A PATTERN LANGUAGE OF CONCEPTUAL DATA MODEL PATTERNS BASED ON FULLY COMMUNICATION ORIENTED INFORMATION MODELING (FCO-IM)

Fazat Nur Azizah*, Guido P. Bakema**.

*School of Electrical Engineering and Informatics, Bandung Institute of Technology, Jln. Ganesha no. 10 Bandung, Indonesia fazat@stei.itb.ac.id

**Faculty of Engineering, Institute of Information Technology, Media and Communication, HAN University of Applied Sciences, Ruitenberglaan 26, 6802 CE, Arnhem, The Netherlands guided.bakema@han.nl

ABSTRACT

A problem in data modeling is that creating a high quality data model is not an easy task and data modelers often deliver low quality results due to unfamiliarity to data modeling principles and standards. This problem can be reduced if they are acquainted with standards which ensure that the data models both meet business needs and be consistent. In this research, we propose the use of data model patterns arranged in a pattern language based on Fully Communication Oriented Information Modeling (FCO-IM) as the modeling approach. In this paper, we present the concepts of a pattern language of conceptual data model patterns (including the concept of Information Grammar for Pattern/IP), as well as a list of conceptual data model patterns based on FCO-IM. We have carried out some tests to prove whether the resulting conceptual data models (FCO-IM information grammar) which are produced using the pattern language hold high quality. The test is carried out on four measurements: syntax correctness, feasible validity, feasible completeness, and feasible comprehension. Based on the test results, we conclude that the conceptual data models are of high quality, particularly in syntax correctness and validity aspect.

Keywords: data modeling, conceptual data model pattern, pattern language, FCO-IM, information grammar for pattern

1. INTRODUCTION

Data modeling is an activity to create a data model by applying formal data model descriptions using data modeling techniques. A data model is a collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints [20].

The quality of the data models is foremost since they provide support to organization’s activities. Nevertheless, creating a good data model is not an easy task. The quality of data models depends in high extent on the skill of the data modelers. Lack of familiarity to data modeling principles and standards causes data modelers delivering low quality results. This problem can be reduced if the data modelers are acquainted with standards which ensure that the data models both meet business needs and be consistent [23].

In this research, we propose the use of conceptual data model patterns arranged in a pattern language [1] as a standard in data modeling. Data model patterns are used in data model design in order to achieve high quality data models. We focus on conceptual data model which is a relatively technology-independent specification of data structures [22] for two reasons: 1) Conceptual data model can be viewed as the translation of business requirements into a technical form of the structures of data; 2) Providing logical and physical data model is a matter of transforming the conceptual data model using established algorithms. Thus, the challenge is how to provide high quality conceptual data models. Pattern language is viewed as a structured way to organize the data model patterns which describes good design practices in a field of expertise, in this case: data modeling, and it can be used by designers to help progress from one problem to another in a logical way [19].

This research employs a fact oriented modeling (FOM) method called Fully Communication Oriented Information Modeling (FCO-IM) [4],[7] as the conceptual data modeling approach. While other modeling approaches: entity relationship modeling (ERM) and object-oriented modeling (OOM), focus to model things in a Universe of Discourse (UoD) and the relationships among them, FOM approach is aimed at modeling the structure of the communication about a universe of discourse. As a leading method in FOM, FCO-IM is claimed to hold FOM basic principles more consequently than other existing FOM methods [7]. It is expected that the use of FCO-IM will give more insights and provide a more powerful concept for the data model pattern. Some of our earlier works on data model pattern based on FCO-IM can be found in [5],[6].

The purpose of this paper is to describe our concept of pattern language of conceptual data model patterns based on FCO-IM as the modeling approach. This concept is expected to be the contribution of this paper especially in data modeling area. The concept also delivers a new approach in modeling the structures of
data which focuses on the quality of the resulting conceptual data models.

2. RELATED THEORIES AND WORKS

2.1. FCO-IM

FCO-IM is a conceptual data modeling approach which is pioneered by NIAM (Nijssen’s Information Analysis Method). FCO-IM is based on NIAM and can be considered as an extended NIAM, claiming to incorporate the basic principles of NIAM even more consequently other available FOM methods [7]. In FCO-IM, information analysis is carried out on fact expressions, i.e. sentences that express concrete facts within a Universe of Discourse (UoD). Examples of fact expression are as the following:

The name of product PAP192 is Johnson paper.
The name of product PEN202 is Goldstein pen.
The name of product DSK401 is Jerry’s disk.

The sentences are accompanied with some business rule statements which provide the rules on what kind of sentences that can be present or not. Examples of business rule statement which are related to the aforementioned fact expressions are:

Product is uniquely identified by product code.
Every product has a name.

The final product of data modeling using FCO-IM is called an Information Grammar (IG). An IG stores fact types (the fact expressions in type level) as well as constraints (the translation of business rule statements in the IG). Parts of a fact type are called roles. Roles of a fact type can be played by either an object type (a representation of real world object) or a label type (a representation of a group of values). To help user to understand an IG better, an information grammar diagram (IGD) is used.

From the aforementioned examples of fact expression and business rule statement, we can create the following IG:

![Diagram of an Information Grammar (IG)](image)

Product is uniquely identified by product code.
Every product has a name.

The IG consists of the following elements:

1. Name of Product, coded as F1, is a Fact Type Expression (FTE) or simply called fact type. F1 is a translation of the fact type expressions.

2. F1 has two roles (the parts between < >). The first role is played by object type Product (which is expressed using Object Type Expression O1). The second role is played by a Label Type Expression (LTE), or simply label type, product name. The object type Product is in fact a nominalized fact type which consists of one role played by label type product code.

3. UC1 and UC2 are constraints involved in the IG. Both are uniqueness constraints. UC1 and UC2 are the translation of business rule statements.

Given proper concrete examples of UoD, the IG can be used to regenerate fact expressions and business rule statements.

The IGD for the example is shown in Figure 1. Roles are presented as rectangles, while object type is shown by a circle surrounding some roles. Label types are shown as dash-lined circles. Uniqueness constraints are presented as two-way arrows over roles. Further description on FCO-IM can be found in [7].

![Figure 1. An example of an IGD](image)

2.2. PATTERN, PATTERN LANGUAGE, AND DATA MODEL PATTERN

In general, a pattern is a type of theme of recurring events or objects that repeat in a predictable manner [18]. Our definition of pattern is based on classical definition of pattern by Christopher Alexander [1],[2]. According to Alexander, each pattern is a three-part rule, which expresses a relation between a certain context, a problem, and a solution [1]. To fully describe a pattern, most pattern authors agree that there are other essential elements required to be described such as forces, rationale, resulting context, examples, etc. [2].

A system which allows its users to create an infinite variety of unique combinations of patterns, appropriate to different circumstances is called a pattern language. A pattern language works much like natural language in which both are finite combinatory systems which allow us to create an infinite variety of unique combinations, appropriate to different circumstances, at will [1]. According to Alexander, a useful pattern is “generative” which means that it works not only as a rule on how to create/generate something, but also, when combined together with other patterns, works to generate other “larger” patterns [1],[2]. Such “emergent behavior” of patterns is similar to the behavior of a natural language.

There are a number of publications with respect to data model patterns. David Hay wrote two books on data model patterns for enterprise information system
Pattern are explanation of how an IG intent is given in rationale. The consequences that Based on Alexander’s classical definition of pattern, we define: a conceptual data model pattern (π) is a relation (R) between problem (P), context (C), and solution (S); each defines a purpose

\[ R = (P, C, S, \pi) \] (1)

**Problem (P)** defines the intent of the pattern, i.e. the goals/objectives that it wants to reach within a context (C) and a system of forces which is required to be balanced by the solution (S). It consists of two elements:

1. **Forces**: things that should be at hand in order to achieve the intent, i.e. typical facts and other things that must be/often present when the problem arises.
2. **Constraints**: the rules to be considered in achieving the intent which restrict what kind of facts that can appear or not.

Meanwhile, **Context (C)** defines a set of statements on conditions in which problem (P) recurs and the solution (S) is desirable.

**Solution (S)** describes a configuration that balances the system of forces within a given context [1],[2]. In our definition of solution, we propose to employ a kind of template to produce FCO-IM IG. While an IG can be viewed as a template to produce user’s fact expressions as well as business rule statements (see again section 2.1), the proposed template is used to produce an IG. We call this template as Information Grammar for Pattern (IGp). This template also show how patterns can be generated from a pattern. See section 3.2 for further explanation on the IGp.

Other elements related to the solution part of a pattern are **rationale** and the **resulting context**. The explanation of how an IGp can be used to achieve the intent is given in rationale. The consequences that follow the application of the pattern is described in resulting context.

There are several extra elements that are used to define a conceptual data model pattern. **Example** and **known uses** are used to provide clearer description of the pattern by giving instances of the application of a conceptual data model pattern. While example explains in full detail of how a pattern is used in a typical case (it can be real or made-up case), known uses explain real world cases in which the problem is found. Every pattern must be given a **name** that embodies the knowledge described within the pattern.

We define the concept of pattern **category**. Some conceptual data model patterns with the same characterizations are grouped together into a category, for instance: patterns that are based on the structure of identification of an object (see section 3.4).

Data model patterns are related to each other in a certain rule (defined in IGp) which is combined together in the form of pattern language (see section 3.2. and 3.3). The relationships among patterns are described explicitly related patterns element. There may also be implicit relation between patterns. This can be explained in this element as well.

### 3.2. INFORMATION GRAMMAR FOR PATTERN (IGp)

Information Grammar for Pattern (IGp) is the central element of the solution of our conceptual data model pattern and therefore, its structure is required to be explored further. The IGp can be seen as an abstracted form of a group of IGs with particular characteristics.

As in IG, IGp is mainly presented in text format. New notations are introduced in addition to FCO-IM notations (see section 2.1 and [7] for full description on FCO-IM notations). An example of IGp is as the following:

```
(object)
[F1 : "There is a(an (object) <(object-id-1#1)|((G1#1)|<((object-id-2#2)|((G1#2)>))*."
G1 : '((object) |<(object-id-1#1)|((G1#1)|<((object-id-2#2)|((G1#2)>))*."
U1 : "(object) is uniquely identified by <(object-id-1#1)|((G1#1)|<((object-id-2#2)|((G1#2)>))*."
```

The notations shown in the above example are defined as the following:

1. **(x)** means x is required to be replaced by something when the IGp is used to generate an IG. Things that can be used to replace x are: a word/term related to a particular UoD; another pattern; or a category of pattern from which a pattern can be chosen to be generated.
2. **[x]** means x is optional; it can be generated 0 or 1 time.
3. \([x]^*\) means \(x\) is optional; it can be generated 0 or n times.
4. \(x\#y\) means \(y\) is a role (defined by a number or an alias of the role) on which \(x\) can be placed.
5. \(X|Y\) means \(X\) or \(Y\) can be generated, but not both (at the same time).

The elements of the above that must be changed by the user when the IG\(_P\) is used to generate an IG are the following:

1. object, object-id-1, and object-id-2 are to be replaced by a word/term related to a particular UoD. object must be replaced, while object-id-1 and object-id-2 depend on whether they are required or not.
2. \(F_1\) is optional. It means that the expression of fact type \(F_1\) can be present or not.
3. On role \#1, role \#2, and so on of fact type \(F_1\), either a term should be used to replace object-id-1, or object-id-2 or a pattern of category G1 must be generated.

An IG\(_P\) is equipped with a diagrammatic version of it. The concept of the diagram is also based on the concept of FCO-IM IGD. The diagram of IG\(_P\) is called the IG\(_P\) Diagram (IG\(_P\)D). We add the following symbols from the concept of IGD:

1. Dashed-lined boxes or lines to indicate that something can be generated or not.
2. Double-lined boxes to indicate that the pattern or term inside it must be generated.

The corresponding IG\(_P\)D for the above IG\(_P\) example is shown in Figure 2.

Figure 2. An example of IG\(_P\)D

The following IGs can be generated based on the above IG\(_P\):

1. IG of the existence of product in a UoD (see example in section 2.1).

   **Product**
   
   \(F_1:\) "There is a product <product code>".
   \(O_1:\) 'product <product code>'
   \(UC1:\) "Product is uniquely identified by product code."

   Several elements of the IG\(_P\) that are changed in this IG are:
   - object is replaced by the term Product.
   - object-id-1 is replaced by the term product code.

   IGD for the case is shown in Figure 3.

   **Figure 3. IGD for Product**

2. IG for a project of department

   **Project of Department**
   
   \(F_3:\) "There is a project of department <Department : O2> <Project : O3>"
   
   \(O_4:\) 'project of department <Department : O2> <Project : O3>''
   \(UC3:\) "Project of Department is uniquely identified by Department, Project."
   \(Department\)
   \(O_2:\) '<department name>'
   \(UC4:\) "Department is uniquely identified by department name."
   \(Project\)
   \(O_3:\) '<project code>'
   \(UC4:\) "Project is uniquely identified by project code."

   Several elements of the IG\(_P\) that are changed in this IG are:
   - object is replaced by the term Project of Department.
   - There are two roles defined for fact type \(F_3\). The first role is played by object type Department which is generated by applying a pattern of category G1 (the result is shown in the text). The second role is played by object type Project which is generated by applying a pattern of category G1 (the result is shown in the text).

   IGD for the case is shown in Figure 4.
Note that the coding of fact type expressions, object type expressions, and roles are subject to comply with the rules of FCO-IM, which should be unique in the entire model. Thus, the numbering can be changed according to the current condition of the modeling.

3.3. THE PATTERN LANGUAGE
An IG_P provides the rule of our pattern language. It shows how a pattern can relate to other patterns. As explained in section 3.2, an IG_P can contain the rules of when to generate a pattern.

We use formal grammar theory [14] to define our pattern language. A formal grammar G is formed by the following quad-tuple:

\[ G = (V, T, P, S) \]  

in which: V is a finite set of variables; T is a finite set of terminals; P is a finite set of productions; and S is the starting symbol in which \( S \in V \). A language \( L(G) \) is defined as the language that can be generated by grammar G.

Applying the theory to data model patterns, we have the following definitions of the elements of the grammar for the pattern language:

1. **Variables (V)**
   - The variables in our grammar consist of:
     - A starting symbol D.
     - All data model patterns and all pattern categories, denoted using the code of the pattern or the category.
     - \( a^+ \) with \( a \) is a term or a pattern or a pattern category which can replace a certain part of UoD. This symbol denotes the following productions: \( a^+ \rightarrow a | a a^+ \)

2. **Terminals (T)**
   - The terminals are the alphabet i.e. places in IG_P in which terms/words of particular UoD can be generated.

3. **Productions (P)**
   - The productions define the rules to generate a language over the variables and the terminals. The productions are created based on the rules defined in the IGP. Consider the example of IG_P in section 3.2. Suppose that it is the IG_P of pattern with code G1P1, the productions resulting from the IG_P are as the following:

\[
\begin{align*}
G1P1 & \rightarrow \text{object object-id-1 | object G1} \\
& \quad | \text{object object-id-1 object-id-2+ | object G1 object-id-2+ | object object-id-2 object-id-2+ | object-id-2+ \rightarrow object-id-2 | object-id-1 G1+ | G1+} \\
& \quad | G1 \rightarrow G1P1 | G1P2 | G1P3 | G1P4 | G1P5 | G1P6 | G1P7
\end{align*}
\]

4. **Starting Symbol (S)**
   - Starting symbol denotes a valid string that can be generated by the grammar. It is indicated with the variable D.

The pattern language not only shows the relationship among the conceptual data model patterns, but also provides a basis for an automation of the conceptual data model patterns in a CASE (Computer Aided Software Engineering) tool. The productions become the basis of parse tree [14] that can be developed for this language. The CASE tool is expected to enhance the use of the new methodology in data modeling activity.

3.4. LIST OF CONCEPTUAL DATA MODEL PATTERNS
We have developed several conceptual data model patterns, arranged into 4 different categories. Each category enfold a certain group of data modeling problems. Group G1 concerns problems on how to identify an object. Group G2 concerns problems on collection of objects. Group G3 concerns problems on the relations of two or more objects. Group G4 concerns problems based on the architecture of object. The idea of pattern G4P1 is based on our work in [15]. Table 3 provides the list of data model pattern in our pattern language. The list provides only the category, the code and name of the pattern. Some of the patterns is described in [6].

<table>
<thead>
<tr>
<th>Pattern Category</th>
<th>Pattern Code</th>
<th>Name of Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code: G1</td>
<td>G1P1</td>
<td>Single Identification Pattern</td>
</tr>
<tr>
<td>Patterns Based on the Identification of an Object</td>
<td></td>
<td></td>
</tr>
<tr>
<td>G1P2</td>
<td>Recursive Identification Pattern</td>
<td></td>
</tr>
<tr>
<td>G1P3</td>
<td>Set Identification Pattern</td>
<td></td>
</tr>
<tr>
<td>G1P4</td>
<td>Generalized Identification Pattern</td>
<td></td>
</tr>
<tr>
<td>G1P5</td>
<td>Synonymy Pattern</td>
<td></td>
</tr>
<tr>
<td>G1P6</td>
<td>Homonymy Pattern</td>
<td></td>
</tr>
<tr>
<td>G1P7</td>
<td>Subtype Pattern</td>
<td></td>
</tr>
<tr>
<td>Code: G2</td>
<td>G2P1</td>
<td>Graph Pattern</td>
</tr>
<tr>
<td>Patterns on Collection of Objects</td>
<td></td>
<td></td>
</tr>
<tr>
<td>G2P2</td>
<td>Sequence Pattern</td>
<td></td>
</tr>
<tr>
<td>G2P3</td>
<td>Parent-Child Pattern</td>
<td></td>
</tr>
<tr>
<td>Code: G3</td>
<td>G3P1</td>
<td>Attribute Pattern</td>
</tr>
<tr>
<td>Patterns Based on the Relation between Two Objects</td>
<td></td>
<td></td>
</tr>
<tr>
<td>G3P2</td>
<td>Mapping Pattern</td>
<td></td>
</tr>
<tr>
<td>G3P3</td>
<td>Assembly-Part Pattern</td>
<td></td>
</tr>
<tr>
<td>G3P4</td>
<td>Supertype-Subtype Pattern</td>
<td></td>
</tr>
<tr>
<td>Code: G4</td>
<td>G4P1</td>
<td>Viewpoints to A Dataset Pattern</td>
</tr>
</tbody>
</table>

3.5. EXAMPLE: ORGANIZATION CHART
In the following chapter, we provide the example of the use of the pattern language of conceptual data model patterns in a small case study on organization chart [7].

Consider the concrete example of an organization chart as shown in Figure 5. In the organization, there is
only one such chart (with the director, Mr. Hofman, as the top most employee).

![Organization Chart]

Figure 5. Organization Chart [7]

Sample of verbalizations of the facts in the case study are:

"Hofman DA supervises Velsen D."
"Velsen D supervises Parel K."

Some business rules statements related to the case are:

"Each employee has a unique name which consists of surname and initials."
"In a company there is only one organization hierarchy (with the director as the top most employee)."

The case shows a typical parent-child relationship in the hierarchy of the employees. Thus, Parent-Child Pattern (G2P3) is used. The identification of the employee shows typical single identification of an object. Thus, Single Identification Pattern (G1P1) is used.

The IGp of Single Identification Pattern is as described in section 3.2. The IGp of Parent-Child Pattern is as the following:

Hierarchy of Employee
F1 : "<(Employee: O1) is higher than <(Employee: O1)."  
Employee O1 : '<surname> <initials>'  
UC1 : "Hierarchy of Employee is uniquely identified by Employee#2."  
UC2 : "Employee is uniquely identified by surname, initials."  
TC1 : "Every Employee must be present either as a parent or as a child in Hierarchy of Employee."  
C1 : "To ensure that the value of a parent node is not the same the value of a child node: in a tuple, Employee#Parent cannot be the same value as Employee#Child."  
C2 : "To ensure that there is only one tree: there must be exactly one value of Employee#Parent which is not in Employee#Child."

The fact type F1 (Hierarchy of Employee), constraints UC1, TC1, C1, and C2 are generated from Parent-Child Pattern; while object type Employee and constraint UC2 are generated from Single Identification Pattern. Note that because of the use of the conceptual data model patterns, there are differences in the expression of fact type F1 in comparison to the original fact expressions, although both expressions are considered of the same meaning (semantic). At the present state of research, we do not consider this difference as a great deal of problem.

4. TESTS ON QUALITY OF CONCEPTUAL DATA MODELS

We have carried out some tests to check whether a conceptual data model (FCO-IM IG) created using the pattern language of conceptual data model patterns retains high quality. The quality of the conceptual data model is measured based on three aspects [16]:

1. Syntactic quality
Syntactic quality is related to the quality of conceptual data model with respect to the syntax of the model – in this case FCO-IM syntax – without considering the meaning of the model. The goal of syntactic quality is syntax correctness which is a measure on whether the model is free from syntax error or not. Syntax correctness is checked by comparing the resulting conceptual data model (IG) with the rules in FCO-IM (see [7]).

2. Semantic quality
Semantic quality is related to the quality of conceptual data model with respect to the UoD to be modeled. There are two goals of semantic quality: feasible validity and feasible completeness. Feasible validity is a measure on whether the statements in the model are relevant and correct in accordance to the statements in the UoD. Feasible completeness is a measure on whether the statements on whether relevant
Underlying statements in the UoD are present in the model. Feasibility aspect is introduced because it is argued that in large models, it is almost impossible to achieve total validity and completeness [16].

Feasible validity and feasible completeness are checked by comparing all regenerated fact expressions and business rules from the resulting IG with the initial verbalizations of facts and business rule statements.

3. Pragmatic quality

Pragmatic quality is related to the quality of the conceptual data model with respect to the audience comprehension on the model. The goal of pragmatic quality is feasible comprehension, which is a measure on whether the audience comprehends the model correctly. Again, the feasibility aspect is used since it is argued that it is almost impossible that all audience comprehend everything properly, especially in large models.

Feasible comprehension is checked by asking several respondents to compare the statements in the initial UoD and whether they can find corresponding statements in the conceptual data model.

We carried out the test on 14 cases provided in [7]. All cases are tested for syntax correctness, feasible validity, and feasible completeness, but only 5 cases are tested for feasible comprehension. Based on the test we conclude the following:

1. For syntax correctness, the average score is 100%. It means that so far, the resulting conceptual data models are free from errors.
2. For feasible validity, the average score is 100%. It means that so far, all statements in the resulting conceptual data models are correct and relevant to the UoD.
3. For feasible completeness, the average score is 88.04%. It means that in average only 88.04% of the statements in the UoD can be modeled in the conceptual data models. Based on examination on the test results, it appears that the statements that cannot be modeled are specific business rule statements. This is tolerable because domain experts can define infinite number of business rule statements which makes it impossible to adopt all of them in the conceptual data model patterns.
4. For feasible comprehension, the average score is 92.46%. It means that in average, the audience comprehended 92.46% of the model properly. Based on examination on the test results, all miscomprehension of the model happens on the business rule statements. The probable cause is the regenerated business rule statements are quiet different from the original statements of the business rules. This is not be helped because in many cases, the business rules cannot directly be translated into a model. Nevertheless, considering that all regenerated fact expressions are correctly comprehended by the respondents, it means that in general the models can be understood well.

All in all, the results of the tests are considered good, which means that the resulting conceptual data models are of high quality. There is a note, however, that the test cases used are small cases (1 to 11 fact types) which provides a controlled situation in which the modeling activity can be handled properly and easily.

5. CONCLUSIONS

In this paper, we present the concepts of a pattern language of conceptual data model patterns (including the concept of IGs), as well as a set of conceptual data model patterns. These results provide the foundation for a new approach in data modeling using FCO-IM as the modeling approach in which standards of best practices in modeling is put forward. The use of such standards is expected to provide more high quality conceptual data models which provide better supports to organization’s activities.

We have carried out some tests to prove whether the resulting conceptual data models which are produced using the pattern language hold high quality. The test is carried out on four measurements: syntax correctness, feasible validity, feasible completeness, and feasible comprehension. Based on the test results, we conclude that the conceptual data models are of high quality, particularly in syntax correctness and validity aspect. Remarks are given to the results of feasible completeness and feasible comprehension. Nevertheless, the problem in the score of feasible completeness is tolerable because only very specific business rule statements that cannot be modeled. The problem in the score of feasible comprehension can also be improved by providing a better way of expressing the business rules.

In the future, we will carry out more experiments on the pattern language of conceptual data model patterns to provide a more sound proof to the new methodology. A further research can be carried out to test whether the new methodology provide a more efficient and effective way of modeling in comparison to traditional way of modeling. The research can also be extended to provide translation of the conceptual data model patterns into logical and physical data models. Based on the concepts, we can also create a CASE tool which will help the use of the pattern language of conceptual data model patterns in data modeling activity.

REFERENCES


