

# Reengineering of a data processing line with Platypus<sup>1</sup>

Alain Plantec, Vincent Ribaud

LISyC, EA3883, Université de Bretagne Occidentale, Brest

{[alain.plantec](mailto:alain.plantec@univ-brest.fr), [vincent.ribaud](mailto:vincent.ribaud@univ-brest.fr)}@univ-brest.fr

## Abstract

This article presents the use of Platypus for the reengineering of a data processing line dedicated to the mechanical study of cod-ends of fishing nets (PREMECS II EU project). Platypus is a STEP-based meta-environment. The STEP technology is used for meta-models and models management.

In this reengineering case, Platypus helps to the development of a specific meta-environment. The meta-environment is exploited in order to rebuild the application and also to produce semi-automatically similar applications for a related domain.

## 1 Introduction

It is common that a software developed for a quite precise field undergoes evolutions aiming at meeting the needs for close or related fields. Gradually one observes an increasing difficulty for the maintenance of the software. The problem, which, although clearly described since the Seventies [Parnas76], remains one major concern in software engineering. The solutions currently considered insist on the importance of the models and place domain-management in the centre of the debate.

This article presents the work done for the reengineering of a scientific application arrived at the end of its maintainability. The solution under development considers right away a multi-dimensional approach: the envisaged process consists in developing a meta-environment dedicated for the implementation of a family of applications.

This article is organized as follows. Section 2 exposes the problems. Section 3 presents the two-staged process of reengineering, the first one developing a prototype of replacement, the second one building the meta-environment. We also briefly describe the Platypus environment. Related work and open issues are drafted.

## 2 Problems

### 2.1 Context

The PREMECS II EU research project [<http://www.ifremer.fr/premecs>] develops a predictive selectivity model of cod-ends. The objective is to envisage the ability of the commercial trawls to capture the fish for which they are intended. A cod-end fish net is modelled by a discrete finite network of meshes. Meshes are made of several directions of twines.

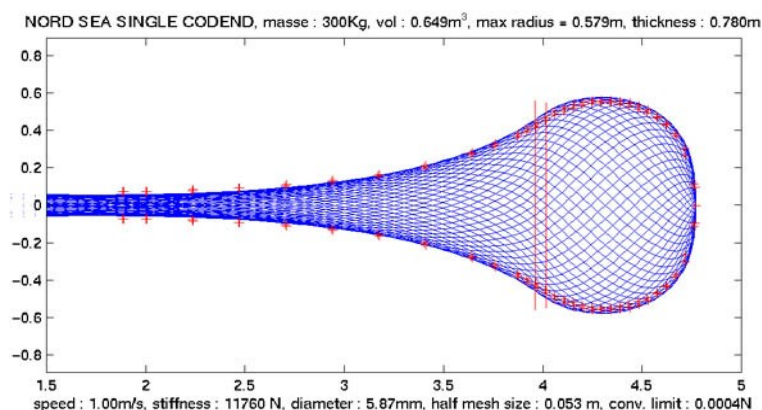


Illustration 1: Model of a cod-end fish net

<sup>1</sup>This work was partially funded with the contract Ifremer-UBO n° LLM2 « Evaluation d'une méthode de rationalisation de l'accès aux données au sein du projet PREMECS II »

Hence, a data processing line of C programs was developed by D. Priour, the PREMECS coordinator, providing: the modelling of the trawls as 3D-structures, numerical simulations and trawl visualizations. The modelling of structures is carried out by the specification of basic elements (such as panels or cables). These elements are decorated by values of mechanical properties, such as twine and mesh parameters.

The existing software is in continual evolution which can be considered according to two levels:

- for the same type of application, various ways of modelling are tested. The variations relate primarily to the properties of the studied models;
- although dedicated to PREMECS project, the developed software was already or will be exploited for other similar applications such as modelling of structures for the aquaculture or floating booms.

From an operational point of view, each version of application can be used by several stakeholders of the PREMECS project.

On the one hand there is a need for a version management of stable applications, and on the other hand a need for the ability to evolve stemming from the nature of the scientific investigations. From the point of view of the software engineering, these two needs can come into conflict. Although the use of object-oriented technology makes it possible to substantially improve quality of such a development, a traditional process of implementation working with components managed in configuration around a single application does not allow the complete stabilization of a version and the satisfaction of the continuous need for evolution.

## 2.2 The envisaged process

The envisaged process consists in developing a meta-environment dedicated to the implementation of a family of applications. The domain of this family is the mechanical study of roped structures. Each application of this family implements a sub-domain such as fishing nets. The meta-environment exploits code generation from object models to produce semi-automatically new versions of applications. The definition of code generators is carried out after expertise and clear definition of the target domain and its instrumentation by/for the users.

In order to set up the meta-environment, we propose a process of reengineering in two stages:

- the first stage aims to build a prototype of replacement of the current solution while being strictly guided by the direct needs of PREMECS; the objective is to have a complete and generic toolkit for the modelling and the 2D and 3D visualizations of fishing nets, the archiving and the data exchange; we have, with this hand-made version, a *specimen* for the sub-domain of cod-ends;
- the second stage consists in setting up the meta-environment specialized for the handling of structures which, starting from a specification of a sub-domain, produces a dedicated PREMECS-like environment; at the end of this stage, one must be able to regenerate a part of the first specimen but also a part of alternatives or applications for other sub-domains.

## 3 The meta-modelling in the heart of the reengineering

### 3.1 The first stage

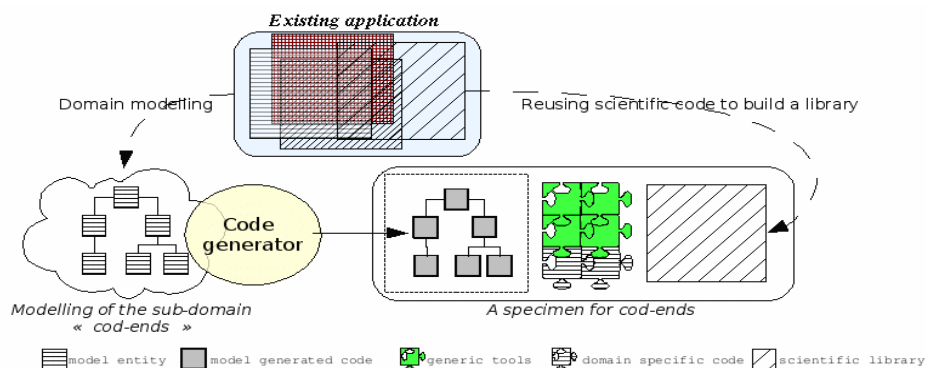


Figure 1- First stage : rebuilding a specimen for cod-end

For this first stage, the idea is, initially, to centre itself on the sub-domain of cod-ends. The objective is to have an application rebuilt around a formal model of the sub-domain. This application mixes generated Squeak code and the C-programmed processing.

As shown in figure 1, the reengineering of the application proceeds first of all of the modelling of the sub-domain. Cod-end models are specified in the formal language EXPRESS. They are exploited to automatically generate a part of the application. Scientific and numerical routines are reorganized into libraries. A set of generic tools (a structured editor of cod-ends, a 2D-editor and an export module towards VRML format for 3D-visualizations) complete the application.

The major part of the components developed for the cod-ends are reusable in very close applications. All these applications share the domain of the definition and 2D and 3D visualization of roped structures (3D-structures assembling 2D-nets). Hence, this first version constitutes a specimen of application in this domain, regarded as the specimen of reference. Delivered version at the end of the contract lacks a mechanism intended to link 2D models with scientific libraries used to compute 3D representation of fishing net. This mechanism is under development.

### 3.2 Work in progress : the second stage

The objective of the second stage is to provide an answer to the evolution need according to both levels described in section 2. The idea is to provide an environment of code generation or *meta-environment* making it possible to automatically generate specimens of a PREMECS-like application.

As shown in figure 2, the role of the meta-environment is to produce the specimens. A specimen corresponds to a version of application for a particular sub-domain. The various specimens of the same sub-domain constitute a line of specimens. The generated specimens of applications thus differ along two axes:

1. the first axis is that determined by the sub-domain;
2. the second axis corresponds to the various versions which one can produce for a particular sub-domain by updating the way of modelling for this sub-domain (add/update/delete elements of structure and their properties).

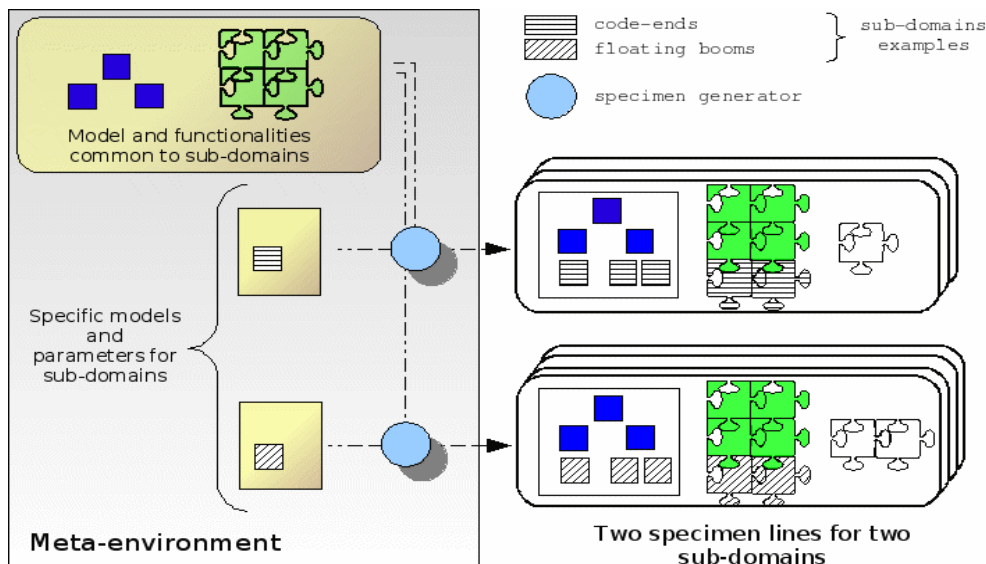


Figure 2: Using the meta-environment

All the specimens, for a certain part, are made with common code. For the same version of meta-environment, this common code is identical from a specimen to another and is managed by the meta-environment. A part of this common code is very strongly related to the handled data (the domain model) and another part with the functionalities of the applications (2D/3D-visualization ...).

For the code generation, the meta-environment is parametrized with a variable part, which corresponds to

the differences from a specimen to another. The variable part of a sub-domain is declared in a specific model. This model contains specialized meta-entities and their code production rules.

A part of the code of a specimen is produced starting from the variable part. This part of code supplements the common code with:

- the classes related to the sub-domain model,
- the specialization and the parametrization of the code implementing the common functionalities.

### 3.3 Working environment

Platypus [<http://cassoulet.univ-brest.fr:8000/Platypus>] is a meta-case tool which embeds within a Squeak system, a meta-modelling environment based on the STEP standard [ISO10303]. The meta-models are specified with the EXPRESS language. In a meta-model, the entities specify the definitions of the concepts of the domain. The constraints specify their semantics locally or globally. The code production rules are expressed by derived attributes.

Platypus provides a semantic analyser and an interpreter for EXPRESS which allow a STEP validation. The lexical and structural validations are carried out by the analysis of the meta-model and the evaluation of the rules. For the semantic validation, the expert of the field works on reports analysis and the computing result of the derived attributes.

During the first stage, a meta-model of a specification language for fishing net has been designed with Platypus. Meta-data management Squeak code of the specimen has been generated with Platypus.

Squeak [<http://www.squeak.org>] is a complete and free Smalltalk system. It is instrumented by a very rich development interface and a set of tools like code browsers, a compiler generator, a version manager.

The meta-environment of a domain (second stage) will be itself a specialization of the Platypus environment. Hence meta-models used for code generation are specialization of the EXPRESS meta-model. Thus, tools of the Platypus framework (taxonomy browser, structured editor, analyser, code generator ...) can be reused and specialized.

## 4 Related work

The generative approach is largely exploited for the reengineering of application. Two kind of approaches are currently employed: model-based approach (mostly relying on UML) and domain-oriented approaches such as tools using UML stereotypes or solutions based on the specification of meta-models (Meta-environment [MetaCase04], DSL [CLRC05]). All these approaches are multi-language: several notations are exploited to model, express the constraints or to specify the production rules of code.

Comparatively, Platypus uses jointly the two approaches and its technology is mono-language. The models, the meta-models, the constraints of integrity and the production rules of code are specified with EXPRESS.

## 5 Open issues

*Framework evolution* - By construction, a domain-oriented environment relies on a fixed part : abstract entities which one finds in all specimens. When the fixed part evolves, object-oriented specialization is not sufficient and a way of mastering abstract entities evolution is lacking.

*Understanding meta-model structure and constraints* - The final user of the meta-environment may find difficult to design models conform to the meta-model constraints. Rules are expressed at the meta-model level and constraints violation refers to meta-concepts and generic terms hard to understand.

*Impedance mismatch* – Specimens can be completed with extra-code (the white puzzle pieces in fig. 2) that has to be plugged with the environment. Generic adaptation techniques are required.

*Acceptance* – Meta-techniques are few used. What are the educational and technical challenges to promote the meta-modelling and meta-programming approach?

## 6 Conclusion

This experience reports presents a work-in-progress about the reengineering of scientific applications based primarily on the modelling of the domain and the automatic generation of code.

This reengineering consists of two stages: the first stage has provided the specification and the implementation of a specimen of application dedicated to the re engineered domain; the second stage aims to provide a meta-environment for the semi-automatic building of lines of specimens for similar sub-domains.

## 7 References

[CLRC05] C. Consel, F. Latory, L. Réveillère, P. Cointe. A Generative Programming Approach to Developing DSL Compilers. Fourth International Conference on Generative Programming and Component Engineering (GPCE), Tallinn, Estonia, September 2005. Springer-Verlag, LNCS 3676.

[ISO10303] ISO 10303-1. Part 1 : STEP overview and fundamentals principles, ISO, 1994.

[MetaCase04] MetaEdit+ Method Workbench 4.0 User's Guide, [www.metacase.com](http://www.metacase.com), 2004.

[Parnas76] D. L. Parnas. On the Design and Development of Program Families. IEEE transactions on software engineering, Vol. SE-2, N° 1, March 1976.