Reusing Geographic E-Services: A Case Study in the Marine Ecological Domain

Patricia Pernich[1] [⋆], Agustina Buccella[1], Alejandra Cechich[1], Maria del Socorro Doldan[2], and Enrique Morsan[2]

[1] GIISCO Research Group
Departamento de Ciencias de la Computación
Universidad Nacional del Comahue
Neuquen, Argentina
patricia.pernich@gmail.com,{abuccel,acechich}@uncoma.edu.ar
[2] Instituto de Biología Marina y Pesquera "Almirante Storni"
Universidad Nacional del Comahue - Ministerio de Producción de Rio Negro
San Antonio Oeste, Argentina
{msdoldan,qmorsan}@gmail.com

Abstract. Software product line engineering aims to reduce development time, effort, cost, and complexity by taking advantage of the commonality within a portfolio of similar products. This may benefit the development of applications in several domains, and particularly in geographic information systems (GIS). Reusing services for GIS challenges developers to produce assets highly adaptable to various service clients and contexts. For that, service variability must carefully be modeled by considering different recommendations such as well-established standards. This paper presents an approach that facilitates variability implementation, management and tracing by integrating product-line and GIS development. Features are separated following standard recommendations and composed on a platform-oriented framework. The presented concepts are illustrated with a case study of a marine ecological system.

Key words: Geographic Information Systems, Software Product Lines, Services, Marine Ecology

1 Introduction

During last years, information about geographic aspects has been widely regarded as crucial part of every information system. With the new emerging technologies, such as GPS devices or remote sensing techniques, the work of capturing geographic information is becoming easier. In addition, there exists hundred of proprietary and open source software to store this type of information in an efficient way. Thus, several organizations are interested in implementing interactive web portals or stand-alone systems that allow users to query and

[⋆] This work is partially supported by the UNComa project 04/E072 (Identificación, Evaluación y Uso de Composiciones Software).

manage this type of information. In Geographic Information Systems (GIS) applications, there exists a set of geographic e-services that several organizations need. Identifying this common set of services and making them flexible to be implemented by the organizations might help reduce costs in development, staffing, maintenance, etc. With this goal in mind, we looked at two main areas of different research lines: *software product line engineering* and *geographic information systems (GIS).*

Within the first area, a system decomposition into high-level components is defined by considering the concepts of reuse and of family of products [1] (introduced firstly by Parnas [2] in 1976). In the literature there are several definitions of software product line engineering. One of them, defined in [3], introduces the concept of product lines as *a set of systems sharing common features and satisfying specific needs of a segment of a market.* The main characteristics involved in this new discipline are [4]: *variability*, in which individual systems are considered as variations of a common part; *architecture-based*, in which the software must be developed by considering the similarities among individual systems; and *two-life cycles approach*, in which two engineerings in every software product line process must be considered: *domain engineering* and *application engineering* [4, 5].

In the second area to be analyzed, GIS have emerged to allow the storage and manipulation of geographic information in spatial domains or with geographic characteristics. The special nature of geographic information results in a set of characteristics that must be taken into account in the development of a GIS application. In addition, there have been many research and industrial efforts to standardize many aspects of GIS technology, particularly by the Open Geospatial Consortium[3] (OGC) and the ISO Technical Committee 211[4] (ISO/TC 211, Geographic Information/Geomatics). For instance, the Service Architecture standard (defined in OpenGIS Service Architecture)[5] and the ISO/DIS 19119[6] std. define a taxonomy of geographic services in which each service of a particular system should be classified into one or more categories (depending on whether it is a simple or aggregate service).

In this work, we introduce a methodology for creating software product lines for GIS applications by combining the advantages of two widely referenced methodologies in the literature [1, 5]. The main goal is to create a software product line taking into account the set of common e-services of GIS software within a specific domain (in this case, the marine ecological domain). These common e-services are firstly defined by using the standard information provided in the ISO 19119. We also propose the creation of a framework to support the generation of new products within the line. The framework acts as a platform allowing developers to reuse implementation of common e-services, instantiate variable e-services, and implement product-specific e-services.

[3] `http://www.opengeospatial.org`
[4] `http://www.isotc211.org/`
[5] The OpenGIS Abstract Specification: Service Architecture, 2002.
[6] Geographic information. Services. International Standard 19119, ISO/IEC, 2005.

This paper is organized as follows: Section 2 describes our methodology to create the product-line for GIS applications and briefly describes related works in this area. Then, we describe the instantiation of the product line applied to an real project. Future work and conclusions are discussed afterwards.

2 A Software Product Line Specification for GIS

For many years, the Open Geospatial Consortium (OGC) and the International Organization for Standardization (ISO) had worked independently to reach overlapping goals, but nowadays both converge towards a common solution. The ISO/TC 211 deals with long-term, abstract, static standards, meanwhile the OGC works on industry-oriented, technology-dependent, evolving standards. Thus, the development of GIS applications can be done by following these standards in order to improve interoperability among systems. However, although software reuse among different systems could be improved when these standards are applied, they are not created for this purpose. Special efforts must be done to reach effective reuse.

Secondly, in the literature there exists several techniques proposing methodologies to develop a software product-line. Some of the most referenced proposals are [1, 3–5]. All of them propose a division into common and variable aspects of the product line, and a set of tasks or activities that must be done to specify and implement these aspects. For example, in [3] and in [4, 5] authors propose the same type of phases but with different names and activities. In [4, 5] the phases, named as *domain engineering and application engineering*, define common and variable aspects, and derive the product line from the platform defined in the first phase. Other recent proposals are analyzed in [6]. Here, authors compare four methodologies that involve specific goals to improve different aspects of a software product line development. For example, FAST (Family-Oriented Abstraction, Specification, and Translation) [7] defines a methodology based on a software engineering process trying to improve some aspects such as production costs, time-to-market, etc.

To the best of our knowledge, the two sets of work, GIS standards and software product lines, are not related in the literature. The creation of software product lines for GIS products is only seen as another application of a product line development. In addition, there are few references applying these concepts together. However, geographic software shares a set of common services that are essential for every application; therefore these common services might be identified and modeled as part of the product line together with different variations.

Our work is based on the creation of a software product line to be used by organizations working within the marine ecological domain that are interested in implementing geographic e-services. In order to build the software product line we combined characteristics of two methodologies defined in [1, 5]. Each methodology presents a set of processes to perfom this task. We have combined both proposals in order to take advantage of their best-defined processes. In particular, we modified the main process, named *domain engineering*, from the software product line engineering framework defined in [5]. We have defined a different set of six subprocesses by applying activities defined in both methodologies. Figure 1 shows these subprocesses together with their main tasks.

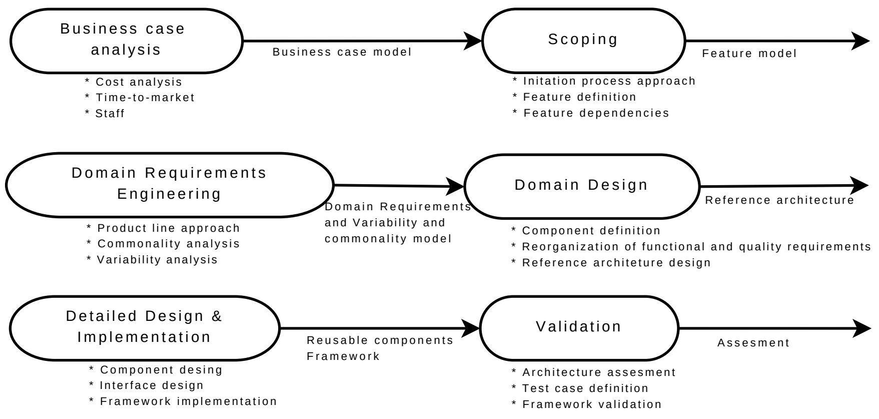

Fig. 1. Domain engineering process.

Following, we briefly describe how these tasks were applied to create the software product line for a GIS application within the marine ecological domain.

- *Business case analysis*: To perform this subprocess, the situation of the different organizations in Argentine working on the marine ecological domain was analyzed. In our case, few organizations had applications involving geographic information. They had used only office software tools in which almost all the tasks were made manually. Therefore, the costs and staff needed were analyzed by taking into account two main phases. In the first one, we analyzed the aspects needed to implement the product line and its supporting application framework. In the second phase, we analyzed what we needed to implement future products by instantiating the framework.
- *Scoping*: We applied an evolutionary approach as starting point for the development of the product line architecture. In this way, we started from a set of existing software products. However, by considering the aspects analyzed in the previous subprocess, we could not take the organizations' software products, which were incomplete and unreliable. Instead, we used a set of related geographic open source software tools for GIS, which provided features needed by our architecture. Then, we defined the features that were candidate for inclusion in the product line. In [1] a *feature is defined as logical unit of behaviour that is specified by a set of functional and quality requirements.* In our work, we firstly defined features by using the ISO 19119

standard as a starting point to define a set of specific features. Table 1[7] shows part of the resultant features. For brevity reasons we do not include here all of them.

Categories	Service	Specific Features
Geographic human interaction	S1. Geographic viewer	S1.1) Show and query data of geographic features. S1.2) Show/hide layers. S1.3) Zoom tool. S1.4) Panning tool. S1.5) Scale. S1.6) Navigation buttons (previous/next)
	S2. Geographic feature editor	S2.1) See, query and edit geographic features graphically. S2.2) Add new geographic features by using the graphic interface
Geographic model/information management services	S3. Feature access service	S3.1) Perform queries to a geographic feature repository. S3.2) Manage data of the geographic features
	S4. Map access service	S4.1) Access to geographic maps
Spatial processing services	S5. Proximity analysis service	S5.1) Obtain all geographic features within a specific area
Temporal processing services	S6. Temporal proximity analysis service	S6.1) Obtain all geographic features within a specified time
Metadata processing services	S7. Statistical calculation service	S7.1) Generate statistics with data of geographic features
	S8. Geographic annotation services	S8.1) Add additional information to a geographic features. S8.2) Add additional information to a map

Table 1. Part of geographic services required by our architecture

Then, we analyzed which services (shown in Table 1) are implemented by the geographic open source tools so we have a basis for implementing e-services into a particular platform. Seven tools were classified into three categories: *thin web clients*, *geographic databases*, and *map servers*. In the first category we analyzed four thin web clients (Mapfish[8], p.mapper[9], Ka-Map[10], and MapBender[11]) which implement the features S1, S2, S5 and S8, and partially implement the features S6 and S7. In the second category we analyzed databases which include geographic analysis. In this case, both MySQL

[7] In the table we can see that the term "feature" has a different meaning. In the standard a "feature" is defined as an abstraction of real world phenomena. We refer this term as "geographic feature" in order to differentiate it from the term "feature" defined by the domain engineering process.

[8] `http://mapfish.org/`

[9] `http://www.pmapper.net/`

[10] `http://ka-map.maptools.org/`

[11] `http://www.mapbender.org/`

GIS[12] and Postgis[13] databases can implement the features S3 and S6. In the third category, the map servers UMN MapServer[14] and GeoServer[15] were analyzed to implement the feature S4 allowing the access to geographic maps.

– *Domain requirements engineering*: We used a minimalist approach in which only the features used in all products are part of the product line. In our case, this approach allowed us to fully implement only common features and let the product-specific features be implemented by each different organization. Thus, our software product line is seen as a *platform* [4]. Table 2 shows the subset of features that are part of the product-line and the subset of features that are product-specific features.

Products/Features	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**S9**	**S10**	**S11**
Product-line	X	X	X	X	X		X	X			
Product 1						X					
Product 2									X		
Product 3										X	
Product 4											X

Table 2. Features that are part of the product-line and specific-product features

As we can observe, features S1-S5 and S7-S8 are part of the product-line. S6 is a product-specific feature that will be implemented only by Product 1. Similarly, features S9, S10 and S11 are part only of each Product 2, 3 and 4, respectively. Feature S9 is a catalogue viewer service within the geographic human interaction category. It allows to locate, browse, and manage metadata about geographic data or geographic services in a catalogue. Features S10 and S11 belong to the spatial processing services category. The first one, named *coordinate conversion service*, allows to change coordinates of a feature from one coordinate system to another one. Finally, feature S11, named *positioning service*, is provided by a position-providing device (like a GPS) to use, obtain and unambiguously interpret position information. In addition, within each feature we determined the commonality and variability models. Figure 2[16] shows the variability model associated to the feature S7.1 of the product-line. In this model we can see that statistics can be represented by using two variants – histograms or tables.

– *Domain design*: Now, it is time to build the reference architecture based on the features defined in the previous subprocesses. We reorganized the features into two sets of requirements to separate functional from non-functional (quality) needs. These sets were the basis to define our architecture's compo-

[12] `http://dev.mysql.com/tech-resources/articles/4.1/gis-with-mysql.html`
[13] `http://postgis.refractions.net/`
[14] `http://mapserver.org/`
[15] `http://geoserver.org/`
[16] We use the graphical notation for variability models defined in [5].

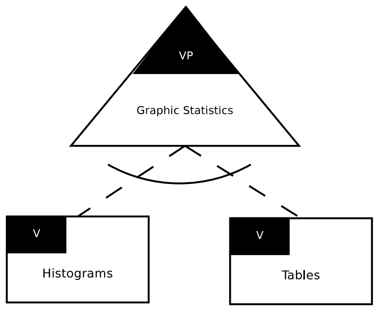

Fig. 2. Variability model of feature S7.1

nents. We chose a layered architectural style to facilitate dealing with modifiability and scalability requirements. The reference architecture is composed of three main layers: *geographic model*, *geographic processing*, and *user interface*. In the first layer we implemented the geographic model that allows the access to a geographic database. The second layer defines features involving processing (features S3-S8 described in Table 1); and the third layer defines the S1 and S2 features in which graphical aspects are involved. Special components in these layers are responsible for performing the operations that must be taken into account to publish, find and bind e-services. The next layers implement services as e-services provided to users or other systems.

– *Detailed design and Implementation*: In addition to define the architecture of the product line, we created an application framework to be used as a platform for each product of the product line. The framework covers the behaviour that is common for all products and allow developers to add product-specific features.
– *Validation*: There are several aspects to analyze within this subprocess. Firstly, some test cases were defined in order to test the framework and the specification of the product line. Secondly, we tested the instantiation of the product line by creating new GIS products (as the one we will describe in the next section).

3 Instantiating the Product Line: A Case Study

In this case, we build the Product 1 (Table 2) containing features of the product line plus a set of product-specific ones (S6). The other products in the software line (Products 2−4) will belong to other organizations within the marine ecological domain (Instituto Argentino de Oceanografía[17], Centro Nacional Patagónico[18], and Laboratorio de Moluscos y Crustáceos belonging to the University of Mar del Plata).

[17] http://iado.criba.edu.ar/web/
[18] http://www.cenpat.edu.ar/

Product 1 emerged from a project between GIISCO research group[19] and the Instituto de Biología Marina y Pesquera "Almirante Storni"[20] (IBMPAS). IBMPAS is responsible for analyzing and storing information about sea surveys in the San Matías Gulf, Patagonia Argentina. Each survey, perfomed once a year (when it is possible), collects information about the population of specific species living in this area. This information is then used for spatial processesing in order to obtain information about spatial distribution of data, population variation patterns in different scales, etc.

Table 3 shows the features of Table 1 redefined according to the requirements of the product. For example, the feature S1.1 is redefined as a set of new features (a-c) to show different geographic features of the sources; and the feature S1.2 allows the user to show and hide layers. In this product we implement several layers including sea zones[21], stations[22], and abundance and density of species[23]. For brevity reasons, we only include here a subset of the layers and of the instantiated features within each feature category.

Following, the variability models must be also instantiated. In order to illustrate this, Figure 3 shows the instantiation in which the feature S7.1 (presented in Figure 2) is implemented by using histograms (not by using a table). The figure shows the component that implements this service (S7.1) and an arrow from the variant point (histogram) indicating that the component uses this variation.

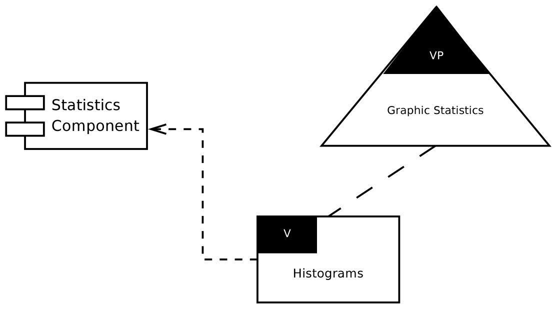

Fig. 3. Instantiated variability model of feature S7.1

Finally, we have to create the architecture based on the reference architecture defined in the last section. The architecture of Product 1 is used to provide a set of different e-services to users of IBMPAS. As an example, we describe here two

[19] `http://giisco.uncoma.edu.ar/`

[20] `http://ibmpas.org/`

[21] A zone is a maritime area bounded and defined with a specific name in the gulf.

[22] A station is a geographic point located within one of the defined zones. In this location the measures of population of species are obtained.

[23] The abundance of a species indicates the number of them in a specific location (in a zone or station, according to the geographical scale chosen). The density displays the number of individuals per m^2.

Categories	**Service**	**Specific Features**
Geographic human interaction	S1. Geographic viewer	S1.1) a) Show zones. b) Show stations within a zone. c) ... S1.2) a) Show/hide the zone layer. b) Show/hide the density layer. c) ...
	S2. Geographic feature editor	S2.1) a) Show a map with the location of zones. b) Show a map with the abundance of species. c) Show a map with the density of a specific species. d)... S2.2) a) Add a station within a specific zone. b) Add the density of species within a specific station. c) ...
Geographic model/information management services	S3. Feature access service	S3.1) a) Query zones of density of species. b) Query zones in which the population of species are higher than a specific value. c) ... S3.2) a) Modify thematic attributes of a zone. b) ...
	S4. Map access service	S4.1) a) Allow to create an image showing the density of species in several zones. b) ...
Spatial processing services	S5. Proximity analysis service	S5.1) a) Obtain the location of stations within a specific zone. b) Obtain the number of species in a specific zone. c) Obtain the density of a determined species in a specific zone. d) ...
Temporal processing services	S6. Temporal proximity analysis service	S6.1) a) Obtain the number of specimens of specific species in a zone at different times. b) ...
Metadata processing services	S7. Statistical calculation service	S7.1) a) Generate a histogram with the density of species. b) ...

Table 3. Instantiated features required in Product 1

of them – *density of a specific species* and *length frequency of a specific species* services. The first one returns information about the number of specimens per m^2 of a species in a specific location. The input of the service is an area of interest defined by the user, which may be a particular geographical zone or area that covers the entire gulf. Depending on the chosen scale, the service shows the number of this species in one zone or all zones in the gulf. The output of the service is a combination of three layers: *the zone* layer, *the station* layer (which are located within zones), and a layer showing *the density of species* at each station. Thus, it is possible to query the population of specific species found in different years and compare the generated maps to detect possible migrations of specimens of a species. In addition, it is possible to analyze different reasons of these migration movements. Figure 4 shows the output map when a user runs the *density of a specific species* service. In this case, data are displayed in all zones of the gulf. As we can see, each point indicates the density of the *Ostrea puelchana* species in a given station. The darker color indicates a greater concentration of the number of specimens per m^2 of the species at that station. The polygons define the covered area by each zone.

The other service, *length frequency of a specific species*, generates statistics data based on the stored information. In this case, from measures of each specimen found for a specific species, the service shows the frequency of occurrence

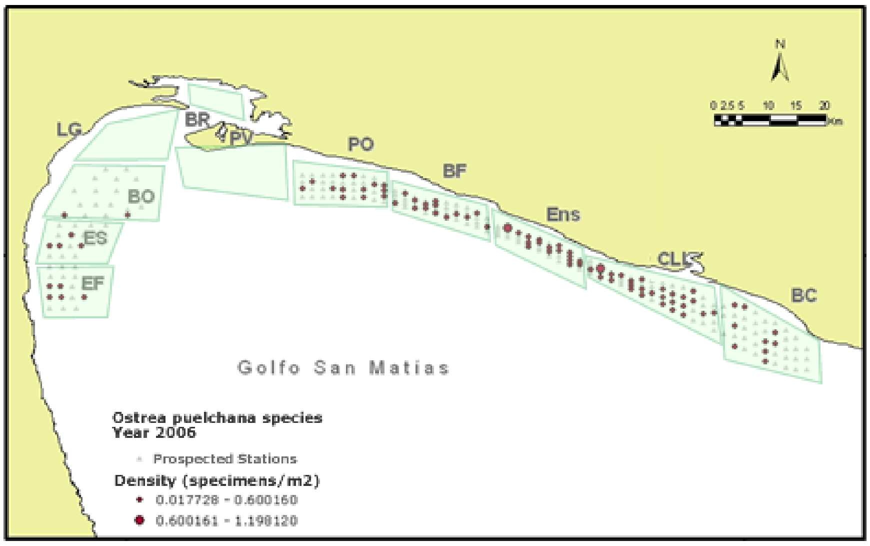

Fig. 4. A map representing the *density of a specific species* service

of each size. Figure 5 displays a histogram of the length frequency of a species named *Vieyra Tehuelche*, in the Northwest zone of the San Matias Gulf.

Each of these two services uses several of the features defined in Table 3. For instance, the *density of a specific species* service uses the features S2.1)a) and c), S3.1)a), S4.1)a) and S5.1)c). Moreover, the *length frequency of a specific species* service uses the feature S7 that is implemented by using histograms.

3.1 Lessons Learned

We tried to ensure validity by using multiple sources of data to build the product line, i.e. interviews, document analysis, and validation sessions. We aimed to show that software product family practices can be applied in several different geographic contexts. Based on the results of the case study (Section 3), we identify the following lessons:

Software product family engineering can be applied to small ecological organizations. Our approach has successfully developed its products in a software product family. The benefits of this model of development have not been directly measured, but one indicator of the success is that the development cost and time-to-market were drastically reduced. Product 1 was available in the short time and the needed staff was fewer than the one needed for the creation of the product line. Of course, benefits would be better quantified whether the organizations follow a defined process, but this was not the case. However, although their processes were ad-hoc in most cases, our approach could be applied by following a set of guidelines. The drawback of our approach is precisely the

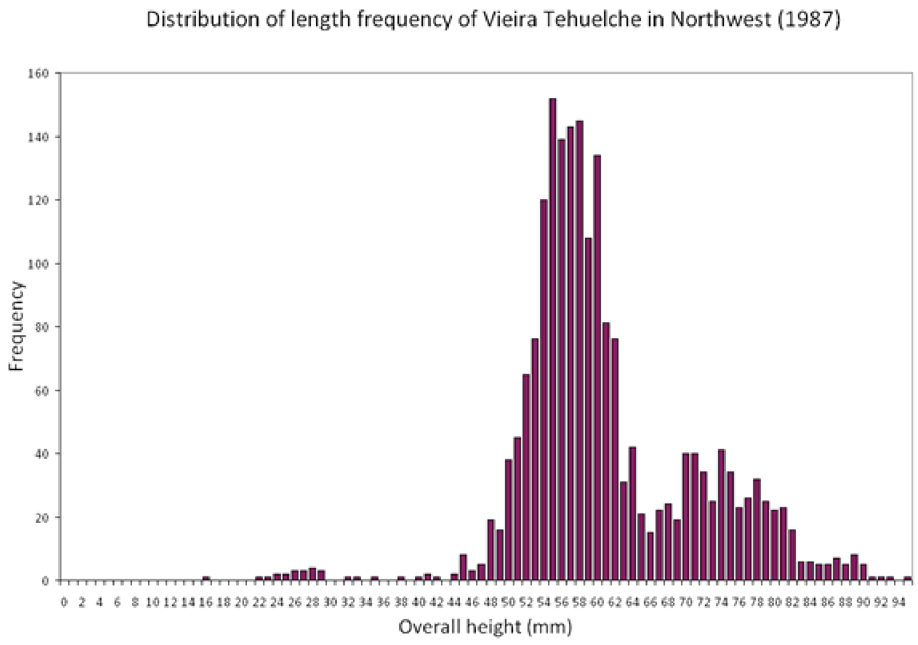

Fig. 5. A histogram representing the *length frequency of a specific species* service

learing effort and time that are spent in setting the right conditions to build the product line.

A software product family development can cross organization borders. Geographic information development is shared among distributed ecological organizations. In fact, the challenge lies in specifying the required variability for each case of software. Extending existing variation points with new variants is easier than creating new variation points from scratch. It is not necessary to specify all possible variants, but it is essential to ensure that the new software implements some mechanisms for all variation points. To address this issue, we are currently analyzing the application of verification configurations models.

Building reusable assets requires variability mechanisms. A software product family requires effective ways of realising and implementing variability mechanisms. From our case, effective variability implementation is as much as important than effective variability management. Both aspects should be considered and we are extending the approach to explicitly include management guidelines. For instance, before we were able to implement the variability model of the product family, we had to externalize the variability information from the product family artifacts and the experts. During this externalization process, engineers and experts found many (previously unknown) conflicts in their own artifacts.

These conflicts were solved following a set of recommendations that should be institutionalized.

4 Conclusion and Future Work

In this work we have introduced a framework for building geographic information systems from a software product line, and we have illustrated the process through a case study. Our work emerges as a solution to different organizations within the marine ecological domain. The main goal is to reduce the development effort involved in the creation of this type of systems. In this way, two main requirements of every GIS software within the ecological domain might be fulfilled. On one hand, *flexibility* might be improved because new products can easily perform changes in the requirements by means of instantiating variable aspects of services. On the other hand, *reusability* might be reached because components of the architecture are created for that purpose.

As future work, the methodology and the framework need more validation by analyzing the process of building new products of the line. However, we are aware that developing management guidelines is also crucial for sucessfully applying the approach.

References

1. Bosch, J.: Design and use of software architectures: adopting and evolving a product-line approach. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA (2000)
2. Parmas, D.: On the design and development of program families. IEEE Transactions on Software Engineering **SE-2**(1) (March 1976) 1–9
3. Clements, P., Northrop, L.: Software Product Lines : Practices and Patterns. Addison-Wesley Professional (August 2001)
4. Van Der Linden, F., Schmid, K., Rommes, E.: Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering. Springer-Verlag New York, Inc., Secaucus, NJ, USA (2007)
5. Pohl, K., Böckle, G., van der Linden, F.: Software Product Line Engineering : Foundations, Principles and Techniques. Springer (September 2005)
6. Matinlassi, M.: Comparison of software product line architecture design methods: Copa, fast, form, kobra and qada. In: Proceedings of the ICSE '04: 26th International Conference on Software Engineering, Washington, DC, USA, IEEE Computer Society (2004) 127–136
7. Weiss, D., Lai, C.T.R.: Software product-line engineering: a family-based software development process. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1999)