Domain Model as Problem-Oriented Architecture Application for Mobile Applications

Mechelle Grace Zaragoza, Haeng-Kon Kim and Ha Jin Hwang

***Abstract*— Domain modeling is an activity that develops a generic model of a family of systems. It has been considered as one of the significant activities in systematic reuse. Frameworks can be used for allowing the design layers, permitting the construction of an intricate structures and reusing development information. In this paper, we will discuss the domain modeling supporting tool that extracts candidate domain model objects to construct frameworks from domain descriptions in a typical text form.**

Index Terms*—— Domain Modeling, Object Identification, Design Pattern, Program Understanding, Components Extraction

I. Introduction

Domain engineering, as explained is in fact also requirements engineering, is basically in need of thoroughly researched development principles, techniques and tools. [1]. Domain engineering support application engineering by producing artifacts necessary for efficiency of application development.[2]

In this process, domain modeling and analysis are intensive activities that require extracting individual domain model elements from information typically text template and construction. Frames are software packages that increase the efficiency of complete application development in a given area. Frames provide not only reuse software that implements the functionality required, but also reuse design, standard cooperation structures that are typically used in applications in this field. Object-oriented technology is the most important technology currently used to develop and use frames.We have developed a domain modeling support tool that extracts candidate domain model object. Objects field extracts candidates could help engineers in the field to provide domain descriptions of the capability of immediate evaluation and validation of source data to compare elements in the field of engineering design. It can also be used to build a field environment. It is based on the extraction of semantics and rules of allocation.

Manuscript received July 16, 2017;

Mechelle Grace Zaragoza, is with School of IT, Catholic University of Daegu Korea. (email:mechellezaragoza@gmail.com).

Haeng-kon Kim is with School of IT, Catholic University of Daegu, Korea.(email:hangkon@cu.ac.kr.).

Ha Jin Hwang is with Department of Business Analytics, Sunway University Business School, Malaysia (email:hjwang@sunway.edu.my).

II. Domain modeling framework

Domain Model is basically a problem-oriented architecture of the application domain that redirects the variations and similarities of the members of the domain. In a domain model of an application domain, an individual target system is done by modifying the domain model. The concept of making target systems from a generic specification and/or architecture has been studied by several researchers. [3]

A. Domain Modeling Method

Object-oriented model of software development is conducive to development and change, the domain modeling approach adopts an object-oriented perspective. The purpose is to apply object-oriented concepts and expand the scope eliminate the limitations of the application. The domain modeling approach is same to other object-oriented methods for the analysis and modeling in a single system. Its novelty, and where it differs from other methods, is the way in which object-oriented methods are extended to model system families. Therefore, the method allows the explicit modeling of similarities and differences in family systems [3]. In a domain model, an application area is represented by several views so that each view represents a different perspective in the field of application. Four views, the aggregation hierarchy, the view of the communication object diagram, the generalization / specialization hierarchy, and the view state transition diagram have similar counterparts in other object-oriented methods for modeling individual systems. On the other hand, in this domain modeling method, aggregation hierarchy is being used to model the types of optional elements, which by some, are being used, but not certainly, all members of the familiar systems. In addition, the generalization / specialization hierarchy is also used to model variations of one type of object, which are used by different members of the system family. The fifth view, the dependency / addict view object, is used to explicitly model the changes introduced in the domain model; Each feature - the optional field requirement - is associated with optional object types and variants required for support. This provides the basis for defining which target systems can be generated from the domain model.

The multiple viewpoints in the field of modeling method are given below: [4]

1) The Hierarchy Combination: The Aggregation hierarchy (AH) is used to break complex entities of the various components into grade to produce a variety of simple objects in hierarchical sheets. Types of objects are essential for all destination systems are optional. At higher levels of the

ISBN: 978-988-14047-5-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

hierarchy, global object types represent subsystems.
2) communication diagram of objects: objects in the real world are modeled as concurrent tasks that communicate through messages. Messages between objects may be weakly coupled or strongly coupled. The hierarchically structured communication object diagram (OCD) shows how objects are communicated. The TOC decomposition levels correspond to the levels of the aggregation hierarchy.
3) State transition diagrams are modeled as a sequential task. The target can be well-defined by a finite state machine and characterized by the state transition diagram, whose implementation is the strictly sequential definition.
4) generalization / specialization hierarchies: Although the requirements of the particular core or optional object type are modified to meet the specific requirements of the target system, the object type may be specialized. Variants of the specified type of domain hierarchy of generalization / specialization objects (GSH).
5) Function / Destination Dependencies: This View all types of objects feature includes (Mandatory Field) required to support the tool. Domain analysis, domain requirements are analyzed and categorized based applications supported on complex targeting systems, optional requirements (required only in some of the target system) and mutually exclusive requirements. In addition, some requirements require other conditions. This view highlights optional features, as it has a number of options and a variety of objects that are needed to support them, which determines the nature of the desired target system. [4]

B. Domain Modeling Environment

An environment for generating target systems from a domain model is mentioned in this article as a domain modeling environment. Such environments are also known as software systems for generator and generator applications. These generators are generally very field specific because they have the structure and code for the integrated scope. In addition, they provide a way to tailor the code to generate a specific target system either by establishing or by an application program written in a specific field area. The objective of this research is to have a method and an independent domain modeling environment in the application area to satisfy.

C. Frames

A framework in the design is a set of classes that work together to perform a set of responsibilities, an abstract design, which is used by modifying existing classes and / or extends by defining new subclasses. While the components of the class library are used individually, classes in a framework are reused as a whole to solve a specific case of a problem. The main difference between a framework and a class library is the knowledge required to use them. User frames should understand the abstract concept of the underlying framework and the internal structure of its classes in order to adapt and expand. Two categories of context are distinguished. Frames that make up a generic application for a domain area are called application frameworks. Executives are those who represent a microarchitecture that includes only a few articles.[5] They work together to perform a set of responsibilities, an abstract design, which is used by modifying existing classes and / or extends by defining new subclasses. While the components of the class library are used individually, classes in a framework are reused as a whole to solve a specific case of a problem. The main difference between a framework and a class library is the knowledge required to use them. User frames should understand the abstract concept of the underlying framework and the internal structure of its classes in order to adapt and expand. Two categories of context are distinguished. Frames that make up a generic application for a domain area are called application frameworks. Executives are those who represent a microarchitecture that includes only a few articles.[5]

III. APPROACH

To provide automated support to domain modeling, domain description semantics need to be extracted and mapped to the domain model elements. Classifying domain description with respect to pre-defined semantic representation structures enables establishment of the semantics of domain description according to the category to which they belong. The establishment of semantic representation structures and classification rules for informal domain description are key features of the approach. The overall framework is illustrated in Fig. 1. The process starts with informal domain description text that is first preprocessed to format this text for use in the other automated processes that follow. This preprocessing supports the attachment of an unique identification number to each domain description sentence.

The analysis is used to establish the syntactic function of each word. This process provides syntactic patterns so they can be used in semantics extraction. Images cases are used to represent the semantic information of the domain descriptions.

The structure of a case scenario is different for each primitive action. It consists of a header that represents the types of action and a body that represents the semantic objects of the concept. By various primitive actions, semantic objects that require may be different. The structure of the box for each primitive action has a different structure to represent its unique meaning.

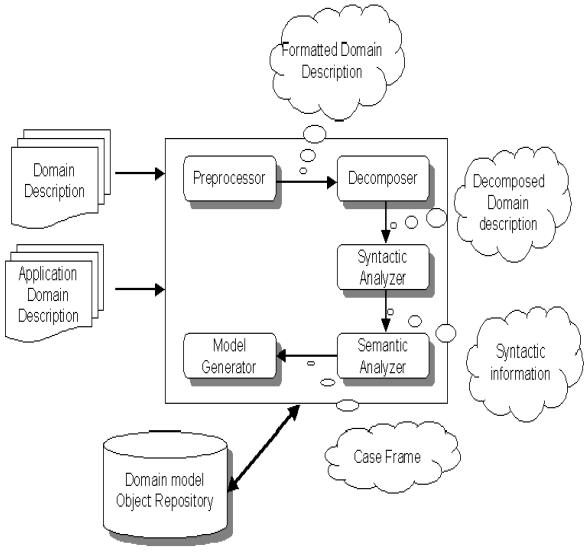

Fig. 1. Framework for Domain Model Object Generation

ISBN: 978-988-14047-5-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

A. Semantic Extraction

The concept of representation of the case framework is usually very useful to represent the thematic role of a word or phrase in a sentence. Each category has a fixed frame structure.
1. Existence: an action connects physical existence or a changing object. In this category, types of functions of entry, exit, Communication and retrieve / store are included.
2. Value: action links the value of the object. The function type of operation and editing is in this category.
3. Status: the action is connected to the object mode. The types of control functions and mode belong to this category.
4. Classification: reports of actions that evaluate the existence, value or state of the object of a given criterion. Identification and tapping belong to this category.

The relationship of the verbs can be classified into two categories:
1. specialization: a relationship that represents the father and the child.
2. Aggregation: a relationship that represents consists.
A classification of the primitive actions for the phrase domain description is described in Fig. 2.

M1: If (primitive action" identification) agent
= common noun)
= <agent> <object pattern>, <verb> action

M2: If (Primitive action= identification (object)
= common noun)

M3: if (Primitive action= identification (domain! =Empty)
=><domain> <object pattern>

M4: if (Primitive action) identification
(condition!)=Empty)= > <condition> condition
M5: if <object pattern>

"noun phrase>< relation preposition><noun phrase>
| <noun>) => noun phrase> object, <noun> object,
<relational preposition> relation
Where,
<relation preposition>= in|of|on

M6: if (Primitive action= Identification{ Object "Object)
(domain)=object)=>relation belong to

Fig. 2. Classification of Primitive Action

Rid: 25

File Name: Domain Description
Action: Search
Agent: The system
Object: Candidtaes
Domain: designated repository of fingerprint information
Method:
Condition

Rid:
File Name:
Action: <header pattern>
Agent: <agent>
Object: <object>
Domain: <domain>
Method: <method>
Condition: <condition>
Where:
<identification header pattern> check|detect|discover|recogncize|search
<Action> Identification header pattern
<Agent> Subject_Noun_Phrase <web pattern 1>
<Agent> Object_Noun_Phrase <web pattern 1>
<vern pattern> verb
<Domain> <Domain Pattern> Noun_Phrase
<Method> <method Pattern> Noun_Phrase
<condition> <condition Pattern> <Clause>
Condition pattern <Clause>
Condition pattern> if
when|during|upon|unless|after|before|one

1. For each case frame in the grammar, attempt an unchored match of the header pattern against the input can not parsed by the grammar
2. Retrive the case frame indexed by a recognized case header
3. Attempt to recognize each required case, as follows:
3.1. If case is marked
3.2. If case marked as pattern
3.3. If case marker cab ne marked eother way

Fig. 3. Model Case Frame Form

Domain Modeling

Before software can be designed we must consider identifying requirements but before requirements can be expressed, one must know what is a domain.[6] The assignment rules are proposed for the identification of the relations between the objects and the objects of the semantic domain model. Once the semantics, different models extracted from the objects can be identified using different allocation rules

IV. DOMAIN MODELING FRAMEWORKS

The system consists of five modules that are shown in Fig. 5. The preprocessor consists of three sub-modules; Formatateur style analyzer files and style corrector. The information obtained and organized, and the existing specifications of existing systems, are accepted as input in this module. From this module, a file containing the Domain Identification Description in each description of the phrase domain and the syntactic information of each word of the phrase is generated. The decomposer comprises two submodules; Solve pronouns and divisor phrases. The pronomènes Resolver replaces the pronoun with the appropriate name resolution according to the rule. The divisor condemns a sentence composed in a simple sentence and without loss of semantics.

ISBN: 978-988-14047-5-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

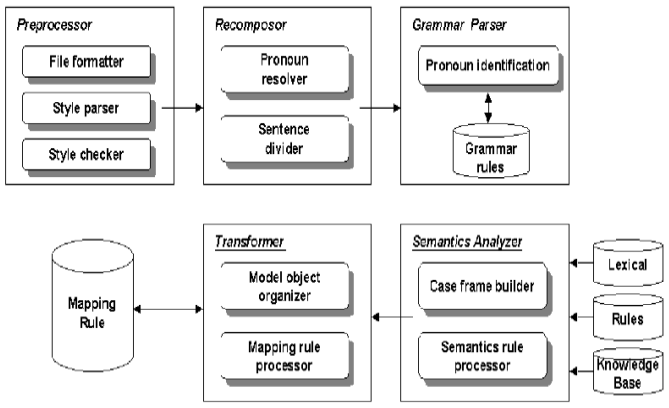

Fig. 4. Framework for Domain Model Object Identification

The syntax analyzer has one sub-module that is syntax identifier. The syntax identifier identifies syntactic objects such as subject, object and predicate from the style parsed output. The identification process is performed according to the pre-defined grammar rules.

The grammar parser has 2 sub-components; semantic rule processor and case frame builder. The semantic rule processor identifies a words or phrases that meets the semantic extraction rules. This identification process uses lexicon information and domain knowledge if available. The identified words or phrases are used by case frame builder to represent a semantic of the domain descriptions sentence.

The transformer has 2 sub-components; mapping rule processor and model objects organizer. The mapping rule processor identifies model objects based on the semantic representation that is case frame and mapping rules. The identified model objects are organized by the model objects organizer so that they can be comprehended by domain descriptions engineer. The system consists of 5 user interface panels that relate to each module. Figure 6 is a one of the panels for extracting semantics.

A. Assessment

The text of two systems was applied, each in separate domains: Fuel Level Monitoring System (FLMS), and Automatic Fingerprint Identification System (AFIS). Figure 7 is an example of the whole process using a typical domain description example sentences. The domain model objects in the FLMS text, are shown in Table 1. This table shows what model objects are described in the domain description text for each domain model objects category. Table 2, shows the ratio between the number of the domain model objects in the FLMS text and the number of domain model objects in generating candidate model objects. [7-8]

Table I. Domain Modeling Elements

Domaim Model Element	Domain Model Elements	Total Number of Model Elements
Object	Controller, motor, water temperature, motor speed, flow, olive valve, fuel, 5 minutes, predefines value	15
Action	Activate, deactivate signal, regulate, operate	7
Relation	Controller of an oil, hot water home heating system, status of fule, home temperature, combustion of furnace system, mim time in 5 minutes	5
Condition	Whenever the temp, falls below desired: once the speed is adequate temp reached is predefined value, after 5 mins: within 5 secs master switch off fuel flow shuts off.	7

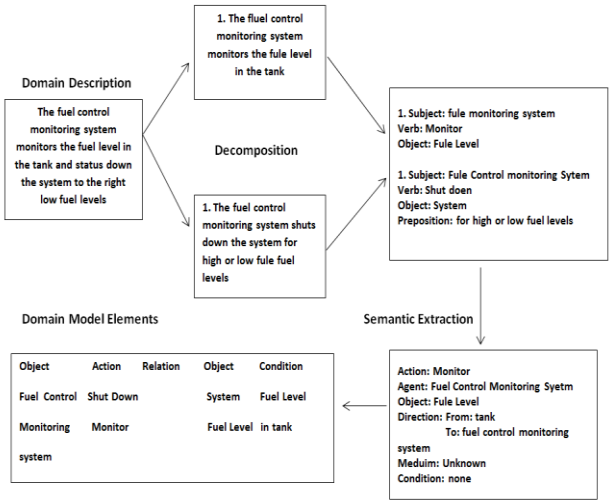

Fig. 6. Semantic Extraction Panel

Table II. Domain Model Objects in FLMS

	Number of Domain Model Elemts	Number of Model Elements	Ratio
Object	15	13	88%
Action	7	6	89%
Relation	5	4	80%
Condition	7	7	100%
Total	34	30	88%

ISBN: 978-988-14047-5-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Table III. Ratio between the Domain Model Objects and System Model Elements in the Generated Candidate Model Objects

Domaim Model Element	Domain Model Elements	Total Number of Model Elements
Object	Electronics ten print submission, fingerprint,ten print card, criminal hostory file, electronic image, remote users,criminal history	22
Action	Scan, user, identify, search, send, submit, store, access, recieve	9
Relation	Fingerprint belongs to electronic ten submission include live scan, ten print card print include palm print, criminal history file has biologival desciptor, file has personal identifier	11
Condition	When received by FBI	1

The second case is when the patterns and sentence structure are not treated with semantic extraction rules. In the text FLMS, the temperature of the house and the preset value is the second case and the operation and temperature of the house have a preset value are the first cases.

V. CONCLUSION

Explains that method has been developed to generate candidate domain model objects to provide automated support for modeling domain descriptions. The main feature of this method is to extract the semantic information using case frames for the primitive actions in each domain description. He presented the approach architecture and a system developed to generate candidate objects, conceptual model.

The main contribution of this research is to develop a methodological framework to provide automatic support for object-based object identification models based on domain text.

The research also contributed to the development of methods of extracting semantic text description field using frames include cases for domain descriptions, verb and name classification structures, and dictionary information. Application of the semantic extraction of the decomposition method provides an ability to decompose several semantic descriptions integrated into the domain descriptions of integrated semantic domains. This capability facilitates the activity of extracting semantic descriptions or the informal domain field of textual descriptions.

ACKNOWLEDGEMENTS

This Research was supported by the MSIP (Ministry of Science, ICT and Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2017-2013-0-0087) supervised by the IITP (Institute for Information & Communication Technology Promotion).

REFERENCES

[1] Bjorner D., Domain Engineering: A Software Engineering Discipline in Need of Research. In: Hlaváč V., Jeffery K.G., Wiedermann J. (eds) SOFSEM 2000: Theory and Practice of Informatics. SOFSEM 2000. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg vol 1963, (2000).

[2] Kim, Haeng-Kon. "Model Driven Engineering for Crop Monitoring Applications."International Journal of Software Engineering and Its Applications,10.6 (2016), pp.125-140.

[3] Carnegie Mellon University, “Domain Analysis,” Feb. (1999), http://www.sei. cmu.edu/domain-engineering/domain-anal. Html.

[4] Hassan Gomaa and Larry Kerschberg, Domain Modeling for Software Reuse and Evolution, http://mason.gmu.edu/~kersch/KBSE_folder/KBSEE_folder/RioCASEConference.html

[5] Object Oriented Database Systems, http://www.upriss.org.uk/db/42033lecture_one.html

[6] Bjørner, Dines. "Domain engineering." Formal Methods: State of the Art and New Directions. Springer London, (2010), pp. 1-41.

[7] Carnegie Mellon University, “Domain Analysis,” http://www.sei. cmu.edu/domain-engineering/domain-anal. html

[8] Preto-Diaz, “Domain Analysis:An Introduction,” http://www.galaxy,net/kpt/VA/Domain,Analysis Into.html

ISBN: 978-988-14047-5-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)