- ORIGINAL ARTICLE -

An SPL Reference Model based on Domain Taxonomies & Standards

Un Modelo de Referencia de Líneas de Producto de Software basado en Taxonomías de Dominio y Estándares

Agustina Buccella[1] and Alejandra Cechich[1]

[1]*GIISCO Research Group, Departamento de Ingeniería de Sistemas, Facultad de Informática, Universidad Nacional del Comahue, Neuquen, Argentina*
{agustina.buccella,alejandra.cechich}@fi.uncoma.edu.ar

Abstract

Building reusable software is always a challenge, even when well-established approaches are applied. Software Product Line (SPL) development is one these approaches, which allows domain modeling be a way of dealing with common and variable aspects of reality. However, domain engineering itself can be complex, many times depending on the domain scope and/or its associated functionality. In this paper, our proposal to SPL development is structured as a leveled reference model built upon standardized semantic resources. This model and its associated process are exemplified through several cases from the field, drawing influencing factors subjectively assessed. Our experiences show that systematically enriching domain engineering may improve SPL development in the practice.

Keywords: Software Product Line Engineering, Domain Analysis, Reference Models, Reusability

Resumen

Crear software reutilizable siempre es un desafío, incluso cuando se aplican enfoques bien establecidos. El desarrollo de una Línea de Productos de Software (LPS) es uno de estos enfoques, ya que permite el modelado de dominios por medio de la definición de aspectos comunes y variables de la realidad. Sin embargo, la ingeniería de dominio en sí misma puede ser compleja, muchas veces dependiendo del alcance del dominio y/o su funcionalidad asociada. En este artículo, nuestra propuesta para el desarrollo de una LPS está estructurada como un modelo de referencia basado en niveles y construido sobre recursos semánticos estandarizados. Este modelo y su proceso asociado se ejemplifican a través de varios casos reales de aplicación, extrayendo factores influyentes evaluados subjetivamente. Nuestras experiencias muestran que el enriquecimiento sistemático de la ingeniería de dominio puede mejorar el desarrollo de una LPS en la práctica.

Palabras claves: Líneas de Productos de Software, Análisis de Dominio, Modelos de Referencia, Reusabilidad

1 Introduction

Since the 80's, the idea of building systems capturing particular requirements has generated a wide set of research proposals in which reuse is one of the major benefits. The domain analysis or domain engineering area was firstly introduced by Neighbors [1] as *the activity of identifying the objects and operations of a class of similar systems in a particular problem domain.* Then, other proposals have improved the basis of this activity by adding techniques and methods that guide software engineers during the development of domain specific systems [2, 3, 4, 5, 6]. Currently, the domain analysis is the first activity of a Software Product Line (SPL) approach [7, 8], in which the construction of domain systems is performed by reusing domain requirements. In this way, software systems are seen as industry products, such as cars or machines, which are built in mass by reducing thus time and costs. At the same time, these products capture specific requirements of a domain making them more adequate for their users. The SPL paradigm involves three main characteristics from a specific domain:

- *Commonalities*: describing services shared by the products to be generated.
- *Variabilities*: describing different options of functionalities among the products to be generated.
- *Two phase development*: including the *domain* and *application* engineering. In the former, commonalities and variabilities are defined resulting in a *software platform*. Meanwhile in application phase, products are built by using this platform; that is, product derivations are performed by reusing the commonalities and instantiating the variabilities (sometimes also by adding product-specific functionalities).

The overall structure and processes of software product line engineering and management have been standardized by the ISO through its ISO/IEC 26550:2015[1] standard. It provides terms and definitions, and describes how the components of a product line reference model fit together. Another standard, ISO/IEC 26551:2016[2], provides the capabilities of tools and methods that support SPL requirements engineering by considering three processes: Product Line Scoping, Domain Requirements Engineering, and Application Requirements Engineering. The major purpose of the Domain Requirements Engineering is to analyze commonality and variability based on the initial features defined in Product Line Scoping; and the major purpose of the Application Requirements Engineering is to define application requirements based on domain requirements assets. Both standards complement each other to define a reference model for software product line engineering.

In addition, sometimes not only domain requirements can be reused (intra-domain reuse), but also domains are related by a series of properties and common services (inter-domain reuse). Proposals in the literature face this challenge by modeling software ecosystems or multiple product lines (MPL). Unfortunately, there are many approaches to deal with MLP modeling, and even MPL capabilities are not clearly established [9, 10, 11]; for instance, improving the integration of components previously developed for different domains still remains an open issue.

In this sense, our previous work has focused on analyzing the geographic domain as a generic one [12, 13] – i.e. a domain that allows to share services among different subdomains depending on their intrinsic characteristics, such as oceanographic, terrestrial, etc. and/or possible human intervention in these areas, such as marine ecology. Precisely, our efforts have been addressed to reuse through a subdomain-oriented development; for instance, in our case, a new SPL for paleobiology [14] was built by reusing artifacts developed for marine ecology [15]. It means that possibilities of reusing go beyond a single set of applications (single-domain oriented), introducing SPL families related by connections among different domains (subdomain oriented). This possibility leads to a series of benefits when developing reusable products, since knowledge and inter-domain reuse can be organized in terms of domain/subdomain-oriented standards and services.

In this paper, we elaborate a reference model for inter-domain reuse, which is also subdomain oriented. By mirroring recommendations from the ISO 26550/51 standards, we introduce here a leveled structure and a process to deal with software product line engineering taking advantage of using standardized services (through taxonomies and/or standards). The model and experiences of use have been developed in cooperation with different institutions during more than 10 years.

This paper is organized as follows. The next two sections briefly introduce our proposal and its process. Then, both are instantiated showing experiences from the field. Section 5 discusses some influencing features subjectively assessed. Finally, we address conclusions and future work.

2 Related Work

Reusability in the context of domain analysis is an active research field being analyzed and studied during the last 30 years. We can place the first works in this area at the end of the 80 decade in which concepts as domain analysis, reusable entities, product derivation, domain languages, etc. have been firstly discussed [2, 3, 4, 5, 16]. Then, at the beginning of the 2000 decade, started the first proposals in what we now call *Software Product Line Engineering* [7, 17, 18, 19]. This field has emerged as a paradigm for domain-oriented software construction focusing on reusability of core assets and providing variability for product generation.

In particular, our work focuses on the SPL development in order to maximize reusability in both inter- and intra-domains. We define and create a set of reusable artifacts providing support to the whole process. To do so, our development is based on the definition and application of *domain taxonomies* and *standards*.

Related to the definition of taxonomies, there exist works creating them as types of services to be used in any software development process. For instance the *Arizona Dictionary and Taxonomy of Human Services*[3] is being developed by members of the Arizona Taxonomy Committee (composed of representatives of departments, cities and towns of the Arizona State) for defining a common language by means of a set of specific services. Another service taxonomy has been developed by the International Foundation for Information Technology (IF4IT)[4] in order to represent different service types used by information technology organizations. As a final example, we can cite the work presented by [20] in which the author defines a service taxonomy based on eight main categories in order to provide a common language for participants (architects, engineers, etc.) of a service-oriented development process. The main goal here is, again, to improve communication within the software process.

On the other hand, in our work we take advantage of using *standards* in order to guide the software development and maximize reusability. The use of standards

[1]ISO/IEC 26550:2015 is the entry point of the whole suite of International Standards for software and systems product line engineering and management - https://www.iso.org/standard/69529.html

[2]ISO/IEC 26551:2016 Software and systems engineering — Tools and methods for product line requirements engineering - https://www.iso.org/standard/69530.html

[3]`https://www.azdes.gov/taxonomy/`

[4]`http://www.if4it.com/SYNTHESIZED/FRAMEWORKS/TAXONOMY/service_taxonomy.htm`

in software developments is a widely recommended practice with known advantages in terms of interoperability, reliability, safety, etc.[5]. In particular, there exist several works promoting the use of standards specially defined in particular domains for developing systems. For example, in the healthcare domain we can cite works in which the DICOM standard[6] is used for different developments. For example, in [21] authors apply this standard for developing a framework for data-driven GUI[7] generation in medical diagnostic, and in [22] authors propose an internet of things (IoT) framework for health sectors in Indonesia. As another example, we can cite the survey presented in [23] in which authors analyze the application of several standards of the industrial automation domain in different parts of software development. Finally, we can cite works presented in [24, 25] in which different domain specific standards are analyzed for management systems. Although we can say that there are several efforts using standard information to define services and assist the development of new products, there is a lack of specific efforts in systematic software reuse. The standards are more related to interoperability than reuse.

In our proposal, a taxonomy-oriented domain analysis is built based on standard information about particular domains being analyzed. This taxonomy covers different aspects such as terminology, services, and conceptual models of the domains. The definition of new aspects to be part of the taxonomy are added by generalization/specialization mechanisms. These mechanisms populate the taxonomy from specific aspects of each particular domain in which the SPL is built. Here, the taxonomy acts as a controlled vocabulary for all participants providing a common language (which result in better communication). Thus, time for learning domain particularities is reduced and, at the same time, it helps stakeholders to bridge the gap among their different skills by reducing the wide spectrum of information sharing [26]. Also, the effective reuse during the creation of new SPLs from existing ones, is benefited from the analysis of reusable services of the existing taxonomy [14].

3 The Reference Model: DT&S_SPL (RM)

The main goal of DT&S_SPL (RM) is to provide a specific environment for promoting reuse from previous SPLs and their resources, which have been created for a domain by specializing into more specific ones. Figure 1 shows DT&S_SPL (RM) as a leveled arrange of components, which can be (1) elements that produce/-constitute architectural assets, (2) domain taxonomies & standards (DT&S) that represent organized domain knowledge; and (3) specific tools that support the different activities. The model must be seen from bottom to top considering that the lowest level – *technologies* – includes the specific supporting tools; and the next one – *domain taxonomies and standards* – includes organized external knowledge that supports all possible architecture for a particular domain.

The levels and components are:

- *Technologies*: This level includes supporting technology for building assets of a particular architecture. The selection of the technology is not only focused on software (frameworks, software systems, languages, etc.), but also hardware, application servers, and so on. In the SPL area there is no a specific set of tools or environments to be applied. The universe of technologies and tools is large and varied due to different technologies can be used to create the same or different assets. For example, for variability modeling and analysis, tools such as those analyzed in [27] might be used. Similarly, there exists a set of open-source frameworks for implementing reusable components, such as those analyzed in [28].

- *Domain Taxonomies and Standards*: Domain specific taxonomies are created to classify elements according to different aspects such as relationships, properties, contexts, etc. [29]. The main goal of these taxonomies is to represent domain knowledge based on the division and specification of the elements involved, which can be anything useful for a domain, such as services, requirements, terms, domain artifacts, etc. Domain taxonomies also work as a controlled vocabulary among all stakeholders by providing a common language. On the same hand, standards provide similar goals but adding more resources, such as terminology, rules, properties, and guides. There are a wide set of available standards from different organizations. We can find general standards created for a wide community, such as the set of ISO Software Engineering Standards (ISO 9126, ISO 14598, etc.), and/or standards for SPL development (ISO/IEC 26550); and more specific standards belonging to particular domains, such as the ISO 19119 std[8] for the geographic domain or CIDOC Conceptual Reference Model (CRM)[9] for the cultural heritage domain.

- *Domain Engineering*: This level includes the main activities for building reusable artifacts within a specific domain. As in any traditional domain engineering phase during SPL development [7], this level is responsible for designing

[5]https://www.iso.org/benefits-of-standards.html

[6]ISO 12052:2017 Health informatics – Digital imaging and communication in medicine (DICOM) including workflow and data management

[7]Graphical User Interface

[8]Services International Standard 19119, ISO/IEC, 2005.

[9]CIDOC Conceptual Reference Model Version 6.0 – `http://www.cidoc-crm.org`

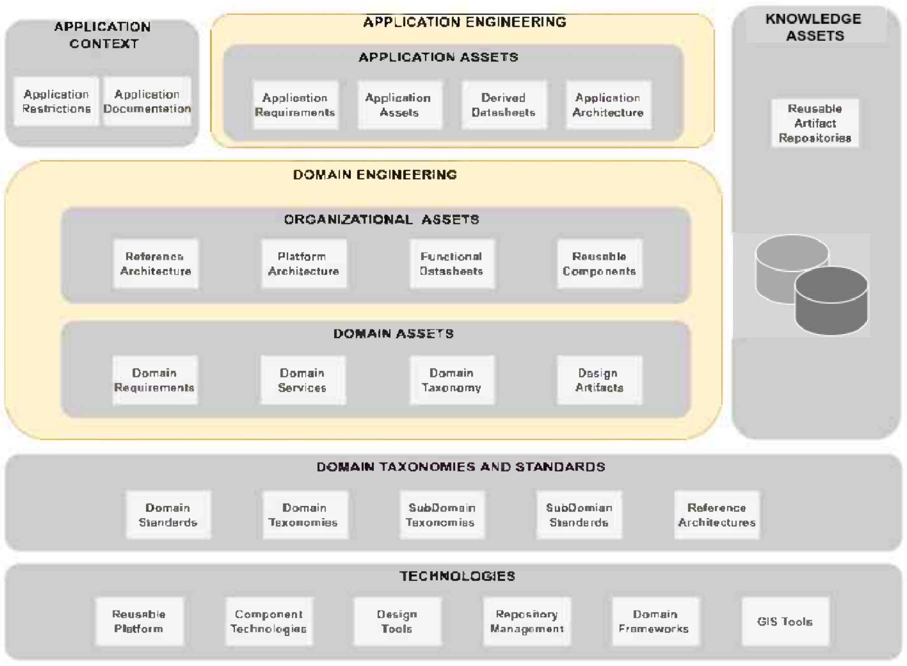

Figure 1: DT&S_SPL (RM) for building reusable LPSs based on domain taxonomies and standards

and implementing the SPL platform together with commonalities and variabilities of the domain. In our approach, we include two main components:

– *Domain Assets*: This component analyzes information received from the *Domain Taxonomies and Standards* and *Knowledge Assets* levels. Here, it is important to determine the information sources to be used (standards, existing applications, domain experts, etc.), analyzing and organizing features or services that a domain should offer, and identifying the set of reusable components that might be used to implement required features. In this way, this component is responsible for designing and extending domain requirements, services, taxonomies and any other software artifact describing the domain. All these outputs will conform the domain assets and they will be stored in the reusable artifact repositories (*Knowledge Assets* transversal level).

– *Organizational Assets*: This component acts as a refinement of the domain assets. It takes the domain analysis information and adapt it to the context of the SPL development. As our approach is based on a subdomain-oriented development, this component takes advantages from available subdomain information sources and knowledge assets (if they exist), to design, extend and implement the SPL platform. The reusable assets created for the subdomain are also stored in the repositories.

• *Knowledge Assets*: This transversal level stores the reusable assets that are taken into account for reusing or creating new ones during the domain and/or application engineering. That is, the repositories are available for software engineers and/or developers to create new domain assets for new or existing domains or subdomains. Previous to create a new domain asset, it is necessary to determine whether existing ones are able to be reused totally, or at least partially.

• *Application Context*: This level involves information about a particular application to be derived. During this process, restrictions and documentation can be used and generated. It is important here that all the documentation belonging to a particular product can be used for others in the same domain; specially when products share some variabilities.

• *Application Engineering*: As in any traditional SPL development, this level involves the activities related to the derivation of new products [7], which are created by reusing the SPL platform (with all generated artifacts), instantiating variabilities, and designing and implementing new specific artifacts.

4 A Process for designing and implementing functionalities

DT&S_SPL (RM) is supported by a process for building functionalities based on a specific domain hierarchy. Our approach is functionality oriented, that is, the activities are focused on building functionalities of a new or existing domains taking into account available components of the model. In Figure 2, we can see the process split into two main branches (from the second to the fourth activities), which respectively denote the task of reusing or creating domain assets. Thus, the process showed in the Figure is defined from the point of view of creating new functionalities for a new subdomain, considering existing domain assets. More specifically, the five activities of the process are:

1. *Processing requirements*: In this step we create the *domain requirements* component of the *domain engineering* level of DT&S_SPL (RM). The inputs are the *requirements of the new subdomain* provided by expert users in the area, and all the information available from the taxonomies and standards on the domain and subdomain (*domain taxonomies and standards* level of DT&S_SPL (RM)). This last information is useful for determining commonalities in the subdomain. The output of this step is a list of the suggested *domain requirements* that must be considered in the SPL.

2. *Building the taxonomy*: This step is responsible for building *domain services* and the *domain taxonomy* of the *domain engineering* level of DT&S_SPL (RM). The inputs of this step are the *suggested requirements* obtained in the previous step, and the *pre-existing taxonomy* created for other domains or subdomains (from the *domain taxonomies and standards* level and the existing repositories). The output of this step is the new taxonomy containing the list of added/reused services of the new subdomain. This step is subdivided into two substeps:

   (a) *Reusing services*: Here, it is important to determine which of the suggested requirements can be translated into a service already defined in the pre-existing taxonomy. To do so, the requirement must be searched into the reusable artifact repositories available in the *knowledge assets* level.

   (b) *Adding services*: When the suggested requirements cannot be matched to a service of the pre-existing taxonomy, the requirements must be added. This addition is not trivial because requirements can be added as a completely new category of the taxonomy or as a new specialization of another pre-existing service (or category). The new taxonomy is again stored in the repositories.

3. *Building functionalities*: In this step, we must create *design artifacts* of the *domain engineering* level of DT&S_SPL (RM). Based on the outputs of the previous step (list of services added/reused in the new taxonomy) and *functional datasheets*[10], a *reference architecture* must be designed. Functional datasheets are software artifacts that represent interactions among (common and variable) services of the taxonomy, and the reference architecture provides a preliminary structure for these interactions. Thus, the output of this step is the set of added/reused functional datasheets for the new subdomain conforming to the reference architecture. As in step 2, it is important to analyze whether the functionality is already defined for previous subdomains or it is a new one. Thus, this step is subdivided into two substeps:

   (a) *Reusing functionalities*: We must identify the functionalities that can be reused. For reusing a functionality, the set of preexisting functional datasheets must be analyzed in order to determine whether the new functionality can be reused completely, partially or it is not possible. To do so, the functionality must be searched in the repositories of the *knowledge assets* level.

   (b) *Designing new functionalities*: A new functionality is designed when it cannot be reused completely. Sometimes, it is possible to reuse a functionality only partially, in this case it is necessary to create a new one with the set of not reused services. The new functionality must also be stored in the repositories.

4. *Deriving components*: In this step, the functional datasheets are analyzed to generate an *abstract component structure* compliant with the reference architecture. This structure is also part of the *domain engineering* level of DT&S_SPL (RM). Thus, as in the previous one, it is important to analyze whether the components' structure can be reused or we have to derive new one. This step is subdivided into two substeps:

   (a) *Reusing components*: In general, when functional datasheets are reused (in the previous step) there exists a component structure for supporting them. These structures must be selected from the repositories (of the *knowledge assets* level of DT&S_SPL (RM)) and organized into the reference architecture.

   (b) *Deriving new components*: When new functional datasheets are created in the previous

[10]Datasheets are design artifacts created to model functionalities including commonalities and variabilities. For a detailed description of datasheets, please refer to [14]

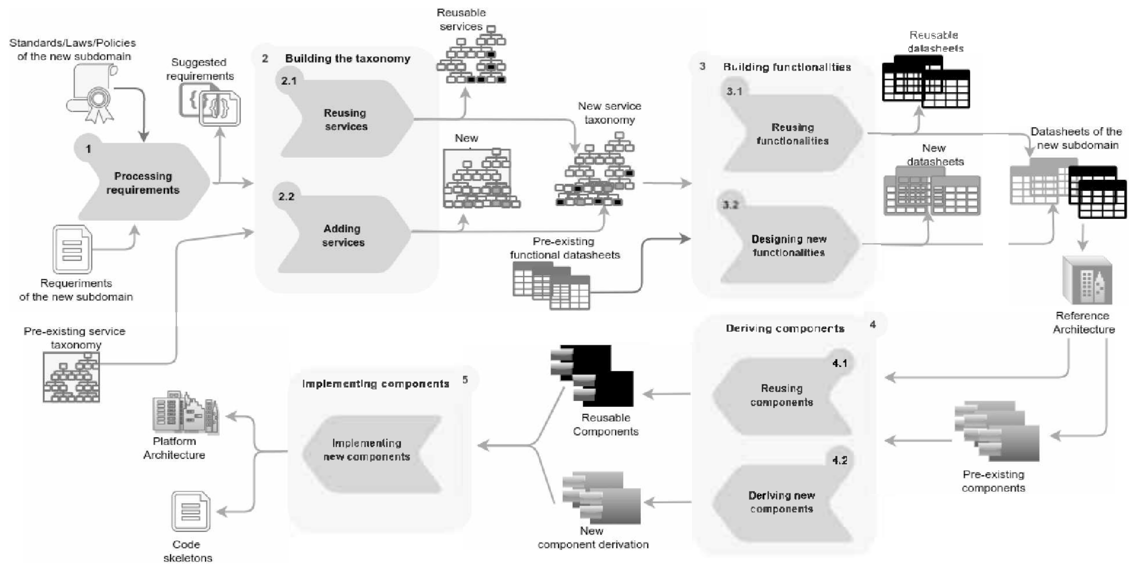

Figure 2: Process for building functionalities based on reusable assets

step, we must derive a new abstract component structure according to the existing one and organize it into the reference architecture. The new structure must be also stored in the repositories.

5. *Implementing components*: In the previous step the reference architecture is used to define software components and their interactions in an abstract way. In this step, we must create a *concrete platform architecture* (as also part of the *domain engineering* level of DT&S_SPL (RM)) to implement all constraints defined during the design activity. This generates a set of code skeletons that represent each concrete software component resulting in the platform architecture. The concrete architecture requires a definition of the all elements that can be identified in the concrete component structures. This includes the component structural information, a loose coupling communication schema, and metadata regarding taxonomy services and variability management.

It is important to highlight that every domain asset generated is stored in the reusable artifacts repositories in order to manage reusability. This repository is used each time an asset is searched for reusability.

5 DT&S_SPL (RM) Instantiation: Cases from the field

In this section, we show the instantiation of DT&S_SPL (RM) when creating SPLs. In our case, we started from the geographic domain and specialized it into more specific subdomains. It is important to highlight that the geographical domain [30] is broad in the sense it includes general aspects applicable to a huge number of products within this domain. For instance, we can find common services such as map panning and zoom, edition of geographic features, layer management, etc. Thus, the creation of an SPL in the geographic domain would be impracticable, since the amount of variability that would have to be defined within each service will become unmanageable. So, it is logical to think about the division into different subdomains, where each one has certain characteristics of the geographic domain, and some special characteristics included in more specific subdomains.

Our proposal is addressed to define common aspects of generic domains (as the geographical one) and specialized into more specific ones sharing and reusing services and functionalities. To do so, we have worked along with organizations involved in the marine ecology [26] and paleobiology [14] subdomains. We analyzed these subdomains by studying upper domains and standards that were arranged into a domain hierarchy. In Figure 3 we can see the standards, domain and subdomains studied for the model instantiation; for example, for the geographic domain we can see the use of several standards defined by the International Organization for Standardization[11] (ISO) and the Open Geospatial Consortium[12] (OGC). The ISO, by means of the ISO Technical Committee 211[13] (ISO/TC211), and the OGC have proposed a series of standards to provide rules and methods for creating interoperable geographic systems. Among the wide range of defined standards, we particularly applied those defining geographic services, a reference architecture, and spatial,

[11] http://www.iso.org
[12] http://www.opengeospatial.org/
[13] http://www.isotc211.org/

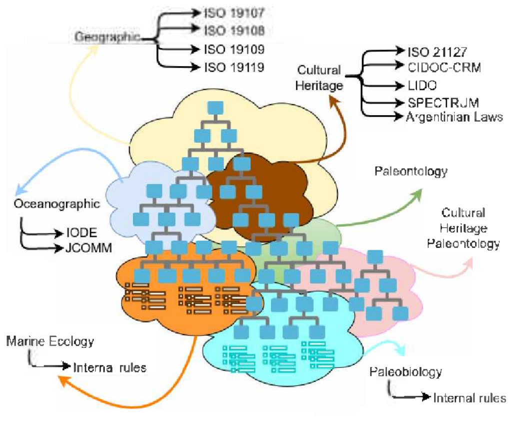

Figure 3: Domains and standards used by the DT&S_SPL (RM) instantiation

temporal and thematic representations. Following, for the cultural heritage subdomain (as an upper domain for the paleobiology one) we applied, for example, the ISO 21127 standard[14] proposing a reference ontology, CIDOC-CRM proposing a conceptual reference model, and LIDO[15] providing an XML schema for describing museum objects. At the same time, for the oceanography subdomain, as an upper domain for marine ecology, we used the information provided by IODE (International Oceanographic Data and Information Exchange[16]) and the Joint WMO-IOC Commission for Oceanography and Marine Meteorology (JCOMM[17]). Finally, in Figure 3 we can see that also internal rules/regulations and/or Argentinian laws were used for extracting information about the subdomains. This information was useful to provide domestic standard processes and regulations of the activities within the organizations. For example, the Argentinean Law 25743[18] defines the way in which cultural heritage must be acquired, moved, and preserved by specific organizations (such as museums) and the state.

Considering the domains and subdomains of our cases, Figure 4 shows the two first levels of the instantiated model. The first one (starting from the bottom) shows the technologies used in order to support the activities for building the software artifacts of the SPLs. For example, at a database level, we applied mongoDB[19] for storing reusable artifacts, such as taxonomies and datasheets that were translated to JSON files, and PostGIS[20] for storing the data of the generated products. For the analysis of the variability models (included in our datasheets) we used our own tool named SeVaTax [31].

Following, in the second level, we can see the domain taxonomies and standards applied. As we aforementioned, in Figure 3 we used standards for the geographic domain and for more specific ones when analyzing the marine ecology and paleobiology subdomains.

In Figures 5 and 6 we can see the main software artifacts built during the domain engineering phase of an SPL development. These artifacts have been created by applying the process showed in Figure 2, in which for each new subdomain, the software artifacts can be reused, extended or created. In our cases, we firstly created an SPL in the marine ecology subdomain [15, 26]. To do so, we started from the geographic domain and specialized it into a set of services, functional datasheets and reusable components. Some years later, we started to work in the paleontology domain; so we applied the same process to generate an SPL into the paleobiology subdomain [14]. Here, we considered the software artifacts previously generated in order to reuse and/or extend services, datasheets and components for this new subdomain. At this point, we were also able to perform some analyses in order to obtain indicators about reusability. As more subdomains are defined into the hierarchy, reusability benefits will be greater.

As an example of the software artifacts generated, in Figure 5 we show the reference architecture, based on the ISO 19119 standard. As we can see, and according to the standard, we defined a three-layer reference architecture involving a *human interaction layer*, responsible for the interaction with the user; a *user processing layer*, responsible for the functionality required by the user; and a *model/information management layer*, responsible for physical data storage and data management. Each layer groups a set of services located into the taxonomy. Thus, in the Figure we also show a little part of the service taxonomy. This part shows some of the human interaction services belonging to the geographic domain (in black), services of the marine ecology subdomain (in red), and services of the paleobiology subdomain (in blue). These services were specialized to show spatial and thematic data to the users by using a map (with polygons in this case), a tabular form, or labels. At the same time, in the right side of the Figure, we can see the knowledge assets that are queried in order to see if a service in a specific subdomain can be reused, or otherwise we have to create a new one as an specialization of existing ones.

Following, in Figure 6 we can see a simplified view of datasheets for the *data visualization* functionality. The first datasheet (Figure 6a) shows this functionality in a general way, that is, by using services of the

[14]Information and documentation – A reference ontology for the interchange of cultural heritage information - ISO 21127:2014

[15]LIDO – Lightweight Information Describing Objects Version 1.0 – `http://network.icom.museum/cidoc/working-groups/lido/what-is-lido/`

[16]`https://www.iode.org/`

[17]`https://www.uk-ioc.org/JCOMM`

[18]Ley de Protección del Patrimonio Arqueológico y Paleontológico - `http://servicios.infoleg.gob.ar/infolegInternet/anexos/85000-89999/86356/norma.htm`

[19]`https://www.mongodb.com/`

[20]`https://postgis.net/`

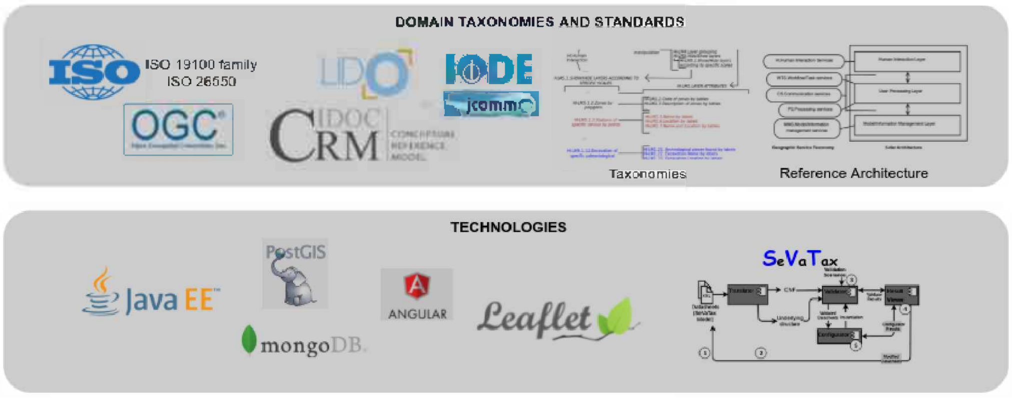

Figure 4: The two firsts levels instantiated according to the domain and subdomains under study

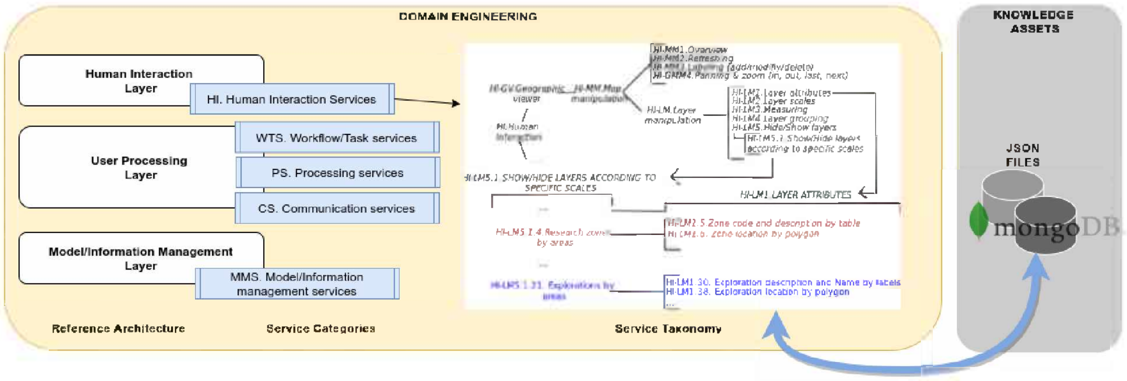

Figure 5: Part of the components generated at the domain engineering level

taxonomy belonging to the geographic domain. Then, these services were specialized into specific ones when required. At the same time, variability was defined in order to provide options for visualizing data as polygons in a map, labels, or in a tabular form. The two other datasheets show specializations for each subdomain. In Figure 6b) we can see the visualization of research zones in which the variability is defined as polygons (as mandatory), and tables and/or labels (as optional). In Figure 6c) for the paleobiology subdomain, we specialized the functionality in order to show explorations that can be visualized by implementing the same options as before. Thus, in this simplified example, we can see some components of the domain engineering level built by our process (Figure 2).

Finally, for the application engineering level we instantiated the SPLs into different products. In our case, for the marine ecology subdomain we derived two products for different institutions – one for the CENPAT[21] center, and the other for the IBMPAS[22] institute. For the paleobiology subdomain we derived also two products for the MCN[23] and the MOZ [24] museums.

In Figure 7a) and b) we show two instantiations of two products, one in the marine ecology subdomain (CENPAT center) and the other in the paleobiology subdomain (MCN museum). The instantiated datasheets show the variabilities selected. At the same time, in Figure 7b) we can see the concrete architecture proposed for implementing the reference architecture presented in Figure 5. Here, each reference layer was mapped to both client- and server-side components. Thus, for implementing this concrete architecture, we used a Client-Server pattern by locating user interaction and lightweight processing in client-side, and heavy processing and querying the database on server-side [32].

Finally, in Figure 8 we can see the results of implementing the *data visualization* functionality for the four products, each of them showing different ways of visualizing explorations for the paleobiology subdomain (Figure 8a) and b)), and research zones for the

[21]Centro Nacional Patagónico – http://www.cenpat.edu.ar/

[22]Instituto de Biología Marina y Pesquera – http://ibmpas.org/

[23]Museo de Ciencias Naturales - Universidad Nacional del Comahue - https://extension.uncoma.edu.ar/index.php/paginas/museo-de-ciencias-naturales/

[24]Museo Olsacher de Ciencias Naturales - http://museoolsacherzapala.blogspot.com/

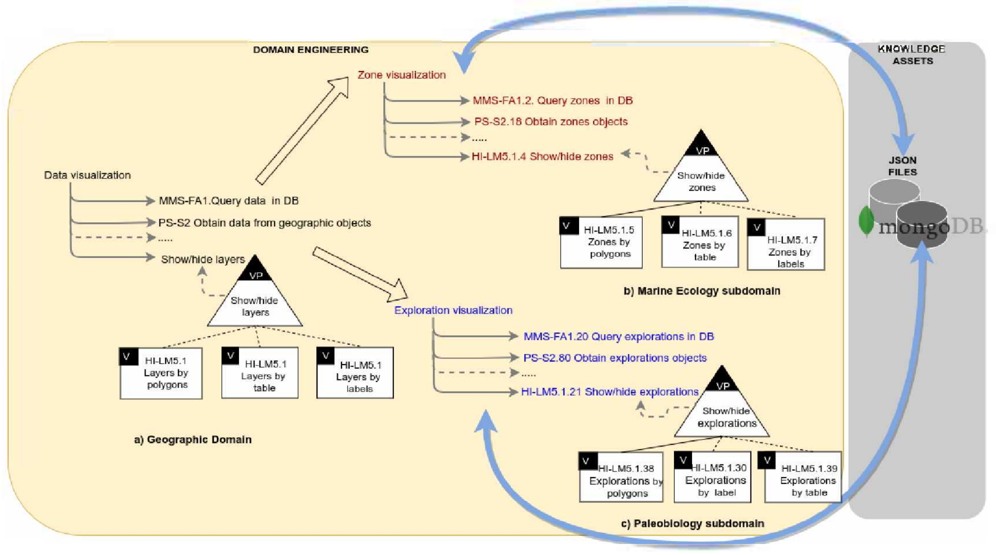
DOMAIN ENGINEERING
Zone visualization
MMS-FA1.2. Query zones in DB
PS-S2.18 Obtain zones objects
.....
HI-LM5.1.4 Show/hide zones
VP
Show/hide zones
V
HI-LM5.1.5 Zones by polygons
HI-LM5.1.6 Zones by table
HI-LM5.1.7 Zones by labels
b) Marine Ecology subdomain
Data visualization
MMS-FA1.Query data in DB
PS-S2 Obtain data from geographic objects
Show/hide layers
Show/hide layers
HI-LM5.1 Layers by polygons
HI-LM5.1 Layers by table
HI-LM5.1 Layers by labels
a) Geographic Domain
Exploration visualization
MMS-FA1.20 Query explorations in DB
PS-S2.80 Obtain explorations objects
HI-LM5.1.21 Show/hide explorations
Show/hide explorations
HI-LM5.1.38 Explorations by polygons
HI-LM5.1.30 Explorations by label
HI-LM5.1.39 Explorations by table
c) Paleobiology subdomain
KNOWLEDGE ASSETS
JSON FILES
mongoDB

Figure 6: Datasheets generated according to the domain and subdomains at the domain engineering level

marine ecology subdomain (Figure 8c) and d)).

6 Analyzing influencing factors

In previous work, we have assessed our proposal from the perspective of facilitating intra- and inter-domain reusability; i.e. when the SPL platform was generated, and when new products were derived for different subdomains [12, 13, 14, 15]. At the same time, some qualitative analyses have already been performed, such as time-to-market and costs [14, 15, 26]. These evaluations were aimed at assessing our reference architecture with respect to relevant aspects (quality attributes), such as those proposed in [33]. Following with these analyses and assessments, here we are focused on analyzing influencing factors that might reveal strengths and pitfalls of our approach, similarly to proposals in [34, 35]. That is, we are focused on the way our proposal transfers knowledge and provides factors helping on the application of an SPL development.

In Table 1, we list some factors that we consider might affect SPL development considering activities and assets of the two phases (domain and application) and people involved in the processes. Therefore, factors are grouped into three categories, possibly containing some same factors, but analyzed from different perspectives (such as the case of *Use of reusable assets*).

This list does not intent to record all possible factors, but just a set of interesting ones. For instance, we have included a factor named *Use of standards* in the Domain Engineering group, because our proposal relies on the combination of standards and committed service taxonomies to help stakeholders define a common vocabulary, and use the same rules and recommendations. Then, collecting opinions about the usefulness of making this information available would make an interesting contribution. Similarly, the *Use of semantic resources* considers the possibility of applying, extending and even creating taxonomies, datasheets, reference architectures, etc. Here usefulness is interpreted

	Influencing factors
Domain Engineering - SPL Platform	• Use of standard information • Use of semantic resources • Variability definition • Use of base technology • Use of reusable assets • Well-established process
Application Engineering - Products	• Use of reusable assets • New requirement management • Variability instantiation • Use of base technology • Well-established process
People	• SPL development knowledge • Domain knowledge • Proactive and good predisposition to work in a team • Previously involved in this SPL development • Knowledge about the base technologies • Good communication and interpersonal skills

Table 1: Influencing factors to be assessed when developing SPLs under DT&S_SPL (RM)

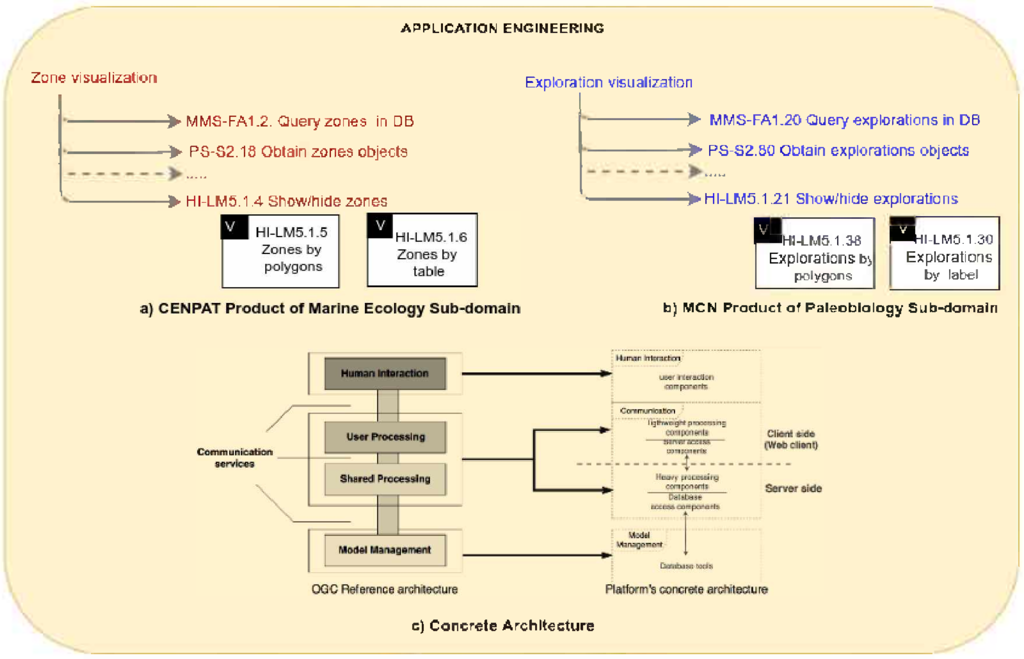
APPLICATION ENGINEERING
Zone visualization
MMS-FA1.2. Query zones in DB
PS-S2.18 Obtain zones objects
.....
HI-LM5.1.4 Show/hide zones
V
HI-LM5.1.5 Zones by polygons
V
HI-LM5.1.6 Zones by table
a) CENPAT Product of Marine Ecology Sub-domain
Exploration visualization
MMS-FA1.20 Query explorations in DB
PS-S2.80 Obtain explorations objects
.....
HI-LM5.1.21 Show/hide explorations
V
HI-LM5.1.38 Explorations by polygons
V
HI-LM5.1.30 Explorations by label
b) MCN Product of Paleobiology Sub-domain
Human Interaction
User Processing
Shared Processing
Model Management
Communication services
OGC Reference architecture
Human Interaction
user interaction components
Communication
lightweight processing components
server access components
Heavy processing components
Database access components
Client side (Web client)
Server side
Model Management
Database tools
Platform's concrete architecture
c) Concrete Architecture

Figure 7: The application engineering level instantiated for deriving products

as a trade-off among different influencing variables, such as effort and time needed for creation, extension, and so on.

As another example, in the Application Engineering group, we defined factors such as *New Requirements Management* to consider the case of newly adding application specific requirements; or the *Use of base technology* to consider influences from the use of a precoded and component-based platform, and the use of preestablished technology as well. Finally, the third group addresses factors from people who participate as stakeholders of the development process (software engineers and/or related profiles). For instance, we considered general factors, such as *Interpersonal skills*, and more specific ones, such as *Previously involved in this SPL*, meaning the existence of previous experience working with our proposal.

In order to subjectively assess these factors, we divided developers (and software engineers) of our team into three groups. The first one included four developers, who worked on the development of the general SPL platform (modeling the geographic domain and reusable assets) during 2010-2016. The second group included five developers, who worked on the development of the SPL platform in the marine ecology subdomain during 2012-2016. Three of these developers belonged to the first group as well. And finally, the third group included five developers working on the paleobiology subdomain during 2017-2021. All members of these three groups were invited to assess the factors defined in Table 1, according to the scale defined in Figure 9. We used a Likert scale for subjectively assessing usefulness of each factor, which was averaged for each group to show the summarization in Table 2.

As we can see, the results of Group 1 were the worst with respect to others. This is because members of Group 1 were the firsts working in the geographic domain and building semantic resources and reusable assets. The other groups could later take advantage from the artifacts already created by extending or just specializing them. However, it is worth noting that although members of Group 1 built the first version of the platform (*Use of reusable assets* is not applicable), they satisfactory started to reuse at application engineering level by deriving products from the recently created platform.

Group 1 expended a lot of effort to create semantic resources that helped improve understandability and reduce ambiguity at the same time. Specifically, service taxonomies and functional datasheets were combined to support the instantiation activity during the product derivation process (*Variability instantiation*). Additionally, services included within mandatory and optional datasheets were classified to guide stakeholders in selecting their specific requirements. As a limitation of the use of these resources, we can again cite here the expertise needed for their understanding. However, although this is true, we found more problems when stakeholders required new functionalities to be included in a new product (*New requirement management*), as this factor requires the intervention of at least two types of stakeholders (expert users and developers). Then, knowledge needed was deeper than in the

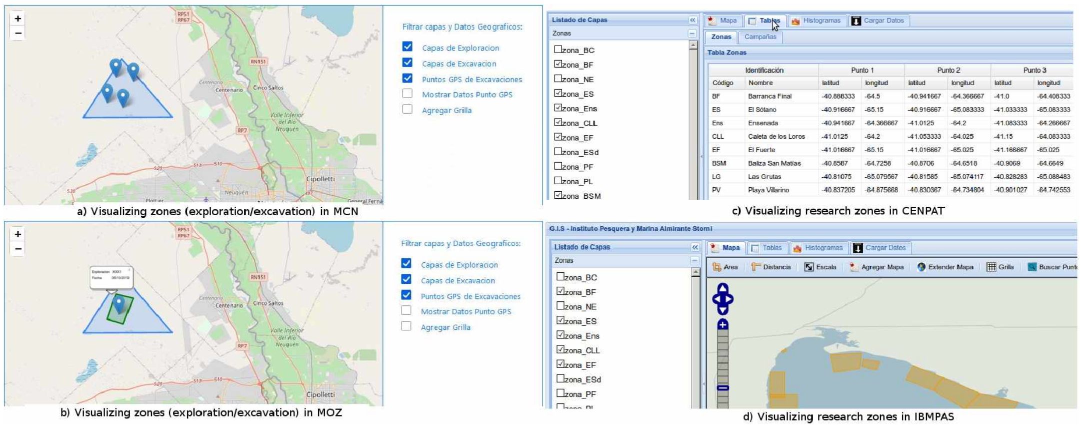
Filtrar capas y Datos Geograficos:
Capas de Exploracion
Capas de Excavacion
Puntos GPS de Excavaciones
Mostrar Datos Punto GPS
Agregar Grilla
a) Visualizing zones (exploration/excavation) in MCN
b) Visualizing zones (exploration/excavation) in MOZ
Listado de Capas
Zonas
zona_BC
zona_BF
zona_NE
zona_ES
zona_Ens
zona_CLL
zona_EF
zona_ESd
zona_PF
zona_PL
zona_BSM
Mapa
Tablas
Histogramas
Cargar Datos
Zonas
Campañas
c) Visualizing research zones in CENPAT
G.I.S - Instituto Pesquera y Marina Almirante Storni
Area
Distancia
Escala
Agregar Mapa
Extender Mapa
Grilla
d) Visualizing research zones in IBMPAS

Tabla Zonas

Identificación		Punto 1		Punto 2		Punto 3	
Código	Nombre	latitud	longitud	latitud	longitud	latitud	longitud
BF	Barranca Final	-40.888333	-64.5	-40.941667	-64.366667	-41.0	-64.408333
ES	El Sótano	-40.916667	-65.15	-40.916667	-65.083333	-41.033333	-65.083333
Ens	Ensenada	-40.941667	-64.366667	-41.0125	-64.2	-41.083333	-64.266667
CLL	Caleta de los Loros	-41.0125	-64.2	-41.053333	-64.025	-41.15	-64.083333
EF	El Fuerte	-41.016667	-65.15	-41.016667	-65.025	-41.166667	-65.025
BSM	Baliza San Matías	-40.8587	-64.7258	-40.8706	-64.6518	-40.9069	-64.6649
LG	Las Grutas	-40.81075	-65.079567	-40.81585	-65.074117	-40.828283	-65.088483
PV	Playa Villarino	-40.837205	-64.875668	-40.830367	-64.734804	-40.901027	-64.742553

Figure 8: Different user interfaces for the *data visualization* functionality in the four derived products

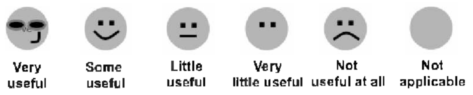

Figure 9: Scale for assessing factors during SPL development

	Influencing Factors	Group1	Group2	Group3
Domain Engineering - SPL Platform	Use of standard information	😎	😎	😎
	Use of semantic resources	●	🙂	😎
	Variability definition	[illegible]	[illegible]	🙂
	Use of base technology	●	😐	😐
	Use of reusable assets	●	🙂	😎
	Well-established process	[illegible]	[illegible]	😎
Application Engineering - Products	Use of reusable assets	😐	🙂	🙂
	New requirement management	[illegible]	😐	😐
	Variability instantiation	🙂	🙂	😎
	Use of base technology	😐	🙂	🙂
	Well-established process	🙂	😎	😎
People	SPL development knowledge	☹️	🙂	😐
	Domain knowledge	😎	😎	😎
	Proactive and good predisposition to work in a team	😎	😎	😎
	Previously involved in this SPL development	●	😎	😐
	Knowledge about the base technologies	🙂	🙂	🙂
	Good communication and interpersonal skills	😎	😎	😎

Table 2: Judgements (avg) for each factor and group

case of the instantiation process, and more valuable when *Domain knowledge* was higher. Undoubtedly, this fact made derivation time longer.

The assessment introduced here revealed perhaps quite obvious conclusions, such as detecting that the more expertise exists in using the proposal, the more effective and satisfying it is. In addition, we are aware that opinions might be biased in favor of DT&S_SPL

(RM), since participants are all members of our team. However, we expect judgements showed in Table 2 point to possible strengths, such as structuring around standardized assets; and pitfalls, such as the needs of building these assets.

7 Conclusion and Future Work

SPL development and management usually requires a considerable effort to deal with complex and multiple domains, many times difficult to approach as well. Our proposal introduces additional elements for dealing with this complexity – semantic resources as taxonomies and standards, which support a reference model for developing intra- and inter-domain SPLs.

In this paper, we have summarized our proposal showing some cases from the field, and preliminary elaborating some factors that might influence the process. Future work is now focused on extending functionality to include services related to data analysis, since geographic systems are inherently great data collectors. Variety in big data systems gives us a space for exploring taxonomies and standards; and in this case, we will approach another subdomain – sustainable agriculture – in cooperation with a national institution in Argentina (INTA[25]). At the same time, we are organizing the set of supporting tools we have developed to facilitate the semantic resources manipulation. This set of open-source tools[26] and some application guidelines will help transfer the model and process to possibly interested SPL developers.

Competing interests

The authors have declared that no competing interests exist.

Authors' contribution

Both authors contributed equally to this research. They discussed the results and contributed to the final manuscript. Both authors read and approved the final manuscript.

Acknowledgements

This work is partially supported by the UNComa project 04/F009 "Reuso de Software orientado a Dominios: Parte II" part of the program "Desarrollo de Software Basado en Reuso: Parte II

References

[1] J. Neighbors, *Software Construction Using Components*. PhD Dissertation, Department of Information and Computer Science, University of California, Irvine, 1981.

[2] R. Prieto-Díaz and P. Freeman, "Classifying software for reusability," *Software, IEEE*, vol. 4, pp. 6 – 16, 02 1987.

[3] R. Prieto-Díaz, "Domain analysis: an introduction," *SIGSOFT Softw. Eng. Notes*, vol. 15, pp. 47–54, Apr. 1990.

[4] G. Arango, "Domain analysis: From art form to engineering discipline," *SIGSOFT Softw. Eng. Notes*, vol. 14, p. 152–159, apr 1989.

[5] G. Arango, "Domain Analysis," in *Encyclopedia of Software Engineering* (J. Marciniak, ed.), vol. 1, pp. 424–434, WILEY, 1994.

[6] K. Czarnecki, *Domain Engineering*, ch. 3. John Wiley & Sons, Inc., 2002.

[7] K. Pohl, G. Böckle, and F. van der Linden, *Software Product Line Engineering: Foundations, Principles and Techniques*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005.

[8] F. van der Linden, K. Schmid, and E. Rommes, *Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.

[9] G. Trujillo-Tzanahua, U. Juárez-Martínez, A. Aguilar-Lasserre, and M. Cortés-Verdín, "Multiple software product lines: applications and challenges," in *Trends and Applications in Software Engineering*, pp. 117–126, Springer, 2018.

[10] K. Manikas and K. M. Hansen, "Software ecosystems – a systematic literature review," *Journal of Systems and Software*, vol. 86, no. 5, pp. 1294 – 1306, 2013.

[11] G. Holl, P. Grünbacher, and R. Rabiser, "A systematic review and an expert survey on capabilities supporting multi product lines," *Information and Software Technology*, vol. 54, pp. 828–852, Aug. 2012.

[12] V. Saldano, A. Buccella, and A. Cechich, "Discovering geographic services from textual use cases," *Journal of Computer Science and Technology*, vol. 10, pp. 61–67, 2010.

[13] P. Pernich, A. Buccella, A. Cechich, M. Arias, M. Pol'la, M. Doldan, and E. Morsan, "Product-line instantiation guided by subdomain characterization: a case study," *Journal of Computer Science and Technology*, vol. 12, pp. 116–122, 2012.

[14] A. Buccella, A. Cechich, J. Porfiri, and D. Diniz Dos Santos, "Taxonomy-oriented domain analysis of gis: A case study for paleontological software systems," *ISPRS International Journal of Geo-Information*, vol. 8, no. 6, 2019.

[15] A. Buccella, A. Cechich, M. Arias, M. Pol'la, S. Doldan, and E. Morsan, "Towards systematic software reuse of GIS: Insights from a case study," *Computers & Geosciences*, vol. 54, no. 0, pp. 9 – 20, 2013.

[16] R. McCain, "Reusable software component construction - a product-oriented paradigm," in *5th AIAA/ACM/NASA/IEEE Computers in Aerospace Conference*, pp. 125–135, 1985.

[17] J. Bosch, *Design and use of software architectures: Adopting and evolving a product-line approach*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 2000.

[18] P. C. Clements and L. M. Northrop, *Software Product Lines : Practices and Patterns*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2001.

[25] https://www.argentina.gob.ar/inta

[26] Some stored at https://github.com/IngSisFAI/ProyectoReuso

[19] L. M. Northrop and L. G. Jones, "Introduction to software product lines adoption," in *Software Product Line Conference (SPLC), 2011 15th International*, pp. 359–359, Aug 2011.

[20] S. Cohen, "Ontology and taxonomy of services in a service-oriented architecture," *MSDN Libary Infrastructure Architectures*, vol. 11, no. 11, 2007.

[21] O. Gambino, L. Rundo, V. Cannella, S. Vitabile, and R. Pirrone, "A framework for data-driven adaptive gui generation based on dicom," *Journal of Biomedical Informatics*, vol. 88, pp. 37–52, 2018.

[22] S. Ariyanti and Kautsarina, "A proposed the internet of things (iot) framework for health sector in indonesia," in *IEEE Region Ten Symposium (Tensymp)*, pp. 282–286, 2018.

[23] V. Vyatkin, "Software engineering in industrial automation: State-of-the-art review," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 3, pp. 1234–1249, 2013.

[24] H. Kohl, "Industry-Specific Standards for Management Systems," in *Standards for Management Systems*, Management for Professionals, Springer, December 2020.

[25] H. Kohl, *Standards for Management Systems: A Comprehensive Guide to Content, Implementation Tools, and Certification Schemes*. 01 2020.

[26] A. Buccella, A. Cechich, M. Pol'la, M. Arias, S. Doldan, and E. Morsan, "Marine ecology service reuse through taxonomy-oriented SPL development," *Computers & Geosciences*, vol. 73, no. 0, pp. 108 – 121, 2014.

[27] M. Pol'la, A. Buccella, and A. Cechich, "Analysis of variability models: a systematic literature review," *Software and Systems Modeling*, vol. 20, pp. 1–35, 08 2021.

[28] J. Feljan, L. Lednicki, J. Maras, A. Petricic, and I. Crnkovic, "Dices technical report: Classification and survey of component models," tech. rep., Unity through Knowledge Fund, 2009.

[29] I. Hunink, E. Rene, S. Jansen, and S. Brinkkemper, "Industry taxonomy engineering: the case of the european software ecosystem," in *Fourth European Conference on Software Architecture: Companion Volume*, ECSA '10, (New York, NY, USA), pp. 111–118, ACM, 2010.

[30] P. Burrough and R. McDonnell, *Principles of Geographical Information Systems*. Oxford University Press, 1998.

[31] M. Pol'la, A. Buccella, and A. Cechich, "Automated analysis of variability models: The sevatax process," in *Computational Science and Its Applications - ICCSA 2018 - 18th International Conference, Melbourne, VIC, Australia, July 2-5, 2018, Proceedings, Part IV*, pp. 365–381, 2018.

[32] M. Arias, A. Buccella, and A. Cechich, "Smooth transition from abstract to concrete spl components: a client-server implementation for the geographic domain," in *Proceedings of the IEEE ARGENCON'16: El Congreso Bienal de la Sección Argentina de IEEE*, (Buenos Aires, Argentina), GRSS: IEEE Geoscience and Remote Sensing Society, 2016.

[33] S. Martínez-Fernández, C. Ayala, X. Franch, H. Marques, and D. Ameller, "A framework for software reference architecture analysis and review," in *Memorias del X Workshop Latinoamericano Ingeniería de Software Experimental, ESELAW 2013*, 01 2013.

[34] D. Smite and C. Wohlin, "Strategies facilitating software product transfers," *IEEE Software*, vol. 28, no. 5, pp. 60–66, 2011.

[35] L. Northrop, "Software product lines: reuse that makes business sense," in *Australian Software Engineering Conference (ASWEC'06)*, pp. 1 pp.–3, 2006.

Citation: A. Buccella and A. Cechich. *An SPL Reference Model based on Domain Taxonomies & Standards*. Journal of Computer Science & Technology, vol. 22, no. 2, pp. 116-128, 2022.
DOI: 10.24215/16666038.22.e09
Received: February 9, 2022 **Accepted:** July 4, 2022.