Domain Requirements Elicitation and Analysis - An Ontology-Based Approach[*]

Yuqin Lee and Wenyun Zhao

Software Engineering Lab, Computer Science and Technology Department,
Fudan University, Shanghai, China
li_yuqin@yahoo.com.cn, wyzhao@fudan.edu.cn

Abstract. Domain requirements are fundamental for software reuse and are the product of domain analysis. This paper presents an approach to elicit and analyze domain requirements based on ontology. Using subjective decomposition method, problem domain is decomposed into several sub problem domains. The top-down refinement method is used to refine each sub problem domain into primitive requirements, which are specified using ontology definition. Abstract stakeholders are used instead of real ones when decomposing problem domain and ontology is used to represent domain primitive requirements. Not only domain commonality, variability and qualities are presented, but also reasoning logic is used to detect and handle incompleteness and inconsistency of domain requirements. In addition, a case of 'spot and futures trading' e-business is used to illustrate the approach.

1 Introduction and Related Work

Requirements engineering (RE) is the branch of software engineering concerned with the real world goals for, functions of, and constraints on software systems. [1] It is generally accepted that the ultimate quality of the delivered software depends on the requirements upon which the system has been built. [3][4] Studies performed at many companies have measured and assigned costs to correct defects in software discovered at various phases of the lifecycle. Generally, the later in the software lifecycle a defect is discovered the more expensive it is to rectify. [5] [6].To achieve large scale reuse of software, a domain is usually mentioned. [2]

Among those domain requirements elicitation and specification approaches, there are mainly three types. One is the traditional method. Domain requirements are presented in natural language. The main drawback of this method is that there are gaps between domain users and requirements engineers. This will be the main source of generating inconsistent and ambiguous requirements. The second method is the scenario-based method. Domain users can take part in elicitation conveniently, and the requirements are the true reflection of the real system. But its disadvantage is the problem how to guarantee a set of scenarios that are the complete requirements of the

[*] This paper was supported by the National High Technology Development 863 Program of China under Grant No.2005AA113120.

system.[7] The overlap of different scenarios is the source of requirements inconsistency. The third solution is knowledge-based methods.[7][8][17] The methods have been researched hotly, but there are few sound achievements yet. Among the third methods, feature-based and ontology-based methods are usually researched. However, the feature-based method[17] concentrates on commonality and variability analysis having a shortcoming in reasoning logic, while ontology-based method[7][8] focusing on representing domain concepts and relations of concepts lack of distinguishing concepts to common or variation sets. Our approach combines the advantages of two kinds of knowledge-based methods, not only representing commonality, variability and qualities clearly, but having reasoning logic which can be used in requirements analysis also.

In this paper, an approach is suggested to systematically develop domain requirements based on ontology. Using subjective decomposition method, domain problem is decomposed into several sub problem domains. Subjectivities are the viewpoints of abstract stakeholders of domain. Top-down refinement method is used to refine each sub problem domain into primitive requirements, which are specified using ontology definition. The advantages of our approach are using abstract stakeholders other than practical ones when decomposing domain, getting commonality, variability and quality features from each sub problem domain, using ontology in domain elementary requirements representation, and using reasoning logic based on ontology relations to analyze completeness and consistency easily.

The paper is organized as follows: Section 2 gives out ontology definition. Section 3 discusses domain requirements metamodel. Section 4 provides requirements elicitation and refinement methods for ontology-based domain requirements. Section 5 discusses domain requirements analysis, focusing on completeness and consistency. Section 6 illustrates our approach by an example of spot and futures e-business domain model. Section 7 draws a conclusion and some suggestions for future work.

2 Ontology Definition

The term ontology was taken from philosophy. According to Webster's Dictionary, an ontology is a branch of metaphysics relating to the nature and relations of being. In knowledge engineering area, ontology was first defined by Neches.[8][9] Another ontology definition is newly defined by Gruber:" Ontology is a formal, explicit specification of a shared conceptualization." [10]

We give a definition of ontology that a quadruple consists of the core elements of ontology, i.e. concepts, relations, hyperspace and theorems.

O={C,R,H,T}. C(Concepts) and R(Relations) are two sets which don't intersect with each other. H stands for hyperspace consisting of concepts and relations. T is the set of ontology theorems.

R consists of the relations as follows:

Is-a: describes the equivalence relation;

Is part of: describes the part and whole relation.

Mutually exclusive: describes the relation of two concepts which can't be in a system contemporary;

Association: describes two concepts having relations with each other; such as pre condition and post condition, etc.

Based on this ontology definition, we have the following definitions or theorems:

Definition 1: functional (one by one mapping) relation. If $c_1, c_2 \in C$, and $r \in R$, $r(c_1, c_2)$. If one c_1 corresponds to only one c_2, then r is functional. Vice versa, if one c_2 corresponding to only one c_1, then r is inverse functional.

Definition 2: symmetric relation. $\exists c_1, c_2 \in C$, if $\exists r \in R$, $r(c_1,c_2)$, then $r(c_2,c_1) \in R$. We call r is symmetric.

Definition 3: Transitive relation. If $c_1,c_2,c_3 \in C$, $\exists r \in R$, $r(c_1, c_2)$ and $r(c_2,c_3)$, then $r(c_1,c_3)$. We call r is transitive.

Definition 4: Homomorphic mapping. Let $O_1= \{C_1, R_1, H_1, T_1\}$, and $O_2= \{C_2, R_2, H_2, T_2\}$ be different ontology, define a homomorphic mapping M from O_1 to O_2 as:

M= {M(C1), M(R1), M(H1), M(T1)}, satisfies the following rules:

1. $M(C_1)$is an one by one mapping. For each $c_1 \in C_1$,then $M(c_1) \in C_2$; if $c_1, c_2 \in C_1$, and $c1 \neq c2$, then $M(c_1) \neq M(c_2)$;
2. $M(R_1)$ is a one by one mapping as $M(C_1)$;
3. $M(H_1)$ is a one by one mapping too;
4. T_1 is a subset of T_2.

Definition 5: Ancestor relation. Let $O_1=\{C_1, R_1, H_1, T_1\}$, and $O_2=\{C_2, R_2, H_2, T_2\}$, $O_1 \neq O_2$; If there is a homomorphic mapping M from O_1 to O_2, we call O_1 ancestor of O_2.

Definition 6: Inheritance relation. Let $O_1=\{C_1, R_1, H_1, T_1\}$, and $O_2=\{C_2, R_2, H_2, T_2\}$, $O_1 \neq O_2$; If O_1 is ancestor of O_2, we say O_2 has an inheritance relation with O_1.

Definition 7: Nesting. Ontology can be nested. This feature will be used in multi-viewpoints domain analysis.

3 Domain Requirements Metamodel

The metamodel (see fig 1) for a domain requirement is adopted from [2] and be improved.

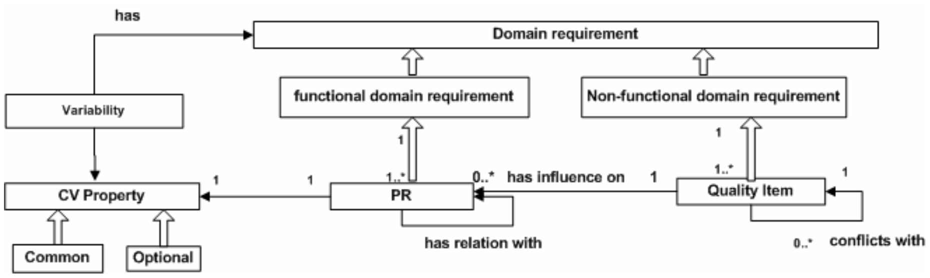

Fig. 1. Metamodel for Domain Requirement

The metamodel has a domain requirement as the central model element. Domain requirements are categorized into functional and nonfunctional requirements. Functional requirements can be refined into PRs (primitive requirements). The common and optional property is relative. The common or optional property of PR is analyzed regarding in how many systems it exists. Optional property is realized as variability of requirements. There are relations between PRs. Nonfunctional requirements have relations with and can influence functional requirements. We use the ontology definition above to represent domain requirements including not only functional and nonfunctional requirements, but also common and variant points in functional requirements. In this method, we categorize common requirements into several sets according to subjects. The variants are realized as configuration or value of dimension in PR level. Nonfunctional requirements are classified as a separate set.

4 Domain Requirements Elicitation and Refinement

Based on ontology description above, domain requirements are elicited and refined. The process consists of three phases. The result of this process is to get PR specifications and relations of PRs.

Phase 1: Define domain terms. To develop precise and consistent domain requirements, the basic terms used in the domain should be defined. [2] The principle of defining terms in a domain is to guarantee them within the domain scope. Domain terms constitute a terminology dictionary, which evolves along with domain decomposition. Domain terms based on ontology contains domain concepts and relations of concepts.

Phase 2: Decompose domain requirements. To decrease complexity, a problem domain is decomposed into several sub-problem domains. Subjective decomposition used in this paper is based on modeling different subjective perspectives of different stakeholders on a system or a domain. [16] In the method illustrated in [16], subjectivity is accounted for in DEMRAL by considering different stakeholders and annotating features by their stakeholders. We don't use different viewpoints of stakeholders directly, but subjectivity is abstracted from a group of stakeholders who have similar needs for the systems.

Domain is specified using the following representation: *D=(S, R, C).*

S= $\{s_1, s_2,..., s_n\}$, s_1, s_2 .... s_n, representing viewpoints of abstract stakeholders dealing with sub-problems of domain;

R: representing the relations of elements of S. We only use mutually exclusive relation in this method;

C: representing constraints of S and R. n is finite. It means the problem domain should be decomposed into finite sub-problem domains. All $s_1 \dots s_n$ constitute the whole problem domain. The relation of s_i and $s_j (i<n, j<n, i \neq j)$ is only mutually exclusive, which means non intersection exists between elements of S.

Phase 3: Refine domain requirements and specify them using ontology. After problem domain is decomposed into several sub-problem domains, the requirements of each sub-problem domain are refined into functional and nonfunctional requirements. After this all functional requirements are refined into PRs.

Top-down method [14] is used to refine functional requirements in a hierarchy structure, which we call function tree. At each level, each function is decomposed into a number of functions at the next level, until a level is reached at which the functions are regarded as elementary functions [15]. The elementary functions are called PRs (primitive requirements). Different levels are related using whole-part association (WPA[17]).

Specify each sub problem domain as $S_i = (C_i, R_i, H_i, T_i)$. According to the definition above, domain requirements can then be: Requirements ={C, R, H, T}

$C= \cup\{C_i\}, i=1..n$; $C_i= \{\cup\{PR_{ij}\}\} \cup Q_i, j=1..r_i$. Q_i represents quality features of sub problem domain i;

$R= \cup\{R_i\}$; $T= \cup\{T_i\}$. *H* is the hyperspace consisting of C and R.

According to the definition above, we get a triple table DR= (sc, re, dc). DR stands for domain requirements; sc and dc stand for source and destination concepts; re stands for relation of sc and dc. One row represents relation of a couple of concepts. To analyze requirements conveniently, a row represents a direct relation of sc and dc.

5 Domain Requirements Analysis

Based on the triple table DR= (sc, re, dc), we can analyze completeness and consistency of requirements.

- Detection and Handling of Requirements Incompleteness

From [18], the REQ represents complete requirements of a domain, if the following formula is satisfied.

$$Domain(REQ) \supseteq Domain\ (NAT) \tag{1}$$

Domain(REQ) stands for requirements of domain, *Domain (NAT)* stands for needs of stakeholders.

Completeness of requirements is relative. We discuss completeness in the point of view whether concepts and relations set are complete.

Definition 1. If c_2 has direct or transitive association with c_1, and $c_1 \in C$, $c_2 \notin C$, then we call concepts is incomplete. The solution is to add c_2 to C.

Definition 2. $\exists c_1, c_2 \in C$, and c_2 has direct association with c_1, i.e. $r(c_1, c_2)$, but $r \notin R$, so we call the relation incomplete. The solution is to add r to R.

Definition 3. Completeness is detected by inherent relation. If two requirements have inherent relation in domain knowledge base, the related requirement should be included in requirements set when the source requirement is selected. The analysis process of completeness is iterative, and it will end until the result concepts and relations set are stable.

- Detection of Requirements Inconsistency

Subject overlap is the main source of inconsistency. We assume that different sub problem domains don't intersect in this paper, so that inconsistency doesn't appear.

6 Case Study

We take spot and futures e-business domain as an example. The domain provides an e-business platform to support multi-to-multi real time transactions in B2B area.

- Requirements Refinement and Specification of Spot and Futures E-business

Phase 1: define domain terms. By analyzing spot and futures e-business domain, the commonalities are that there are order, match, settlement and delivery phases. The variations exist in each phase. Many different areas use the B2B trading method, so contract which is the base unit for transaction has different features for different commodities. Several match modes are supported, such as: forward, special performance, auction, etc. We get a terminology dictionary including all these terms.

Phase 2: Decompose e-business domain requirements. The abstract stakeholders are roles responsible of trading, settlement and delivery. The problem domain can be decomposed into three sub problem domains: trade, settlement and delivery.

D=(S,R,C)

S={trade, settlement, delivery}.

R: mutually exclusive relation between elements of S, complying with the whole transaction flow.

C: constraint of S and R. S has three elements, and is thus finite. The elements of S deal with different phases of transaction flow, so they don't intersect with each other.

Phase 3: Refine e-business domain requirements and specify them by ontology. After using the refinement method, the requirements tree of spot and futures transaction domain is in figure 2.

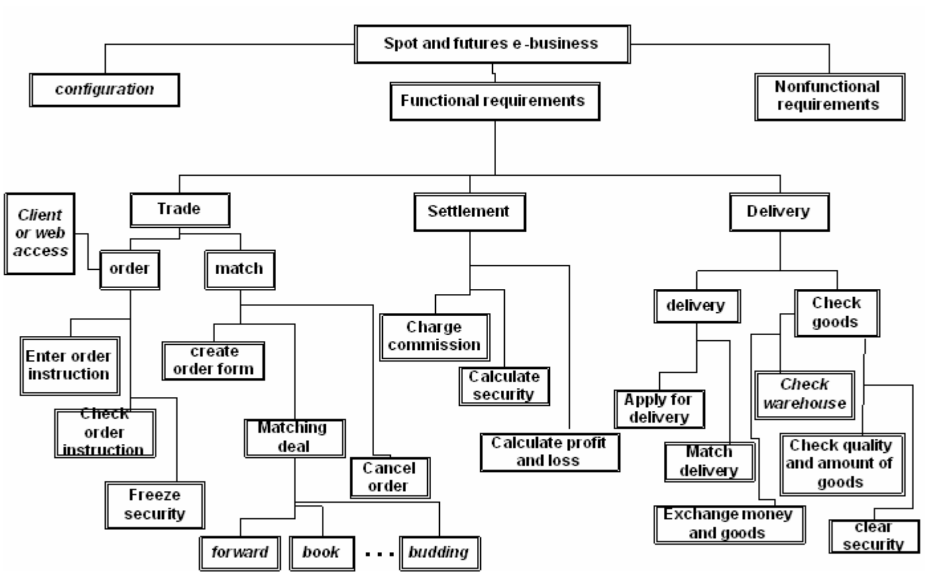

Fig. 2. Requirements Tree of Spot and Futures Transaction

In figure 2, the italic represents variant points. The leaves are primitive requirements. We only specify these leaves and their relations using ontology.

The relations of PRs are {is-a, is part of, is precondition of, is post condition of, transition}. The theorems are corresponding to those defined in the ontology.

Requirements of the case are represented as follows:

$C= \bigcup\{C_1,C_2,C_3, Q\}$.
C_1= {order, match}
△*order*={client access, web access}
⦿*order*={enter order instruction, check order instruction, freeze security}
match={ create order form, ●match deal, cancel order}
●*match deal*= {forward, special field, book, talk, list, auction, bidding}
C_2={charge commission, calculate security, calculate profit and loss}
C_3={apply for delivery, match delivery, ○check special field , exchange money and goods, check quality and amount of goods, clear security}
○*check special field*= optional { check warehouse in special field}
Q= {order transaction should complete in 0.1 seconds, etc}.
Where,
Q--quality features i.e. nonfunctional requirements of the domain
△-- mutually exclusive relation
⦿--whole and part relation
●--alternative option, every element is optional ,but at least one is selected
○—optional

- Requirements Analysis of Spot and Futures E-business Domain

The domain is decomposed by using subject-orient method according to different business phases, so time guarantees sub problem domains don't overlap. This method decreases inconsistency of requirements from different sub problem domains. We focus on completeness analysis.

After requirements refinement, we get triple table DR= (sc, re, dc) representing domain requirements, where sc and dc are nodes of requirements tree, and re is branch of the tree representing direct relation between primitive requirements and implied relations between brother nodes.

Take sub problem settlement as example, we get the table as the follows.

Table 1. Requirements table of sub problem settlement

• Sc	• re	• dc
charge commission	is part of	settlement
calculate security	is part of	settlement
calculate profit and loss	is part of	settlement
charge commission	is precondition	calculate security
calculate security	is precondition	calculate profit and loss

After analysis according to rules above, we can get a stable requirements table.

7 Conclusion and Future Work

The domain requirements stand for requirements for families of similar systems in one area, but it's difficult to get domain requirements from domain users because there are gaps between domain users and software developers. In this approach, we use ontology to represent domain requirements so that domain users can take part in requirements elicitation easily. Using subjective decomposition method, domain problem is decomposed into several sub problem domains. Subjectivities are the viewpoints of abstract stakeholders of domain. Top-down refinement method is used to refine each sub problem domain into primitive requirements, which are specified using ontology definition. The requirements specification is precise and easy to analyze. A case of spot and futures e-business is used to illustrate our approach. The advantages of our approach are using abstract stakeholders other than practical ones when decomposing domain, and getting commonality, variability and quality features form each sub problem domain, and using ontology in domain elementary requirements representation, and using reasoning logic based on ontology relations to analyze completeness and consistency easily.

In this paper, mutual exclusion is the predicted relation. In reality, subjects may crosscut many aspects of a domain so that intersection often exists. We will improve the approach to deal with more complex situations, and to develop map rules from the domain model into reusable architecture and components.

References

1. Axel van L.,Handling obstacles in goal-oriented requirements engineering, IEEE Transactions on software engineering, vol.26, NO.10, pp978-1005, Oct 2000.
2. M.Moon, An approach to developing domain requirements as a core asset based on commonality and variability analysis in a product line, IEEE Transactions on software engineering, vol.31, NO.7, pp 551-569, Jul 2005
3. Brooks, F.P., No Silver Bullet: Essence and Accidents of Software Engineering, IEEE Computer, 20, 4 (April 1987), 10-19.
4. Hrones, J.,Defining Global Requirements with Distributed QFD. Digital Technical Journal 5, 4, 36-46.
5. W. James, Effectiveness of Elicitation techniques in distributed requirements engineering, Proceedings of the IEEE Joint International Conference on RE, 2002.
6. Davis, Alan M. Software Requirements: Objects, Functions, and States. Englewood Cliffs, NJ: Prentice-Hall, 1993.
7. Zhi, Jin. Ontology-based requirements elicitation automatically. Chinese J. Computers. Vol.23,No.5, May 2000. pp486-492.
8. R.Q.Lu, Ontology-based requirements analysis Journal of Software. 2000,11(8). 1009~1017.
9. NechesR, Enabling technology for knowledge sharing.AIMagazine,1991,12(3):36〜56.
10. GruberTR. A translation approach to portable ontology specifications. Knowledge Acquisition,1993,5(3):199〜220.
11. Gruninger,M; Lee,J.; Introduction to the ontology application and design section, guest editors-communications of the ACM, Feb,2002, Vol 45, No,2 , pp39-41.
12. G.Booch, The Unified Modeling Language Reference Manual. Addison-Wesley, 1999
13. M.Jackson. Problem Frames: Analyzing and Structuring Software Development Problems. Addison-Wesley, 2001
14. D.Knuth. The Art of Computer Programming. Addison-Wesley, 1973

15. Xuefeng.Z, Inconsistency Measurement of Software Requirements Specifications an Ontology-Based Approach. Proceedings of the 10th IEEE ICECCS. 2005
16. Krzysztof Czarnecki, Feature Modeling, July, 1998, pp1-31.
17. Z. wei, M.Hong, A feature-oriented domain model and its modeling process. JOS. 2003. Vol. 14, No. 8. pp1345-1356.
18. Konstantin K., David L.,P., On Documenting the Requirements for Computer Programs Based on Models of Physical Phenomena. models3 august, pp1-14.
19. James.C, Commonality and Variability in Software Engineering. IEEE software 1998 November. pp37-45