Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

Data & Knowledge Engineering 47 (2003) 167–205

DATA & KNOWLEDGE ENGINEERING

www.elsevier.com/locate/datak

An analysis of structural validity in entity-relationship modeling

James Dullea [a], Il-Yeol Song [b,*], Ioanna Lamprou [b]

[a] *The Boeing Company, Boeing Phantom Works, Philadelphia, PA 19142, USA*
[b] *Drexel University College of Information Science and Technology, Philadelphia, PA 19104, USA*

Received 4 November 1999; received in revised form 8 February 2001; accepted 12 September 2002

Abstract

We explore the criteria that contribute to the structural validity of modeling structures within the entity-relationship (ER) diagram. Our approach examines cardinality constraints in conjunction with the degree of the relationship to address constraint consistency, state compliance, and role uniqueness issues to derive a complete and comprehensive set of decision rules. Unlike typical other analyses that use only maximum cardinality constraints, we have used both maximum and minimum cardinality constraints in defining the properties and their structural validity criteria yielding a complete analysis of the structural validity of recursive, binary, and ternary relationship types. Our study evaluates these relationships as part of the overall diagram and our rules address these relationships as they coexist in a path structure within the model. The contribution of this paper is to provide a comprehensive set of decision rules to determine the structural validity of any ERD containing recursive, binary, and ternary relationships. These decision rules can be readily applied to real world data models regardless of their complexity. The rules can easily be incorporated into the database modeling and designing process, or extended into case tool implementations.
© 2003 Elsevier B.V. All rights reserved.

Keywords: Entity-relationship modeling; Structural validity; Recursive relationships; Binary relationships; Ternary relationships; Cardinality constraints

* Corresponding author. Tel.: +1-2158952489; fax: +1-2158952494.
E-mail addresses: james.dullea@boeing.com (J. Dullea), song@drexel.edu (I.-Y. Song), il27@drexel.edu (I. Lamprou).

0169-023X/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/S0169-023X(03)00049-1

1. Introduction

Entity-relationship (ER) modeling [5] is the foundation of various analysis and design methodologies for the development of information systems and relational databases [29]. A key measure of success in the design of these models is the level that they accurately reflect the real world environment. A model can be a very complex abstract structure, and designers are highly prone to making small mistakes that incorporate inconsistencies into the structure. There is various supporting empirical evidence [2,6,11,12] that concludes errors, mistakes, or inconsistencies made in the early stages of the software development life cycle are very expensive to correct. Boehm [3] states that the cost difference to correct an error in the early phases as opposed to the post-implementation phase is on the order of a ratio of 1:100. Left undetected these inconsistencies become very costly to correct, so early discovery of an error is highly desirable.

The structural validity of an ER diagram (ERD) is concerned whether or not a given ERD contains any constructs that are contradictory to each other. An ERD with at least one inconsistent cardinality constraint is structurally invalid [22]. Direct treatment of structural validity as a concept is rare in the literature. Some structural validity rules are partly discussed in other publications, such as [1,10,24,27,30], but they do not exhaustively cover all the combinations of maximum and minimum cardinality. Indirect treatment of the subject materials, are usually confined to making buttress points concerning other issues. Structural validity in the literature is almost always treated as a property of a study; in this paper we treat it as the object of our study. In our study we evaluate each relationship type as part of the overall diagram and address how they coexist with other relationship types within the model.

Deciding whether a particular ERD is valid or not is sometimes a difficult issue to many database designers. For example, how do we know whether the diagrams shown in Fig. 1 are valid? Fig. 1 shows three simple ERD containing recursive, binary and ternary relationship types. These diagrams seem to be a plausible representation of a set of semantics, but they are actually structurally invalid from many perspectives. The work presented in our paper will show why these diagrams are invalid and allows for the evaluation of more complex diagrams than presented here.

This paper is organized into six sections. In Section 2, the concepts of semantic and structural validity, both necessary for a valid design, are defined and discussed. In Section 3, we focus on recursive relationships. In the literature, only two types of recursive relationships—symmetric and asymmetric [8,30] and their validity [8] are discussed. In this section, we discuss five different types of recursive relationships by further defining the concept of an asymmetric association into hierarchical, circular, and mirrored relationships. We also introduce the concept of a hybrid relationship by combining the concepts of a circular and hierarchical association. By expanding the work of [9], we present a complete taxonomy of recursive relationships and decision rules for checking the validity of those various recursive relationships. Section 3 presents five rules and three corollaries to evaluate the structural validity of standalone recursive relationship types. Section 4 encapsulates our work on the structural validity of binary relationships. We show that standalone binary relationships and binary relationships existing in an acyclic path are always structurally valid. In this section, we examine binary relationships, as they exist in an ERD with other binary relationship types. Our holistic view presents six rules and two corollaries to evaluate ERDs containing binary relationships. These rules also play an important role in the analysis of ternary relationships. Section 5 explores the effects of ternary relationship structures on the va-

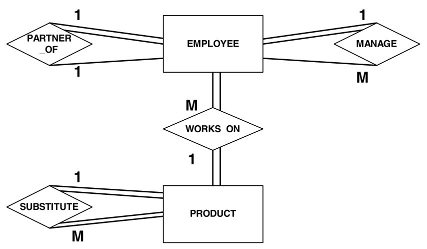


Figure 1a (Unary Example)

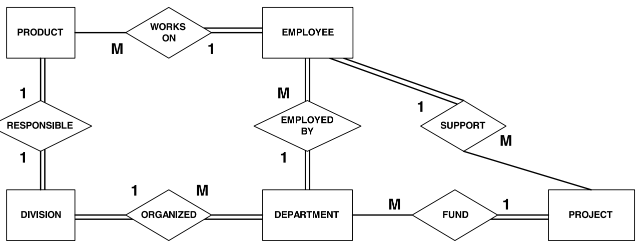


Figure 1b (Binary Example)

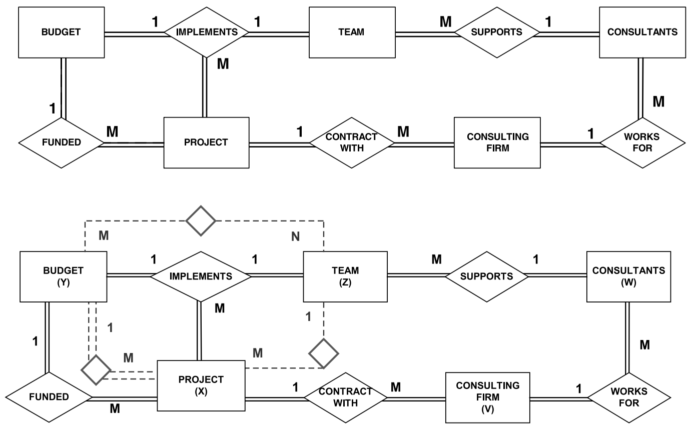


Figure 1c (Ternary Example)

Fig. 1. Three plausible but structurally invalid ERD: (a) an ERD with invalid recursive relationships; (b) an ERD with invalid binary relationships and (c) an example of an invalid multi-path entity relationship diagram with a constrained ternary relationship showing the effects of the embedded relationships.

lidity of the diagram. We extensively explore the embedded binary relationships derived from the ternary relationship to perform our path analysis. Ternary relationships are also examined for standalone validity and the redundancy [7] of embedded binary relationships in conjunction with imposed constraining binary relationships. Both single and multiple constraining relationships are explored. In addition, we consider the effects of minimum cardinality on constraining binary relationships. Section 3.2 states the conclusions of our research, and Appendices summarize the complete set of rules to test the structural validity of recursive, binary, and ternary relationships in any ERD.

1.1. Related works

In this section, we review literatures related to the use of cardinality constraints and validity in ER modeling.

The original entity-relationship model proposed by Chen [5] only used the maximum cardinality constraints. The maximum cardinality constraint describes the maximum number of data instances that may occur for that entity to participate in the relationship. Generally four choices are available to a binary relationship: ‘one-to-one’, ‘one-to-many’, ‘many-to-one’, or ‘many-to-many’. Scheuermann et al. [23] introduced the idea of the participation constraint to the model adding yet another concept that is important to our research and to the concept of structural validity. The Scheuermann paper defined the minimum cardinality constraint on the data model as the minimum number of data instances that must occur for that entity to participate in the relationship. Two choices are available: zero and one, and they correspond to the terms ‘optional participation’ and ‘mandatory participation’. Maximum and minimum cardinality constraints are the keystone concepts to the analysis of structural validity, because taken together they represent the structural constraints on the model [10].

Song and Jones in [18,25] discuss implicit binary relationships embedded in various ternary relationships. They show that the implicit cardinality of any binary relationship in any ternary relationship is $M : N$. They further present a structural validity rule called Explicit Binary Permission rule. The rule states a binary relationship cannot be imposed where the binary cardinality is less than the cardinality specified by the ternary. Jones and Song [19] discuss the decomposition of ternary relationships into binary relationships and show that not every lossless decomposition is always functional-dependency-preserving. Both Jones [18,19] and McAllister [22] address the structural consistency issues related to ternary relationships. They specifically look at maximum cardinality constraint issues in ternary relationships and examine the effect of related and unrelated binary relationship constraints on the ternary relationship. Their analysis of the structural validity of the complex ternary relationship is solely based on the maximum cardinality constraint of both the ternary relationship and the related binary relationships. One of the contributions of our paper is that we analyze the structural validity by analyzing both maximum and minimum cardinality constraints and ternary relationships along with any constraining binary relationships, as they exist holistically within the ERD.

Various mechanisms were introduced to visually represent the information requirements. The diagramming mechanisms evolved into many different forms that satisfied the need to communicate with either the user community or the implementation community, while allowing a great deal of latitude in between for other mechanisms to be introduced. Ferg [12] performs an analysis on three notations: information engineering (IE), Merise, and Chen notation. Each one illustrates different views on showing the cardinality constraints that exist on the relationships. Liddle [21] expands the previous analysis by examining and describing additional models such as the semantic binary data model, the structural model, the semantic association model, the Nijssen’s information analysis methodology, and the binary relationship model. Song [26] further compares and contrasts various ER diagramming notations. In our paper we use the Elmasri and Navathe’s notation [10] to describe the diagrams we are presenting. The main reason for using the Elmasri notation as opposed to a newer notation such as UML, is that the ER diagramming is still a widely acceptable mechanism in representing information and modeling data. The Elmasri notation also uses the ‘look here, look across’ notation for expressing minimum and maximum cardinalities, respectively. Since ternary relationships are discussed in depth in this paper, we are required to use the ‘look here’ notation for the participation constraint. A ‘look across’ notation such as used in the UML does not effectively represent the semantics of participation constraints imposed on relationships where the degree is higher than binary.

During the late 1980s and into the 1990s, numerous papers were written that enriched the concept of cardinality constraints. Habrias [13] introduces the concept of static and dynamic cardinality constraints. They believe that two views exist, first being the view of the present state and what happens in the static environment, and the second view referring to what happens in a changing or dynamic environment. He models both the static and dynamic to allow analysis of the processes that occur within a relation between two chronological states. Thalheim [28] discusses the ambiguity of properties of *n*-ary relationship models and analyzes the theoretical basis of its cardinality constraints. He presents ways of constructing and manipulating them in order to use them in practical database tasks. For that purpose he formalizes concepts and introduces a way of representing cardinality constraints for *n*-ary relationships in a diagram. The formalization of these concepts aids in "detecting" inconsistencies in the cardinality constraints, leading to the improvement of maintenance processes and the enhancement of design decisions. Zhou [32] introduces the concept of valences in conjunction with cardinality constraints. The concept of valence constraints adds richness to the model's semantics. The purpose of this is to discover evolving relationships as the model matures or becomes more complex. The valence constraints impose restrictions making the boundaries of the maximum cardinality constraints more specific. Valence bundles describe the instance sets of a relationship and the various states that the model can represent. Hartmann [14] introduces the concept of global cardinalities that supports lower and upper bounds on cardinalities adding richness to the constraints imposed on relationships. In later work, Hartmann [15] extends the concept of global cardinalities with *int-cardinalities* to allow gaps or non-contiguous enumerated lists in the sets of permitted cardinalities. He [16] further presents an efficient method for checking consistency and logical implications in cardinality constraints. Applying his methods permits the designer to discover interesting properties of schemata as well as conflicts among the particular constraints. Dullea and Song [9] present the classification of recursive relationships as well as the criteria influencing the structural validity of recursive relationships in entity-relationship data models. Finally, some notable works address the integration of object oriented concepts and data modeling. Ye [31] introduces a way of managing cardinalities in object-oriented systems through an object-oriented language. Calvanese [4] analyzes the concept of inheritance in relation to cardinality constraints in an object-oriented environment.

In concluding the related works section, we would like to note that, even though much research was performed on the use of cardinality constraints, the research on the structural validity using both minimum and maximum cardinality has not been performed. We believe that our paper makes a significant contribution on the correct use of ERDs.

2. Validity

An ER model is composed of entities, the relationships between entities, and constraints on those relationships. Entities may be chained together in a series of alternating entities and relationships, or may participate singularly with one or more relationships. The connectivity of entities and relationships is called a *path*. Paths are the building blocks of our study in structural validity analysis. Paths visually define the semantic and structural association that each entity has simultaneously with all other entities or with itself within the path. The terms, structural and semantic validity, are defined as follows.

Definition. An entity-relationship diagram is structurally valid only when simultaneous consideration of all structural constraints imposed on the model does not imply a logical inconsistency in any of the possible states. An entity-relationship diagram is semantically valid only when each and every relationship exactly represents the modeler's concept of the problem domain. An entity-relationship diagram is valid when it is both semantically and structurally valid.

In data modeling, validity can be classified into two types: semantic and structural validity. A semantically valid ERD shows the correct representation of the application domain being modeled. The diagram must communicate exactly the intended concept of the environment as viewed by the modeler. Since the semantic validity is application-dependent, we cannot define generalized validity criteria. Therefore, we do not consider semantic validity in this paper.

The structural validity of an ERD is concerned whether or not a given ERD contains any constructs that are contradictory to each other. An ERD with at least one inconsistent cardinality constraint is structurally invalid [15]. An ERD represents the application semantics in terms of maximum and minimum cardinality constraints. The driving force behind cardinality constraint placement is the semantics of the model.

The values and placement of these constraints must be robust enough to convey the business rules exactly intended by the modeler while being consistent with respect to the whole model in order to reflect the real world environment [13,16,28]. The model is not just a set of individually constrained relationships pieced together between sets of entities, but a holistic view of the representation domain. Each set of cardinality constraints on a single relationship must be consistent with all the remaining constraints in the model and over all possible states [13,21]. More restricted methods of cardinality constraints have been proposed, that allow upper and lower bounds, or an algorithm to define the constraint [13–16,31,32]. These constraints are more restrictive than the relaxed method of using 'one' and 'many' as their boundary. Although the restricted constraints are richer in the semantic meaning of the model, they are considered subsets of the relaxed cardinality constraints. Hartmann [15] states that the "set of restricted cardinality (int-cardinality) constraints is consistent whenever the corresponding relaxed set of ordinary cardinality constraints is consistent". For the purpose of our own analysis, we address the relaxed set of cardinality constraints because it includes the more restrictive subset.

A recursive relationship, that is, the association between role groups within a single entity, is evaluated as a standalone conceptual object. It is semantically invalid when the concept does not reflect the business rules as defined by the user community. A recursive relationship is structurally invalid when the cardinality and participation constraints do not support the existence of data instances as required by the user and renders the entire diagram as invalid. We will show that standalone binary relationships are always valid and are only evaluated holistically with respect to all other relationships in the diagram. Ternary relationships are evaluated for structural validity both as a standalone conceptual object and holistically as they behave with other relationships. Our study shows that binary constraining relationships play an important role in the standalone evaluation of ternary relationships while the embedded binary relationships holistically determine the structural validity of the entire diagram.

In general, a structurally invalid diagram reflects semantically inconsistent business rules. In order for a model to be valid all the paths in the model must also be valid. Our analysis will investigate those types of structural paths in an ERD that are critical to the validity of the entire diagram. Our study focuses on both the standalone structural validity of recursive and ternary relationships and the holistic structural validity of recursive, binary, and ternary relationships represented in a constraint-compliant ERD.

3. Recursive relationships

Section 3.1 presents our taxonomy of recursive relationships and Section 3.2 presents the complete decision rules for checking the validity of recursive relationships.

3.1. The taxonomy of recursive relationships

A *recursive relationship* is defined as an association between instances as they take on *roles* within the same entity. Roles play an important part in the examination of structural validity, especially for recursive relationships [12,32]. A *role* is the action or function that the instances of an entity play in a relationship [4]. In a recursive relationship, a set of instances may take on a single role or multiple roles within the same relationship. Examining these roles allows us to classify all recursive relationships into symmetric or asymmetric associations while further classifying asymmetric relationship types into hierarchical, circular, and mirrored associations. The complete classification of recursive relationships we consider in this paper is shown in Fig. 2 as follows.

A recursive relationship is *symmetric* or *reflexive* when all the instances participating in the relationship take on a single role and the semantic meaning of the relationship is exactly the same for all the instances participating in the relationship independent of the direction in which it is viewed. These relationship types are called bi-directional. For example, if instance I_1 is associated with another instance I_2 in the same entity through relationship R and the I_2 is associated with I_1 using exactly the same semantics of relationship R then the relationship is symmetric. Relationships such as "partner of", "spouse of", and "sibling of" are examples of symmetric relationships.

A recursive relationship is *asymmetric* or *non-reflexive* when there is an association between two *different* role groups within the same entity and the semantic meaning of the relationship is different depending on the direction in which the associations between the role groups are viewed. These relationship types are called unidirectional. For example, the instances of an entity EMPLOYEE associated through a relationship called SUPERVISES contain two roles groups. The role groups are "supervisor employees" and "supervised employees". In one direction the relationship is viewed as 'supervisor employees *supervise* supervised employees' while in the other direction the relationship is viewed as 'supervised employees are *supervised by* supervisor employees'.

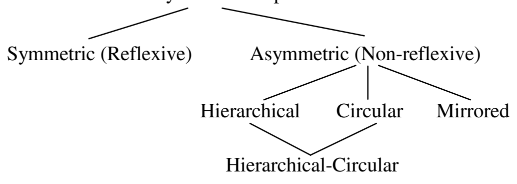


Fig. 2. A taxonomy of recursive relationships.

Previous studies [8,30] of what here is described as a symmetric or reflexive relationship implicitly suggested that the only other relationship type to be investigated would be an asymmetrical relationship. In this study we further classify the concept of an asymmetric relationship type into three *non-reflexive* recursive relationship types: hierarchical, circular, and mirrored relationships.

A recursive relationship is *hierarchical* when a group of instances within the same entity are ranked in grades, orders, or classes, one above another. It implies a beginning (or top) and an end (or bottom) to the ranking scheme of instances. An example of a hierarchical recursive relationship is the entity EMPLOYEE and the relationship SUPERVISES. In this relationship some employees supervise other employees. There is usually an employee at the top who is not supervised by any other employee and employees at the bottom who do not supervise any employees.

A recursive relationship is *circular* when an asymmetrical recursive relationship has at least one instance that does not comply with the ranking hierarchy. The relationship is unidirectional in that it can be viewed from two directions with different semantic meaning. An example of this occurrence would be an entity representing a continuous-supporting help desk environment. Envision the enforcement of a business rule where the help desk phone system rotors to the next person available depending on the position of their assigned desk location. For example, consider N individuals supporting the phones. As a call is received, the phone system assigns the responsibility of the call to the next available person in numeric order. If person N is unavailable, the phone system assigns the call to person 1 or the next available person after person 1. Thus person 1 is backed up by person 2, person 2 is backed up by person 3, and so on, while person N is backed up by person 1. The relationship in this example is completely circular because no hierarchy was established between help desk personnel. We can also develop a hybrid by introducing a hierarchy to the help desk instances. If the business rules were changed to allow only senior personnel to backup many other junior help desk individuals, then we would have *a hierarchical–circular* relationship. This type of relationship is common among decision-makers where a management employee of a lower rank assumes the responsibilities of a more senior manager during their absence.

Another question that arises in the modeling of recursive relationships is whether an instance can be associated with itself. This event is impossible in relationships above degree one but could happen in special cases of a recursive relationship and we call this special event a *mirrored relationship*. A mirrored relationship exists when the semantics of a relationship allow an instance of an entity to associate with itself through the relationship. For example certain individual contributors in an organization could be self-managed while other individuals report up the management chain. Two relationships would be required to model the supervisory concept. Fig. 3 shows the diagram that addresses this issue.

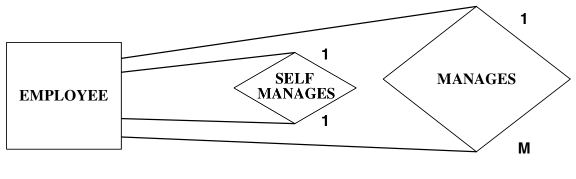


Fig. 3. An entity with two asymmetrical recursive relationships addressing self-management issues.

Table 1
Valid recursive relationship types as they relate to the cardinality constraints

Type of relationship		Direction of relationship	Participation constraints	Cardinality constraints	Example	
					Relationship	Role(s)
Symmetrical (reflexive)		Bi-directional	Optional–optional Mandatory–mandatory	1–1 M–N	Spouse of	Person
Asymmetrical (non-reflexive)	Hierarchial	Uni-directional	Optional–optional	1–M 1–1	Supervises Is supervised by	Manager–employee
			Optional–optional Optional–mandatory Mandatory–mandatory	M–N	Supervises Is supervised by	Manager Employee
	Circular	Uni-directional	Optional–optional Mandatory–mandatory	1–1	Backs up Is backed up by	Help desk
	Hierarchial circular	Uni-directional	Optional–optional Optional–mandatory	1–M	Backs up Is backed up by	Decision-maker
			Optional–optional Optional–mandatory Mandatory–mandatory	M–N	Supervises Is supervised by	Manager–employee
	Mirrored	Uni-directional	Optional–optional	1–1	Self-manages Manages self	CEO

Table 1 summarizes each recursive relationship by its directional properties, the combination of minimum and maximum cardinality constraints, and examples. In our diagrams throughout this paper, we use 'one (1) and many (M)' notation for maximum cardinality, and a single line to indicate optional participation and a double line to show mandatory participation [10,12,26]. The words "mandatory" and "optional" are used in our tables to indicate mandatory (or total) and optional (or partial) minimum cardinality, respectively. Also, the notation $|E|$ represents the number of instances in entity E.

3.2. Decision rules for validity checking of recursive relationships

In this section, we analyze recursive relationships from the point of view of cardinality and participation constraints. Table 1 pairs each valid set of constraints with the different recursive relationship types. For example, symmetrical relationships are supported by 'one-to-one' or 'many-to-many' cardinality constraints coupled with either 'optional–optional' or 'mandatory–mandatory' participation constraints. Only these constraint combinations are valid for a symmetrical relationship. These constraints, along with others, also are valid in hierarchical and hierarchical–circular relationships, as shown in the table. We review all combinations of constraints and identify the valid recursive relationship types discussed in Section 3.1, and shown in

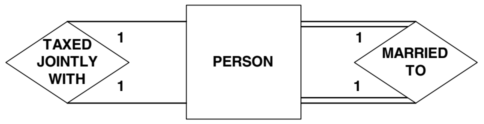


Fig. 4. Valid recursive relationships.

Table 1, through examples and discussion. Those minimum and maximum cardinality constraints that yield invalid structures are demonstrated through formal proofs.

3.2.1. One-to-one recursive relationships

There are three cases that require investigation in one-to-one recursive relationships. They are 'mandatory–mandatory', 'optional–optional', and 'optional–mandatory'. The 'mandatory–optional' case is a reverse image of the 'optional–mandatory' case and does not require further analysis.

3.2.1.1. 1:1 Mandatory–mandatory. This type of relationship is valid for a symmetrical relationship. In a 'mandatory–mandatory' relationship each instance of the role group must participate fully in the relationship. When the relationship is 1:1, each instance must be paired with one and only one other instance. An example of this symmetrical pairing of instances is shown in Fig. 4 representing of a group of married persons with the relationship MARRIED_TO. Every person in the group is married to one and only one other person in the group.

3.2.1.2. 1:1 Optional–optional. In an 'optional–optional' relationship each instance of the role groups can optionally participate in the relationship. This case makes no restrictions on the number of roles that is contained in the entity, so both symmetrical and asymmetrical relationships are supported. In a 'one-to-one' symmetrical relationship, instances of the role group are paired with one and only one other instance in the same role group. If the mandatory constraint is removed from the relationship then additional non-participating instances can be included in the entity without affecting the validity. In Fig. 4, married individual, have the option of being taxed either jointly or separately. This is an example of a symmetric relationship where the entity is PERSON and the relationship is TAXED_JOINTLY_WITH can take on a minimum cardinality constraint of 'mandatory–mandatory' or 'optional–optional' when the maximum cardinality is 'one-to-one'. For the 'mandatory–mandatory' case all of the instances in PERSON must file their taxes jointly with their spouse while in the 'optional–optional' case, shown in Fig. 4, at least one pair of married persons may decide to file their taxes separately [28].

3.2.1.3. 1:1 Mandatory–optional or optional–mandatory. In a 'mandatory–optional' relationship each instance of one role group must participate in the relationship while the instances of the other role group can optionally participate in the relationship. This implies the existence of two different role groups in the entity and therefore the relationship must be asymmetric. Symmetric relationships of any type cannot support 'mandatory–optional' relationships because an instance participating in a symmetric relationship is mapped to another instance in the same role group. By the definition of a symmetrical relationship the instance being 'mapped to' must also have the reflexive property of being able to map back to its paired instance. This would be impossible if the participation is optional. Asymmetrical relationships also have a similar instance-mapping problem, as we will demonstrate.

Theorem 1. *A* 1:1 *recursive relationship structure with a minimum cardinality constraint of 'mandatory–optional' is an* ***invalid*** *structure.*

Proof. Suppose role group 1 ($rg1$) contains I_j instances (where $j = 1, 2, 3, \dots, n$) and represents all the instances in entity E. That is,

$$|rg1| = |E| = n \tag{1}$$

Because of the mandatory participation, each instance in $rg1$ is mapped totally to role group 2 ($rg2$) and $|rg2|$ is at least equal to n. Therefore,

$$|rg2| \geqslant n \tag{2}$$

If the instances of $rg2$ are only partially mapped to the instances of $rg1$ then the instances in $rg2$ may contain at least one additional instance that is not mapped to instances of $rg1$. Therefore, in the strictest form

$$|rg2| > n \tag{3}$$

$rg2$ also represents all the instances in entity E, therefore from (3) we can derive

$$|E| > n \tag{4}$$

Eqs. (1) and (4) are inconsistent because $|E|$ cannot be equal to n and greater than n. Therefore the structure is invalid. □

The proof of Theorem 1 demonstrates that a 1:1 recursive relationship with a minimum cardinality constraint of 'mandatory–optional' or 'optional–mandatory' is invalid for both symmetrical and asymmetrical relationships.

Rule 1. *Only* 1:1 *recursive relationships with mandatory–mandatory or optional–optional minimum cardinality constraints are structurally valid.*

Corollary 1. *All* 1:1 *recursive relationships with mandatory–optional or optional–mandatory minimum cardinality constraints are structurally* ***invalid***.

3.2.2. One-to-many recursive relationships

There are four cases requiring investigation in one-to-many recursive relationships. The minimum cardinality constraints are mandatory–mandatory, mandatory–optional, optional–mandatory, and optional–optional. 'One-to-many' recursive relationships are always asymmetrical relationships because the non-reflexive relationship requires two role groups.

3.2.2.1. One-to-many mandatory–mandatory. In a 'one-to-many' recursive relationship the 'mandatory–mandatory' case is invalid as stated in the following rule and demonstrated in the following proof.

Theorem 2. *A 'one-to-many' recursive relationship where both minimum cardinality constraints are mandatory is an invalid structure.*

Proof. Given that role groups 1 and 2 are both contained in entity E, they participate fully in the relationship R. Because of the $1 : M$ constraint placed on R, the relationship R must support the case of $rg1$ containing fewer instances than $rg2$, therefore, in the strictest form

$$|rg1| < |rg2| \tag{1}$$

Because of the mandatory constraint on R the instances of $rg1$ represent all the instances in E, therefore,

$$|rg1| = |E| \tag{2}$$

Also because of the mandatory constraint on R the instances of $rg2$ represent all the instances in E, therefore,

$$|rg2| = |E| \tag{3}$$

$$\text{If } |rg1| = |E| \quad \text{and} \quad |rg2| = |E|$$

$$\text{Then } |rg1| = |rg2| \tag{4}$$

Eqs. (1) and (4) are inconsistent, therefore the structural representation is invalid. □

3.2.2.2. One-to-many mandatory–optional. Also in the 'one-to-many' recursive relationship, the 'mandatory–optional' case is invalid as stated in the following rule and demonstrated in the following proof.

Theorem 3. *Any 'one-to-many' recursive relationship where the minimum cardinality constraint is mandatory on the 'one' side and optional on the 'many' side is an invalid structure.*

Proof. Consider role groups 1 and 2 that are contained in entity E and that participate in a $1 : M$ relationship R. If $rg1$ on the 'one' side of a 'one-to-many' relationship is mapped with optional participation to $rg2$ on the 'many' side, then the relationship R must support the case of $rg1$ containing fewer instances then $rg2$. Therefore, in the strictest form

$$|rg1| < |rg2| \tag{1}$$

Because of the mandatory cardinality constraint on the role group 1 the number of instances in $rg1$ must equal the number of instances in entity E. Therefore,

$$|rg1| = |E| \tag{2}$$

Combining Eqs. (1) and (2) implies that $|E| < |rg2|$, but this is inconsistent because the number of instances in a role group can not exceed the total number of instances in the entity containing that role group. □

3.2.2.3. One-to-many optional–mandatory. A hierarchical–circular recursive relationship type supports the occurrence of the 'optional–mandatory' case in a 'one-to-many' recursive relationship. An example would be where there are two roles being played by the instances of the entity.

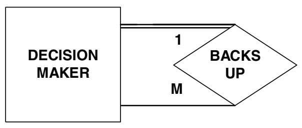


Fig. 5. An example of a one-to-many optional–mandatory recursive relationship.

First, every instance participates as an employee and employees are expected to be able to make decisions about the area of their expertise. Some employees have more than one area of expertise and are required to backup other employees during their absence. The relationship is hierarchical because all employees participate in the relationships of being backed-up at the bottom of hierarchy but only some employees at the top of the hierarchy backup all other employees. The relationship is circular because someone in the tree structure circles back to backup the instance at the top. Fig. 5 is an example of this type of relationship. This relationship is similar to the help desk example but allowing an instance to backup more than one other instance.

3.2.2.4. One-to-many optional–optional. Howe [17, p. 137] implies that "the most senior employee is regarded as self-supervised" and allows the 'many' side to be mandatory. Using this method only implies that an employee can be self-managed and may lead to a different interpretation. We feel the concept of role uniqueness should be explicitly stated in the diagram and the more appropriate way to model the example would be with two relationships, both being 'optional–optional'. An example of a recursive relationship specifically addressing the self-management issue is shown in Fig. 6.

Fig. 6 demonstrates the validity of the 'optional–optional' case of a $1:M$ relationship. It is optional for the manager role group because only some instances are managers while others are not. It is optional for the employee role group because as previously stated at least one manager at the higher level is an employee that is not managed by other instances in the entity. From Theorems 2 and 3, proofs, and the accompanying discussion the following rules and corollaries can be stated about $1:M$ relationships.

Rule 2. *For $1:M$ or $M:1$ recursive relationships optional–optional minimum cardinality are structurally valid.*

Rule 3. *For $1:M$ recursive relationships of the hierarchical–circular type, optional–mandatory minimum cardinality are structurally valid.*

Corollary 2. *All $1:M$ or $M:1$ recursive relationships with mandatory–mandatory minimum cardinality constraints are structurally* ***invalid***.

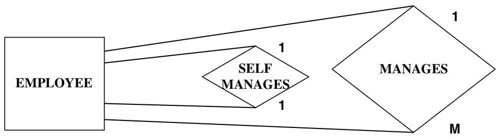


Fig. 6. An entity with two asymmetrical recursive relationships addressing the self-management issue.

Corollary 3. *All* 1 : *M or M* : 1 *recursive relationships with mandatory participation constraint on the 'one' side and an optional participation constraint on the 'many' constraints are structurally* ***invalid***.

3.2.3. Many-to-many recursive relationships

There are three cases requiring investigation in many-to-many recursive relationships. The minimum cardinality constraints are mandatory–mandatory, mandatory–optional, and optional–optional.

3.2.3.1. Many-to-many mandatory–mandatory. A 'mandatory–mandatory' many-to-many relationship is valid for a symmetrical relationship while as stated before the asymmetrical recursive relationship cannot be totally mandatory. In a 'mandatory–mandatory' relationship each instance of a role group must participate fully in the relationship. When the relationship is a 'many-to-many' relationship then each instance must be paired with one or more instances in the role group. Because they are symmetric, each pairing is reflexive, meeting the requirement of a 'mandatory–mandatory' relationship. An easy-to-understand valid example would be an entity of PERSON composed of groups of brothers and sisters with the SIBLING_OF as the relationship. We could extend this concept to many business examples, such as groups of employees teaming with each other to work on projects. Employees would work only on one project with many employees represented through the recursive relationship TEAMED_WITH and the binary relationship WORKS_ON connected to the entity PROJECT.

3.2.3.2. Many-to-many mandatory–optional. In the example of a hierarchical tree for a 'many-to-one' relationship using the entity EMPLOYEE with different levels of management, we showed that the minimum cardinality constraint is 'optional' on both sides of the relationship. This occurred because some employees were at the bottom of the management chain and did not manage anyone, and one employee, the highest manager, was at the top of the chain not managed by anyone. If we change this example to a company that is completely employee-owned and some of these employees form the board of directors that manage the senior manager, then this is an example of a 'many-to-many', mandatory–optional recursive relationship. In this modified example, some employees (but not all) take on the role of manager satisfying the optional side of the relationship. The mandatory side of the relationship is satisfied because all employees are managed even the most senior manager and the owner-employees. This is an example of a 'many-to-many' relationship because the senior manager is managed by more than one owner–employee and manages more than one employee in the hierarchy. This type of relationship with these constraints is valid.

3.2.3.3. Many-to-many optional–optional. Our earlier example of an entity PERSON composed of groups of only brothers and sisters with the SIBLING_OF as the relationship is a valid 'mandatory–mandatory' 'many-to-many' recursive relationship and can serve as a model for the totally optional case. If we add individuals to the entity PERSON that have no brothers or sisters then these individuals can not participate in the relationship SIBLING_OF. Of more importance is that they do not participate on both sides of the relationship because they have no brothers or sisters, and no other instance has them as brother or sister. The relationship is optional on both sides. This is an example of a valid symmetric relationship.

Rule 4. *All recursive relationships with many-to-many maximum cardinality are structurally valid regardless of minimum cardinality constraints.*

Rule 5. *All recursive relationships with optional–optional minimum cardinality are structurally valid.*

3.3. Analysis of Fig. 1a

From Section 3.2, we have developed five decision rules for determining the validity of a recursive relationship and three corollaries that quickly identify invalid recursive relationships. Appendix A summarizes each validity rule and corollary for recursive relationships with an example. The relationship types that are addressed by each rule are also included.

In Fig. 1a of this paper the ERD contained three recursive relationships: PARTNER_OF, MANAGE, and SUBSTITUTE. PARTNER_OF is invalid because it violates Corollary 1. It was shown in this paper by formal proof presented for Theorem 1 that a 'one-to-one' recursive relationship with a 'mandatory–optional' participation constraint is invalid. SUBSTITUTE is invalid because it violates Corollary 2. The proof of Theorem 2 states that a 'one-to-many' recursive relationship with a 'mandatory–mandatory' participation constraint is invalid. MANAGE is invalid because it violates Corollary 3. The proof of Theorem 3 states that a 'one-to-many' recursive relationship with a 'mandatory–optional' participation constraint is invalid. ERD containing these types of recursive relationships are invalid.

4. Binary relationships

A binary relationship is an association between the role group of one entity with the role group of another entity. It is a relationship of degree two and involves only two entities associated by one relationship [15]. With respect to the relative number of instances in the two entities allowed by a relationship, certain simple binary structures can support only one of three possible solutions: $|E_1| = |E_2|$, $|E_1| < |E_2|$, or $|E_1| > |E_2|$ while others can support all three. The key to determining the possible relative values of the number of instances in each of the two entities is the maximum and minimum cardinality constraints placed on the relationship [15,22]. For example, in a 1:1 mandatory–mandatory binary relationship, the number of instances in each entity participating in the relationship must always be equal to each other by the definition of the cardinality constraints on the relationship [4,31]. Changing the cardinality constraints can relax the restrictions allowing the inequalities to vary in either direction. For example, in a 1:1 optional–optional binary relationship, the relative number of instance values between E_1 and E_2 can be $|E_1| = |E_2|$, $|E_1| < |E_2|$, or $|E_1| > |E_2|$ depending on how many instances in each entity do not participate in the relationship [20]. This type of relationship, when occurring in a cyclic path, is self-adjusting with respect to the relative values of the instances in each entity in a path allowing the relative values to vary to accommodate other restrictions placed on the whole path. This accommodation is similar to what occurs to entities that are restricted. In a 'many-to-one' mandatory–mandatory relationship the entity on the 'One' side accommodates the relationship by having less instances than the entity on the 'many' side.

Table 2
Structural restrictions of a binary relationship

Case	Maximum cardinality	Minimum cardinality	Restrictions on the number of instances between entities
B1	1:1	Mandatory–mandatory	$\|E_1\| = \|E_2\|$
B2	1:1	Mandatory–optional	$\|E_1\| < \|E_2\|$
B3	1:1	Optional–mandatory	$\|E_1\| > \|E_2\|$
B4	1:1	Optional–optional	Self-adjusting
B5	M:1	Mandatory–mandatory	$\|E_1\| > \|E_2\|$
B6	M:1	Mandatory–optional	Non-restrictive
B7	M:1	Optional–mandatory	$\|E_1\| > \|E_2\|$
B8	M:1	Optional–optional	Self-adjusting
B9	1:M	Mandatory–mandatory	$\|E_1\| < \|E_2\|$
B10	1:M	Mandatory–optional	$\|E_1\| < \|E_2\|$
B11	1:M	Optional–mandatory	Self-adjusting
B12	1:M	Optional–optional	Self-adjusting
B13	$M : N$	Mandatory–mandatory	Self-adjusting
B14	$M : N$	Mandatory–optional	Self-adjusting
B15	$M : N$	Optional–mandatory	Self-adjusting
B16	$M : N$	Optional–optional	Self-adjusting

In considering both cardinality constraints there are 16 possible combinations of a simple path consisting of just two entities and one relationship between them. Table 2 identifies each of the 16 simple paths and the restrictions that apply to these paths. Having a restriction does not make the path structurally invalid; it only identifies the structural constraints that are imposed on the path. A path will become structurally invalid only when it cannot support all the structural constraints imposed on the path simultaneously. Our analysis will consider two types of paths: acyclic and cyclic.

4.1. Acyclic paths

Acyclic paths are paths that do not recur back on a previous entity. Acyclic paths are open-ended paths where each entity only has a direct structural effect on an adjacent entity in the path. An entity may have a transitive relationship with a non-adjacent entity in an acyclic path, but this is a semantic issue and not a structural issue. The only structural issue in a transitive relationship is that connectivity has been established. Fig. 7 shows an example of an acyclic path with the simple binary relationship of E_1 associated with E_2 through R_{12}. As shown in Table 2 this path of a simple binary relationship is valid, independent of the cardinality constraints imposed on the single relationship. We can create another acyclic path by appending R_{23} associating E_2 with E_3 as shown in Fig. 8. Even though the two relationships share the common entity E_2 and there may be

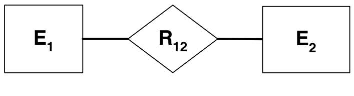


Fig. 7. An example of an acyclic path with two entities.

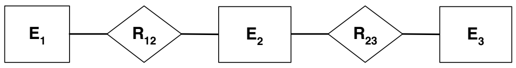


Fig. 8. An example of an acyclic path with three entities.

a transitive semantic relationship between them, they are structurally mutually exclusive of each other.

In an acyclic path the maximum and minimum cardinality constraints of two relationships sharing the same entity are independent of each other. Either relationship can take on any of the 16 possible cardinality combinations of Table 2 without affecting the structural validity of the other relationship. Since any single simple binary relationship is always structurally valid and each binary relationship in an acyclic path is structurally independent of any other binary relationship, an acyclic path containing binary relationships is always structurally valid.

Rule 5. *An acyclic path containing all binary relationships is always structurally valid.*

4.2. Self-adjusting relationships and cyclic paths

A cyclic path is a closed path having the capability of starting with and ending with the same entity. This capability extends itself to any entity within the cyclic path. Unlike acyclic paths and because of the nature of a closed path, it is possible for an entity in a cyclic path to have indirect structural associations with other non-adjacent entities in the path. Relationships in a cyclic path can be structurally dependent on each other.

Table 3 shows the four possible conditions that can exist for a binary relationship derived from Table 2. By examining the possible generalized combinations of these four conditions we can develop a set of heuristics that determine the structural validity of a cyclic path containing a self-adjusting relationship.

Theorem 4. *Any cyclic path of binary relationships containing a self-adjusting relationship* (*see Table* 3) *will be structurally valid.*

Proof. Given a cyclic path with entities $E_1, E_2, E_3, \ldots, E_{(N-1)}, E_N$ with relationships $R_{12}, R_{23}, R_{34}, \ldots, R_{(N-1)N}, R_{N1}$ and R_{N1} is self-adjusting. The acyclic path between $E_1, E_2, E_3, \ldots, E_{(N-1)}, E_N$ is always valid as stated in our Rule 5 and will yield an inequality (or equation) between the number

Table 3
Four possible conditions for binary relationships

Cases	Restrictions on the number of instances between entities
B4, B6, B8, B11, B12, B13, B14, B15, B16	Self-adjusting
B1	$\lvert E_1\rvert = \lvert E_2\rvert$
B2, B9, B10	$\lvert E_1\rvert < \lvert E_2\rvert$
B3, B5, B7	$\lvert E_1\rvert > \lvert E_2\rvert$

of instances in E_1 and E_N. Since the binary relationship R_{N1} is self-adjusting, the simple binary relationship between E_N and E_1 will accommodate the restrictions placed on E_N and E_1 by the acyclic path between $E_1, E_2, E_3, \ldots, E_{(N-1)}, E_N$. $\square$

From Table 2 there are three structure types that are self-adjusting. They are 'optional–optional', {many-to-one when 'optional' on the 'one' side}, and 'many-to-many'. The following three rules can be developed from the proof of Theorem 4.

Rule 6. *A cyclic path that contains all binary relationships, and one or more 'optional–optional' relationships is always structurally valid.*

Rule 7. *A cyclic path that contains all binary relationships, and one or more 'many-to-one' relationships with 'optional' participation on the 'one' side is always structurally valid.*

Rule 8. *A cyclic path that contains all binary relationships and one or more 'many-to-many' relationships is always structurally valid.*

4.3. Opposing relationships

In the absence of a self-adjusting relationship in a cyclic path, each relationship in the path must take on one and only one of the equality conditions, $|E_1| = |E_2|$, $|E_1| < |E_2|$, or $|E_1| > |E_2|$. The equality condition $|E_1| = |E_2|$ has a neutral effect on the path and will be discussed in the next section. In this section, we look at the conditions $|E_1| < |E_2|$ and $|E_1| > |E_2|$, and their behavior in a cyclic path. Fig. 9 demonstrates the concept of opposing relationships. In Fig. 9 the entity EMPLOYEE is associated with the entity DEPARTMENT in two ways. The diagram is structurally valid because the two relationships comply simultaneously with the restrictions in Table 2. From Table 2 (Case B5) $R_{\text{MEMBER\_OF}}$ states that $|E_{\text{EMPLOYEE}}| > |E_{\text{DEPARTMENT}}|$ and from (Case B3) R_{MANAGES} also states that $|E_{\text{EMPLOYEE}}| > |E_{\text{DEPARTMENT}}|$. Since they both reflect the same inequality with respect to EMPLOYEE and DEPARTMENT, and there are no other imposed constraints on the diagram then the cyclic path is structurally valid.

With respect to cyclic path {EMPLOYEE–DEPARTMENT–EMPLOYEE} the relationships are opposing as one moves through the path. They oppose each other in that reading from left to right EMPLOYEE is greater than DEPARTMENT and DEPARTMENT is less than EMPLOYEE.

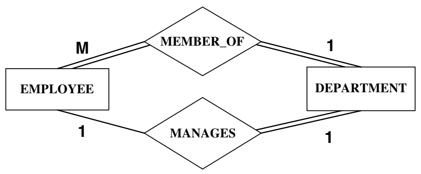


Fig. 9. An example of two entities in a cyclic path.

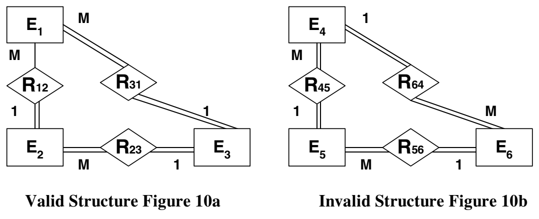


Fig. 10. An example of valid and invalid structures in a cyclic path.

Theorem 5. *In the absence of a self-adjusting or equivalent relationship* ($|E_1| = |E_2|$) *in a cyclic path there must exist at least one set of opposing relationships for the path to be valid.*

Proof. Consider Fig. 10a. It represents a valid structure because it supports a valid set of instances for each entity. From Table 2 we can see that R_{12} corresponds to case B7 and R_{23} corresponds to case B5 yielding the following inequalities:

$$R_{12} \quad \text{from} \quad \text{B7 } |E_1| > |E_2| \tag{1}$$

$$R_{23} \quad \text{from} \quad \text{B5 } |E_2| > |E_3| \tag{2}$$

$$\text{Concluding that} \quad |E_1| > |E_3| \tag{3}$$

Also from Table 2, R_{31} corresponds to case B9

$$R_{31} \quad \text{from} \quad \text{B9 } |E_3| < |E_1| \tag{4}$$

Since inequality (3) is opposing inequality (4) in this path, then the diagram is a valid structure and can support a valid set of semantics that is represented by the combination of these relationships. □

Another view of the proof is that $|E_3| < |E_1|$ from (4) implies $|E_1| > |E_3|$ (4a). Since (4a) is equivalent to (3), the E_1–E_2–E_3 sub-path yields the same inequality association as the sub-path E_1–E_3. Therefore, the structure can support a valid set of instances.

Proof. Consider Fig. 10b. It is an invalid structure because it cannot support at least one valid set of instances for each entity. From Table 2 we can see that both R_{45} and R_{56} correspond to case B5 yielding the following inequalities:

$$R_{45} \quad \text{from} \quad \text{B5 } |E_4| > |E_5| \tag{5}$$

$$R_{56} \quad \text{from} \quad \text{B5 } |E_5| > |E_6| \tag{6}$$

$$\text{concluding that } |E_4| > |E_6| \tag{7}$$

Also from Table 2 we can see that R_{64} also corresponds to case B9

$$R_{64} \quad \text{from} \quad \text{B5 } |E_6| > |E_4|. \quad \square \tag{8}$$

This cyclic path contains no opposing relationships. They are all flowing in one direction. Inequality $|E_6| > |E_4|$ (8) also implies the inequality $|E_4| < |E_6|$ (8a). Since inequality (7) is not equivalent to (and in contradiction of) inequality (8a) while representing the same entities (E_4 and E_6), this structure cannot support any set of data instances and, therefore, is an invalid structure. Figs. 9 and 10 demonstrate that a cyclic path without *self-adjusting* relationships can be valid if and only if the inequalities are supported holistically throughout the diagram. This requires the presence of at least one relationship whose inequality solution is *opposing* the inequality solution of another relationship in the cyclic path. A set of *opposing* relationships consists of at least one binary relationship from this group [{1 : M mandatory–mandatory}, {1 : M mandatory–optional}, or {1:1 mandatory–optional}] and at least one other binary relationship from [{M : 1 mandatory–mandatory}, {M : 1 optional–mandatory}, or {1:1 optional–mandatory}].

Rule 9. *Cyclic paths containing at least one set of* ***opposing*** *relationships are always valid.*

Corollary 4. *Cyclic paths containing no opposing relationships and no self-adjusting relationships are structurally* ***invalid*** *and called a Circular Relationship.*

4.4. Neutral effects

The presence of a 1:1 mandatory–mandatory (Case B1) relationship in a cyclic path with more than two relationships has a neutral effect with respect to the other relationships and the validity of the path. The neutrality of a B1 type relationship can be shown by taking any valid cyclic path P with entities $E_1, E_2, E_3, \ldots, E_{(N-1)}, E_N$ with relationships $R_{12}, R_{23}, R_{34}, \ldots, R_{(N-1)N}, R_{N1}$ where $N > 2$. By inserting an entity E_{N+1} and a B1 type relationship R_{N+1} after R_N to cyclic path P will have no effect on the validity of the path because $|E_{N+1}| = |E_N|$ and the structural relationship between E_1 and E_N is equivalent to the structural relationship between E_1 and E_{N+1} thus maintaining the consistency with all other entities in the path. Similarly it can be shown that adding a B1 type relationship to an invalid path can continue to maintain the path's invalidity; therefore it is neutral in its effect.

Corollary 5. *The presence of a* {'*One-to-one*' *mandatory–mandatory*} *relationship has no effect on the structural validity* (*or invalidity*) *of a cyclic path containing other relationship types.* (*This corollary applies to all the above rules.*)

A cyclic path containing all B1 type relationships is always structurally valid because each entity has the same number of instances as all the others in the path supporting the maximum and minimum cardinality constraints placed on the relationships.

Rule 10. *A cyclic path containing all* '*one-to-one*' *mandatory–mandatory binary relationships is always structurally valid.*

4.5. Analysis of Fig. 1b

In Section 4 we developed six decision rules for determining the validity of a binary relationship and two corollaries that quickly identify invalid binary relationships. Appendix B summarizes each validity rule and corollary for binary relationships with an example.

In Fig. 1b of this paper the ERD contains three cyclic paths that require evaluation. The cyclic path PRODUCT–EMPLOYEE–PROJECT–DEPARTMENT–DIVISION–PRODUCT is a valid path because it contains at least one opposing relationship and meets the evaluation criteria for Rule 10. From Corollary 5 the one-to-one relationship RESPONSIBLE has no effect on the validity of the path. The path PRODUCT–EMPLOYEE–DEPARTMENT–DIVISION–PRODUCT is an invalid path according to Corollary 4 and 5. It does not have any opposing or self-adjusting relationships. The path EMPLOYEE–PROJECT–DEPARTMENT–EMPLOYEE is also an invalid path governed by Corollary 4. It is a circular relationship.

5. Effects of ternary relationships on validity

In this section, we examine the criteria to evaluate the validity of an ERD containing paths with binary and ternary relationships. *Ternary* relationships are associations involving three entities. In a ternary relationship the association of an instance from each entity participating in the relationship is represented as a triple [10,22,27].

Membership in the triple implies that instance pairs of two entities are associated with an instance from the remaining entity [20]. Consider the example of a ternary relationship with entities X, Y, and Z, each containing one instance $X1$, $Y1$, and $Z1$, respectively. The triple represents the association of the instance pair $(Y1, Z1)$ with $X1$, $(X1, Z1)$ with $Y1$, and $(X1, Y1)$ with $Z1$. Continuing to populate the entities with instances, the number of triples can increase by the combinations of allowable 'instance pairs to instances' in the relationship. The allowable pairing is only restricted by the maximum and minimum cardinality constraints on the relationship [12,14].

5.1. Ternary relationships as members of acyclic paths

Like binary relationships, ternary relationships can be part of an acyclic or cyclic path. Only two entities of the ternary relationship participate in the path. When evaluating the validity of an acyclic path containing a ternary relationship, only the binary relationship between the two entities in the path is used. The binary relationship between two entities of a ternary relationship is derived from the ternary relationship and can either be explicit or implicit. Since our previous Rule 6 states that an acyclic path containing all binary relationships is always valid and since a derived binary association represents the binary relationship between two entities in a ternary relationship, an acyclic path containing a ternary relationship is always structurally valid.

Rule 6 (Restated). *If a path containing binary and ternary relationships is an acyclic path, then the path is always structurally valid.*

5.2. Ternary relationships as members of cyclic paths

Song and Jones [18,19,25] have performed comprehensive studies of the simultaneous coexistence of ternary and binary relationships, and the interrelationships created by their joint coexistence. They identified three types of binary-ternary coexistence relationships: unconstrained ternary relationships with derived binary relationships between entities, ternary relationships further constrained by explicit binary relationships, and unrelated binary relationships between two entities of a ternary relationship. In their analysis they established rules to determine the standalone validity of these three relationship types. The purpose of the next three sections is to investigate and establish rules to determine the overall validity of an ERD containing these three relationship types.

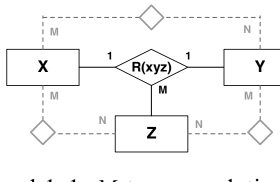
5.2.1. No explicit binary relationships (constraining or unrelated) between entities

In this section we examine the criteria for evaluating the validity of cyclic path containing a ternary relationship with no explicit binary relationships between entities. Between any two entities in a ternary relationship, when there is no explicit constraining binary relationship, there is an implicit binary relationship that describes the association between the instances of only these two entities. Jones and Song [18] established the *implicit binary cardinality* (*IBC*) rule stated as follows.

> In any given ternary relationship, regardless of ternary cardinality, the implicit cardinalities between any two entities must be considered ‘many-to-many’, provided that there is no explicit restrictions on the number of instances that can occur.

The rule implies that the maximum cardinality constraints of a ternary relationship have no effect on the cardinality of implicit binary relationships provided that there are no explicit restrictions on the ternary relationship. The following example demonstrates that the implicit binary relationship between any of the two entities tends towards a ‘many-to-many’ relationship.

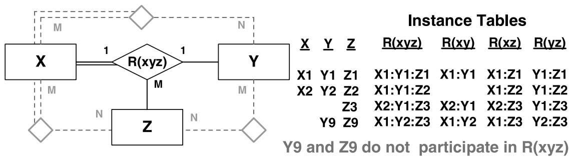
A $1:1:M$ ternary relationship allows one of the entities’ instances to participate in the instance pairings of the other entities more than once. The minimal set of triples to demonstrate a $1:1:M$ relationship is: $X1:Y1:Z1$ and $X1:Y1:Z2$, showing that the instance pair $\{X1, Y1\}$ is associated with more that one instance in entity Z. Adding two additional triples ($X2:Y1:Z3$ and $X1:Y2:Z3$) as shown in Fig. 11 will allow us to draw some inference about maximum cardinality constraints of the embedded binary relationships. An embedded (or implicit) binary relationship is a relationship between two entities in a ternary that is not explicitly modeled in the diagram but can be derived between two entities as they participate in the ternary relationship. The observed binary



Instance Tables

X	Y	Z	R(xyz)	R(xy)	R(xz)	R(yz)
X1	Y1	Z1	X1:Y1:Z1	X1:Y1	X1:Z1	Y1:Z1
X2	Y2	Z2	X1:Y1:Z2		X1:Z2	Y1:Z2
		Z3	X2:Y1:Z3	X2:Y1	X2:Z3	Y1:Z3
			X1:Y2:Z3	X1:Y2	X1:Z3	Y2:Z3

Fig. 11. An unconstrained $1:1:M$ ternary relationship showing three embedded binary relationships.



Instance Tables

X	Y	Z	R(xyz)	R(xy)	R(xz)	R(yz)
X1	Y1	Z1	X1:Y1:Z1	X1:Y1	X1:Z1	Y1:Z1
X2	Y2	Z2	X1:Y1:Z2		X1:Z2	Y1:Z2
		Z3	X2:Y1:Z3	X2:Y1	X2:Z3	Y1:Z3
	Y9	Z9	X1:Y2:Z3	X1:Y2	X1:Z3	Y2:Z3

Y9 and Z9 do not participate in R(xyz)

Fig. 12. A 1 : 1 : M ternary relationship with minimum cardinality constraints imposed showing there is no effect on the embedded binary relationships.

relationships $R(xy)$ between entities X and Y in the above set of triples is also ‘many-to-many’, and likewise entities X and Z from $R(xz)$, and entities Y and Z from $R(yz)$, as shown in the instance tables in Fig. 11. This example demonstrates that the maximum cardinality constraints for the implicit binary relationships embedded in a $1:1:M$ ternary relationship tends towards $M:N$. Jones [18] shows this phenomenon extends to all other combinations of maximum cardinality constraints for ternary relationships.

The above example in Fig. 11 considered the minimal number of instance triples to satisfy the maximum cardinality constraints. Adding two instances ($Y9$ and $Z9$) to entities Y and Z does not have any effect on the maximum cardinality constraints of the embedded binary relationships as shown in the instance tables in Fig. 12. The conclusion is that the minimum cardinality constraints have no effect on the IBC rule.

Rule 12. *A cyclic path containing a ternary relationship where there are no explicitly restricting binary relationships on the ternary's entities is always structurally valid, regardless of maximum or minimum cardinality constraints on the ternary relationship.*

5.2.2. The effects of explicit unrelated binary relationships involved with a ternary relationship

An explicit binary relationship between two entities participating in a ternary relationship can be unrelated to the ternary relationship. When that binary relationship imposes a concept different from the concept being presented by the ternary and does not constrain the instances participating in the ternary relationship, it is considered to be *unrelated* [28].

For example, if an association exists between the owner of a book, the title of the book, and store where the book is purchased, then a ternary relationship can be used to model this association. Fig. 13a shows how that relationship would be diagrammed. If we introduce an additional relationship that is independent of the OWNERSHIP relationship, such as the reader of the book, then the binary relationship READ between Person and Book in Fig. 13b is an explicit *unrelated* relationship. Owning and reading a book are two different concepts. In an unrelated relationship

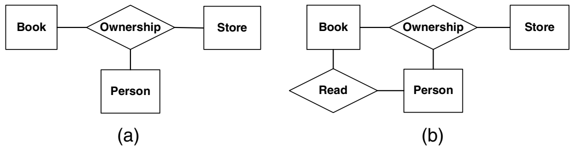


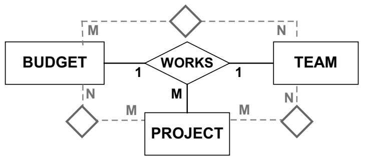
Fig. 13. (a) A temporary relationship with no explicit relationships and (b) a temporary relationship with an explicit unrelated binary relationships.

the maximum and minimum cardinality of the binary is independent of the ternary. From a structural validity viewpoint, unrelated binary relationships imposed on two entities in a ternary relationship, behave in the same manner as a ternary involved in a cyclic path and should be evaluated in the same manner.

5.2.3. The effects of explicit constraining binary relationships imposed on ternary relationships

There are real world scenarios involving three entities that cannot be modeled by using only a simple ternary relationship, because the implicit cardinality between any two entities is ‘many-to-many’, but can be modeled using the combination of a ternary relationship and a constraining binary relationship. A constraining binary relationship on the ternary relationship is used to further define the association between two entities and is considered part of the ternary relationship. An explicit constraining relationship restricts the instance groupings of the ternary relationship by directly constraining the possible instance pairings allowed between the two entities. Fig. 14 shows a simple ternary relationship that associates PROJECT, BUDGET, and TEAM.

Fig. 14 shows an $M : 1 : 1$ relationship where each Project is associated with a Team and Budget combination. The functional dependencies $\{\text{Budget, Project}\} \rightarrow \{\text{Team}\}$ and $\{\text{Team, Project}\} \rightarrow \{\text{Budget}\}$ are derived from the $M : 1 : 1$ ternary relationship. The instance table in Fig. 14 shows a possible set of instance triples that conform to the diagram and the requirements. The instance table reflects the ‘many-to-many’ embedded binary relationships between each pair of entities in the ternary relationship. Assume a recent change in business policy imposes the following additional constraint on the model. Projects can only be funded from a single budget. At first, we may try to model this rule by modifying the ternary relationship. If we introduce the functional dependency $\{\text{Budget, Team}\} \rightarrow \{\text{Project}\}$, we violate the condition that ‘Teams may work on more than one project’. Relaxing our constraints is certainly not the answer to our problem either. Modifying the ternary association to create another simple ternary relationship does not accomplish our objective. The reason is the new requirement’s association between Project and Budget is only a binary association and does not directly involve the entity Team as shown by its functional dependency $\{\text{Project}\} \rightarrow \{\text{Budget}\}$. This is a constraining binary relationship that is considered part of the ternary relationship because it is subset of the ternary relationship. Jones and Song [18] established the explicit binary permission (EBP) rule for evaluating potential combinations of explicit constraining binary relationships allowed to be imposed on a ternary relationship. Their EBP rule states: “For any given ternary relationship, a binary relationship cannot be imposed where the binary cardinality is less than the cardinality specified by the ternary, for any specific entity”. Table 4 summarizes the allowable and disallowable binary impositions on the different cardinality constraints of a ternary relationship.



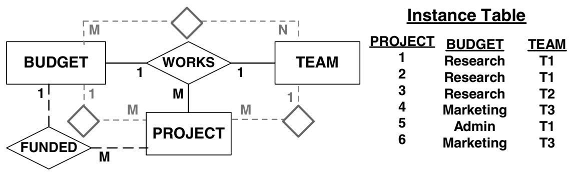
Instance Table

PROJECT	BUDGET	TEAM
1	Research	T1
2	Research	T1
3	Research	T2
3	Marketing	T3
4	Marketing	T3
5	Admin	T1
5	Research	T2
6	Marketing	T3

Fig. 14. A simple ternary relationship with resulting embedded binary relationships and a possible set of instance triples.

Table 4
Allowed and disallowed constraining binary impositions [18]

1:1:1 ternary relationships			
Any cardinality of a binary relationship can be imposed on a 1:1:1 ternary relationship			
1 : 1 : *M* ternary relationships			
X	*Y*	*Z*	Allowed
M	*N*		Yes
M	1		Yes
1	1		Yes
1	*M*		Yes
M		*N*	Yes
M		1	No
1		1	No
1		*M*	Yes
	M	*N*	Yes
	M	1	No
	1	1	No
	1	*M*	Yes
1 : *M* : *N* ternary relationships			
X	*Y*	*Z*	Allowed
M	*N*		Yes
M	1		No
1	1		No
1	*M*		Yes
M		*N*	Yes
M		1	No
1		1	No
1		*M*	Yes
	M	*N*	Yes
	M	1	No
	1	1	No
	1	*M*	No
M : *N* : *P* ternary relationships			
Only binary cardinality of *M* : *N* can be imposed on a *M* : *N* : *P* ternary relationship			
This imposition is redundant since the ternary relationship implicitly establishes the cardinality constraints			



PROJECT	BUDGET	TEAM
1	Research	T1
2	Research	T1
3	Research	T2
4	Marketing	T3
5	Admin	T1
6	Marketing	T3

Fig. 15. A ternary relationship with an explicit constraining binary relationship (FUNDED) and the resulting embedded binary relationships with a possible set of instance triples.

The constraining binary relationship in our example is allowable according to the EBP rule. Fig. 15 shows the ternary relationship from Fig. 14 with the explicit binary relationship imposed, the resulting maximum cardinality constraints of the embedded binary relationships, and a possible instance table to reflect the instance triples of the diagram. Important to our analysis of structural validity are the changes that occur to the embedded binary relationships due to the imposition. The embedded relationship between Project and Budget follows the constraining $M:1$ relationship as expected but the additional embedded relationship between Project and Team also changes to $M:1$ because of the imposition. The embedded relationships change because of the additional binary FD(s) imposed on the ternary relationship and the additional binary FDs that may be derived from the imposition on the ternary relationships. Jones [18] analyzed the effects of an imposed constraining binary relationship on the embedded binary relationships in a ternary relationship and their results are presented in Table 5.

Fig. 16 is another example ERD showing a ternary relationship without an imposed constraining relationship. This diagram contains a ternary relationship and two cyclic paths, *XYZWVX*

Table 5
Effects of single binary imposition on a ternary relationship [18]

Ternary relationship $X:Y:Z$	Imposed constraining binary relationship	Effect on the embedded relationships
1:1:1	$X:Y$ is $M:1$	$X:Y$ is $M:1$
		$X:Z$ is $M:1$
		$Y:Z$ is $M:N$
	$X:Y$ is $1:M$	$X:Y$ is $1:M$
		$X:Z$ is $M:N$
		$Y:Z$ is $M:1$
	$X:Y$ is 1:1	$X:Y$ is 1:1
		$X:Z$ is $M:1$
		$Y:Z$ is $M:1$
$M:1:1$	$X:Y$ is $M:1$	$X:Y$ is $M:1$
		$X:Z$ is $M:1$
		$Y:Z$ is $M:N$
	$X:Z$ is $M:1$	$X:Y$ is $M:1$
		$X:Z$ is $M:1$
		$Y:Z$ is $M:N$
	$Y:Z$ is $M:1$	$X:Y$ is $M:N$
		$X:Z$ is $M:N$
		$Y:Z$ is $M:1$
	$Y:Z$ is $1:M$	$X:Y$ is $M:N$
		$X:Z$ is $M:N$
		$Y:Z$ is $1:M$
	$Y:Z$ is 1:1	$X:Y$ is $M:N$
		$X:Z$ is $M:N$
		$Y:Z$ is 1:1
$M:N:1$	$X:Z$ is $M:1$	$X:Y$ is $M:N$
		$X:Z$ is $M:1$
		$Y:Z$ is $M:N$
	$Y:Z$ is $M:1$	$X:Y$ is $M:N$
		$X:Z$ is $M:N$
		$Y:Z$ is $M:1$

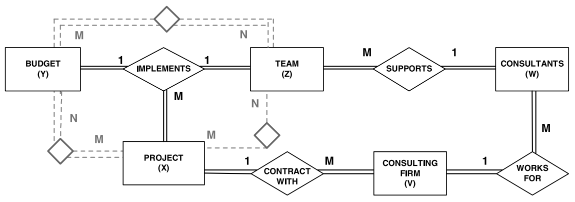


Fig. 16. An example of a valid multi-path ERD containing an unconstrained ternary relationship showing the embedded binary relationships.

and *XZWVX*, that require evaluation. Both cyclic paths involve embedded binary relationships. First, the ternary is evaluated for structural validity. Since in this example there are no constraining relationships on the ternary, the ternary relationship is structurally valid. The analysis of the cyclic paths for structural validity is restricted only to the explicit or implicit binary relationships in the paths. Both cyclic paths in Fig. 16 contain at least one 'many-to-many' and therefore according to Rule 6 these paths are structurally valid. This diagram is structurally valid because all the paths and the ternary relationship are structurally valid.

From the concepts presented in this section, we conclude that first the ternary relationship must be evaluated for its structural validity and the embedded binary relationships between the entities must be determined. When that step is completed we can identify and evaluate all the paths for structural validity. In evaluating the ternary relationship we use the following rule and corollary.

Rule 13. *If the maximum cardinality constraints for a constraining binary relationship imposed on a ternary relationship is greater than or equal to the maximum cardinality constraints of the ternary relationship between the two involved entities then the constrained ternary relationship is valid.*

Corollary 6. *If the maximum cardinality constraints for a constraining binary relationship imposed on a ternary relationship is less than the maximum cardinality constraints of the ternary relationship between the two involved entities, then the constrained ternary relationship is* ***invalid****.*

5.2.4. Analysis of Fig. 1c

This is an appropriate point to analyze Fig. 1c. This figure is similar to Fig. 16 except that the ternary relationship is constrained by the binary relationship FUNDED. The labels V to Z are used in place of the entity names for readability. We will use the notation $R(xyz)|R(xy)$ to indicate that $R(xy)$ is a constraining relationship on $R(xyz)$. This diagram contains the constrained ternary relationship $R(xyz)|R(xy)$ and two cyclic paths, *XYZWVX* and *XZWVX*. The first step again is to analyze the ternary relationship with the constraining relationships imposed on it. $R(xy)$ is a $M:1$ constraining relationship and from Table 4 or the EBP rule we conclude it is a valid imposition on a $M:1:1$ ternary relationship. Therefore the ternary is structurally valid. In our analysis of the cyclic paths we only use the binary relationships. In Fig. 1c both cyclic paths *XYZWVX* and *XZWVX* are invalid according to our rules. They are called circular relationships as described in Section 4 and according to Corollary 4 these cyclic paths are invalid. This ERD in Fig. 1c is invalid.

5.2.5. The effects of multiple constraining relationships on structural validity

In the previous section, we showed that the imposition of a constraining relationship affects the embedded binary relationships between the entities of the ternary relationship and consequently can affect the validity of the ERD. If necessary an additional constraining relationship can be imposed on an already constrained ternary relationship to further define the modeler's concept of the real world environment. Care must be taken when imposing an additional constraining relationship on an already constrained relationship. The first constraining relationship imposes additional derived functional dependencies on the functional dependencies already imposed by the ternary relationship. A second constraining relationship cannot redefine already defined functional dependencies, although a valid imposition may indirectly further define the derived embedded cardinality between two entities. The only entities in a ternary relationship available to be constrained are those that have an embedded 'many-to-many' cardinality constraint between them. These entities do not have binary type functional dependencies between them and are therefore unconstrained with respect to a binary relationship.

For example, in a 1:1:1 ternary relationship $R(xyz)$, if we choose any two entities, such as X and Y, and impose a 1:1 constraining relationship between X and Y, then we introduce two additional functional dependencies on the ternary relationship.

Original FDs on $R(xyz)$	Imposed FDs from $R(xy)$
$(X, Y) \to Z$	$Y \to X$
$(X, Z) \to Y$	$X \to Y$
$(Y, Z) \to X$	

Two additional FDs are derived: $Y \to Z$ and $X \to Z$. From this information each embedded binary relationship in the ternary relationship is defined by a functional dependency, therefore no additional constraining relationships can be imposed.

In another example of a 1:1:1 ternary relationship $R(xyz)$ we choose any two entities, such as X and Y, and impose an $M:1$ constraining relationship between X and Y. This would impose one binary functional dependency on the ternary and one additional FD is derived: $X \to Z$.

Original FDs on $R(xyz)$	Imposed FDs from $R(xy)$	Derived FD
$(X, Y) \to Z$	$X \to Y$	$X \to Z$
$(X, Z) \to Y$		
$(Y, Z) \to X$		

From this information only two embedded binary relationships in the ternary relationship are constrained by binary functional dependencies. The embedded binary association between entities Y and Z is still 'many-to-many' allowing an additional constraining relationship to be imposed if necessary. We have three options to analyze. The relationship between Y and Z can be $M:1$, $1:M$, or 1:1. The only other option of imposing $M:N$ would be redundant. The following are the functional dependencies for the three cases.

$R(xyz)$ constrained by $R(xy)$	Imposed $R(yz)$	FDs from $R(yz)$	Derived FD
$(X,Y) \to Z$	Many-to-one	$Y \to Z$	$Y \to X$
$(X,Z) \to Y$			
$(Y,Z) \to X$	One-to-many	$Z \to Y$	$Z \to X$
$X \to Y$			
$X \to Z$ (derived)	One-to-one	$Y \to Z$	$Y \to X$
		$Z \to Y$	$Z \to X$

In the ‘many-to-one’ case for $R(yz)$ the derived FD $\{Y \to X\}$ in combination with the FD $\{X \to Y\}$ from the explicit previously imposed relationship redefines the relationship $R(xy)$ and therefore is invalid. The same argument applies to the ‘one-to-one’ case. The only valid imposition is the ‘one-to-many’ case because this imposition does not redefine any explicitly defined relationship. It does further define the embedded relationship $R(xz)$ to be ‘one-to-one’. Table 6 from [18] shows the results of a second constraining binary imposition on ternary relationships for all possible relevant cases.

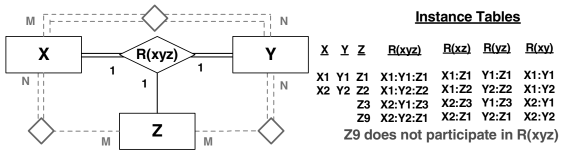
Rule 14. *If a second constraining binary relationship is required to further define an already constrained ternary relationship, then it can only be imposed between two entities where the maximum cardinality constraint is ‘many-to-many’ and the effect of the second constraining relationship cannot redefine any previously defined explicit relationships or relax any previously derived binary relationships for the imposition to be structurally valid.*

5.2.6. The effects of minimum cardinality on implicit and explicit binary relationships

5.2.6.1. Implicit binary relationships. The previous section did not mention minimum cardinality constraints. The purpose of this section is to explore the effects of minimum cardinality on implicit relationships and explicit constraining relationships. Imposing an optional participation constraint on one (or more) of the entities in a ternary relationship restricts at least one instance of the entity from participating in the ternary relationship. We will continue to use instance tables as an analysis methodology and generalize from the results. In Fig. 17, we use four triples to reveal the ‘many-to-many’ embedded binary relationships in the ternary. We use instance $Z9$ in entity Z to represent the set of instances that do not participate in the ternary relationship. We extract the embedded binary relationships $R(xz)$, $R(yz)$, and $R(xy)$ from the instance triples of the ternary relationship $R(xyz)$. Since $Z9$ did not participate in $R(xyz)$ it follows that it does not participate in $R(xz)$ and $R(yz)$. We conclude that when evaluating structural validity, the minimum cardinality constraints of the embedded binary relationships must follow the minimum cardinality constraints on the entities of the ternary relationship regardless of the ternary’s maximum cardinality. The driving reason is that the instance pairs in the embedded relationships must be a subset of the instance triples of the ternary relationship.

Table 6
The effects of multiple binary imposition on ternary relationships [18]

Ternary relationship $X:Y:Z$	Imposed constraining binary relationship	Embedded binary relationship	Additional imposed constraining binary relationship	Resultant embedded binary relationship
1:1:1	$X:Y$ is $M:1$	$X:Y$ is $M:1$	$Y:Z$ is $1:M$	$X:Y$ is $M:1$
		$X:Z$ is $M:1$		$X:Z$ is 1:1
		$Y:Z$ is $M:N$		$Y:Z$ is $1:M$
	$X:Y$ is $M:1$	$X:Y$ is $M:1$	$X:Z$ is $1:M$	$X:Y$ is $M:1$
		$X:Z$ is $M:1$		$X:Z$ is 1:1
		$Y:Z$ is $M:N$		$Y:Z$ is $1:M$
M:1:1	$X:Y$ or $Z:X$ is $M:1$	$X:Y$ is $M:1$	$Y:Z$ is 1:1	$X:Y$ is $M:1$
		$X:Z$ is $M:1$		$X:Z$ is $M:1$
		$Y:Z$ is $M:N$		$Y:Z$ is 1:1
	$X:Y$ or $Z:X$ is $M:1$	$X:Y$ is $M:1$	$Y:Z$ is $M:1$	$X:Y$ is $M:1$
		$X:Z$ is $M:1$		$X:Z$ is $M:1$
		$Y:Z$ is $M:N$		$Y:Z$ is $M:1$
	$X:Y$ or $Z:X$ is $M:1$	$X:Y$ is $M:1$	$Y:Z$ is $1:M$	$X:Y$ is $M:1$
		$X:Z$ is $M:1$		$X:Z$ is $M:1$
		$Y:Z$ is $M:N$		$Y:Z$ is $1:M$
	$Y:Z$ is 1:1	$X:Y$ is $M:N$	$X:Y$ or $X:Z$ is $M:1$	$X:Y$ is $M:1$
		$X:Z$ is $M:N$		$X:Z$ is $M:1$
		$Y:Z$ is 1:1		$Y:Z$ is 1:1
	$Y:Z$ is $M:1$	$X:Y$ is $M:N$	$X:Y$ or $X:Z$ is $M:1$	$X:Y$ is $M:1$
		$X:Z$ is $M:N$		$X:Z$ is $M:1$
		$Y:Z$ is $M:1$		$Y:Z$ is $M:1$
	$Y:Z$ is $1:M$	$X:Y$ is $M:N$	$X:Y$ or $X:Z$ is $M:1$	$X:Y$ is $M:1$
		$X:Z$ is $M:N$		$X:Z$ is $M:1$
		$Y:Z$ is $1:M$		$Y:Z$ is $1:M$
$M:N$:1	$X:Z$ is $M:1$	$X:Y$ is $M:N$	$Y:Z$ is $M:1$	$X:Y$ is $M:N$
		$X:Z$ is $M:1$		$X:Z$ is $M:1$
		$Y:Z$ is $M:N$		$Y:Z$ is $M:1$
	$Y:Z$ is $M:1$	$X:Y$ is $M:N$	$X:Z$ is $M:1$	$X:Y$ is $M:N$
		$X:Z$ is $M:N$		$X:Z$ is $M:1$
		$Y:Z$ is $M:1$		$Y:Z$ is $M:1$



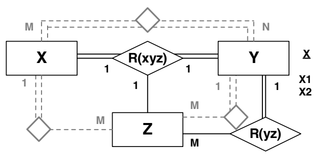
Instance Tables

X	Y	Z	R(xyz)	R(xz)	R(yz)	R(xy)
X1	Y1	Z1	X1:Y1:Z1	X1:Z1	Y1:Z1	X1:Y1
X2	Y2	Z2	X1:Y2:Z2	X1:Z2	Y2:Z2	X1:Y2
		Z3	X2:Y1:Z3	X2:Z3	Y1:Z3	X2:Y1
		Z9	X2:Y2:Z1	X2:Z1	Y2:Z1	X2:Y2

Z9 does not participate in R(xyz)

Fig. 17. A 1:1:1 ternary relationship with optional participation on Z and related instance tables.

5.2.6.2. Explicit binary relationships. In this section, we examine minimum cardinality constraints with respect to explicit binary constraining relationships. We use the concept of minimality with instance tables and functional dependencies to perform our analysis. In our tables let $Z9$ represent the set of instances that do not participate in the relationship. We will first examine the imposition



Instance Tables

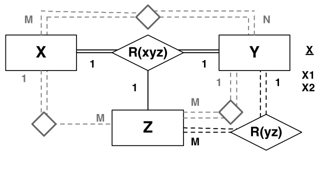
X	Y	Z	R(xyz)	R(yz)	R(xyz)|R(yz)	R(xz)	R(xy)
X1	Y1	Z1	X1:Y1:Z1	Y1:Z1	X1:Y1:Z1	X1:Z1	X1:Y1
X2	Y2	Z2	X1:Y2:Z2	Y2:Z2	X1:Y2:Z2	X1:Z2	X1:Y2
		Z3	X2:Y1:Z3	Y1:Z3	X2:Y1:Z3	X2:Z3	X2:Y1
		Z9	X2:Y2:Z1				

Z9 does not participate in R(xyz)

Fig. 18. A ternary relationship with optional participation on Z and a constraining relationship $R(yz)$ imposed on $R(xyz)$.

of a constraining relationship on a ternary relationship where the minimum cardinality of the binary follows the minimum cardinality of the ternary. Consider Fig. 18 with an imposition of a 'many-to-one' binary constraining relationship between entities Y and Z, and the following possible scenario. This imposition adds the functional dependency $Z \rightarrow Y$ to the relationship removing $Y2 : Z1$ from instance table $R(yz)$ and $X2 : Y2 : Z1$ from $R(xyz)$ as shown in Fig. 18. We also lose $X2 : Z1$ from $R(xz)$ because of the change in the embedded cardinality between Z and X caused by the constraining relationship (See Table 5). The remaining instance groupings in table $R(xyz)|R(yz)$ still contain the minimal number of triples ($X1 : Y1 : Z1$) to represent the ternary relationship and $R(xy)$ remains a 'many-to-many' embedded relationship. In the instance table in Fig. 18 $R(xz)$ and $R(yz)$ correctly reflect 'many-to-one' relationships as shown in Table 5. Fig. 18 and the accompanying instance tables show the effect the constraining relationship $R(yz)$ has on $R(xyz)$. If we remove $Z9$ from the instance table and impose mandatory participation on Z from both the ternary and the constraining binary, we find the results to be the same as in the instance tables of Fig. 18. We conclude that when the minimum cardinality constraint of the constraining binary relationship follows the minimum cardinality of the ternary the resulting embedded binary relationships remain to be subsets of both the original and the constrained ternary relationships.

We further find that from our definition of a constraining relationship the instance pairs of $R(yz)$ are always a subset of the instance triples of $R(xyz)$. If $R(yz)$ contains an instance pair that is not a subset of $R(xyz)$, then the binary relationship is *unrelated* to the ternary relationship and not a constraining relationship. Consider the case where the minimum cardinality constraint is 'optional' on one of the entities of the ternary relationship that is being constrained. In Fig. 19 the proposed constraining relationship $R(yz)$ is 'mandatory' forcing $Z9$ to participate in $R(yz)$ without participating in $R(xyz)$. $Y1 : Z9$ in $R(yz)$ is not a member of $R(xyz)$, therefore, $R(yz)$ is not a subset of the $R(xyz)$, and therefore, $R(yz)$ cannot be a constraining relationship. If the intent is for that relationship to be constraining, then it is semantically and structurally invalid because it is unrelated to the ternary relationship. Fig. 19 shows this example of an invalid constraining relationship $R(yz)$.



Instance Tables

X	Y	Z	R(xyz)	R(yz)
X1	Y1	Z1	X1:Y1:Z1	Y1:Z1
X2	Y2	Z2	X1:Y2:Z2	Y2:Z2
		Z3	X2:Y1:Z3	Y1:Z3
		Z9	X2:Y2:Z1	
				Y1:Z9

R(yz) is not constraining because the instances of R(yz) are not a subset of the instances of R(xyz)

Z9 does not participate in R(xyz)

Fig. 19. A 1:1:1 ternary relationship with optional participation on Z ($Z9$ does not participate) and an invalid constraining relationship $R(yz)$ with mandatory participation.

The case where the ternary is fully mandatory and the proposed constraining relationship is 'optional' on one side, presents a similar issue. The driving force in the resulting constrained relationship is the ternary relationship not the constraining relationship. In this case the ternary relationship is allowing all instances to participate in the relationship. The constraining binary relationship is in conflict with the ternary relationship in that it restricts at least one instance that is not restricted by the ternary. In this case the constraining relationship is redefining the ternary relationship not further defining it. We state the following rule and corollary.

Rule 15. *The minimum cardinality constraints of a constraining relationship must follow the minimum cardinality constraints of the ternary relationship being constrained for it to be structurally valid.*

Corollary 7. *If the minimum cardinality constraints of a constraining relationship do not follow the minimum cardinality constraints of the ternary relationship, then the constrained ternary relationship is* ***invalid****.*

6. Conclusion

In this paper, we have discussed the structural validity of ERD. The structural validity of an ERD is concerned with whether or not a given ERD contains any constructs that are contradictory to each other. An ERD with at least one inconsistent cardinality constraint is structurally invalid. In this paper we have developed rules that allow us to check whether a given ERD is structurally invalid. In defining the properties and validity criteria, while most previous analyses used only maximum cardinality constraints, we have used both maximum and minimum cardinality constraints. Analyses with minimum constraints, in addition to maximum constraints, result in more complete criteria for checking structural validity in ERD. We have shown that the use of maximum constraints alone cannot completely determine the structural validity. The absence of using the minimum cardinality constraint in determining structural validity does not give full consideration to the role each instance plays in each relationship and makes the incorrect assumption that all instances always participate in each relationship. We believe that our analysis has yielded a complete and comprehensive set of decision rules to determine the structural validity of any ERD containing recursive, binary, and ternary relationships.

For recursive relationships, we have first presented the complete classification of recursive relationships and then the criteria that contribute to the validity of modeling recursive relationships. We have classified all recursive relationships into symmetric and asymmetric. Asymmetric recursive relationships are further classified into hierarchical, circular, and mirrored. We have identified their directional properties and their validity in terms of maximum and minimum cardinality constraints. Our decision rules used concepts of role uniqueness, path connectivity, and cardinality constraints. Five rules and three corollaries were established to determine structural validity of recursive relationships. Appendix A summarizes each rule and corollary with valid and invalid examples.

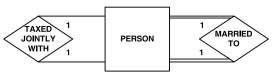
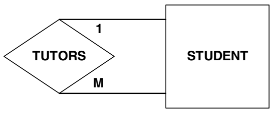
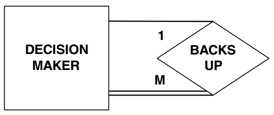
For the structural validity of any ERD containing binary relationships, we have used the notions of path connectivity and cardinality constraints. Both acyclic and cyclic paths were investigated and all combinations of minimum and maximum cardinality constraints were analyzed. The concepts of opposing and self-adjusting relationships were introduced to determine cyclic path validity. Six rules and two corollaries were established to determine structural validity of binary relationships. Appendix B states each rule and corollary with examples.

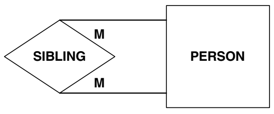
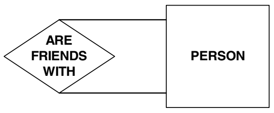
Finally, we have also performed a complete investigation of the structural validity of ternary relationships. We improved the analysis of structural validity of ternary relationships in two aspects. First, while previous approaches used only maximum cardinality constraints, we considered both minimum and maximum constraints. Second, while previous approaches analyzed only in standalone ternary relationships, our study considered the ternary relationship as part of the overall diagram. That is, our rules addressed ternary relationships as they coexist with other relationships in a path structure within the model. These two points allowed us to develop comprehensive rules for checking validity of more complex business concepts. Our analysis yielded four additional rules and two corollaries. We summarized them with examples in Appendix C.

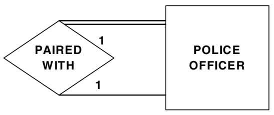
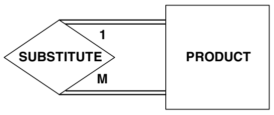
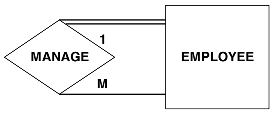
The contribution of this paper is to provide a comprehensive set of decision rules to determine the structural validity of any ERD containing recursive, binary, and ternary relationships. The rules presented in this paper are easy to use to evaluate the structural validity of complex ERDs containing recursive, binary, and ternary relationships. The 15 rules and seven corollaries add value to the information management analysis and design process insofar as they provide a standalone, application-independent tool that can easily be automated to evaluate the structural validity of any ERD regardless of complexity. We believe that this analysis is a completed effort and can be readily implemented in its current form providing an adequate foundation for the evaluation of structural validity in entity relationship modeling. These rules can be extended to the analysis of other diagramming techniques used in data modeling and the analysis of diagrams in the object-oriented model [4,31].

Appendix A

Summary of validity rules for recursive relationships with examples

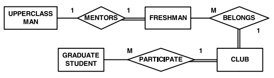
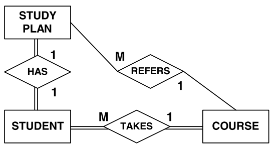
Validity rules for recursive relationships	Valid example
Only 1:1 recursive relationships with mandatory–mandatory or optional–optional minimum cardinality constraints are structurally valid. Valid for symmetrical and completely circular relationships.	 TAXED JOINTLY WITH 1 1 PERSON 1 1 MARRIED TO
For 1:M or M:1 recursive relationships optional–optional minimum cardinality are structurally valid. Valid for asymmetrical relationships only.	
For 1:M recursive relationships of the hierarchical–circular type, optional-mandatory minimum cardinality are structurally valid. Valid for hierarchical–circular relationships only.	

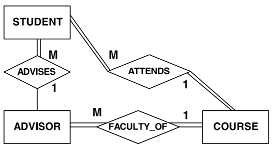
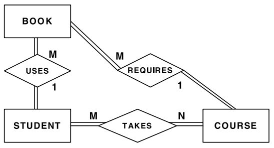
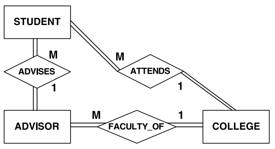
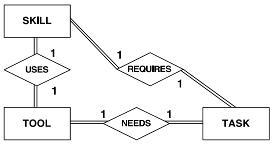
Validity rules for recursive relationships	Valid example
All recursive relationships with many-to-many maximum cardinality are structurally valid regardless of minimum cardinality constraints. Valid for symmetrical, hierarchical, and hierarchical–circular relationships.	
All recursive relationships with optional–optional minimum cardinality are structurally valid. Valid for symmetrical and asymmetrical relationships.	

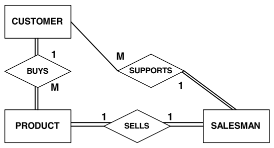
Validity corollaries for recursive relationships	Invalid example
All 1:1 recursive relationships with mandatory–optional or optional–mandatory minimum cardinality constraints are structurally *invalid.*	
All 1:M or M:1 recursive relationships with mandatory–mandatory minimum cardinality constraints are structurally *invalid.*	
All 1:M or M:1 recursive relationships with mandatory participation constraint on the 'one' side and an optional participation constraint on the 'many' constraints are structurally *invalid.*	

Appendix B

Summary of validity rules for binary relationships with examples

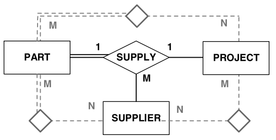
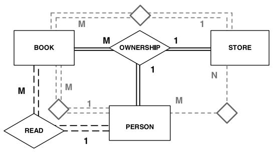
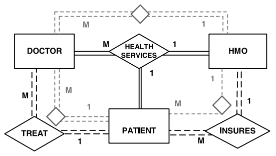
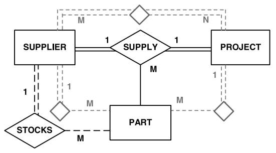
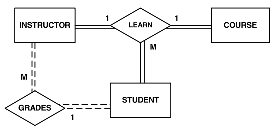
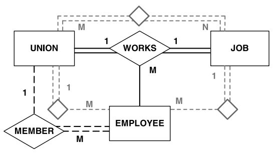
Validity rules for binary relationships	Valid example
An acyclic path containing all binary relationships is always structurally valid.	 UPPERCLASS MAN 1 MENTORS 1 FRESHMAN M BELONGS 1 GRADUATE STUDENT M PARTICIPATE 1 CLUB
A cyclic path that contains all binary relationships, and one or more 'optional–optional' relationships is always structurally valid.	 STUDY PLAN 1 HAS 1 M REFERS 1 STUDENT M TAKES 1 COURSE

Validity rules for binary relationships	Valid example
A cyclic path that contains all binary relationships, and one or more ‘many-to-one’ relationships with ‘optional’ participation on the ‘One’ side is always structurally valid.	 STUDENT M ADVISES 1 M ATTENDS 1 ADVISOR M FACULTY_OF 1 COURSE
A cyclic path that contains all binary relationships, and one or more ‘many-to-many’ relationships is always structurally valid.	
Cyclic paths containing at least one set of ***opposing*** relationships are always valid. Explanation. A set of *opposing* relationships consists of at least one binary relationship from this group [{‘one-to-many’, mandatory–mandatory}, {‘one-to-many’, mandatory–optional}, or {‘one–to-one’, mandatory–optional}] and at least one other binary relationship from [{‘many-to-one’, mandatory–mandatory}, {‘many-to-one’, optional–mandatory}, or {‘one-to-one’, optional–mandatory}].	 STUDENT M ADVISES 1 M ATTENDS 1 ADVISOR M FACULTY_OF 1 COLLEGE
A cyclic path containing all ‘one-to-one’ binary relationships that are either all ‘mandatory–mandatory’ or at least one ‘optional–optional’ minimum cardinality constraint is always structurally valid.	

Validity corollaries for binary relationships	Invalid example
Cyclic paths containing no *opposing* relationships and no self-adjusting relationships are structurally *invalid* and are called a circular relationship. Explanation of opposing relationship: See Rule 9.	
The presence of a {‘one-to-one’ mandatory–mandatory} relationship has no effect on the structural validity (or invalidity) of a cyclic path containing other relationship types. (This corollary applies to all the above rules.)	

Appendix C

Summary of validity rules for ternary relationships with examples

Validity rules for ternary relationships	Valid example
A cyclic path containing a ternary relationship with no explicit restricting binary relationships on the ternary's entities is always structurally valid, regardless of maximum and minimum cardinality constraints on the ternary relationship.	
If the maximum cardinality constraints for a constraining binary relationship imposed on a ternary relationship is greater than or equal to the maximum cardinality constraints of the ternary relationship between the two involved entities, then the constrained ternary relationship is valid.	
If a second constraining binary relationship is required to further define an already constrained ternary relationship, then it can only be imposed between two entities where the maximum cardinality constraint is 'many-to-many' and the effect of the second constraining relationship can not redefine any previously defined explicit relationships or relax any previously derived binary relationships for the imposition to be structurally valid.	 DOCTOR HEALTH SERVICES HMO TREAT PATIENT INSURES M 1 M 1 1 M 1 M 1 1 1 M
The minimum cardinality constraints of a constraining relationship must follow the minimum cardinality constraints of the ternary relationship being constrained for it to be structurally valid.	
Validity corollaries for ternary relationships	**Invalid example**
If the maximum cardinality constraints for a constraining binary relationship imposed on a ternary relationship is less than the maximum cardinality constraints of the ternary relationship between the two involved entities, then the constrained ternary relationship is *invalid*.	
If the minimum cardinality constraints of a constraining relationship do not follow the minimum cardinality constraints of the ternary relationship, then the constrained ternary relationship is *invalid*.	

References

[1] R. Barker, *CASE * METHOD*$^{®}$: Entity Relationship Modeling, Addison-Wesley, New York, 1990.

[2] Boehm, W. Barry, Software Engineering, IEEE Transactions on Computers 25 (12) (1976) 1226–1241.

[3] Boehm, W. Barry, Software Engineering Economics, Prentice-Hall, Englewood Cliffs, NJ, USA, 1981.

[4] D. Calvanese, M. Lenzerini, On the interaction between ISA and cardinality constraints, in: Proceedings of the 10th IEEE International Conference on Data Engineering (ICDE'94), Houston, TX, USA, IEEE Computer Society Press, Silver Spring, 1994, pp. 204–213.

[5] Chen, Peter, The entity-relationship model—toward a unified view of data, ACM Transactions on Database Systems 1 (1) (1976) 9–36.

[6] E.B. Daly, Management of Software Engineering, IEEE Transactions on Software Engineering 3 (3) (1977) 229–242.

[7] J. Dullea, I.-Y. Song, An analysis of cardinality constraints in redundant relationships, in: Proceedings of the 6th International Conference on Information and Knowledge Management, Las Vegas, Nevada, USA, 10–14 November 1997, pp. 270–277.

[8] J. Dullea, I.-Y. Song, An analysis of structural validity in unary and binary relationships in entity relationship modeling, in: Proceedings of the 4th International Conference on Computer Science and Informatics, Research Triangle Park, NC, USA, 23–28 October, 1998, pp. 329–334.

[9] J. Dullea, I.-Y. Song, A taxonomy of recursive relationships and their structural validity in ER modeling, in: Proceedings of the 18th International Conference on Conceptual Modeling (ER'99), Paris, France, 15–18 November 1999, pp. 384–398.

[10] Elmasri, Ramez, B. Shamkant, Navathe, Fundamentals of Database Systems, third ed., Addison-Wesley, Menlo Park, CA, USA, 2000.

[11] M. Fagan, Design and code inspections and process control in the development of programs, IBM Report IBM-SDD-TR-21-572, December 1974.

[12] S. Ferg, Cardinality constraints in entity-relationship modeling, in: Proceedings of the 10th International Conference on Entity-Relationship Approach (ER'91), 23–25 October, 1991, San Mateo, California, USA, ER Institute, 1991, pp. 1–30.

[13] H. Habrias, P. Legrand, A description of rules through occurring/synthetic cardinalities, in: Proceedings of the 7th International Conference on Entity-Relationship Approach (ER'88), Rome, Italy, November 16–18, 1988, North-Holland, Amesterdam, 1989, pp. 509–525.

[14] S. Hartmann, Global Cardinality Constraints, in: Proceedings of the Workshop Challenges of Application and Challenges of Design, 15th International Conference on Conceptual Modeling, Cottbus, Germany, October 9, 1996, Brandenburg Technical University, 1996, pp. 196–206.

[15] S. Hartmann, On the Consistency of Int-cardinality Constraints, in: Proceedings of the 17th International Conference on Conceptual Modeling (ER'98), Singapore, November 16–19, 1998, Lecture Notes in Computer Science 1507, Springer, Berlin, 1998, pp. 150–163.

[16] S. Hartmann, On interactions of cardinality constraints, key, and functional dependencies, in: K.-D. Schewe, B. Thalheim (Eds.), Proceeding of the Foundations of Information and Knowledge Systems, First International Symposium (FoIKS 2000), Burg, Germany, 14–17 February 2000, Lecture Notes in Computer Science 1762, Springer 2000, pp. 136–155.

[17] D.R. Howe, Data Analysis for Data Base Design, second ed., Edward Arnold, London, GB, 1989.

[18] Jones, H. Trevor, Il-Yeol Song, Analysis of binary/ternary cardinality combinations in entity-relationship modeling, Data and Knowledge Engineering 19 (1) (1996) 39–64.

[19] Jones, H. Trevor, I.-Y. Song, Binary equivalents of ternary relationships in E-R modeling: A logical decomposition approach, Journal of Database Management 11 (2) (2000) 12–19.

[20] M. Lenzerini, G. Santucci, Cardinality constraints in the entity-relationship model, in: Proceedings of the 3rd International Conference on Entity-Relationship Approach (ER'83). Anaheim, California, USA, Elsevier Science Publishers B.V., North Holland, 1983, pp. 529–549.

[21] Liddle, W. Stephen, W. David, Embley, N. Scott, Woodfield, Cardinality constraints in semantic data models, Data and Knowledge Engineering 11 (1993) 235–270.

[22] McAllister, J. Andrew, Complete rules for *n*-ary relationship cardinality constraints, Data and Knowledge Engineering 27 (1998) 255–288.
[23] Scheuermann, Peter, Gerd Schiffner, Herbert Weber, Abstraction capabilities and invariant properties modelling within the entity-relationship approach, in: P.P. Chen (Ed.), Entity-Relationship Approach to Systems Analysis and Design, North Holland Publishing Company, Amesterdam, 1980.
[24] Silberschatz, Abraham, Henry F. Korth, S. Sudarshan, Database System Concepts, fourth ed., McGraw-Hill, New York, 2001.
[25] I.-Y. Song, T.H. Jones, An analysis of binary relationships within ternary relationships in ER modeling, in: Proceedings of the 12th International Conference on Entity-Relationship Approach (ER'93), Arlington, Texas, USA, December 15–17, 1993, pp. 265–276.
[26] Song, Il-Yeol, Mary Evans, E.K. Park, A comparative analysis of entity-relationship diagrams, Journal of Computer and Software Engineering 3 (4) (1995) 427–459.
[27] Teorey, J. Toby, Database Modeling and Design, 3rd ed., Morgan Kaufmann Publishers, San Francisco, CA, USA, 1999.
[28] B. Thalheim, Fundamentals of cardinality constraints, in: Proceedings of the 11th International Conference on the Entity-Relationship Approach (ER'92), Karlsruhe, Germany, 7–9 October 1992, Lecture Notes in Computer Science 654, Springer, Berlin, 1992, pp. 7–23.
[29] Thalheim, Bernhard, Entity Relationship Modeling: Fundamentals of Database Technology, Springer-Verlag, Berlin, 2000.
[30] Tillman, George, A Practical Guide to Logical Data Modeling, McGraw-Hill, New York, 1993.
[31] X. Ye, C. Parent, S. Spaccapietra, Cardinality consistency of derived objects in DOOD systems, in: Proceedings of the 13th International Conference on the Entity-Relationship Approach (ER'94) Business Modelling and Re-Engineering, Manchester, UK, 13–16 December 1994, Lecture Notes in Computer Science 881, Springer, Berlin, 1994, pp. 278–295.
[32] J. Zhou, P. Baumann, Evaluation of complex cardinality constraints, in: Proceedings of the 11th International Conference on the Entity-Relationship Approach (ER'92), Karlsruhe, Germany, 7–9 October 1992, Lecture Notes in Computer Science 654, Springer, Berlin, 1992, pp. 24–40.

James Dullea is an Information and Knowledge Management Scientist employed by The Boeing Company. He received his Bachelor of Science degree in Economics and his Master of Science in Statistics both from Villanova University. His Ph.D. in Information Science and Technology was awarded in 1998 by Drexel University. Dr. Dullea also teaches Information Management courses at the graduate schools of Drexel University and Villanova University. His research areas of interest are data and object modeling, data warehousing, data mining, information integration, and logistics.

Il-Yeol Song received his M.S. and Ph.D. degrees in Computer Science from Louisiana State University in 1984 and 1988, respectively. He is a professor in the College of Information Science and Technology at Drexel University, Philadelphia, PA. His current research areas include conceptual modeling, data warehousing, OLAP, web-based systems, and OO analysis & design with UML. He has published over 90 refereed technical articles in various journals and international conferences. Dr. Song has won three teaching awards from Drexel University: *Exemplary Teaching Award* in 1992, *Teaching Excellence Award* in 2000, and the *Lindback Distinguished Teaching Award* in 2001. He received a *Research Scholar Award* from Drexel University in 1992 and won nine Sigma Xi research competition awards. He served as a program co-chair of CIKM'99, DOLAP'98, and DOLAP'99. He serves as a program co-chair for ER2003. He is an associate editor for the Journal of Database Management.

Ioanna Lamprou is a graduate student at Drexel University, Philadelphia, PA majoring in Information Systems. She has received her Bachelor's degree in Elementary/Special Education from Holy Family College, Philadelphia, PA. She is currently interested in data and object modeling, database implementation strategies, data warehousing, and data mining.