# ReqGIS Classifier: A Tool for Geographic Requirements Normalization

Viviana E. Saldaño

Proyecto de Investigación Área Ingeniería de Software
Unidad Académica Caleta Olivia – Universidad Nacional de la Patagonia Austral
vivianas@uaco.unpa.edu.ar

Agustina Buccella, Alejandra Cechich

Grupo de Investigación en Ingeniería de Software del Comahue (GIISCo)
Departamento de Ciencias de la Computación
Universidad Nacional del Comahue
{abuccel, acechich}@uncoma.edu.ar

**Abstract.** Component Based Software Development (CBSD) is a development process based on components' reuse. One of the main difficulties for developers is selecting the most suitable component that fit in their development systems. In this paper we describe a software tool, named ReqGIS, which supports our methodology for improving components' identification in a geographic information environment. In particular, we introduce a new component named *AlgSim*, which completes the automation of the whole methodology. It starts analyzing user requirements specified by use cases and returns the best fitting geographic service category corresponding to those requirements.

**Keywords:** DSBC, Off-The-Shelf (OTS), GIS services, geographic component selection.

## 1. Introduction

Software reuse has been incremented during last years, becoming a common practice for software products development. In particular, Component Based Software Development (CBSD) is based on components' reuse which have been developed at different times, by different people and possibly with distinct goals of use [21]. In this context, one of the main difficulties for developers is searching and selecting the most suitable components. It is known that, a wrong component selection will impact through all the software development life cycle. Therefore, searching and selecting OTS (Off-The-Shelf) components [3] are quite important.

A key mechanism which is responsible of searching and selecting components is the mediator process. In this context, a client who requires a specific component service may interrogate a mediator service for the references to those components

which supply the required service. Another key issue is standardizing components' information. Service supply can be standardized so that compositions are stored in an easy access repository. The same should happen for services demand, which should also be expressed in standard terms to make search easier.

Thus, two models can be identified: *demand* and *supply*. The *supply model* concerns gathering and storing components' information in a repository in a standard way. On the other hand, the *demand model* involves identifying required services based on user requirements. The connection between these two models is the mediator service which is responsible of mapping the required services with components implementing them.

In this work, we are interested in geographic services which are necessary for implementing geographic information systems. In the last ten years, many GIS software companies have begun supplying software components to satisfy GIS software developers' needs. Therefore, a methodology and its supporting tool for facilitating the demand model and the identification of the correct components shall be very useful in this context.

The work presented in this paper is an extension of works previously presented in [17, 18, 19], in which we have proposed a methodology for improving the component identification process. In particular, this work is presented as a complement to the supply model presented in [6, 7, 8] where a publication service is defined to facilitate selection of requested components.

In this paper, we describe our supporting tool, named ReqGIS, which implements the process for searching and selecting geographic components automatically. Requirements of GIS developers are processed and classified according to a geographic services category. After classifying requirements, a mediation service is invoked to find references to components which fit in the required functionality. In this context, we have developed a geographic-services taxonomy, a use-case knowledge extraction process, and a supporting tool which classifies requirements according to service categories defined in the taxonomy.

This paper is organized as follows: next section describes a methodology for geographic services identification. Section 3 describes the supporting tool developed to classify geographic services. Then, in Section 4 we apply the whole process in a real example. Future work and conclusions are discussed afterwards.

## 2. Methodology for Geographic Services Identification from User Requirements

In this section we describe our methodology [18, 19] to classify services specified in textual use cases. This methodology implements the *demand model* in which a client (developer) who requires a specific component service shall ask a mediation service to find the references to those components which provide the required service category. Thus, the main goal is to identify required services from use cases in order to find the correct GIS components that provide these services. Figure 1 shows the main steps of the methodology.

As we can see, the input of the methodology are use cases. The developer provides a use case in which the main functionality required is described. In our approach, to take advantage of the natural language and to avoid ambiguities, we have analyzed use case proposals involving a restricted natural language. Thus, we have selected the proposal presented by Cockburn [4], in which templates are applied to specify the behavior within use cases. In addition, we have restricted the language in these use cases by applying a controlled natural language which structures sentences in a particular way [9]. Here, the SVDPI (Subject, Verb, Direct object, Preposition, Indirect object) pattern is applied as follows: "Sentence structure must be simple"… "Subject… verb … direct object … preposition … indirect object".

In this way, these two proposals [4, 9] are combined in order to maximize the understanding of use cases for common users and to provide, at the same time, a notation in which the automatic analysis and validation are possible.

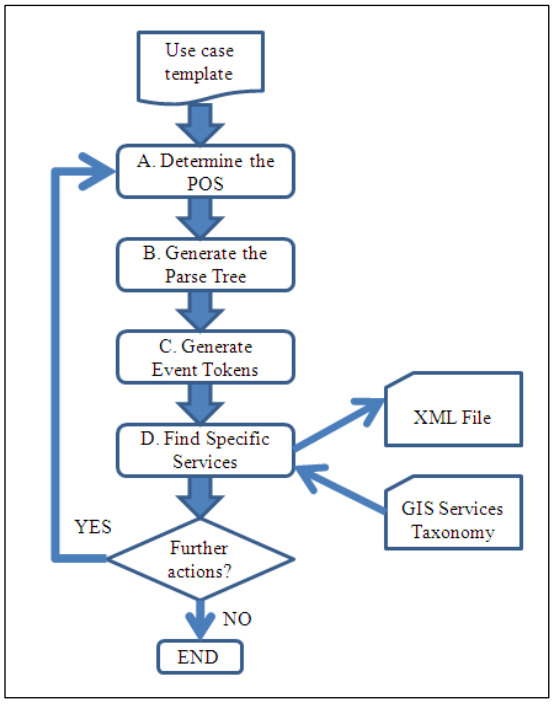


**Fig. 1.** Steps for extracting GIS services from use cases

In addition, in Figure 1 we can see a *GIS Services Taxonomy* component used to classify the GIS services. This taxonomy has been built by using the information provided by ISO 19119 std. This standard was developed by the Open Geospatial Consortium (OGC) and the International Standardization Organization (ISO). It proposes a geographic services classification that shall be used for all the systems compliant to this International Standard. The standard defines six categories grouping human interaction, model/information management, workflow/task management,

processing, communication, and system management services. In a previous work [17], we have defined this taxonomy based on the ISO 19119 std. In addition, in order to support the matching process between user requirements and services categories, we have defined a list of keywords which describe services provided by each category [19]. In Table 1 we can see part of this taxonomy. The first column of the table is the category as defined in the standard and the second and third columns denote the keywords for service description. For instance, within the Human Interaction category, main verbs to describe services here are interact, locate, manage, etc.; and the representative objects can be catalogue, map, chain, etc.

**Table** 1**.** Fragment of GIS Taxonomy

| Category | Service Description | |
|---|---|---|
| | Main Verb | Representative Object |
| Human Interaction | interact<br>locate<br>browse<br>manage<br>view<br>display<br>overlay<br>query<br>animate<br>calculate<br>edit | catalogue<br>metadata<br>feature<br>coverage<br>map<br>spreadsheet<br>service<br>chain<br>workflow<br>view<br>perspective<br>texture<br>symbol<br>structure<br>dataset |

The other component that we can see in Figure 1 is the *XML File* component which is used to store the result of the mapped service.

According to Figure 1 the main steps of our methodology are:

A. *Determining the POS (part-of-speech)*: It analyzes each word and specifies the type (verb, noun, etc.) and the role of each of them within the sentence in which they are defined.

B. *Generating the parse tree:* Different parse trees are created according to the sentences of the main scenario of the use cases.

C. *Generating event tokens*: Event tokens are created by finding main verbs and representative objects within each sentence of the parse tree.

D. *Finding specific services:* Each event token is processed to get the corresponding geographic category according to the GIS Services Taxonomy.

The methodology applies linguistic tools to build a parse tree in which actions of predefined textual use cases are identified. Then these actions are used to discover the required GIS services. The method takes a use case specification and processes each step of the main scenario (main part of use case template) by performing the A-D steps.

The software tool implementing all the methodology, named ReqGIS, was partially implemented and described in previous works [18, 19]. Steps A-C of the methodology (Figure 1) have been implemented by using FreeLing Tool Suite [5, 15]. The tool reads a sentence (of the main scenario of the use case) and returns a parse tree with necessary information to generate the Event Token (step C). However, step D had to be made manually, that is, the developer was responsible of understanding the event token and finding the specific service in the taxonomy. Therefore, in this work, we present the *AlgSim component* which completes the implementation of the ReqGIS tool. This component implements step D by using the event token as input and returning the corresponding geographic category. The result is stored in an XML file aforementioned, which shall be used to find mappings between user requirements and the information of OTS components published on the Web. With the *AlgSim* component we fully automate the whole *demand process*. In the next section we describe the ReqGIS tool in detail, and in particular the *AlgSim* component.

## 3. ReqGIS: Requirements Classification Tool for GIS Services

The main goal of the ReqGIS tool is to automate the process of classifying developers' GIS requirements and speed up the demand process. In this way clients will find the most suitable component in less time.

The requirements' classification tool has been created by reusing components available on Internet. Figure 2 shows these main components that work together in order to support the steps of our methodology (Figure 1). Following, we describe each of the ReqGIS tool's components:

**FreeLing Component**. As we have described in the last section, FreeLing [15] is an open-source multilingual language processing library providing a wide range of language analyzers for several languages. It offers text processing and language annotation facilities to natural language processing application developers, simplifying the task of building those applications. In ReqGIS, Freeling performs steps A-C of our methodology. It receives a use case main scenario step (an english sentence in SVDPI format) and returns a parse tree of the sentence, with the corresponding syntactic analysis. This parse tree is then used to build the event token, taking the words tagged as top and direct object for event token's main verb and representative object respectively.

**WordNet::Similarity Component.** It is a freely available Perl software package that makes it possible to measure the semantic similarity and relatedness between a pair of concepts [16]. It provides six measures of similarity and three measures of relatedness, all of which are based on the lexical database WordNet. One of the relatedness measures calculated is Adapted Lesk Algorithm, which is the measure

used by AlgSim component in order to find the most related category to the event token.

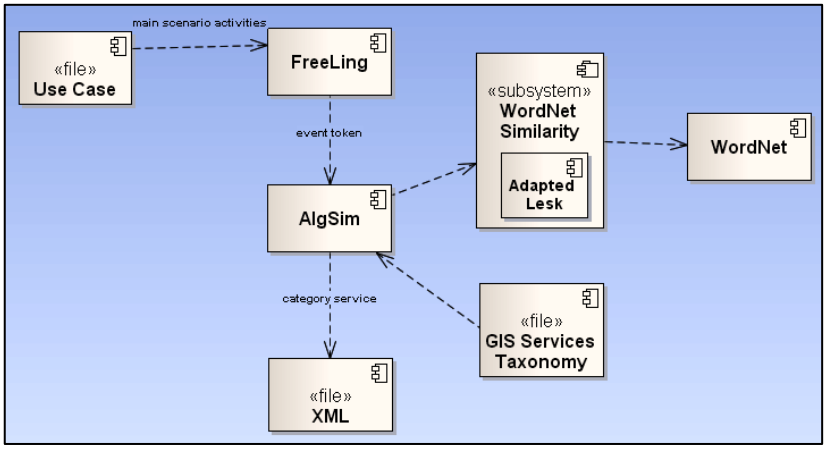


**Fig. 2.** ReqGIS component diagram.

**Adapted Lesk Algorithm.** It is a module included in WordNet::Similarity package and it is based on Lesk algorithm which disambiguates words in short phrases. Adapted Lesk Algorithm [12] measures relatedness by evaluating words' relations in WordNet.

**WordNet Component.** WordNet is a large lexical database of English [2], arranged semantically. This package provides semantic information to WordNet::Similiarity component in order to compute similarity and relatedness measures.

**AlgSim Component**. This component, written in Perl, has been developed to achieve the main goal of classifying the required services. It implements the step D of our methodology, by taking as input the event token and processing it to obtain the required geographic category service. In order to perform this task, it performs an iterative process, shown in Figure 3, accessing information stored in the GIS Services Taxonomy and using services provided by WordNet::Similarity, AdaptedLeskAlgorithm and WordNet components. In fact, it calculates the average category relatedness for each category and selects the category with the highest relatedness. In order to calculate each category average relatedness, it computes verbs' relatedness and objects' relatedness, by evaluating pairs of verbs (category verb, event token verb) and objects (category object, event token object) within each category. After selecting the most suitable category the algorithm stores the result in an XML file.

## 4. Case Study

In this section we present a case study in order to show how our methodology and the classification tool work. The specification was provided by a local organization of

Comodoro Rivadavia in Argentina. As it was in the Spanish language, we have translated it by considering our specification of use cases [18, 19].

```
given eventToken (verb, object)
for each taxonomy category {
    for each category verb {
        calculate SIMILARITY(categoryVerb, tokenVerb)
        }
        calculate categoryVerbsSimilarityAverage
    for each category object {
        calculate SIMILARITY(categoryObject, tokenObject)
        }
        calculate categoryObjectsSimilarityAverage
    calculate categorySimilarityAverage
    }
category = category with highest categorySimilarityAverage
store (verb, object, category)
return category
```

**Fig. 3.** Similarity algorithm to calculate highest relatedness

Table 2 shows a resultant use case in which a service to modify a coordinate of an electric line is presented.

**Table 2.** Fragment of Textual Use Case

| | | |
|---|---|---|
| Main Scenario | 1 | User selects electric line |
| | 2 | User modifies coordinate attribute |
| | 3 | System displays updated electric line |

This use case is the input of the ReqGIS tool, which applies the four steps of our methodology (Figure 1) to each action defined in the main scenario of the use case specification.

Steps A-C are performed together by the *FreeLing* component. Considering the second action in the main scenario of the use case *"User modifies coordinate attribute"*, the component creates a parse tree classifying each word of the sentence. Figure 4 shows this tree. Then, ReqGIS creates the *Event Token* by finding main verbs and representative objects within each sentence of the parse tree. That is, the tool takes the root node in the parse tree tagged as **top** as the *event token main verb*, i.e. "modify", and the sentence direct object as the *event token representative object*, which is the word tagged as **dobj** in the parse tree, i.e. "attribute". So, the resultant event token is *"modify attribute"*.

Finally, in step D, the *AlgSim* component takes the event token as input, and after processing, it returns the name of the corresponding service category needed to accomplish our functional requirements. In addition the component also stores this result in an XML file.

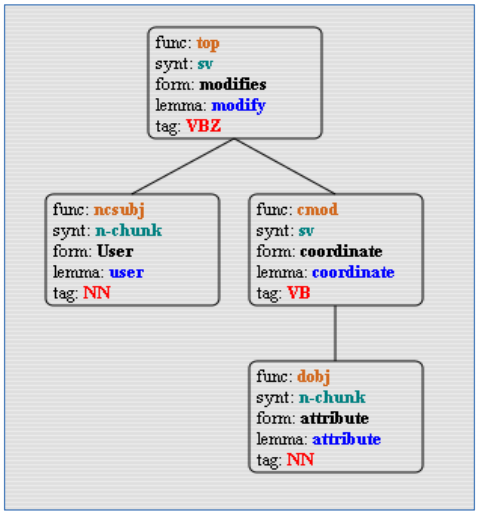


**Fig. 4.** Parse Tree for: "User modifies coordinate attribute"

AlgSim's user interfaces are shown in Figure 5. In the first one, the user enters *event token's main verb* and *event token's representative object*. The second user interface shows the services category matching the event token, in this case, *Processing-Metadata Services* category.


**Fig. 5.** AlgSim user interfaces

In order to appreciate in more detail the process implemented by the AlgSim component, and in particular the relatedness measuring, we include a sample table (Table 3) with the scores values for each combination of Event Token Verb / Category Verb and Event Token Object / Category Object. For example to compute Category Verbs Average, the process calculates relatedness between each verb in Human Interaction category of the GIS Service Taxonomy (Table 1) and the "modify" verb (which is the Token Verb). For instance, for the first pair of verbs

(interact, modify) we can see that the calculated relatedness is 158. The same process is applied to each pair of verbs and objects of the use case against the taxonomy.

These measures are taken for all categories in the GIS Services Taxonomy, calculating the average value for each category. The chosen category is the one having the highest average relatedness value. In our case study, the *AlgSim* component selects the *Processing-Metadata Services* category. The mediator service will have to map this category against the offered services to determine the components that provide them.

**Table 3.** Examples of relatedness measures between verbs and objects in category *Human Interaction* and event token "*modify attribute"*

| Category | Category Verbs | Token Verb | Mea-sure | Category Verbs Average | Category Objects | Mea-sure | Token Object | Category Objects Average |
|---|---|---|---|---|---|---|---|---|
| Human Interaction | interact | modify | 158 | 72,54 | Catalogue | 35 | attribute | 54 |
| | locate | | 30 | | Metadata | 8 | | |
| | browse | | 23 | | Feature | 127 | | |
| | manage | | 12 | | Coverage | 8 | | |
| | view | | 95 | | Map | 76 | | |
| | display | | 70 | | Spreadsheet | 13 | | |
| | overlay | | 30 | | Service | 38 | | |
| | query | | 33 | | Chain | 31 | | |
| | animate | | 12 | | Workflow | 9 | | |
| | calculate | | 122 | | View | 64 | | |
| | edit | | 213 | | Perspective | 64 | | |
| | | | | | Texture | 60 | | |
| | | | | | Symbol | 98 | | |
| | | | | | Structure | 179 | | |

## 5. Conclusion and Future Work

In this work we have shown our methodology for improving the demand model used by GIS developers. In particular, we have focused on the ReqGIS tool which has been implemented to make all the process automatically. The main goal is to improve the mechanisms to find required services in existing GIS components. We have analyzed several lexical analysis tools and we have developed a GIS classification tool, reusing and adapting some open source components. As future work, we will go on working on the combination with the methodology defined for publishing GIS services in order to complete the mapping between supply and demand models.

## References

[1] Armour, F., Miller, G.: Advanced Use Case Modeling Volume One, Software Systems. Addison-Wesley Longman Publishing Co. Inc., Boston, MA, USA, (2001)

[2] Banerjee, S., Pedersen, T.: An Adapted Lesk Algorithm for Word Sense Disambiguation Using WordNet. In: Gelbukh, A. (ed.) Computational Linguistics and Intelligent Text Processing. LNCS, vol. 2276, pp. 117--171. Springer, Heidelberg (2002)
[3] Cechich A., Réquilé A., Aguirre J., Luzuriaga J.: Trends on COTS Component Identification. In: 5th IEEE International Conference on COTS-Based Software Systems, pp. 90--99. IEEE Computer Science Press, Orlando (2006)
[4] Cockburn, A.: Writing Effective Use Cases. Addison-Wesley Pub Co, (2001)
[5] Freeling Home Page, http://garraf.epsevg.upc.es/freeling/
[6] Gaetan, G., Cechich, A., Buccella, A.: Un Esquema de Clasificación Facetado para Publicación de Catálogos de Componentes SIG. In: XIV Congreso Argentino en Ciencias de la Computación. Chilecito, La Rioja, Argentina, (2008)
[7] Gaetan, G., Cechich, A., Buccella, A.: Aplicación de Técnicas de Procesamiento de Lenguaje Natural y Web Semántica en la Publicación de Componentes para SIG. In: X Argentine Symposium on Software Engineering. Mar del Plata, Argentina, (2009)
[8] Gaetan G., Cechich A., Buccella A.: Extracción de Información a partir de Catálogos Web de Componentes para SIG. In: XV Congreso Argentino en Ciencias de la Computación, pp. 891--900. (2009)
[9] Graham, I.: Object-Oriented Methods: Principles and Practice. Addison-Wesley, (2000).
[10] Kholkar, D., Krishna, G., Shrotri, U., and Venkatesh, R.: Visual Specification and Analysis of Use Cases. In: SoftVis'05: Proceedings of the 2005 ACM symposium on Software visualization, pp. 77--85. ACM, New York (2005)
[11] Kulak, D., Guiney, E.: Use Cases: Requirements in Context. Addison-Wesley Longman Publishing Co. Inc., Boston (2003)
[12] Lesk, M.: Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone From a Ice Cream Cone. In: 5th ACM International Conference on System Documentation, pp. 24-26. ACM, Toronto (1986)
[13] OGC. Topic 12: OpenGIS Service Architecture. Open GIS Consortium, (2002)
[14] OMG. UML Superstructure Specification, v2.1.2. OMG Formal Document 2007-11-02, (2007).
[15] Padró, L.; Collado, M.; Reese, S.; Lloberes, M.; Castellón, I. FreeLing 2.1: Five Years of Open-Source Language Processing Tools
[16] Pedersen, T., Patwardhan, S., Michelizzi, J.: WordNet::Similarity – Measuring the Relatedness of Concepts. Demonstration Papers at HLT-NAACL 2004, pp. 38—41. Boston, Massachusetts (2004)
[17] Saldaño, V., Buccella, A., Cechich, A.: Una Taxonomía de Servicios Geográficos para facilitar la identificación de componentes. In: XIV Congreso Argentino en Ciencias de la Computación. Chilecito, La Rioja, Argentina, (2008)
[18] Saldaño, V., Buccella, A., Cechich, A.: Descubrimiento de Servicios Geográficos a partir de Casos de Uso Textuales. In: XV Congreso Argentino en Ciencias de la Computación. Jujuy (2009)
[19] Saldaño, V., Buccella, A., Cechich, A.: Discovering Geographic Services From Textual Use Cases, Journal of Computer Science & Technology, Vol. 10 - No. 2 – June 2010 - ISSN 1666-6038
[20] Spivey, J.: The Z Notation: A Reference Manual. Prentice Hall, 1992.
[21] Szyperski, C.: Component Software-Beyond Object-Oriented Programming. Addison-Wesley, 1998
[22] Whittle, J., Jayaraman, P.: Generating Hierarchical State Machines from Use Case Charts. In: 14th IEEE International Requirements Engineering Conference (RE'06), pp 16-25. Washington, DC, USA, IEEE Computer Society (2006)