meets all the acceptance criteria of a user. In the case of web applications, different alternatives have emerged to obtain the expected results, among the most relevant are the use of templates, the programming of customizations to existing systems, and the reuse of code; however, all these strategies require an extra effort

to meet the goal.

The use of a model transformation chain allows for minimizing the construction time of applications that are under the same context, which allows focusing efforts on software analysis and design. With the implementation of a transformation chain, it is possible to semi-automatically generate a final product which in this specific case is the source code of a web application.

To build models it is necessary to express them in some way. Modeling languages allow capturing the information or knowledge of a system or structure defined by a set of consistent rules. These rules are used to interpret the meaning of the components of said structure. Creating a domain-specific language is worthwhile when it allows a particular type of problem or solution to be expressed more clearly than other existing languages.

The Eclipse Modeling Project focuses on the evolution and advancement of model-based technology development within the opportunities provided by Eclipse, providing a unified set of modeling frameworks, tools, and implementation standards.

With the work done using the EMF framework and more specifically the metamodeling with Ecore, it is possible to show the construction of our core of the final case study, which is the generation of web application source code through transformations. Although the developed components are the initial ones, they serve to introduce us to the world of the MDE in a practical way, to later apply the results of the research required for our general objective.

Taking into account the case study proposed for the construction of the domain metamodel, it is planned to maintain this scenario in the subsequent phases of the model transformation chain that is being built as the objective of this project.

References

- [1] G. Martínez Villalobos, G. D. Camacho Sánchez, D. A. Biancha Gutiérrez, *Diseño de framework web para el desarrollo dinámico de aplicaciones*, *Scientia et Technica* 16 (2010) 178–183.
- [2] A. Velasco, J. Aponte, *Automated fine grained traceability links recovery between high level requirements and source code implementations*, *ParadigmPlus* 1 (2020) 18–41.
- [3] M. Fischer, F. Imgrund, C. Janiesch, A. Winkelmann, *Strategy archetypes for digital transformation: Defining meta objectives using business process management*, *Information & Management* 57 (2020) 103262.
- [4] H. Florez, E. Garcia, D. Muñoz, *Automatic code generation system for transactional web applications*, in: *International Conference on Computational Science and Its Applications*, Springer, 2019, pp. 436–451.
- [5] B. Nuseibeh, S. Easterbrook, *Requirements engineering: a roadmap*, in: *Proceedings of the Conference on the Future of Software Engineering*, 2000, pp. 35–46.
- [6] J. Rothenberg, L. E. Widman, K. A. Loparo, N. R. Nielsen, *The nature of modeling*, *Artificial Intelligence, Simulation and Modeling* (1989).
- [7] H. Florez, M. Leon, *Model driven engineering approach to configure software reusable components*, in: *International Conference on Applied Informatics*, Springer, 2018, pp. 352–363.
- [8] C. F. Pons, R. S. Giandini, G. Pérez, *Desarrollo de Software Dirigido por Modelos: conceptos*

teóricos y su aplicación práctica, Universidad Nacional de La Plata. Facultad de Informática, 2010.

- [9] H. Florez, Model transformation chains as strategy for software development projects, in: *The 3rd International Multi-Conference on Complexity, Informatics, and Cybernetics: IMCIC 2012*, 2012, pp. 1–10.
- [10] D. Sanchez, H. Florez, Model driven engineering approach to manage peripherals in mobile devices, in: *International Conference on Computational Science and Its Applications*, Springer, 2018, pp. 353–364.
- [11] Eclipse, Ecore components, <https://download.eclipse.org/modeling/emf/emf/javadoc/2.9.0/overview-summary.html>, 2012.
- [12] P. Gómez, M. Sánchez, H. Florez, J. Villalobos, Co-creation of models and metamodels for enterprise architecture projects, in: *Proceedings of the 2012 Extreme Modeling Workshop*, 2012, pp. 21–26.