International Journal of Conceptions on Computing and Information Technology
Vol. 6, Issue. 2, November' 2018; ISSN: 2345 - 9808

Transformation of software components into a reusable software components based on elaborate Lexicon Extended Language

Brice Evrard TAREHY
Laboratory for Mathematical and Computer Applied to the Development (LIMAD)
University of Fianarantsoa
Fianarantsoa, Madagascar
tarehybriceevrard@gmail.com

Thomas MAHATODY
Laboratory for Mathematical and Computer Applied to the Development (LIMAD)
University of Fianarantsoa
Fianarantsoa, Madagascar
tsmahatody@gmail.com

Josvah Paul RAZAFIMANDIMBY
Laboratory for Mathematical and Computer Applied to the Development (LIMAD)
University of Fianarantsoa
Fianarantsoa, Madagascar
jp_josvah@yahoo.com

Abstract— **This article discusses about software component transformation as a reusable software component based on eLEL or elaborate Lexicon Extended Language. A reusable software component based on eLEL is a software component that possesses the information necessary for its reusability and its research, such as notion, behavior, methods, attributes or even circularity. The principle of the transformation is then to integrate in the simple software component the principles of the lexicon eLEL. For that, we proceeded in three stages, which are the determination of the structure of the software component, the establishment of the Universe of Discourse and the extraction of information needed from the Universe of Discourse to get the reusable software component based on eLEL. To validate the transformation algorithm, we conducted a case study on simple software components. In addition, we obtained eLEL-based reusable software components with the necessary information for reuse and will allow the developer to facilitate its research during a development of software.**

Keywords- elaborate Lexicon Extended Language; reuse of software component; software component; software component transformation.

I. INTRODUCTION

The search for information takes an important place in information systems. It's important to know how to model the information because it's necessary to be able to easily access it. In the same way, the search for components is fundamental for the engineering of component-based systems. Software component selection is very costly because it potentially imposes the path of markets with hundreds of software components and which are described with potentially very different formats. Software packages, libraries, executables, files, databases, or configuration items (parameters, scripts, Executable file) are software components [7,8]. In the end, selection becomes very time-consuming, to the point of threatening the gains originally conferred by this type of approach [3].

The most important benefit of component-based development is the reuse of existing software components in the construction of new systems, which allows companies to save time and effort in software development [12]. However, the problem is that all software components have not the same levels of reusability and the lack of structuring limits the definition of software components and makes it difficult to find [4]. So,we have proposed a new software component reuse environment based on the eLEL lexicon (elaborate Lexicon Extended Language) that allows to create a software component shaped for reusability from the moment of its creation [11]. The problem is how to transform existing software components into a reusable component based on eLEL.

In this article, we will propose a solution to this problem by transforming a software component into a reusable software component based on elaborate Lexicon Extended Language (eLEL). For that, we will first see the reusable software component based on eLEL, then we will propose an algorithm for transforming a software component into a reusable software component based on eLEL and finally before concluding and giving several perspectives, we will see some case studies to validate our transformation algorithm.

II. THE REUSABLE SOFTWARE COMPONENT BASED ON ELABORATE LEXICON EXTENDED LANGUAGE

Software development is a stimulating, knowledge-intensive activity [1] and to develop a complex software system, the reuse of software components is the activity that stands out as a methodological and technological solution. To facilitate reuse of software component, it's necessary to have a reuse environment that allows to create software components and to describe them by their names, descriptions and behaviors.

The Lexicon of Extended Language or eLEL [9] is a set of symbols (signs) conforming to the definition of (Eco, 1976)

[2]. Indeed, according to (Eco, 1976), a sign is something that can be interpreted to substitute for something [2]. According to (Hjelmslev, 1963), a sign defined as an entity that has an expression and a content, but an eLEL symbol is a simple coding system with six entities [5]:

• The term is the symbol itself;

• The notion is the meaning or definition of the symbol or term;

• Behavioral responses describe the action to be performed or executed on the symbol;

• Circularity describes the tacit relationship or link between the symbols;

• Attributes characterize the symbol;

• And methods allow you to access or manipulate the symbol.

eLEL is essential because it allows to define a set of possible development processes based on the reuse of the components of the database. eLEL provides essential support for reuse because it guides the search and assembly of compatible software components through its circularity concept or the relationship between each symbol.

(Tarehy et al., 2017) have created an eLEL-based reuse environment [11] that integrates the notion, typology, and behavior of eLEL [9] into the process of creating a software component. This process makes it possible to specify and facilitate the search for a software component as a reusable software component in the management of the reuse environment and to facilitate the specification, understanding and adaptation of the software component by referring to its behavior. The created software component then provides 92% information about the reusable software component, so the developer only needs 8% of the information needed to get complete information about the reusable software component.

The Fig. 1 schematically shows the reuse environment managing with eLEL-based components.

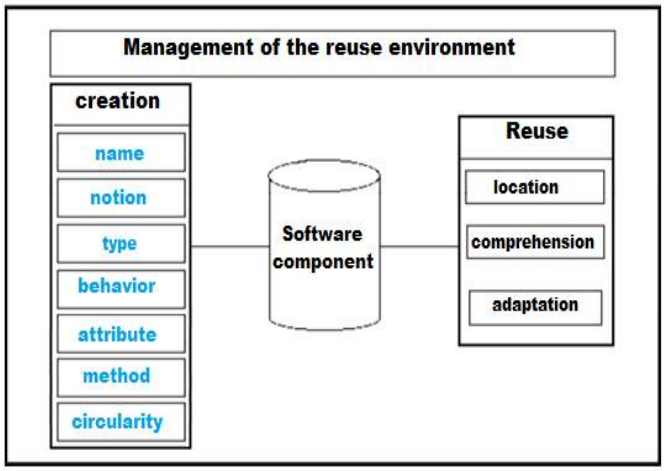

Fig. 1: Managing the reuse environment with eLEL-based components

III. Algorithm of Transformation of Software Component to Reusable Software Component based on eLEL

A software component is difficult to reuse if he doesn't have a structure which is capable of formulizing his semantics and his relationship with the another software components. To solve this problem, we need to transform the software component to reuse software component based on eLEL because the eLEL defines each software component with its behaviors [9] and classify each software component to facilitate its search and reusability[11].

The basis of the transformation algorithm is the construction principle of eLEL. Building eLEL requires the following two principles [6]: the first one is to maximize the lexicon term used to describe the notion, the behavioral response, the method and the attribute of a new term; this is the principle of closure or circularity. The second is to minimize the use of terms outside the UofD.

To be able to identify the information on a software component, it is important to know that a software is the collection of computer programs, procedures, rules, and associated documentation and data.

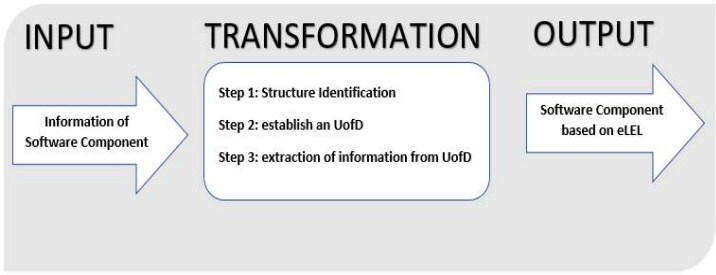

Figure 1. Transformation of software component into reusable component based on eLEL

Detail of Algorithm transformation:

- **INPUT:**
  - Identify component information
- **TRANSFORMATION:**
  - **Step 1: Structure identification**
    - Does the component have a name, a notion, a type, a behavioral response, an attribute, a method, or a circularity?
  - **Step 2: Establish an Universe of Discourse(UofD)**
    - What type of component is it?
    - Who is affected by this component?
    - Where is the component most useful?

- When is the component needed?
- How to use the component?
- Why is this component used?
- How much component will be related to this component?

➢ Deduction if the component's type is **subject, object, verb or state**

- **Step 3: Extraction of information from UofD**
  - **If the component's type is subject:**
    - **Description:** a subject is an entity having an important role in an application
    - **Notion:** who is the subject? What are its characteristics? What are the objects he manipulates?
    - **Behavior:** these are the definitions or meanings of the actions made by the subject
    - **Attribute:** subject characteristic
    - **Method:** these are the operations done to manipulate the attributes
    - **Circularity:** describes the relationships or links between each symbol
  - **If the component's type is object:**
    - **Description:** an object is an entity manipulated by a subject
    - **Notion:** what is its use? What are its characteristics? What are the other objects with which it is linked?
    - **Behavior:** what are the actions done on this object?
    - **Attribute:** characteristic of the object
    - **Method:** action taken to access or modify an object
    - **Circularity:** describes the relationships or links between each symbol
  - **If the component's type is verb:**
    - **Description:** characteristic run by an object having impacts on its environment
    - **Notion:** who intervenes when an event occurs? What is the object manipulated by the subject? What is the goal to complete?
    - **Behavior:** what are the environmental impacts?
    - **Attribute:** subject or object affected by the verb component
    - **Method:** action done by the subject on the object to accomplish the objective
    - **Circularity:** describes the relationships or links between each symbol
  - **If the component's type is state:**
    - **Description:** attribute containing values at different times during system execution
    - **Notion:** what does it represent? What action does it take?
    - **Behavior:** how to identify other states that may result from the current state?
    - **Attribute:** object or subject that can change state
    - **Method:** action done to have the current state
    - **Circularity:** describes the relationships or links between each symbol

OUTPUT:

- Reusable software component based on eLEL having type: subject or object or verb or state

IV. CASE STUDY

To validate the transformation algorithm, we will use as case studies the amazon's web service software component "amazonwebservice.wsdl" in order to transform them into a reusable software component based on eLEL. We extracted the information for the input of the algorithm using the principle of (Ian Somerville, 2007) which indicates the typical set of software composition [10].

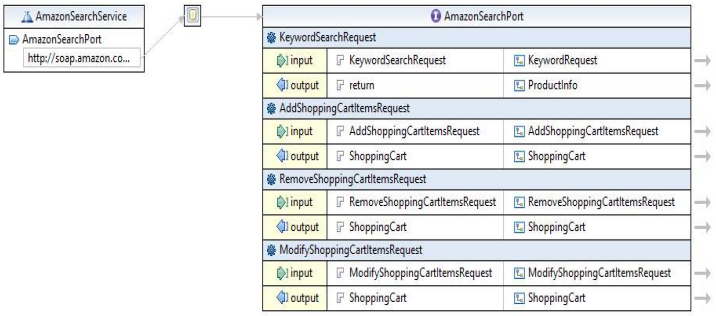

Figure 2. The amazon's web service component
http://cs.au.dk/~amoeller/WWW/webservices/AmazonWebServices.wsdl

INPUT:

TABLE I. INFORMATION OF AMAZON'S WEB SERVICE COMPONENT

AmazonSearchPort	
Identification	**Classification**
KeywordSearchRequest	Function
AddShoppingCartItemsRequest	Function
RemoveShoppingCartItemsRequest	Function
ModifyShoppingCartItemsRequest	Function

TRANSFORMATION:

- **Step1: Structure identification**
  - The software component has a method
  - The software component hasn't name, notion, behavior, attribute and circularity
- **Step2: establish UofD**
  - It is a Object-type component that allows user to make search
  - The first concerned is the user
  - When the user makes a search
  - The AmazonSearchPort is useful when user making a search, add shopping cart items, Remove shopping cart items , Modify shopping cart items.
  - The user must enter a keyword to do a search and then do the corresponding actions to the results.
  - The AmazonSearchPort is useful because it allows the possibility to make a search, to add shopping cart items, to remove shopping cart items , to modify shopping cart items
  - The AmazonSearchPort is in relation with the KeywordSearchRequest, the AddShoppingCartItemsRequest, the RemoveShoppingCartItemsRequest and the ModifyShoppingCartItemsRequest.
- **Step3: Extract information from UofD to get the eLEL-based software component**
  - The AmazonSearchPort software component is a object-type component:
    - Name : AmazonSearchPort
    - Notion : Amazon Search Port is a system that allows a user to make search and then perform actions against the search results
    - Behavior: Actions are made by the user in relation to the results of his research
    - Attribute: Keyword, items, cardId
    - Method: KeywordSearchRequest, AddShoppingCartItemsRequest, RemoveShoppingCartItemsRequest , ModifyShoppingCartItemsRequest
    - Circularity: AmazonSearchPort is linked to user, user is linked to KeywordSearchRequest, to AddShoppingCartItemsRequest, to RemoveShoppingCartItemsRequest , and ModifyShoppingCartItemsRequest

OUTPUT:

TABLE II. AMAZONSEARCHPORT SOFTWARE COMPONENT BASED ON ELEL HAVING OBJECT-TYPE

Name	AmazonSearchPort
Notion	Amazon Search Port is a system that allows a user to make search and then perform actions against the search results
Behavior	Actions are made by the user in relation to the results of his research
Attribute	Keyword, items, cardId
Method	KeywordSearchRequest, AddShoppingCartItemsRequest, RemoveShoppingCartItemsRequest, ModifyShoppingCartItemsRequest
Circularity	The AmazonSearchPort is linked to user, user is linked to KeywordSearchRequest, to AddShoppingCartItemsRequest, to RemoveShoppingCartItemsRequest, and ModifyShoppingCartItemsRequest

V. Conclusion and Future Work

In this article, we created an algorithm that can transform a simple software component into a reusable component based on eLEL. The basis of the algorithm is the process of creating the eLEL lexicon starting from the UofD [9]. The principle is simple, we have component information [10] and the first step is to determine its structure in relation to the lexicon eLEL, then we go to step two to establish the UofD [6] and the method we used is the method description of the needs by asking the relevant questions to obtain the most relevant informations possible. Finally for step three, we proceed to the extraction of relevant informations that we can categorize according to the type of software component (Subject, Object, Verb and State) to determine the notion, the behavior, the attributes, the methods and the circularity component [11]. We can say then that after the transformation, the software component has the basic information for a good reuse and that will then facilitate its search [11].

For the rest of our work, we will create a reusable software component databases and for that we have to create a metamodel of reusable software component based on eLEL able to generate code and we will create an application with a recommendation system to facilitate its search and its reuse.

References

[1] U. Apte, C.S Sankar, M. Thakur and J.E Turner, Reusability-Based Strategy for Development of Information Systems: Implementation Experience of a Bank, MIS Quarterly, December 1990, pp. 421-433

[2] U. Echo. Theory of semiotics. Bloomington: Indiana University Press.P.4, 1976.

[3] C. G. En, H. Baraçli, « A Brief Literature Review of Enterprise Software Evaluation and Selection Methodologices: A Comparison in the Context of decision-Making Methods », Proceedings of the 5th International Symposium on Intelligent Manufacturing Systems, 2006.

[4] G. Guzelian, C. Caveat and P. Ramadour, Component Design, and Reuse: A Goal Approach.

[5] L. Hjelmslev, "Prolegomena to a theory of Language ". Madison: University of Winsconsin Press.P.120, 1963.

[6] B. Karin, J.C.S.P. Leite, "Lexicon based ontology construction". Engineering for MultiAgent Systems II, Springer, 2004.

[7] P.A. Muller and N. Gaertner, Modelisation objet avec UML, Eyrolles, p.231, deuxième édition 2000.

[8] T.H. Nguyen, A. Abran, R. Bayard and O. DeChantal, Les concepts de réutilisation du logiciel et la pratique institutionnelle dans les entreprise québecoises, 2000, http://www.lrgl.uqam.ca/publications/pdf/290.pdf .

[9] J.L Razafindramintsa., T. Mahatody and J.P. Razafimandimby, Elaborate Lexicon Extended Language with a lot of conceptual information. International Journal of Computer Science, Engineering and Applications (IJCSEA). Vol 5, N°6, 2015.

[10] I. Sommerville, International computer science series, Pearson Education – 2007.

[11] B.E. Tarehy, J.L. Razafindramintsa, T. Mahatody, J.P. Razafimandimby, Reuse environment based on elaborate Lexicon Extended Language, IEEE xplore, 2017, ICCC 18th 2017, Romania, 06 pages, http://ieeexplore.ieee.org/document/7970417/

[12] L. Wang and P. Krishnan, "A Framework for Checking Behavioral Compatibility for Component Selection", Australian Software Engineering Conference, pp. 49-60, 2006.