Software Environment Modelling Method

Qiuying Li¹,²,⋆ and Ruize Sun¹,²

¹School of Reliability & Systems Engineering, Beihang University, China
²Science & Technology on Reliability & Environmental Engineering Laboratory, Beijing, China

*Corresponding author email: li_qiuying@buaa.edu.cn

Abstract. The influence of software environment and its changes on software functions and performance is becoming more and more important. Through literature investigation, this paper studies the concept, composition and characteristics of software environment, and analyzes the relationship between software environment and software requirements. This paper also presents the environment modelling method based on feature model and ontology model, and proposes the procedure and the important steps, together with the differences between the two models. Finally, case studies are given for each model.

Keywords: Software environment; Environment modelling; Feature model; Ontology model; Procedure.

1. Introduction

Software environment is an important factor affecting software operation. In recent years, with the vigorous development of the internet, the application scenarios of software continue to expand. People hope that software can work normally not only in a certain environment, but also that when environment changes, the software can perceive such changes and adjust itself to ensure functions and performance. This kind of software is so-called self-adaptive software, which refers to the software that can monitor changes in the environment and the system itself, and can dynamically modify its behavior and structure in response to changes. No matter what kind of self-adaptive software, its adaptation is based on “collection-decision-control”, and it is essential for the modelling and expression of environmental information and environment changes.

Software environment is open, dynamic, uncontrollable and unpredictable. Existing researches on software environment are mainly divided into the description of state information of environment and changing rules of environment. The description of state information helps to further improve software requirements and generate more comprehensive and effective test cases in software development and testing, quickly and accurately locate faults during software maintenance. In addition, environment itself is independent, which can change autonomously, and is not dependent on the existence of software. Therefore, the environment model can be reused on different software, that is, the description of environment should be reusable.

[1] defined software environment as all the contexts that can be observed by software system, including the physical environment, deployment environment, user interaction environment, etc. [2] described environment as a combination of clock and resources, where clock is used to maintain the environmental running sequence and the resource represents an entity or variable that may change. [3] pointed out that environment behavior is a function of the time, state and operation. [4] used feature model to express the environment, and the resulting model meets the minimum set of environmental features defined by the feature rules whose values are updated by sensors. [5] used UbiFEX to divide
features into entity and information characteristics. [6] modeled environment based on ontology language and used semantic web technology and OWL-DL language. [7] used threat models to model security threats in environment. [8] modeled environment as a random process and environmental change through the changes of random variables. [9] divided the degree to which the environment can meet the software functional requirement and made assumptions based on the idealization degree of environment.

In general, the existing methods of describing environment have a relatively small scope of application, only applicable to certain types of software (such as web service software), and have limited description capabilities. Also their characterizing the changing rules of environment is relatively simple and insufficient. To solve these problems, researchers suggest environment models based on different types and aspects of environment, such as feature models, ontology models, threat models, probabilistic processes, time series languages, and element hierarchical models, but these models are only applied on specific software examples without forming a concrete process.

This paper studies the concept, composition and characteristics of software environment, presents environment modelling method based on feature model and ontology model, explains the important steps, and compares the difference of these two models. Finally, case studies are given for each model.

2. Environment Modelling Based on Feature Model

2.1. Overview

Feature model mainly describes the state of environment at a certain moment. Feature engineering is a method of mapping system from problem space to solution space. Thus feature model is a realization of feature engineering method, which is mainly composed of features and feature combination rules. The essence of building a feature model is to find features and determine the relationship between features. Features are derived from software requirements or functions. The meaning of features can be understood from the following two aspects: product features are used to describe product characteristics, that is, each relatively independent subset of requirements, and the relationship between features. The refined relationship mainly includes four types: mandatory, optional, xor and or. The first two describe the inclusion relationship between parent-child features, and the latter two describe the selection relationship between parent-child features. Constraint relations include require and exclude, which are used to express compatibility between features.

Environment feature model is a structured model that refers to software requirements and extracts features and the relationship between features. It can express the variability of environment and output possible environment configurations [10].

2.2. Modeling Process

The flow chart of feature model method is shown in figure 1.

![Figure 1](image_url)

**Figure 1.** The flow chart of environment feature model method.

Step 1: Extract environmental features. Software requirements play an important reference in determining the characteristics of environment. Although the requirements are not detailed and specific, but the level of requirement details determines the pros and cons of feature model. After giving the description of requirements to a certain extent, the next step is to search entities that affect the expression of requirements in physical environment and interactive environment where the
software is located. Entities include, but not limited to, physical objects perceived or controlled by software and software users. Then clarify the specific information that the entity has on software behavior.

Step 2: Determine the relationship between features. Features can be divided into environmental entity features and information features.

Step 3: Build model. The relationship between features can be organized into a tree diagram, where features are nodes of the tree, and relationships between features are connecting lines between nodes.

Step 4: Output environment configuration set. Environmental feature configuration represents a possible state of environment at a certain moment. But one thing should be noted that the output of feature model is static. In other words, when we model environment according to feature model, we can only guarantee that the current environment at a certain moment is a set of feature configurations, but environment value over time cannot be obtained by feature model [11].

2.3. Case Study

Here take an adaptive software as an example. The software is used for sewage treatment, and its main function is to apply the best dosage according to water quality and temperature. The sensor can give the state of the environment/water quality during operation, and the software controls the delivery of drugs through calculation. The modeling process is given as follows:

1) Determine the environmental entity that affects the realization of the function according to the software function: sewage and drug delivery device.

2) Determine the features describing environmental entities as shown in table 1.

| Feature name   | Value type | Continuity   | Value range     | Unit       | Data source   |
|----------------|------------|--------------|-----------------|------------|---------------|
| Temperature    | numerical  | continuous   | [0,100]         | Celsius    | temperature sensor |
| PH             | numerical  | continuous   | [0,14]          | PH sensor  |               |
| Density        | numerical  | continuous   | [0.5,10]        | kg m^-3    | density sensor |
| Turbidity      | numerical  | continuous   | [0,100]         |           | turbidity meter |

The features of drug delivery device are shown in table 2.

| Feature name           | Value type | Continuity | Value range               | Unit   |
|------------------------|------------|------------|---------------------------|--------|
| Working mode           | character  | discrete   | [intermittent, continuous] |        |
| Power consumption mode | numerical  | discrete   | [1,5,10]                  | kilowatt |
| Raw material quality   | character  | discrete   | [quality, qualified, unqualified] | |
| Raw material adequacy  | character  | discrete   | [equal, sufficient, insufficient] | |

3) Analyze the relationship between features and organize them into feature models.

4) According to feature model, output environment configuration set as shown in table 3.

| Configuration Feature | Configuration 1 | Configuration 2 | Configuration 3 |
|-----------------------|-----------------|-----------------|-----------------|
| Temperature           | 20              | 25              | 15              |
| PH                    | 5.6             | 6.5             | 8               |
| Turbidity             | 5               | 3               | 5               |
| Density               | default         | 1.3             | default         |
| Working mode          | intermittent    | continuous      | intermittent    |
| Power consumption mode| 5               | 10              | 1               |
| Raw material quality  | qualified       | unqualified     | quality         |
| Raw material adequacy | insufficient    | sufficient      | equal           |
3. Environment Modelling Based on Ontology Model

3.1. Overview
Ontology is a special type of term sets with structural characteristics, which is actually a formal expression of conception sets and their relationships in a specific field. Currently, OWL is the most commonly used language for describing ontology. An ontology is composed of individuals, attributes and classes. An individual refers to an object of interest in the domain and is an instance of a class. Attributes refer to the binary relationships on individuals, and are divided into object attributes and data attributes. The former describe the relationships between individuals, such as inclusion and association, and the latter describe the attributes of the individuals, such as the physical parameters of the individuals. One class can be seen as a collection of individuals. Both classes and attributes have a hierarchical structure. For example, “Class cat” is a subclass of “Class animal”, and “Attribute father” is a sub-attribute of “Attribute parents”. The so-called attribute reasoning is to reason about the undefined but existing relationships between two individuals based on the hierarchical relationships.

3.2. Method Process
The environment ontology model is a model that describes the environment as an ontology, which can use ontology knowledge to describe the knowledge in the environment and infer concepts based on ontology language rules. The model consists of individuals, classes and attributes. The class in environment ontology are similar to the parent-child feature definition in feature model. The flow chart of ontology model method is shown in figure 2.

1) Clarify the scope of software environment, that is, the scope of ontology domain, which mainly refers to software requirements to make judgments about which objects in software environment need to be considered instead of making judgment from environment itself.
2) Environment ontology creates classes from top to bottom. There are no multiple levels of environment classes. Each class is an environment module composed of several entities with the same attributes or functions in the environment.
3) Clarify the individuals in each class, and an individual is an entity under the class.
4) Clarify data attributes and object attributes. Investigate the entity parameter of software behavior or relationship as data attributes of the entity class, along with the relationships between entities and how to express the relationships.
5) Classes, individuals and attributes are combined into a hierarchical model according to relationships. According to this model, the attributes of any individual can be inferred [12].

3.3. Case Study
Take a tourist schedule software which is a self-adaptive software as an example. The software lists tourist attractions in the city, then recommends travel plans to these areas based on weather and user’s current location. The software mainly includes the following functions: browsing
weather information (provided by two service providers), obtaining location information (including the current locations of tourists and recommended routes and time to tourist attractions), traffic information (location of bicycle rental stations and public transportation, the location of train station and the number of trains that need to be taken) and ticket service. The ontology model is shown as follows:

1) Create class. There is only one class in this example: travel planning service.
2) Investigate individuals in the category: weather service provider 1, weather service provider 2, location, bicycle service, bus service, travel mode service and ticket service.
3) Clarify data attributes, which are shown in table 4.

Table 4. Environment data attributes.

| Individual          | Data attribute       | Type     | Individual          | Data attribute       | Type     |
|---------------------|----------------------|----------|---------------------|----------------------|----------|
| Weather service provider | Current temperature | numerical | Bus service         | Station location    | character |
|                     | Wind rating          | numerical |                    | Recommended bus     | character |
|                     | Response time        | numerical |                    | Payment rules       | character |
| Travel mode service | Mode                 | character | Ticket service      | Payment port        | logical  |
|                     | Weather Port One     | logical   |                    |                      |          |
|                     | Weather Port Two     | logical   |                    |                      |          |
|                     | Bicycle port         | logical   |                    |                      |          |
|                     | Bus port             | logical   |                    |                      |          |
| Bicycle service     | Station location     | character | Location service    | Location information| character |
|                     | Payment rules        | character |                    | Distance to target  | numerical |
|                     | Location port        | logical   |                    | Recommended route    | vector   |

4) Build the ontology model.
5) Output configuration sets that meet the ontology inference rules shown in table 5.

Table 5. Configuration sets.

| Individual          | Data attribute       | Type     | Individual          | Data attribute       | Type     |
|---------------------|----------------------|----------|---------------------|----------------------|----------|
| Weather service provider 1 | Weather Type | cloudy   | Weather service provider 2 | Weather Type | cloudy   |
|                     | Current temperature  | 15°C     | Wind rating          | Wind rating          | 2        |
|                     | Wind rating          | 3        | Response time        | Response time        | 0.7s     |
| Travel mode service | mode                 | bicycle  | Station location     | Recommended bus      | false    |
|                     | Weather Port One     | true     | Bus service          | Payment rules        | false    |
|                     | Weather Port Two     | true     |                      |                      |          |
|                     | Bicycle port         | true     | Location port        | false                |          |
|                     | Bus port             | false    |                      |                      |          |
| Bicycle service     | Station location     | XX district XX Road XX no. | Location information | Location information |          |
|                     | Payment rules        | XXXXXXXXXX | Distance to target  |                       | 10 km    |
|                     | Location port        | true     | Recommended route    |                       | vector   |
| Ticket service      | Amount               | 100 yuan |                      |                      |          |
|                     | Payment port         | normal   |                      |                      |          |

From table 5, it can be seen that there are similarities between ontology model and feature model. Ontology model is more suitable for environment modeling of web service software, because web service software is object-oriented and similar to the structural characteristics of ontology model.
4. Conclusion
Feature model and ontology model mainly describe the state of environment at a certain moment. However, the application range of feature model is relatively wider, and the concept of the feature is more general, while the ontology model is mainly used for web service software, whose scope of application is relatively small. We can use ontology knowledge to reason about the attributes in the domain, which has stricter regulations and stronger expressive capabilities for the relationships between environmental entities. But the shortcomings of these two methods are also obvious, feature model is not completely independent of software model, and the perception of the environment is always demand-driven, and the effect of the model is greatly related to the perfection of the demand.

Acknowledgments
This work was supported in part by the Defense Key Laboratories Support Program of China (Grant No. WDZC2019601A303).

References
[1] Zhao TQ, Zhao HY, Zhang W, et al 2018 Survey of model-based self-adaptation method Journal of software vol 29 no 1 pp 23-41 (in Chinese)
[2] Jiang HT, Liu Y, Lin HS, et al. 2016 Environment model of self-adaptive software and its implementation Journal of Frontiers of Computer Science & Technology vol 10 no 3 pp 301-310 (in Chinese)
[3] Wang QX, Shen JR and Mei H 2004 An introduction to self-adaptive software Computer science vol 10 pp 168-171+178 (in Chinese)
[4] Wang JC and Gao Z 2002 Research on software requirement analysis Computer engineering and design vol 8 pp 18-21 (in Chinese)
[5] Fernandes P, Werner C and Murta L G P 2008 Feature Modeling for Context-Aware Software Product Lines Proceedings of the Twentieth International Conference on Software Engineering & Knowledge Engineering (SEKE'2008) San Francisco, CA, USA, July 1-3
[6] Alférez G H and Pelechano V 2011 Context-Aware Autonomous Web Services in Software Product Lines International Software Product Line Conference (IEEE Computer Society)
[7] Cheng B H C, Sawyer P, Bencomo N, et al. 2009 A Goal-Based Modeling Approach to Develop Requirements of an Adaptive System with Environmental Uncertainty Proceedings of the 12th International Conference on Model Driven Engineering Languages and Systems
[8] Moreno G A, Câmara J, Garlan D et al 2015 Proactive self-adaptation under uncertainty: a probabilistic model checking approach Proceedings of the 10th Joint Meeting on Foundations of Software Engineering pp 1-12
[9] D'Ippolito N and Kramer J 2014 Hope for the best, prepare for the worst: multi-tier control for adaptive systems Proceedings of the 36th International Conference on Software Engineering ACM
[10] Bencomo N, Sawyer P, Blair G et al 2008 Dynamically Adaptive Systems are Product Lines too: Using Model-Driven Techniques to Capture Dynamic Variability of Adaptive Systems Software Product Lines, 12th International Conference vol 2
[11] Neskovic S and Matic R 2015 Context Modeling based on Feature Models Expressed as Views on Ontologies via Mappings Computer Science and Information Systems vol 12 no 3 pp 961-977
[12] Shaaban A M, Schmittner C, Gruber T, et al 2019 Ontology-Based Model for Automotive Security Verification and Validation Proceedings of the 21st International Conference on Information Integration and Web-based Applications & Services