Incorporating Business Requirements and Constraints in Database Conceptual Models

Khaled M. Khan **Mahesha Kapurubandara** **Urvashi Chadha**

School of Computing and Information Technology
University of Western Sydney,
Locked bag 1797 South Penrith DC
NSW 1797 Australia
Email: {khaled, mahesha, urvashi}@cit.uws.edu.au

Abstract

Entity relationship (ER) approach is predominantly used for conceptual modelling of database systems in terms of entities and their relationships. The approach does not provide sufficient support for incorporating business constraints and their impact on the entity relationships, thus leaving a gap between the requirements elicitation and database implementation. This paper makes an attempt to bridge this gap by proposing a construct that would enable the system architects to illustrate business requirements and constraints at the conceptual model with minimal effort. To do this, we have proposed an approach called *attribute-oriented* business requirements and constraints (BRCs). We classify five different categories of attribute-oriented BRCs for binary relationships. Based on this approach, we propose a construct to enhance the modelling capabilities and expressiveness of the ER approach without introducing any conflict with the current features. Our main contribution in this paper is a new construct to specify the business rules and constraints along with the systems requirements in database conceptual modelling.

Keywords: database conceptual modelling, entity-relationship approach, business rules, attribute-oriented constraints

1 Motivation

From a systems analysis perspective, requirements and constraints could be classified into two major groups: *systems requirements and constraints (SRC)*; and *business requirements and constraints (BRC)*. *SRCs* usually define the functional aspects of a system such as what a system is supposed to achieve by down playing the business constraints. The traditional systems development methodologies focus primarily on functional requirements such as what a system is supposed to do. Although business constraints and requirements are taken under active consideration during the analysis of SRCs, these are unfortunately not specified in the artefacts along with the functional design layout of the system. A systems requirement for a database initially contains business entities, their attributes and the relationship topology among the entities. Systems analysts rarely model the 'business conditions' under which a system does certain things. Although analysts make the design decision based on business requirements and constraints, these information are not captured in the model in any form for a later usage. Most of the BRCs are not available for further consultation at the implementation stage.
BRCs on the other hand commonly express behavioural rules and constraints that govern SRC such as under which 'business conditions' a system achieves its functionalities. Some SRCs such as referential constraints, participation constraints, structural constraints and constraints on relationship types are well addressed in the existing features of some modelling tools like ER model (Chen 1976), Enhanced ER model (Elmasri, & Navathe 2001). ER modelling approach initially proposed by (Chen 1975) is used to model a database system in terms of entities, attributes and relationships. This model is sufficient to represent the basic semantics of the real world such as entities and their relationships. However, it has limited features to capture and express the business requirements and constraints which define the conditions and policies under which a relationship is governed. We believe that the BRCs which eventually dictate the formation of a relationship should be specified along with the conceptual model. Despite the importance of incorporating BRCs in the conceptual model, the issue has received less attention. Usually BRCs are either at the back of the analyst's mind or hidden away in notes and documents. These are often addressed and revised during the implementation stage of the database development.
To address the BRC issues, different approaches have been proposed by various researchers. One such approach uses object constraint language (OCL) (Coronato, Cinquegrani, & Pietro 2002) emphasising on constraints using a textual model and incorporating it at the implementation level. OCL provides a mechanism of incorporating the business rules only at the implementation level. Attempts have also been made to use OCL to express business constraints in Unified Modelling Language(UML)(Coronato et al. 2002), (Warmer, & Kleppe 1999). However, the approach does not address the issue of expressing BRCs at the analysis level. Substantial work has already been carried out to incorporate non-functional requirements (NFR) in conceptual models such as in (Chung, Nixon, & Yu 1994), (Coronato et al. 2002), (Cysneiros, Leite, & Neto 2001a), (Cysneiros, Leite, & Neto 2001b), (Cysneiros, & Leite 2001c) and (Khan 2003). The published reports focus basically on non-functional requirements such as confidentiality, security, reliability and so on.

Virtually a very few addresses the issue of incorporating business constraints in the database conceptual model at the analysis level. We need an approach that would enable designers to express business rules and constraints in one single artefact which could later be consulted during the implementation of the system. Considering the importance of incorporating BRCs in the conceptual model, we believe that there is a strong need to provide a unified model that would

ensure the propagation of the BRCs from analysis to implementation level. Integrating BRCs into conceptual models would be a major step towards the quality improvement of the database development.
We, in this paper, concentrate on incorporating BRCs into entity relationship diagrams typically used in database modelling. Our aim is to extend the existing ER modelling technique, and introduce a new construct that would express business requirements and constraints along with the conceptual model. We employ an analytical approach in our research to spell out various categories of business rules and constraints which are critical to the establishment of relationship between entities in database modelling. Our main contribution in this paper is a new construct to specify the BRCs along with the systems requirements in database conceptual modelling. The new construct is based on a notion of *attribute-oriented* BRCs. We classify this type of BRCs into five different categories. The proposed construct is also integrated with the existing features of the ER approach without introducing any conflict with ER diagrammatic notations. We demonstrate our ideas with some examples.
This paper is organised as follows. Section 2 highlights the motivation of this study, defines the new construct and continues to discuss its integration to the ER approach. Section 3 cites some examples to illustrate how the business requirements can be integrated into ER diagrams using our proposed constructs. Finally we close in section 4 with some conclusions.

2 Proposed framework

To tackle the problems mentioned in the proceeding section, we need to develop some rules and their associated notations to express BRCs around the main constructs of entities and relationships in ER approach. To address these we employ an analytical approach which is based on the followings:

1. *Defining rules for attribute-oriented BRCs* in determining relationship between entities in database modelling.

2. *Defining a new construct* along with appropriate parameters to capture business requirements and constraints.

3. *Integrating the new construct* in the existing ER diagram keeping the original principles of entities, their attributes, and relationships intact as suggested in Chen's model (Chen 1975).

2.1 Defining rules for attribute-oriented BRCs

A business requirement is a statement that defines or constrains some aspect of the business. It is intended to assert business structure or to control or influence the behavior of the business. A business requirement expresses specific constraints on the creation, updating, and removal of persistent data in an information system. For example, a purchase order cannot be made if the customer's credit rating is not high. Similarly, only research students are eligible for university scholarships. In the first example, the system goal is: *making a purchase order*. The corresponding BRC is: *credit rating must be high*. The functionality is determined by the BRC in this case. To model this in the current ER approach, we create two entities, one is called *Purchase order* and the another one is *Customer*. We make a relationship between these two entities based on the BRC. The actual BRC is not captured in the ER diagram at all. At the implementation level we again consult the BRC to implement the conceptual model.
A binary relationship in an ER approach is predominantly governed by the attributes of the participating entities. The 'business conditions' imposed by the attributes, in some way, determine the establishment of the relationship. These attributes may belong to either entities of the relationship type or the relationship itself. Our focus is specifically on the category of BRCs that determines whether a relationship between two entities would exist or not, depending on the value of the attributes. The BRCs that are very much dependent on the value of attributes are referred to as *attribute-oriented* BRCs in this paper. As we have seen in the previous example of customer and credit rating, the attribute *credit rating* determines the relationship between a customer and a purchase order. As the attributes determine the relationships between entities, this can be termed as a *control*. Thus the *control* specifies the constraints or BRCs related to a relationship. In this paper we restrict our focus only on BRCs for binary relationships. A much closer look at the ER approach pertaining to different business scenarios focusing specifically on relationships could lead us to the following conclusions: a binary relationship between two entities can be determined by at least one of the five following different categories of *attribute-oriented* BRCs:

- **Category (i) BRC**: *attributes of only one entity*;
- **Category (ii) BRC**: *attributes of both entities*;
- **Category (iii) BRC**: *attributes of the relationship itself*;
- **Category (iv) BRC**: *attributes of the relationship itself plus attributes of only one participating entities*; and
- **Category (v) BRC**: *attributes of the relationship itself plus at least one attribute of both participating entities.*

We discuss each of these with examples based on the assumption that the elicitation and formalisation of business requirements and constraints have already been completed.

2.1.1 Category (i). Relationship determined by attributes of only one entity.

To illustrate this category of attribute-oriented BRC, we discuss a simple example of an accommodation allocation database. Assume a BRC specification of a database segment is described in natural language as: 'only *full time* students are allowed to lease a campus accommodation.' The italic text within the quotes indicates the value of an attribute.
In ER approach, all we could model is the systems requirements, but we cannot specify the BRC along with the SRC. The ER diagram for this database segment is shown in Figure 1. To keep the ER diagram simple, cardinalities, connectivities, and participation constraints have been omitted in our illustrations throughout this paper.
Assume that the possible values of the attribute *studyMode* could be either *part-time* or *full-time.* From the specification, we understand that the attribute *studyMode* determines the existence of a relationship between a student instance and an accommodation instance. If the value of the attribute *studyMode* of a student entity is *part time*, there would not be any relationship between this student

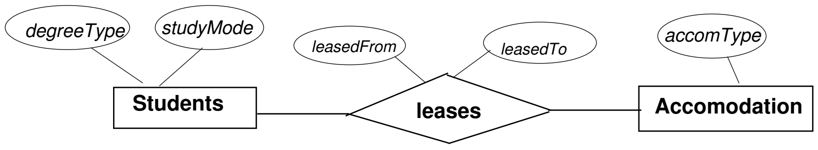


Figure 1: ER diagram for the student accommodation database.

instance and an accommodation instance. In this particular BRC, the value of the attribute $studyMode$ determines whether there would be any relationship instance between a student and an accommodation. Note that the attribute of the entity accommodation does not influence this relationship –at least the BRC specification does not dictate so. In its current form of the ER diagram in Figure 1, it does not show the enforcement of the BRC at all. One could argue that this type of constraints could be tackled with the subtyping of the entity in the existing ER approach. In that case, it leads to two separate relationship types. The business requirements could not be specified in that case either. The BRCs that triggered the subtyping of the entity is lost. It also may lead to creation of multiple subtypes of an entity in a relationship type. We believe that the business rules should be well specified at the conceptual level to capture the rationale for making a relationship.

2.1.2 Category (ii). Relationship determined by attributes of both entities.

In this category of attribute-oriented BRC, at least one attribute of both participating entities dictates the relationship. In other words, this category consists of at least two BRCs –one for each of the participating entities. More than two BRCs could be included in such category for a particular relationship. However, the condition is that each of the participating entities must enforce at least one BRC.
We redefine the systems requirement of the previous database segment, and describe a different BRC for the database as: 'Students with degree type *research* must lease *non-shared* accommodation; whereas *non-research* students can lease only *shared* accommodation.' This particular relationship purely depends on two attributes: the degree type of the student and the type of the accommodation. To establish a relationship instance of *lease*, two BRCs belonging to both entities must be enforced. In ER approach, the systems requirement for this database segment could be modelled by introducing two separate relationship types, however, the BRCs could not be expressed along with the model in the current form of ER approach.

2.1.3 Category (iii). Relationship determined by the attribute of the relationship itself.

Further analysis show that in some cases attributes of a relationship itself determine the relationship. Consider another example of a BRC: 'An accommodation could only be leased for a period of at least *one year*.' This specific relationship is determined by the rental period that is derived from descriptive attributes. Descriptive attributes are used to record information about the relationship rather than about any one of the participating entities. The ER diagram in Figure 1 represents this database segment.
In this category, the relationship *leases* is determined by the attributes of the relationship itself: $leasedFrom$ and $leasedTo$. This category could be handled and represented by domain constraint in the current practice. However, this could only be achieved at the implementation level of the database design. At the conceptual level we could unlikely specify the domain constraints of the attributes.

2.1.4 Category (iv). Attributes of the relationship plus attributes of only one entity.

This category of BRC comprises attributes of the relationship itself and attribute of only one participating entity. In other words, it is the combination of category (i) and (iii) BRCs. Consider this BRC: 'Students with degree type *research* must lease an accommodation for at least *nine months*'. The ER diagram for the database segment is shown in Figure 1. The BRCs remain unspecified in this diagram. The attribute $degreeType$ of the entity student and the attributes $leasedFrom$ and $leasedTo$ of the relationship itself determine the relationship between the student and accommodation. This could be tackled by defining the domain constraints of the attribute at the implementation stage of the database design.

2.1.5 Category (v). Attributes of the relationship plus attributes of both participating entities.

This is the most restrictive category of BRC. It consists of at least one BRC enforced by each of the participating entities and the BRC imposed by the relationship itself. In other words, it is a combination of category (ii) and (iii). Let us consider the example of this category: 'Students with degree type *research* must lease *non-shared* type accommodation for at least *six months*.' Each entity has one BRC and the relationship itself has one BRC. The ER diagram of this database segment is shown in Figure 1. The diagram only expresses the systems requirements, but it does not express the BRCs at all. Again, this category could be addressed by defining domain constraints of the attributes at the implementation level. In that case, unless these rules are captured at the conceptual model, these BRCs would not be available at the later stage for further consultation.
What we see here that five different BRCs are modelled with a single ER diagram which simply cannot distinguish one BRC from another. The ER diagram in Figure 1 fails to express different attribute oriented BRCs which are vital to create the relationship instances between entities. At the implementation stage, it would be difficult to identify various BRCs from the diagram cited in Figure 1.
The first three categories described above are the primitive categories of BRCs, whereas, the last two categories are the combination of the primitive categories. The category (v) is the most restrictive one, whereas, category (i) is the most simplest one.

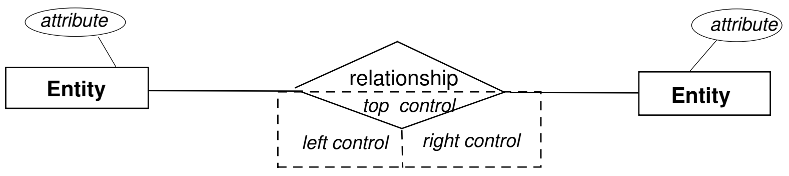


Figure 2: The control construct to accommodate BRCs in ER diagram

2.2 Defining the new construct for BRCs

Now the question is: where to specify these different categories of business rules and constraints in a database conceptual model and in which form. In the ER context, there are two possibilities –tag them either with the entity of concern or with the relationship related to the BRC. If we specify BRC with the entity, it will influence all relationships established with that particular entity which is not desirable at all. It will hardly achieve our objectives if we specify BRC with the entities. It is the relationship type in which BRC needs to be specified because the BRC of the participating entities essentially determines a relationship between two entities.
To accommodate the five different categories of attribute-oriented BRCs that control the relationship instances in the ER approach, we propose the following construct for the first three categories.
A binary relationship type can have three controls namely *left, right* and *top*. Thus the *left control* of a relationship specifies the BRC governed by the attribute of the entity on the left side of a binary relationship. *Right control* can be defined in the similar fashion to represent the BRC imposed by the attribute of the right side entity of the relationship. The *top control* includes BRC directly related to the attributes of the relationship itself. Each of the control includes various information such as the reference number of the BRC in relation to the entire database schema, priority level in a relationship if more than one control is used, and the rule expression of the BRC in structured format. Following is the standard structure for the proposed control construct.

$Control :: left, right, top$
$\{ \quad BR\,ref:$
$Priority\ level:$
$Rule\ expression:\}$

The field $BR\,ref$ is the reference number that points to the BRCs listed in the requirements document of the entire database schema. The *Priority level* field captures the order of the three controls in a relationship if more than one control exists. The field *Rule expression* stores an expression derived from a BRC statement expressed in a structured format. Due to space limitation we are only using the field *Rule expression* in our examples throughout the paper. The format uses following standard operators to evaluate a BRC expression:
$>$ (greater than), $>=$ (greater than or equal to), $<=$ (less than or equal to), $<$ (less than), $+$ (addition), $-$ (substraction), $*$ (multiplication), $/$ (division), $<>$ (not equal to), $=$ (equal to).
The standard operands are: attribute names of the entity types or relationship type, and constant values. The string or character constants are used in single quotes. The integration of this construct in graphical form into the ER notational framework is proposed in the next section.

2.3 Integrating the new construct into ER diagram

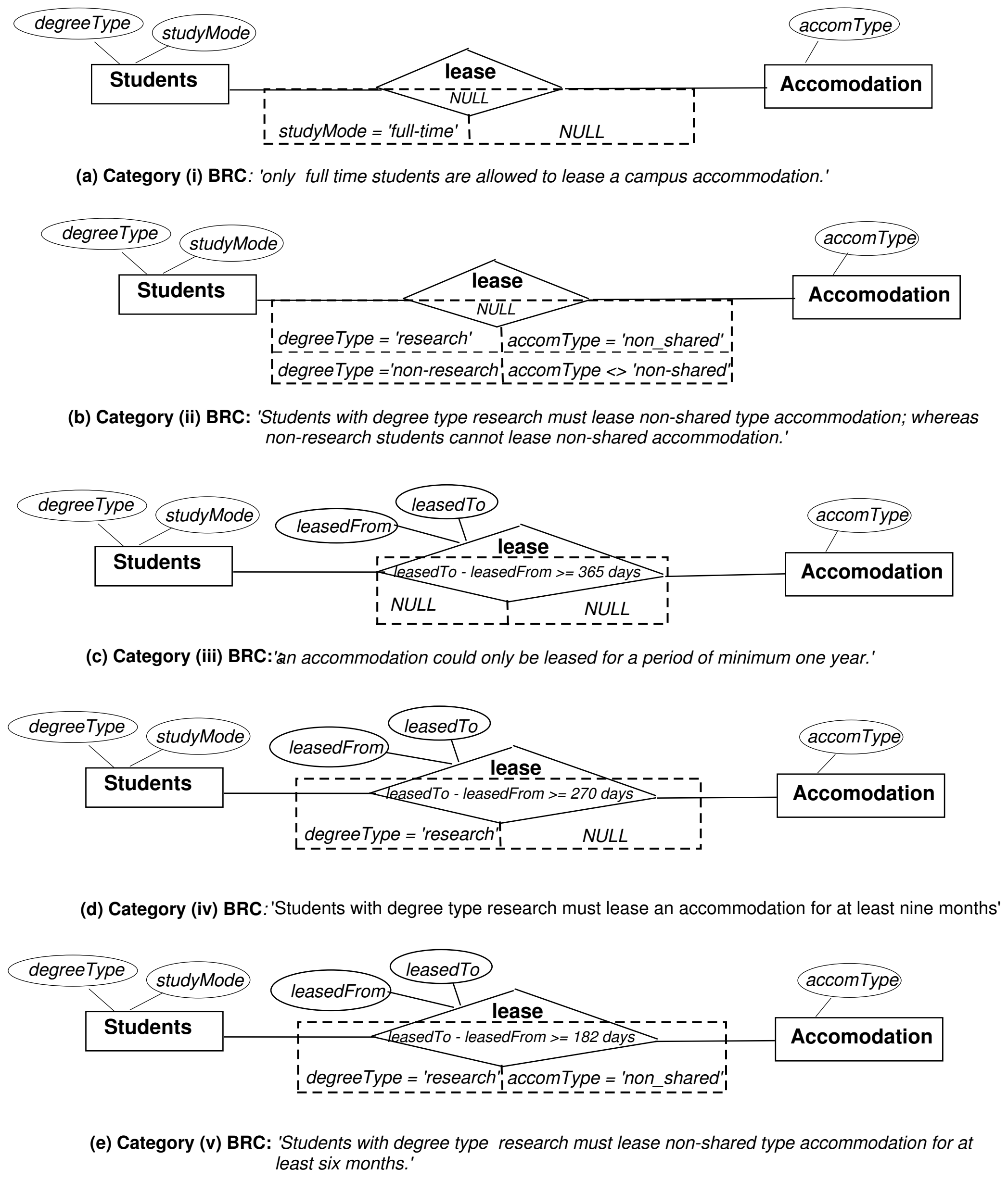
Representing the BRCs imposed by the relevant entities or relationship would certainly enhance the comprehensibility of the entire ER diagram. The ER approach highlights the fact that the categories of BRCs discussed in this paper would be more appropriate if linked with the relationship type rather than with the entities in the ER diagram. The integration of our proposed construct with the relationship symbol of ER diagram makes the database more enhanced. In addition to the current features of the ER approach, we propose a rectangle attached to the diamond symbol for the relationship type to represent our control construct for BRCs. It has a vertical line in the middle to separate the left and right controls, and the overlapped portion between the diamond and rectangle symbols are used to show the top control.
Figure 2 cites such a layout. Using the proposed construct, we can now model all BRCs discussed in the previous subsection. Five BRCs in five different categories are shown in Figure 3. The diagrams are self explanatory.

3 Applicability of the proposed construct

In this section we illustrate the applicability of the model with an example. The example is about a car rental company. Clients can rent cars from the company's different branches. To make a booking, the client must deposit 25 percent of the total amount for the rent and must have a valid driving licence. The car can only be picked up within 24 hours of the rental start date. A car cannot be rented if its odometer exceeds 30000 km.
In this paper we restrict our study to three categories of BRCs given below. BRCs for the car rental example are categorised accordingly:

- **Category (i) BRC**: Relationship in where attributes of one entity control the relationship.
  BRC 1: A car cannot be rented if its odometer exceeds 30000 km
- **Category (ii) BRC**: Relationship in where attributes of both participating entities in a binary relationship control the relationship.
  BRC 2: A customer must have a valid driver's licence and a 25% deposit to rent a car.
- **Category (iii) BRC**: Relationship in where a descriptive attribute of the relationship itself determines the relationship
  BRC 3: The car must be picked up within 24 hours of the rental start date.

Figure 4 depicts the three BRCs in three different categories defined for the car rental database system. Figure 4 shows that two BRCs have been specified in two controls for the relationship 'booked for' -one BRC is imposed by the entity *Client* and the other one is by the entity *Rental*. The top control has a


degreeType
studyMode
Students
lease
NULL
studyMode = 'full-time'
NULL
accomType
Accomodation
degreeType = 'research'
accomType = 'non_shared'
degreeType ='non-research
accomType <> 'non-shared'
leasedFrom
leasedTo
leasedTo - leasedFrom >= 365 days
leasedTo - leasedFrom >= 270 days
leasedTo - leasedFrom >= 182 days

(a) Category (i) BRC: *'only full time students are allowed to lease a campus accommodation.'*

(b) Category (ii) BRC: *'Students with degree type research must lease non-shared type accommodation; whereas non-research students cannot lease non-shared accommodation.'*

(c) Category (iii) BRC: *'an accommodation could only be leased for a period of minimum one year.'*

(d) Category (iv) BRC: 'Students with degree type research must lease an accommodation for at least nine months'

(e) Category (v) BRC: *'Students with degree type research must lease non-shared type accommodation for at least six months.'*

Figure 3: All five categories of BRCs using the Control Construct

value $NULL$. The relationship type 'assigned to' between the $Rental$ and Car entities are associated with one BRC, which is enforced by the attribute owned by the entity Car. The category (iii) BRC has been specified in the relationship type 'picked for', other remaining controls are $NULL$. In the diagram we did not specify the fields BRC ref, and the priority level to keep this diagram simple.

The example cited in Figure 4 clearly illustrates that the new construct can be used adequately to incorporate the attribute oriented BRCs into the ER approach. This brief example demonstrates that the BRCs could be adequately integrated on to the ER diagram. The graphical representation presented here is overly simplistic,and we aim to formalise this approach in our future research.

Database designers can be benefitted with this construct in several ways. *Firstly*, they can specify various attribute-oriented business rules and constraints along with the systems requirements with this construct in ER approach. *Secondly*, any potential conflict between BRCs, and between BRCs and SRCs could easily be tracked down at an early analysis stage. *Thirdly*, implementation of the SRCs and BRCs could be more consistent with the design artefact. *Fourthly*, all stakeholders including users could understand the BRCs more clearly at the conceptual level, hence, communication between all stakeholders are enhanced. *Finally*, the BRCs that drive the design decision at the analysis level could be revisited and re-consulted with at the implementation level if any confusion occurs. BCRs are readily available along with the conceptual model in our approach.

In this paper we have used Chen's ER model (Chen 1975) to illustrate our discussions. The graphical notation of the construct is based on the diamond in the Chen's model. A similar representation can be adopted with other models such as Crow's foot model. We acknowledge that more research work needs to be done in this area. Several open issues still remain unsolved such as: (i) how to propagate the defined BRC in conceptual model to the database implementation; (ii) How to handle BRCs for ternary relationship type; (iii) how to best address the graphical notations to accommodate BRCs in ER and other approaches, and (iv) how to formalise BRCs as data model.

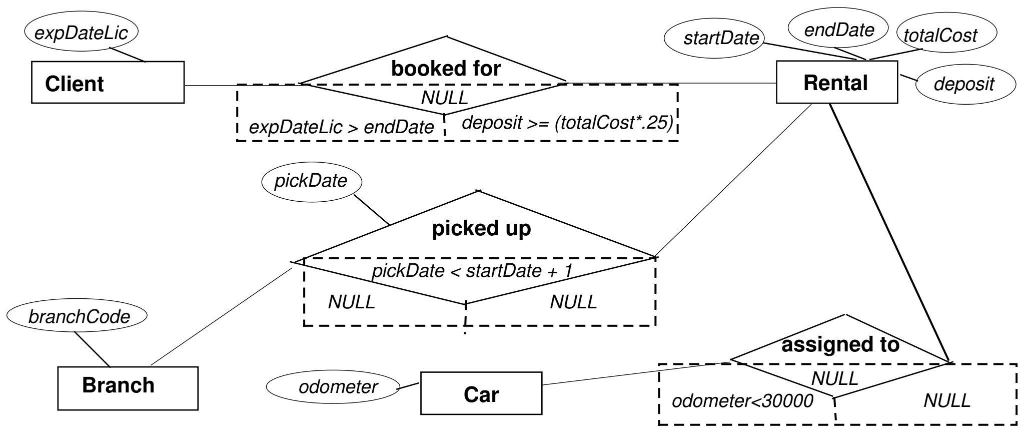


Figure 4: Illustration of BRCs using ER diagram

4 Conclusion

The construct proposed in this paper is capable of enhancing the modelling capabilities and expressiveness of entity relationship model to incorporate attribute oriented business requirements and constraints. It utilises a simple construct as a vehicle to help the user to incorporate a class of business rules namely, attribute oriented business requirements and constraints, in the early stages of the analysis. It allows the user a great latitude in specifying the business rules much earlier in the database design process.
The proposed construct shows how to capture business rules by embedding them in the ER approach. In order to do that, we have extended the existing ER diagram with a new construct. We have categorised attribute-oriented business rules and constraints related to relationship type in a conceptual database modelling context. Although the proposed construct is easy to use, its usefulness is not negligible. The main aim of this paper was not to focus on complex mechanism, but to suggest a simplistic way to satisfy an urging requirement. The work presented in this paper is in an initial stage of our research. Future work will aim to increase its credibility through formalising the approach and experimenting it on a large case study.

References

Chen, P. (1975), The entity relationship model: towards a unified view of data. *In* 'ACM Proceedings of the International Conference on Very Large Databases, Farmingham, Mass., September, pp. 22-24.

Chen, P. (1976), The entity-relationship model - toward a unified view of data. *ACM Transactions on Database Systems*, 1(1), 9–36.

Chung, L., Nixon, B., & Yu, E. (1994), Using Quality Requirements to Drive Software Development. *In* 'Workshop on Research issues in the Intersection between Software Engineering and Artificial Intelligent.' Sorrento, Italy, May 16-17 1994.

Coronato, A., Cinquegrani, T. & Pietro, G.(2002), Adding business rules and constraints in component based applications. *In* 'Proceedings of Symposium on Distributed Objects and Applications', Springer-Verlag, pp. 948–964.

Cysneiros,L., Leite, J., & Neto, J. (2001a), A framework for integrating non-functional requirements into conceptual models. *Requirements Engineering Journal*, 6(2), 97–115. Springer-Verlag.

Cysneiros, L., Leite, J., & Neto, J. (2001b), Using UML to reflect non-functional requirements. *In* 'Proceedings of the 11th Annual IBM Centers for Advanced Studies Conference (CASCON)', November.

Cysneiros, L., & Leite, J. (2001c), Using the language extended lexicon to support non-functional requirements elicitation. *In* 'Proceedings of the 5th Workshop on Requirements Engineering' Buenos Aires, November.

Elmasri, L., & Navathe, S. (2001), *Fundamentals of Database Systems.* Benjamin-Cummings Inc.

Greenspan, R., Feblowitz, M., & Wild, C. (1997), A decision making methodology in support of the business rules lifecycle. *In* 'Proceedings of the International symposium on Requirements Engineering', IEEE Computer Society, pp. 236–246.

Khan, K.(2003), Integrating security properties with systems design artefacts. *In* 'Proceedings of the Conference on Information Systems Development.'Melbourne, Kluwer Academic Publishers.

Warmer, J., & Kleppe, A. (1999), *The object constraint language: precise modeling with UML.* Addison-Wesley publishing, Reading, Massachusetts.