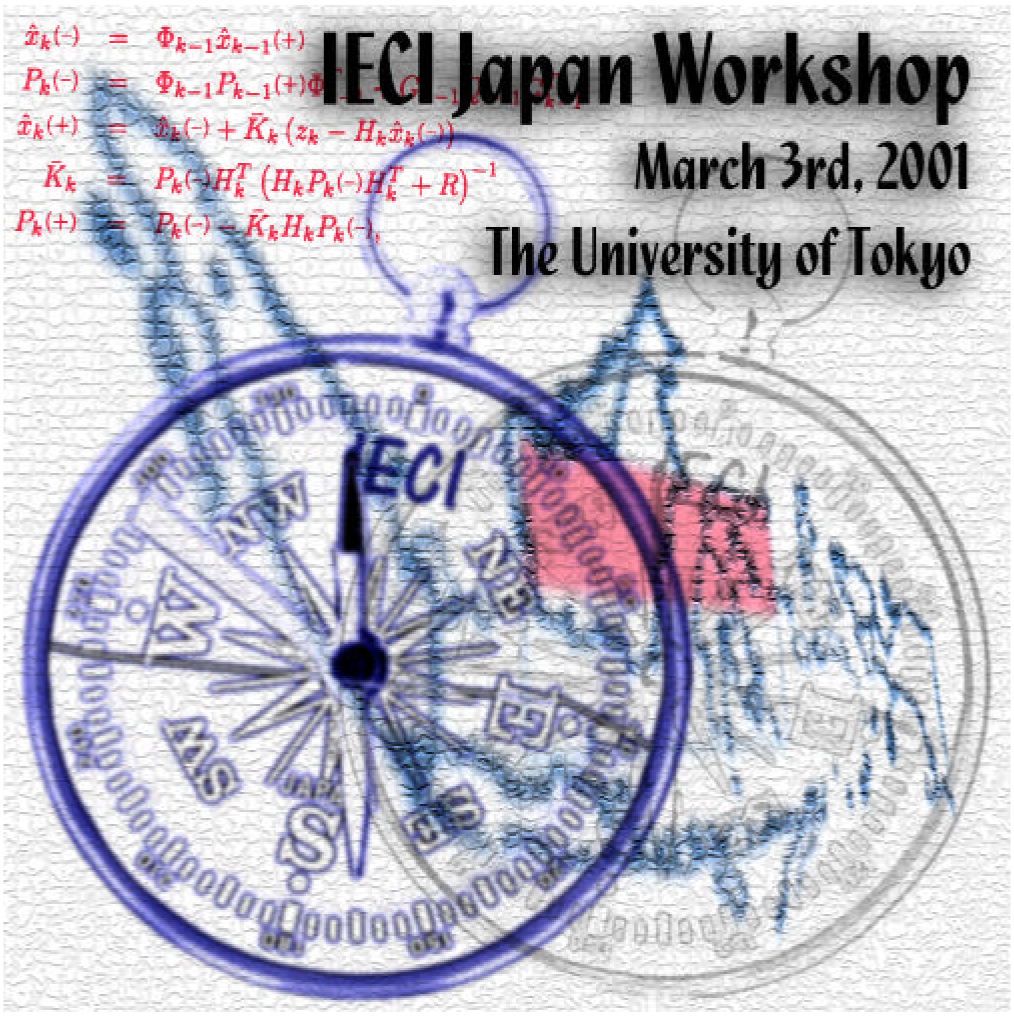
IECI Chapter Japan Series Vol. 3 No. 1, 2001 ISSN 1344-7491

Proceedings of the IECI Japan Workshop 2001

IJW-2000


$\hat{x}_k(-) = \Phi_{k-1}\hat{x}_{k-1}(+)$
$P_k(-) = \Phi_{k-1}P_{k-1}(+)\Phi$
$\hat{x}_k(+) = \hat{x}_k(-) + \bar{K}_k(z_k - H_k\hat{x}_k(-))$
$\bar{K}_k = P_k(-)H_k^T(H_kP_k(-)H_k^T + R)^{-1}$
$P_k(+) = P_k(-) - \bar{K}_kH_kP_k(-)$
IECI Japan Workshop
March 3rd, 2001
The University of Tokyo
IECI
NW
W
SW
S
SE
E
NE

Supported by
Indonesian Society on Electrical, Electronics, Communication and Information (IECI)
Indonesian Students Association (PPI)
Institute for Science and Technology Studies (ISTECS)

Organized by
Indonesian Society on Electrical, Electronics, Communication and Information (IECI) Japan

In Cooperation With
The University of Tokyo

Object Based Formal Specification: Methodological Support for Specifying Requirements in Object Model Creation Process

Romi Satria Wahono and Behrouz H. Far
Department of Information and Computer Sciences,
Graduate School of Science and Engineering, Saitama University

Abstract:
Requirement acquisition is considered as one of the most important activities in software development. Most faults found during testing and operation result from poor understanding or misinterpretation of requirements. We propose an approach where end users take an active role in analysis by specifying requirements using Object Based Formal Specification (OBFS). We use OBFS to guide end users in describing their problem. This approach is a first important step for solving the difficulties and ill-defined tasks in the object model creation process. In this paper we present OBFS and its roles to be methodological support for specifying requirements in object model creation process, including identification process and object refinement with inheritance process.

Keywords: Requirement Engineering, Object Model Creation Process, Object-Oriented Analysis and Design

1. INTRODUCTION

Requirement acquisition is considered as one of the most important activities in software development. Most faults found during testing and operation result from poor understanding or misinterpretation of requirements. In spite of progress in analysis techniques, Computer Aided Software Engineering (CASE) tools support, prototyping, early verification and validation. Software development still suffers from poor requirements acquisition.

In the traditional approach to software analysis, system analyst interviews end users to capture requirements. We propose an approach where end users take an active role in analysis by specifying requirements using *Object Based Formal Specification* (OBFS). We use OBFS to guide end users to describe their problem. This approach will be a first important step for solving the difficulties and ill-defined tasks in the object model creation process, including identification of objects, relationships, attributes, behaviors and organization of objects with inheritance. This approach also takes advantage of end users' domain knowledge.

In this paper we present OBFS and its roles to be a methodological support for specifying requirements in object model creation process, including identification process (object identification, association identification, attribute identification, behavior identification), and object refinement with inheritance process.

2. OBJECT BASED FORMAL SPECIFICATION and ITS MODEL

Figure 1 shows our strategy to formulate requirement specification for solving the object model creation process. We propose an approach where end users take an active role in analysis by specifying requirements using OBFS. We use OBFS to guide end users in describing their problem. OBFS is composed of *Description Statements* (*DS*), *Collaborative Statements* (*CS*), *Attributive Statements* (*AS*), *Behavioral Statements* (*BS*), and *Inheritance Statements* (*IS*).

$$OBFS = DS \oplus CS \oplus AS \oplus BS \oplus IS$$

Each OBFS statement consists of Subject (S), Verb (V), and Object (O) as well as the English (E) natural language.

$$\begin{aligned}
DS &= \{reqID, reqName, Language, Description\} \\
CS &= \{(S_1,V_1,O_1)_{cs},(S_2,V_2,O_2)_{cs},(S_3,V_3,O_3)_{cs},...\} \quad \text{and} \\
AS &= \{(S_1,V_1,O_1)_{as},(S_2,V_2,O_2)_{as},(S_3,V_3,O_3)_{as},...\} \\
BS &= \{(S_1,V_1,O_1)_{bs},(S_2,V_2,O_2)_{bs},(S_3,V_3,O_3)_{bs},...\} \\
IS &= \{(S_1,V_1,O_1)_{is},(S_2,V_2,O_2)_{is},(S_3,V_3,O_3)_{is},...\}
\end{aligned}$$

$$\forall OBFS \in E$$

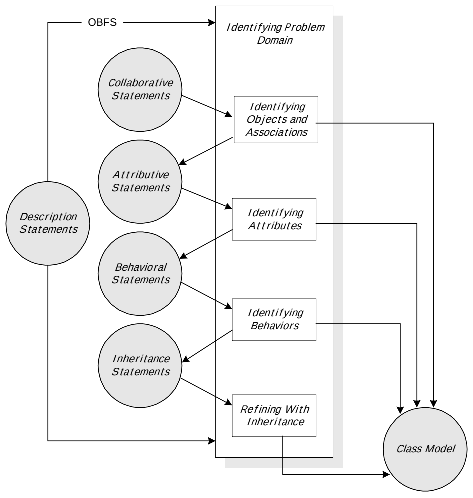


Figure 1: Object Based Formal Specification (OBFS)

Definition 2.1 (*Object-Based Formal Specification (OBFS)*): Object-Based Formal Specification (*OBFS*) is a semi-formal requirements template used to reveal ambiguity, incompleteness, and inconsistency in an object-oriented software system, and to guide end users take an active role while describing their problem statements. OBFS is composed of description statements (*DS*), collaborative statements (*CS*), attributive statements (*AS*), behavioral statements (*BS*), and inheritance statements (*IS*).

2.1. Description Statements (DS)

Description Statements (DS) are used to guide writing an overview of the system that we want to build. DS contain four kinds of elements: *Requirement ID*, *Requirement Name*, *Language*, and *Description* (Figure 2). The description statements should state what is to be done and not how it is to be done. It should be a statement of needs, not a proposal for a solution.

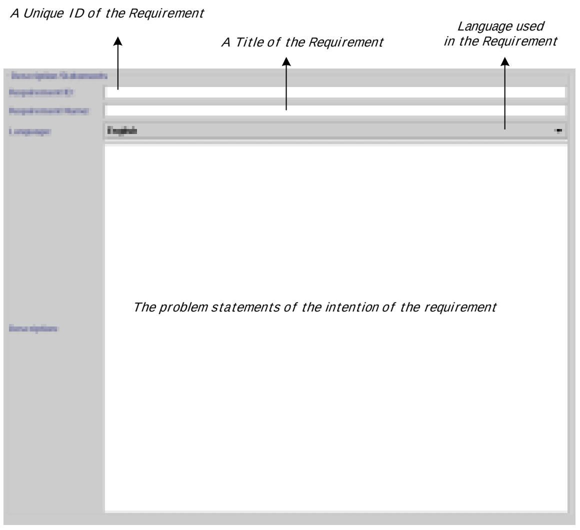

A Unique ID of the Requirement
A Title of the Requirement
Language used in the Requirement
Description Statements
Requirement ID
Requirement Name
Language
English
The problem statements of the intention of the requirement
Description

Figure 2: Description Statements Shell

Definition 2.2 *(Description Statements (DS))*: A description statement is an OBFS statement used to write an overview of the system that we want to build, which consists of *Requirement ID*, *Requirement Name*, *Language*, and *Description*.

2.2. Collaborative Statements (CS)

Collaborative statements (CS) are used to identify objects, and association between objects. The first step in object model creation process is to identify relevant objects and its associations with the application domain. Objects include physical entities and all objects must make sense in the application domain. All objects are explicit in the CS, and objects are corresponding to nouns that are identified from collaborative statements.

Any dependency between two or more objects in the CS is an object association. A reference from one object to another is also an association. Associations show dependencies between objects at the same level of abstraction as the objects themselves. Associations can be implemented in various ways, but such implementation decisions should be kept out of the analysis model to preserve design freedom. Associations often correspond to verbs or verb phrases.

CS consists of Subject (*S*), Verb (*V*), and Object (*O*) as well as the English (*E*) natural language.

$CS = \{(S_1,V_1,O_1)_{cs},(S_2,V_2,O_2)_{cs},(S_3,V_3,O_3)_{cs},...\}$ and

$$\forall CS \in E$$

S_{cs} and O_{cs} will be identified as a tentative object (OBJ_t), and V_{cs} will be identified as a tentative association (ASS_t) in terms of object-oriented paradigm.

$\forall CS \in E\ [S_{cs} \Rightarrow OBJ_t]$ and $\forall CS \in E\ [O_{cs} \Rightarrow OBJ_t]$

$$\forall CS \in E\ [V_{cs} \Rightarrow ASS_t]$$

Definition 2.3 *(Collaborative Statements (CS))*: A collaborative statement is an OBFS statement, which has a tuple {S_{cs},V_{cs}, O_{cs} }. An Object (*OBJ*) is derived from S_{cs} and O_{cs}, and association between object (*ATT*) is derived from V_{cs}.

2.3. Attributive Statements (AS)

Attributive statements (AS) are used to identify object attributes. Attributes are properties of individual objects. Attributes usually correspond to nouns followed by possessive phrases, and sometimes are characterized by adjectives or adverbs. AS must contain properties of each object identified at the previous step.

AS consists of Subject (*S*), Verb (*V*), and Object (*O*) as well as the English (*E*) natural language.

$AS = \{(S_1,V_1,O_1)_{as},(S_2,V_2,O_2)_{as},(S_3,V_3,O_3)_{as},...\}$ and

$$\forall AS \in E$$

O_{as} will be identified as a tentative attribute (ATT_t) in the term of object-oriented paradigm. And S_{as} is identified and refined objects (*OBJ*) from tentative object (OBJ_t), as the final result of object identification's process.

$$\forall AS \in E\ [O_{as} \Rightarrow ATT_t]$$

$$\forall AS \in E\ [S_{as} = OBJ]$$

Definition 2.4 *(Attributive Statements (AS))*: An attributive statement is an OBFS statement, which has a tuple {S_{as},V_{as}, O_{as} }. S_{as} is an identified object (OBJ), and V_{as} is a constant word, which shows that O_{as} is an attribute of S_{as}. The object's attribute (ATT) is derived from O_{as}.

2.4. Behavioral Statements (BS)

Behavioral statements (BS) are used to identify object behaviors. Behavior is how an object acts and reacts, in terms of its state changes and message passing [Booch, 1991]. BS contain behaviors of each object identified at the previous step.

BS consists of Subject (*S*), Verb (*V*), and Object (*O*) as well as the English (*E*) natural language.

$$BS = \{(S_1,V_1,O_1)_{bs},(S_2,V_2,O_2)_{bs},(S_3,V_3,O_3)_{bs},...\}$$

and $\forall BS \in E$

O_{as} will be identified as a tentative behavior (BEH_t) in the term of object-oriented paradigm. And S_{bs} is identified and refined objects (*OBJ*) from tentative object (OBJ_t), as the final result of object identification's process.

$$\forall BS \in E\ [O_{bs} \Rightarrow BEH_t]$$

$$\forall BS \in E\ [S_{bs} = OBJ]$$

Definition 2.5 *(Behavioral Statements (BS))*: An behavioral statement is an OBFS statement, which has a tuple {S_{bs},V_{bs}, O_{bs} }. S_{bs} is an identified object (OBJ), and V_{bs} is a constant word, which shows that O_{bs} is a behavior of S_{bs}. The object's behavior (BEH) is derived from O_{bs}.

2.5. Inheritance Statements (IS)

Inheritance statements (IS) are used to organize classes by using inheritance, to share common object attributes and behaviors. Inheritance provides a natural classification for kinds of objects and allows for the commonality of objects to be explicitly taken advantage of in modeling and constructing object systems. Inheritance is a relationship between classes where one class is the parent class of another. IS provides sentences that have is-a-kind-of relationship.

Inheritance can be added in two directions, bottom up (generalization) and top down (specialization).

- Bottom Up (*Generalization*): By generalizing common aspects of existing classes into a superclass. We can discover inheritance from the bottom up by searching

for classes with similar attributes, associations, or behaviors.
- Top Down (*Specialization*): By refining existing classes into specialized subclass.

Definition 2.6 (*Generalization and Specialization*): Generalization and Specialization are relationships between concepts. Any type of A, each of whose objects is also an instance of a given type B, is called a specialization (or subtype) of B and is written as $A \subset B$. B is also called the generalization (or supertype) of A.

However, mainly we use bottom-up (generalization) concepts as a basic approach to build IS. IS consists of Subject (*S*), Verb (*V*), and Object (*O*) as well as the English (*E*) natural language.

$$IS = \{(S_1, V_1, O_1)_{is}, (S_2, V_2, O_2)_{is}, (S_3, V_3, O_3)_{is}, \ldots\} \text{ and } \forall IS \in E$$

O_{is} will be identified as a tentative superclass (SCL_t) in the term of object-oriented paradigm. And S_{is} is identified and refined objects (*OBJ*) from tentative object (OBJ_t), as the final result of object identification's process.

$$\forall IS \in E \; [O_{is} \Rightarrow SCL_t]$$

$$\forall IS \in E \; [S_{is} = OBJ]$$

Definition 2.7 *(Inheritance Statements (IS))*: An Inheritance statement is an OBFS statement, which has a tuple $\{S_{is}, V_{is}, O_{is}\}$. S_{is} is an identified object (OBJ), and V_{is} is a constant word, which shows that O_{is} is a superclass of S_{is}. The subclass (CLS) is derived from S_{is}, and the superclass (SCL) is derived from O_{is}.

3. IMPLEMENTATION

In this research, object model creation process is viewed as a society of software agents that interact and negotiate with each other. We have devised six types of agents: requirement acquisition agent, object identification agent, attribute identification agent, association identification agent, behavior identification agent, and object refinement agent (Figure 3). Each agent is an intelligent in its own field and may interact with its human counterpart or behave autonomously. This system is named *OOExpert* [Romi et al, June 1999] [Romi et al., July 2000].

- *Requirements acquisition agent* manages the task concerning the requirements acquisition from OBFS.
- *Object identification agent* manages the task concerning the object identification.
- *Attribute identification agent* manages the task concerning the identification of object attributes.
- *Association identification agent* manages the task concerning the identification of associations between the identified objects.
- *Behavior identification agent* manages the task concerning the identification of object behaviors.
- *Object refinement agent* manages the task concerning to refine objects and organize classes by using inheritance to share common structure

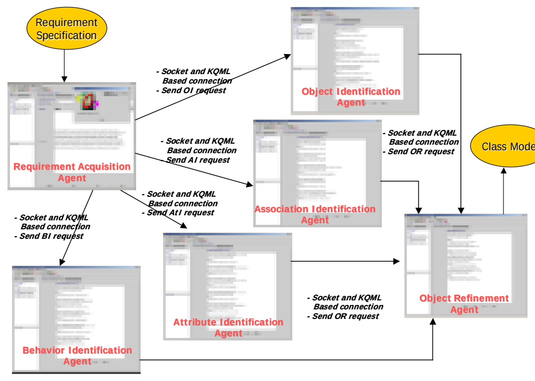

Requirement Specification
Requirement Acquisition Agent
- Socket and KQML Based connection - Send OI request
Object Identification Agent
- Socket and KQML Based connection - Send AI request
Association Identification Agent
- Socket and KQML Based connection - Send OR request
Class Model
- Socket and KQML Based connection - Send AtI request
- Socket and KQML Based connection - Send BI request
Attribute Identification Agent
- Socket and KQML Based connection - Send OR request
Object Refinement Agent
Behavior Identification Agent

Figure 3: *OOExpert* Agents

4. CONCLUSION

We propose an approach where end users take an active role in analysis by specifying requirements using Object Based Formal Specification (OBFS). In this paper we present OBFS and its roles to be a methodological support for specifying requirements in object model creation process, including identification process and object refinement with inheritance process. The models and implementation for using OBFS are also presented.

5. REFERENCES

[Booch, 1991] Grady Booch, "Object-Oriented Analysis and Design with Application", *Benjamin/Cummings*, 1991.

[Booch et al., 1999] Grady Booch, James Rumbaugh, and Ivar Jacobson, "The Unified Modeling Language User Guide", *Addison-Wesley*, 1999.

[Holland et al., 1996] Ian M. Holland and Karl J. Lieberherr, "Object-Oriented Design", *ACM Computing Surveys*, Vol. 28, No. 1, March 1996.

[Liang et al., 1998] Ying Liang, Daune West, and Frank A. Stowell, "An Approach to Object Identification, Selection and Specification in Object-Oriented Analysis", *Information Systems Journal*, Vol. 8, No. 2, 1998, pp. 163-180, Blackwell Science Ltd., 1998.

[Romi et al, June 1999] Romi Satria Wahono and B.H. Far, "OOExpert: Distributed Expert System for Automatic Object-Oriented Software Design", *Proceedings of the 13th Annual Conference of Japanese Society for Artificial Intelligence*, pp.456-457, Tokyo, Japan, June 1999.

[Romi et al., July 2000] Romi Satria Wahono and Behrouz H. Far, "Hybrid Reasoning Architecture for Solving Object Class Identification Problem in the OOExpert System", *Proceedings of the 14th Annual Conference of Japanese Society for Artificial Intelligence*, Tokyo, Japan, July, 2000.

[Rumbaugh et al., 1991] James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, and William Lorenson, "Object-Oriented Modeling and Design," *Prentice Hall*, 1991.

[Rumbaugh et al., 1999] James Rumbaugh, Ivar Jacobson, and Grady Booch, "The Unified Modeling Language Reference Manual", *Addison-Wesley,* 1999.

BIOGRAPHY of AUTHOR

Romi Satria Wahono, Received B.Eng. and M.Eng degrees in Information and Computer Sciences in 1999 and 2001, respectively, from Saitama University. He is currently a researcher at the Indonesian Institute of Sciences (LIPI), and a Ph.D. candidate at the Department of Information and Computer Sciences, Saitama University. The research fields of his interests are Multi Agent Systems, Reasoning System, Software Engineering, and Object-Orientation. He is a member of the ACM, IEEE Computer Society, The Institute of Electronics, Information and Communication Engineers (IEICE), Japanese Society for Artificial Intelligence (JSAI), and Indonesian Society on Electrical, Electronics, Communication and Information (IECI).

Behrouz Homayoun Far, Received BSc. and MSc. degrees in Electronic Engineering in 1983 and 1986, respectively, from Tehran University, Iran. He has received his Ph.D. degree from Chiba University - Japan, in 1990. He is currently an Associate Professor at the Department of Information and Computer Sciences, Saitama University - Japan. The research fields of his interest are qualitative reasoning, automatic programming and distributed AI. Dr. Far is a member of the ACM, IEEE Computer society, Japanese Society for Artificial Intelligence, IEICE and Information Processing Society of Japan.