Application of The Agent-oriented Analysis to A Software System in Problem Cause Identification using Fuzzy Decision Tree

Rui-Yang Chen*

Department of Business Administration, Aletheia University, Taiwan.

*Corresponding author: a168.jacky@msa.hinet.net

Abstract:
A problem cause identification software system is usually very complex and may contain multi-roles dealing processes such as R&D, customer and manufacturing. The causes of a problem may have cognitive uncertainty properties, which need to be analyzed for the purpose of algorithm-based automatic problem-solving. In this study, an application of the scenario-driven agent-oriented analysis to a software system in problem cause identification is proposed. A case study is conducted to demonstrate the problem-solving algorithm for problem cause identification of a notebook computer product. The major contributions of this research include: (1) The proposal of an autonomous problem-solving system base on a scenario-driven agent-oriented approach. (2) The proposal of a complex system of problem cause identification based on the fuzzy data.

Keywords:
Fuzzy Decision Tree; Agent-oriented; Scenario-driven; DFX; Problem Cause Identification

1. INTRODUCTION

Traditionally, the customer complaints have only been processed by the RMA (Return Merchandise Authorization) department, which usually do not deal with cognitive uncertainties such as vagueness associated with product problems and cause perceptions.

Two issues may arise from the above practice.

The first issue is about the redesign of a product based on customer’s complaints with the consideration of all the downstream related attributes for product design, manufacturing, sales, and so on. For this purpose, the concept of design for X (DFX) plays a key role by considering the most important X item constraints synchronously in the design stage and integrates related processes of new product development [1]. More recently, for manufacturing, quality, maintainability, reliability, and assembly, researchers have focused their studies on design for the X item [2–4].

The second issue is the automated mapping of cognitive uncertainty attributes with thinking and perceptions of different process stages. The related classification problems need to be measured, analyzed and tracked into the customer complaint process. However, the product problems reported by customers may have observation errors, uncertainty, subjective judgments. Then, the obtained data usually involve
fuzzy descriptions [5].

In the notebook computer industry, the R&D engineer plays a very important role. Some manufacturers will ask their vendors to do the maintenance works of the sold products for their customers. As timely new product development becomes critical due to the shorter product life cycle, a software system for quickly analyzing and sending customer complaints to the related product design engineers plays a very important role. The software system should have the capability for problem cause identification with respect to customer complaints [6]. The problem cause identification software system is usually very complex. It involves multi-roles dealing processes such as R&D, manufacturing and customer. Also it may have cognitive uncertainties which need to be properly analyzed to identify causes of the problem with fuzzy symptoms.

To address the above-mentioned issues, an application of the scenario-driven agent-oriented analysis to a software system in the problem cause identification is proposed in this research. It includes the following two parts:

1. It provides a quantitative methodology based on the fuzzy decision tree for the problem cause identification system.

2. It provides a case study using a scenario-driven agent-oriented analysis for analyzing a software system in the problem cause identification for the notebook computer product.

The rest of the paper is organized as follows. Section 2 reviews literature for fuzzy decision tree, agent-oriented analysis and DFX. Section 3 describes the problem formulation. Section 4 presents the proposed theoretical approach of the fuzzy decision tree algorithm for the problem cause identification. A scenario-driven agent-oriented analysis approach for a problem cause identification system is provided in Section 5, and some conclusions are given in Section 6.

2. LITERATURE AND PREVIOUS WORK

2.1 The fuzzy decision tree methodology

Decision trees classify data by sorting them with the tree from the root to the leaf nodes. On the other hand, a fuzzy decision tree allows data to follow down simultaneously multiple branches of a node with different satisfaction degrees ranged in [0, 1]. A fuzzy decision tree is a generalization of the crisp case. Fuzzy sets defining the fuzzy terms used for building the tree are imposed on the algorithm. One of the main objectives of the fuzzy decision tree induction is to generate a tree with high accuracy of classification for unknown cases. Experimental results [7] show that the selection of attributes is an important factor to influence the accuracy of classification. In [8], the construction method for the fuzzy decision tree is designed for a classification problem with attributes and classes represented in fuzzy linguistic terms. According to the above literature, the fuzzy representation is becoming more and more popular in dealing with problems of uncertainty, noise, and inexact data [9, 10]. It has been successfully applied to problems in many industrial areas.

2.2 DFX

DFX means that one needs to consider the most important X item constraints synchronously in the design stage. More recently, researchers have focused their key point such as manufacturing, quality,
maintainability, reliability and assembly in design for the X item [2, 4, 11]. Besides this, a comprehensive information management system is necessary to implement these two key functions: design for manufacturability and manufacturing information feedback to design [12]. Since 1990s, the speed and the flexibility in the new product development also had received increased priority [13, 14]. While the innovation and environmental impact of a new product is emphasized, it accelerates the eco-innovation product design by integrating case-based reasoning and the TRIZ method [15].

Although the DFX literature is widely distributed over many different disciplines, DFX is applied to design for the problem cause identification data in this study. However, these data are usually vague and unclassified. For example, when a R&D engineer is designing the battery of a product and thinking about possible causes, there are three symptoms associated with the product problem: \{ Battery life too short, electrify not full and voltmeter incorrect \}. The vague data expressions in the problem cause identification are illustrated in Table 1. For example, the probability ascertained for the symptom “electrify not full” is between 0 and 1 of fuzzy values.

### 2.3 Agent-oriented analysis

The problem cause identification software systems are usually very complex and difficult to analyze due to the interaction between different functional departments. Agent-oriented software engineering is one of the most recent contributions to the field of system analysis [16–19]. It has several benefits compared to existing analysis approaches such as procedure-oriented, module-oriented and object-oriented [20, 21], in particular, extant approaches fail to adequately capture autonomous problem-solving behavior, the richness of an agent’s interactions, and the complexity of an agent system’s organizational structures [22, 23]. But first, what is an agent? An agent, also called a software agent or an intelligent agent, is a piece of autonomous software. Modeling complex expert systems often requires using the behavior of intelligent agent that can be trained within simulation system [24, 25]. Complex expert system may be a very time consuming and complex process. Multi-agent systems fulfill autonomous needs related to automating decision-making procedure [26]. A single agent can’t solve complex problems. Some pervious papers have applied multi-agent system for mobile robot complex system [27].
In this paper, an agent-oriented analysis approach from Wooldridge [12] is presented. The approach provides an agent-specific set of concepts through which a software engineer can analyze a complex large-scale system as shown in Figure 1.

It may seem strange to think of a computer system as being defined by a set of roles, but the idea is quite natural when adopting an organizational view of the world. Considering the problem causes identification such as a customer, who may have a role such as “consumer”, “R&D engineer”, and so on. A role is defined by three attributes: responsibilities, permissions, and protocols. Responsibilities determine functionality and are perhaps the key attribute associated with a role. An example of the responsibility associated with the role of a R&D engineer might be the assigned responsibility to identify the causes when a customer complaint problem has occurred. In order to realize responsibilities, a role is usually associated with a set of permissions. For example, a maintenance engineer role might have the associated permission to identify the causes of a product problem. Finally, a role is also identified with a number of protocols, which define the ways that it can interact with other roles. For example, a manufacturer role might have the protocols “response result” and “deal problem” associated with it.

3. PROBLEM FORMULATION

Normally, some data produced by problem cause identification are difficult to express, and they are vague and unclassified. We can use the fuzzy set and the decision tree method to deal with the above-mentioned two issues. Fuzzy set theory deals with the vagueness. Decision trees classify the data important index by sorting these data with a tree from the root to the leaf. A fuzzy decision tree combines the fuzzy set with the decision tree. A major contribution of the fuzzy decision theory is its capability of representing vague and unclassified data. For example, a designer’s knowledge about how to identify for the maintenance cause data is usually vague and there is no crisp boundary for it. The cognitive uncertainties and the classification method for knowledge comprehensibility can be well represented by Yuan’s fuzzy decision tree methodology [8].

The problem formulation could be represented as follows:

A Maintenance problem item \( p_i \) is described by a collection of the symptom \( K_j \). This a symptom gets one of the mutually exclusive values \( V_j = \{V_{j1}, ..., V_{jk}\} \) from the R&D engineer’s past experiences, and each process node is classified into only one of the mutually exclusive data \( D_i = \{D_{i1}, ..., D_{im}\} \) from the maintenance cause data.

Nomenclature

\[ P = \{p_i\}, \text{ maintenance problem item } i \]
Table 2. The fuzzy data

<table>
<thead>
<tr>
<th>LCD screen</th>
<th>show up regularly</th>
<th>less DPI</th>
<th>causes</th>
</tr>
</thead>
<tbody>
<tr>
<td>H M L</td>
<td>H M L S W M</td>
<td>0.3</td>
<td>Display chip damage</td>
</tr>
<tr>
<td>0.1 0.2 0.6</td>
<td>0.7 0.1 0.5 0.3 0.7</td>
<td></td>
<td>Power cord touch failures</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.3</td>
<td>Membrane transistor damage</td>
</tr>
</tbody>
</table>

$k_i = \{k_{i1}, ..., k_{ij}\}$, symptoms of the problem, where $j$ is the symptom of the problem item $i$

$v_{jn} = \{v_{j1}, ..., v_{jn}\}$, values of the symptom, where $n$ is the value of the symptom item $j$

$d_{im} = \{d_{i1}, ..., d_{im}\}$, causes of the problem, where $m$ is the cause of the problem item $i$

An example of the data with fuzzy values is shown in Table 2. It lists the maintenance causes of a LCD screen.

It has three symptoms:

- $S = \{\text{screen has an image pictures, show up irregularly, less DPI}\}$, and each symptom has three values:
  - $(V_{j1})$ Screen has an image pictures $= \{\text{high (H), medium (M), low (L)}\}$
  - $(V_{j2})$ Show up irregularly $= \{\text{high (H), medium (M), low (L)}\}$
  - $(V_{j3})$ Less DPI $= \{\text{strong (S), weak (W), medium (M)}\}$

It has three classes:

- $(D_{im})$ causes $= \{\text{Display chip damage, Power cord touch failures, Membrane transistor damage}\}$

From the above statements, one heuristics definition of the process node for classifying a class of cause relationships in the fuzzy decision tree could be found. In the definition, it has two key points: one point is how to decide the symptom to classify the class of cause (see definition1); the other point is how to select the necessary symptoms (see definition2)?

**Definition 1:** Let $R$ denote a relational probability, it attributes a symptom to the class of cause. $R$ is defined by

$R(K_{ij} = V_{jn}, D_{im})$

**Definition 2:** Compare the values of $R$ with one another with different symptoms. If the value is bigger, then its priority is higher.

$\text{MAX}\{R(K_{i1} = V_{1n}, C_{1m}), R(K_{i2} = V_{2n}, C_{1m}), ..., R(K_{ij} = V_{jn}, C_{1m})\}$ (1)

From the above description, there are two questions which need to be solved:

2. How to analysis the complex large-scale system in the problem cause identification.

This study will focus on solving the above two questions. The framework of this study is given in Figure ??.

### 4. IDENTIFY CAUSES AND DEPARTMENT RESPONSIBILITY OF PRODUCT PROBLEM

#### 4.1 An algorithm using use fuzzy decision tree method

We define a classification accuracy rate for each symptom, which is attributed to a class of causes. If it could not be classified clearly, it shows that there is a probability of classification ambiguity. This
probability is desired to be lower for ease of classification. If the classification accuracy rate is higher than the standard value (0.9) of the classification accuracy rate, then the symptom will be classified to that class of causes. A fuzzy decision tree is a generalization of the crisp case. Before giving the definition, we denote the following symbols:

\[ U(X) \] is the family of all fuzzy subsets defined on \( X \), and the cardinality measure of a fuzzy subset. \( x \) is given finite data of \( U(X) \).

\[ S = \text{Attribute, } C = \text{Classes} \]

\( S \) is defined by the membership of a fuzzy subset \( U_S \) which takes values in the interval \([1, 0]\) in a universe of \( U(X) \). For \( x \in U, U_S(x) = 1 \) means that \( x \) is definitely a member of \( S \) and \( U_S(x) = 0 \) means that \( x \) is definitely not a member of \( S \), \( S \) is a crisp set.

\[
M(S) = \sum_{x \in U} S(x), \text{Probabilitysummation for } S
\]  

\[
M(S \cap C) = \sum_{x \in U} \min(U_S(x), U_C(x)); \quad (3)
\]

The intersection \( S \cap C \) is defined by probability summation for \( S \) and \( C \).

Then, we describe the definition of algorithm as follows.

First, determine the symptoms which belong to a particular a class of cause (see definition3):

**Definition 3**: Given the fuzzy data, the accuracy probability of the classification can be defined as

\[
S(S, C) = \frac{M(S \cap C)}{M(S)} = \frac{\sum_{x \in U} \min(U_S(x), U_C(x))}{\sum_{x \in U}(U_S(x))} \quad (4)
\]

The fuzzy subsethood \( S(S, C) \) measures the degree with the accuracy probability of the classification to which \( S \) is a subset of \( C \).

It is transferred to the paper formulation:

\[
S(S_{ij} = V_{jk}, C_{im}) = \frac{M(S_{ij} = V_{jk} \cap C_{im})}{M(C_{im})} \quad (5)
\]

It measures the classification probability of the mapping attributes (symptoms) which belong to a certain class (the maintenance cause), and it replaces the traditional decision tree entropy with the accuracy probability of classification.

The definition is similar to \( R(K_{ij} = V_{jm}, D_{im}) \)
For example, \( S(\text{screen has an image pictures } = \text{H}, \text{Display chip damage}) = P(\text{screen has an image pictures } = \text{H}, \text{Display chip damage}) / P(\text{screen has an image pictures } = \text{H}) = 0.93 \).

Second, select the necessary data class (see definition 4):

**Definition 4**: Given a set of fuzzy data, the probability of classification ambiguity could be defined as

\[
G(S) = \frac{\sum_{j=1}^{j} G(S_j)}{j}
\]

Given fuzzy attribute values \( Z \) for a specific \( j \) attribute, the possibility of classification an attribute to a class of maintenance causes:

\[
G(S_{j, Z}) = \sum_{Z=1}^{n} (S_{Z} - S_{Z+1}) \ln Z
\]

\( S_{Z} \) represents the accuracy probability of classification in different classes by normalization of the \( Z \) attribute value.

\[
G(S_j) = \frac{\sum_{j=1}^{j} G(S_j)}{j}
\]

For example:

- \( S(\text{screen has an image pictures } = \text{H}, \text{Display chip damage}) = 0.93 \);
- \( S(\text{screen has an image pictures } = \text{M}, \text{Power cord touch failures}) = 0.86 \);
- \( S(\text{screen has an image pictures } = \text{L}, \text{Membrane transistor damage}) = 0.61 \);
- Accuracy probability of classification by normalizing = (1, 0.92, 0.66);
- \( G(\text{screen has an image pictures } = \text{H}) = (1-0.92)\ln 1+(0.92-0.66)\ln 2+(0.66-0)\ln 3 = 0.9 \);
- \( G(\text{screen has an image pictures } = \text{M}) = 0.81 \);
- \( G(\text{screen has an image pictures } = \text{L}) = 0.86 \);
- As result: \( G(\text{screen has an image pictures}) = (0.9+0.81+0.86)/3 = 0.85 \);
- Gain as the same procedure:
- \( G(\text{Show up irregularly}) = 0.88 \); \( G(\text{Less DPI}) = 0.66 \).

### 4.2 Process using Scenario-driven analysis method

This is a Scenario-driven analysis method that exists within a real-world process to provide a level of flow to which roles can interact. The role understands how to interact with other roles and also understands the permission relationship associated with the event. They make response to the inquisition from database and algorithm result.

The Scenario-driven analysis of the problem cause identification is given in Figure ??.

### 5. A PROBLEM CAUSE IDENTIFICATION SYSTEM BASED ON THE AGENT-ORIENTED APPROACH

#### 5.1 A scenario-driven agent-oriented analysis approach

The scenario-driven analysis in Figure 3 contains such components such as roles, event, flow and algorithm. These scenarios correspond to the physical requirement statements in the real-world. So, it is important that how to analyze the system with system analysis approach that interact to achieve the
requirement statement. The system analysis approach must adequately capture autonomous problem-solving of an algorithm such as cause identification of problem and the complexity of a role’s interactions such as problem dealt respond. In this paper, it provides a case study using a scenario-driven agent-oriented analysis for analyzing a software system in problem cause identification for notebook compute product. The approach is differential to agent-oriented analysis approach from Michael Wooldridge [12]. The approach is intended to allow that an analyst explore problem-solving of an algorithm autonomously from a scenario of requirement statements in the agent-based system. The Scenario-driven agent-oriented analysis approach is given in Figure 4.

5.2 Case study

In this paper, it provides a case study using a scenario-driven agent-oriented analysis for analyzing a software system in problem cause identification for notebook compute product. For reasons of brevity, we omit some details, and aim instead to give a general flavor of the analysis.

The analysis step of the approach can now be summarized:

Step1. Transfer scenario-driven process to agent-oriented system.

For example, it transfers process such as “identify” to system such as “identify agent”.

Step2. Define the roles in the system.

By means of an illustration, consider the above same example such as “identify agent”. A roles definition is comprised of a set of role schemata in the system. An exemplar instantiation is given for the maintenance engineer role in Figure 5. This schema indicates that maintenance engineer has permission to search identified result from database (that indicates whether the symptom with cause by symptom values is classified or misclassified), and retrieve data (that indicates which the role is intended to retrieve data of cause and responsibility department). In addition, the role has permission to define problem.
Step 3. For each role, find the associated protocols. Protocols are the patterns of interaction that occur in the system between the various roles.

Step 4. Using the interaction model as a basis and elaborate the roles model. By means of an illustration, consider the protocol identify, which forms part of the maintenance engineer role (Figure 6). This states that the protocol identify is initiated by the role maintenance engineer and involves the role department. The protocol involves maintenance engineer calculating algorithm in the method named fuzzy decision tree, and results in database being informed about the value of classification.

Step 5. For each process, find the associated algorithm. Algorithm is the problem-solving of interaction that occur in the system between the various roles for a protocol. An exemplar instantiation is given for the algorithm fuzzy decision tree of protocol identify in Figure 7. The source of method comes from Cezary’s approach [28].

Step 6. Iterate steps (1)–(5).
In this paper, an application of the scenario-driven agent-oriented analysis to a software system in problem cause identification is proposed. The key concepts in this approach are roles, which have associated with them responsibilities, permissions and protocols. Roles can interact with one another in certain problem-solving algorithm such as fuzzy decision tree, which are defined in the protocol “identify” of the respective roles “maintenance engineer” and “database”.

The major contributions of this research include: (1) The proposal of an autonomous problem-solving system based on a scenario-driven agent-oriented approach. (2) The proposal of a complex system of problem cause identification based on the fuzzy data.

References


