# Early Definition of Frozen and Hot Spots in the Development of Domain Frameworks

Simone Nasser Matos
Computer Science Dept.
Federal Technological University of Paraná and
Aeronautics Institute of Technology
+55 42 32204827
simone@pg.cefetpr.br

Clovis Torres Fernandes
Computer Science Dept.
Aeronautics Institute of Technology
+55 12 39475983
clovistf@uol.com.br

## ABSTRACT
In this work, we present a research project in which a set of responsibilities are used to determine the frozen and hot spots in initial phases of the domain framework development process. The purpose is to provide the framework developer with domain understanding earlier than expected, as well as with opportunities for reusing software artifacts from the example applications.

## Categories and Subject Descriptors
D.2.1 [**Requirements/Specifications**]: Methods.

## General Terms
Design.

## Keywords
Determination of Frozen and Hot Spots, Domain Frameworks, Responsibility-driven Method, Reuse of Model

## 1. INTRODUCTION
A framework is a group of subsystems, components and subframeworks, which can be extended and adapted to a particular domain. They may be classified as application, domain, and support frameworks [9]. While application and support frameworks are basically concerned about internal problems of software development, domain frameworks have as objective to support the development of application directed to users and products in specific domains, such as games, manufactures etc.

The main part of the domain framework design consists of determining which points must be classified as frozen or stable spots or as hot or flexible spots [5]. Frozen spots correspond to aspects that are common to all software products in a domain, whereas hot spots represent aspects that are specific to just some software products in a domain.

Several approaches are found in the literature with the purpose of helping the process of determining those points. In order to carry out this study some of them were examined [3, 5, 2]. Those approaches usually contemplate activities in which frozen and hot spots of a framework must be identified directly from the class structures already or from code fragments of the sample applications. Both ways of identifying spots start much later in the life cycle, once the spots will only be identified when the classes are already in existence, or they are associated to a class structure or they are already codified. As a result, a delay at the understanding of which spots should be kept stable or flexible usually occurs.

How to provide the framework developer with domain understanding earlier than expected, as well as with opportunities for reusing software artifacts from the example applications? How to determine frozen and hot spots in initial phases of a development process of domain frameworks?

## 2. RELATED WORKS
For Johnson [3], Pree [5] and Silva [6] frozen and hot spots are identified through the class model of the example applications under analysis for developing the framework. These kinds of identification are started much later in the life cycle, once the spots will only be identified when the classes are already in existence and are associated to a class structure.

According to Braga [1], in order to find frozen and hot spots it is used information present in a pattern language. The considered elements are the following: Context, Problem, Forces, Structure, Participants, Example, Following Patterns. Braga offers a process for identifying the frozen and hot spots in which it is analyzed the graphs of the example applications for identification and classifying the patterns, which can be optional or obligatory, but do not specify which characteristics of the graph should be analyzed to infer its classification.

The works of Landin and Niklasson [4] and Ben-Abdallah et al. [8], identify frozen and hot spots during the phase of analysis. It is also emphasized that those works utilize the reuse in the analysis phase, for it allows increasing consistency during the construction of use case, class, and sequence framework model.

For Landin and Niklasson [4], the idea is that all the requirements of the example applications be found. Right after, all these requirements are analyzed, dividing them in the following categories: functional and non-functional requirements of framework, and functional and non-functional specific requirements for each example application. From the requirements classified it is possible to identify the classes that will satisfy it.

Ben-Abdallah et al. offers a process for determining frozen and hot spots from UML use case, class, and sequence diagrams, in which it is analyzed the name of the use case and classes of the example applications in order to build the structure of the framework [8]. From the analysis of requirements and classes names it is built the requirements and class model of the framework. The comparison of names is based in four situations

as following: identical or synonym, variation of a concept, generalization of the specific names and, finally, they may be classified as distinct. The first situation represents the frozen spots and the others the hot spots.

In the work of Ben-Abdallah et al. it is not established how a requirement classified as frozen and hot spots will be placed in the class diagram, that is, the unification process is made separately. It was also observed that the analysis by name of use cases many times may lead to a wrong classification of frozen and hot spots, since it is analyzed just one element of the requirement. For instance, when we have a requirement *Present Options* and another *Show* from different example application, the projectist when analyzing this requirement may infer that its classification will be synonym, but they represent in fact different functionalities.

Hanenberg et al. [2] states that, by means of the join points of aspect-oriented programming, some hot spots may be determined. The join points are the very specific ones of the execution of a system. The definition of the spots is made by analyzing the code and details of implementation. Another point to be observed concerning the proposal of Hanenberg et al. is that this may make the work of the framework developer more difficult, since each programmer has a specific way to codify, which may require a longer period of time and considerably more effort for understanding the artifacts codified.

# 3. THE RESEARCH

## 3.1 Brief Description of the Research Proposal

Determination of frozen and hot spots in initial phases of a domain framework development process by using the concepts of responsibility, set theory and functions.

## 3.2 Importance of the Research Problem in the Field

This work presents a responsibility-driven method, called MDB-FHS, that aims to identify frozen and hot spots in initial phases of the domain framework development process. The purpose is to define most of the spots earlier, from the artifacts of the analysis phase of the sample applications used as development basis of domain frameworks, for instance, from use case diagrams.

The goal is to increase the reuse of the analysis artifacts of sample applications and perfect the precision in identifying the spots, as well as to contribute for the creation of a requirements and class model of the framework. As a result, more time is available for the refinement of the framework in the subsequent phases, including the definition of new spots obtained usually from the class structures.

The development of this research will fulfill the demand for a method that allows the developer to have a more precision for the identification of frozen and hot spots, analyzing the set of responsibilities, besides providing guidelines for the developer to create the requirements and class model of the framework.

It also offers a set theory-based model and it may be implemented. Thus, it is offered a semi-automatized process that may facilitate the spot classification. Another advantage of the method is that it may be executed iteratively, being possible to create a requirements model for different subsystems concomitantly and allowing the segmentation of the activities.

The method is based in the reuse of the example application models and artifacts obtained in the analysis phase, which may increase the consistency during the creation of requirements and class models for the framework.

## 3.3 Hypotheses

The hypotheses raised for the research project proposed are the following:

- The definition methods for frozen and hot spots that have as main reference the name of the requirement or class to accomplish its classification may be substituted by the analysis of a set of responsibilities, which might improve the precision of the classification process.
- The manual comparison process may be substituted by a semi-automatized classification process. It is expected that this process will facilitate the spot classification.
- The classification for the classes does not depend on the previously classified requirements. So, it is possible to substitute this process by a dependent one in which each responsibility will be associated to a method. This will facilitate the work of determining if the classes will be of stability or flexibility ones.

# 4. PROPOSED SOLUTION

The main objective that has guided this research is to face the problems of the current literature approaches, especially in the following problematic points:

P1) Analysis by name of use cases and classes for the identification of frozen and hot spots.
P2) Process not dependent on the previously classified requirements to determinate the classification of classes.
P3) Analysis just of class diagrams already prepared.
P4) Absence of an automated process.
P5) Non-existence of a method in the literature that keeps the data of the requirements and classes, obtained during the analysis to build the framework. This may be a major difficulty for a posterior work of improvement in the model.

After studying the matters listed above, it was verified that the concepts of the Responsibility-Driven Design (RDD) [7] and set theory could help solving the problems. From the union of these concepts was created the method MDR-FHS (Responsibility-Driven Method – Frozen and Hot Spots), which may be applied to the initial phases of the framework elaboration process.

The method MDR-FHS supposes that all the responsibilities that will be analyzed have gone through a pre-processing, where semantics problems were solved. It allows classifying sets of use cases into three categories: equals, different and similar. The sets found represent the probability that the sets of responsibilities belong to the equals, different and similar categories. Despite this probability, it is still necessary that the projectist analyze the sets obtained during the construction of the requirements and class model of the framework, for it may occur cases in which they were classified as equal but are homonym, or classified as different but are synonym.. The advantage of utilizing the proposed method is that more than one functionality is analyzed – responsibilities – of the requirement for its classification, through

a semi-automatized process for the implementation of the procedures presented in Figures 1 and 2.

Considering the procedure in Figure 1, the difficulty of making a more exact classification of the requirements is given in the item e.), in which it is compared the set of responsibilities and not one element that is the requirements name, solving the point P1). It is also possible to realize that the analysis will initially occur from the requirements and not from the class diagram, solving the point P3).

From the moment the set of equals, different and similar were generated, the necessary information to build the framework requirements model is obtained and kept. This process enables the model to be adaptable and extensible, solving the point P5). Thus, the union of sets will form the requirements model of the framework. This process may be repeated, allowing refinement and improvement of the model, having as base the information obtained from the union of the sets.

Considering n example applications, the requirements model for the domain framework is build through the procedure describe bellow: //where: n is the total amount of example applications.

i.) Choose one example application, called k. // where: $1 \leq k \leq n-1$.
ii.) Choose one subsystem, called $s_p$, of k for the analysis. // where: $p \in \{1,..,\omega\}$ , $\omega$ is the total amount of subsystems belonging to the k, and $1 \leq p \leq \omega$
iii.) Choose another example application, called a. //where: $a \leq n$ e $a \neq k$.
iv.) Verify whether $s_p$ exists in a. If the result is true, then the steps bellow should be followed:
  a.) Choose one requirement, called $r_j$ , of $s_p$ to be analyzed. // where: $j \in \{1,..,f\}$ , $f$ is the total amount of requirements belonging to k, and $1 \leq j \leq f$
  b.) Remove of $r_j$ its set of responsibilities called re. //where: re is set of responsibilities
  c.) Choosing one requirements, $r_q$ , of a. //where: $q \in \{1,..,\Re\}$, $\Re$ is the total amount of requirements belongs a, and $1 \leq q \leq \Re$.
  d.) Remove from $r_q$ its set of responsibilities called re´.
  e.) Compare re to re´ by name.
    - If the comparison is equal then classify the requirements belonging to the set of equals. If the comparison is different then classify the requirements belonging to the set of different. Otherwise, it belongs to the set of similar.
      - One set of responsibilities is equal when : $(re \subset re') \wedge (re' \subset re)$
      - One set of responsibilities is different when: $re \cap re' = \varnothing$
      - One set of responsibilities is similar when: $(re \cap re' \neq \varnothing) \wedge (re - re' \neq \varnothing) \wedge (re' - re \neq \varnothing)$
  f.) Return to the Step c.) and incrementing q.
  g.) Return to the Step a.) and incrementing j.
v.) Return to the Step iii.) and incrementing a.
vi.) Create the requirements model framework for the set equals, different, similar of $s_p$. This step establishes guidelines such as: classification, type and reuse procedure, as well as the alias for the sets.
vii.) Return to the Step ii.) and incrementing p.
viii.) Return to the Step i.) and incrementing k.

**Figure 1.MDR-FHS method procedure to create the requirement model of the framework.**

i.) Choose one of the sets created in Figure 1. The chosen set is called $c_f$ . //where: $c_f$ ={equals}.
ii.) Choose one requirement, called $r_j$ , of $c_f$ //where: $j \in \{1,..,f\}$, and $f$ is the total amount of requirements, and $1 \leq j \leq f$
iii.) Remove from $r_j$ its set of responsibilities called re.
iv.) Choose one example application, called k. //where: $1 \leq k \leq n$ and n is the total amount of the example application
v.) Choose one class, called $c_d$, of k. // where: $1 \leq j \leq y$ and y is the total amount of methods in class. Each $c_d$ has one set of methods and each method is associated to a responsibility.
vi.) Remove the set of responsibilities from $c_d$ denominated re´.
vii.) Compare re to re´ by name .
  - If all the methods of re are contained in the class $c_d$, without existing additional methods in class.
    - Reuse $c_d$ for the class model framework. • In this case, the class $c_d$, is classified as of stability.
  - If all the methods of re are contained in the class $c_d$, with additional methods that were not part of the responsibilities. These methods were created by the modularization process of the method to reduce its complexity.
    - Reuse $c_d$ for the class model of the framework. • In this case, the class $c_d$, is classified as of stability.
  - If all the methods of re are contained in class $c_d$ , with any methods that do not belong to re, but are not additional methods, we have two situations to be analyzed:
    - **Case 1 –** When the methods that do not belong the re, but also belongs to the responsibilities classified as equals, then the class will be classified as of stability and it is possible to reuse $\mathbf{c_d}$.
    - **Case 2 –** When the methods that not belong the re, also do not belong to the responsibilities classified as equals, then the class will be classified as of flexibility and reuse $\mathbf{c_d}$. The class $c_d$ goes by a refinement process.
viii.) Store the reused or created class in class model of the framework.
ix.) Return to the Setp v.) and incrementing d.
x.) Return to the Setp iv.) and incrementing k.

**Figure 2. MDR-FHS method procedure to create the class model for the set equals.**

The point P4) was solved because the whole procedure was formalized using the set theory, which was susceptible of being implemented using the relational model. Consequently, there may be a time reduction for classifying frozen and hot spots.

The proposed method also allows a process dependent on the initial classification for the creation of the class model. The class model for an example application has a model in which each method is associated to a responsibility.

In this work, it will was demonstrated the procedure for creating the class model of the framework for the set of equals (see Figure 2). It is necessary to emphasize that the set of equals were created from the procedure, described in Figure 1. So, the point P2) was also solved in this case, because for the creation of the class model the process is dependent on the classification of the requirements obtained in the procedure of Figure 1. This may be observed on the steps v.) to vii.) illustrated in Figure 2.

From the moment the classes for the sets of equals, different and similar were generated, it will be obtained the information that constituted the class model of the framework. This model may also be utilized in a posterior refinement or improvement of the framework.

## 5. CONTRIBUTIONS

This research proposal has the following main contributions:

- Set of steps that facilitate classifying the set of responsibilities, allowing a more precise classification process.
- Set of guidelines that help the construction of the requirements and class model of the framework.
- Process that may be semi-automatized making possible the shortening of the classification time of requirements and classes of sample applications.
- Model that may be adapted or extended, allowing the domain framework improvement.

## 6. BRIEF ANALYSIS OF THE RESULTS

The way to determine the frozen and hot spots demonstrated in this work is different from the ones accomplished in other approaches found in the literature. This work is not concerned in analyzing either class diagrams or an element of the requirement such as its name. It is concerned in analyzing sets of responsibilities associated to requirements instead.

The analysis accomplished by the set of responsibilities allowed better precision for classifying the requirements and early determination of frozen and hot spots. It differs from the analysis by name, defined by Ben-Abdallah [8] and Landin and Niklasson [4], which many times may lead to a wrong classification, since one element is analyzed, whereas in the responsibility-driven analysis several elements are considered, namely the set of its responsibilities.

The classification process was facilitated, once the MDR-FHS is based on the set theory, it is possible to implement it in a relational model. Thus, the time for the identification of frozen and hot spots can be substantially shortened.

Due to the way the method was formalized, it is possible to accomplish refinements in the framework. Thus, the framework may be extended, considering other examples applications that had not been initially analyzed.

The method MDR-FHS also differs from the literature approaches because it contemplates the rules to classify the sets of requirements, namely responsibilities, and the reuse of analysis models and artifacts, where it is possible to analyze each subsystem concomitantly, thus facilitating in posterior phases the detailed identification of collaboration among subsystems.

During the elaboration of the requirements model of the framework for car racing games, there was not much change in the elaboration of the model pertaining to either the set of equals or different**,** because they had already initially been classified. The only part of the method in which the model refinement really was put in action was during the construction of the requirement model for the sets of similar. It was necessary to analyze whether existed sets of functionalities with different options, however with the same functionality.

The reuse is given through the requirements model of each example application. This helped diminish the impact in the identification of the frozen and hot spots of the framework, as well as facilitated the early comprehension of the domain.

## 7. RESEARCH EVALUATION METHODS

The method is being experimented in the elaboration of two frameworks: Car Racing Games and Criticity Method. However, more detailed experiments should be accomplished in order to evaluate the level of improvement in development process of frameworks that it is obtained with the method here proposed.

A group of metrics, called GM-F, is being developed for the evaluation of MDR-FHS method. The GM-F was divided in the following two parts: Framework Requirements Metrics (MR-F) and Framework Project Metrics (MP-F). Both MR-F and MP-F are based on the Goal/Question/Metric Method, or simply GQM. Using the GM-F will be possible to measure the requirements and classes reuse of the applications examples, as well as to confront the amount of frozen and hot spots found out in initial phases and the ones spots were found detected at level of class model.

## 8. REFERENCES


[1] R. T. V. Braga, P. C. Masiero. A process for framework construction based on a pattern language. *In Proc. 26th Annual International Computer Software and Applications Conference*, *IEEE Computer Society*, pages 615-620, 2002.

[2] S. Hanenberg, R. Hirschfed, R. Unland, K. Kawamura. Applying Aspect-Oriented Composition to Framework Development – A Case Study. In. Proc. *1st International Workshop on Foundations of Unanticipated Software Evolution*, Barcelona, Spain, 2004.

[3] R. E. Johnson. How to design frameworks. *In. Object-Oriented Programming Systems, Languages and Applications Conference*, Washington Proceedings, 1993.

[4] N. Landin, A. Niklasson. *Development of Object-Oriented Frameworks. Relatório Técnico*, Lund Institute of Technology, Lund University, 1995.

[5] W. Pree. *Design Patterns for Object-Oriented Software Development*. Addison-Wesley, Reading, Mass, 1995.

[6] R. P. Silva. *Support for the Development of Frameworks and Components*. Masters Thesis, Universidade Federal do Rio Grande Sul (UFRGS), pages 1-262, 2000.

[7] R. Wirfs-Brock, A. Mckean. *Object Design: Roles, Responsabilities, and Collaborations*. Addison Wesley, 2003.

[8] H. Ben-Abdallan, N. Bouassida, F. Gargouri. A. Ben-Hamadou. A UML-based Framework Design Method. *In Journal of Object Technology*, v. 3, n. 8, p. 98-119, 2004.

[9] M. Fayad, D. Schmidt, R. Johnson. Building Application Frameworks – Object-Oriented Foundations of Framework Design, Wiley, 1999.