Recent Enhancements to ORM

Terry Halpin[1] and Matthew Curland[2]

[1] INTI International University, Malaysia
[2] ORM Solutions, USA
{t.halpin,mcurland}@live.com

Abstract. Fact-oriented modeling approaches such as Object-Role Modeling (ORM) employ rich graphical notations for capturing business constraints, and validate their data models with domain experts by verbalizing the models in natural language, and by populating the relevant fact types with concrete examples. This paper discusses several recent enhancements to ORM, including the following: further constraints on supertype link roles and their relevance to restricted mandatory role constraints; inclusive-or constraints on roles hosted by different types; refinements to the concept of independent object types; additional kinds of reference schemes and associated uniqueness constraints; and verbalization of further constraint cases involving subtyping, additional reference scheme patterns, uniqueness and frequency constraints involving unaries, and external uniqueness constraints involving *n*-ary fact types. The paper also includes some discussion of how these enhancements have been supported, or are soon to be supported, in the Natural ORM Architect (NORMA) tool.

1 Introduction

Fact-oriented modeling approaches formulate data models in terms of fact types whose fact instances are expressed in natural sentences easily understood by business users of the information systems for which the data models are designed. A fact type corresponds to a set of typed predicates of arity one (e.g. Person smokes), two (e.g. Person was born on Date), or higher (e.g. Person hired Person on Date). Since all facts are expressed as relationships, this differs from Entity relationship (ER) modeling [2] and class diagramming within the Unified Modeling Language (UML) [18], which encode some facts in attributes (e.g. Person.isSmoker, Person.birthdate). Fact-orientation's attribute-free nature facilitates validation with domain experts by verbalizing the model's fact types, constraints and derivation rules in natural sentences, and populating the fact types with concrete examples. It also promotes semantic stability (e.g. no remodeling is needed to talk about an attribute). Moreover, fact-orientation's graphical constraint notation for data modeling is much richer than that of industrial ER or UML.

The family of fact-oriented modeling approaches include various dialects such as Object-Role Modeling (ORM), Natural-language Information Analysis Method (NIAM) [20], Fully-Communication Oriented Information Modeling (FCO-IM) [1] and the Predicator Set Model (PSM) [16]. This paper focuses on recent enhancements to second generation ORM (ORM 2) [6]. Overviews of ORM may be found in [8, 9],

Y.T. Demey and H. Panetto (Eds.): OTM 2013 Workshops, LNCS 8186, pp. 467–476, 2013.
© Springer-Verlag Berlin Heidelberg 2013

and a detailed coverage in [15]. The rest of this paper is structured as follows. Section 2 considers constraints on supertype link roles and their relevance to restricted mandatory role constraints. Section 3 considers inclusive-or constraints on roles hosted by different types. Section 4 refines the concept of independent object types. Section 5 discusses additional kinds of uniqueness and frequency constraints, and their verbalization support in the Natural ORM Architect (NORMA) tool [3]. Section 6 concludes by summarizing the main contributions, identifying future research areas, and listing the references cited.

2 Constraints on Supertype Link Roles

In Figure 1(a) the role of using a company car is optional for employees, but mandatory for executives. The restricted mandatory role constraint is displayed as a footnoted textual constraint. In ORM, mandatory role and or-constraints over subtyping links are understood to apply to *supertype link roles* (i.e. the supertype roles of the implied, instance-level, identity relationships linking subtype and supertype instances). For example, the exclusive-or constraint in Figure 1(b) applies to the supertype roles of the subtyping identity fact types shown with dashed lines in Figure 1(c). We recently extended ORM and NORMA to allow other kinds of constraint to apply to supertype link roles. Among other things, this enables restricted mandatory role constraints to be expressed graphically, as shown by the subset constraint in Figure1(d), which is interpreted as in Figure 1(e). In NORMA we automatically verbalize this constraint as follows: **If some** Employee is **some** Executive **then that** Employee uses **some** CompanyCar.

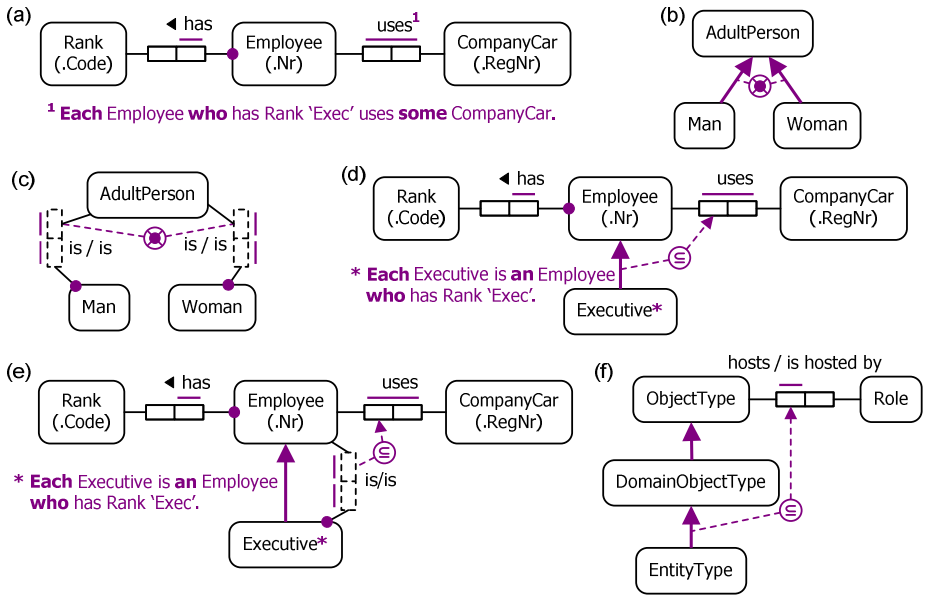

Fig. 1. Using subset constraints on subtyping link roles for restricted mandatory constraints

Another example is provided by Figure 1(f), which is extracted from a recent draft of a common metamodel for fact-based modeling being developed by the fact-based modeling working group. Here the subtypes are asserted rather than being derived, and the subset constraint ensures that each entity type hosts some role.

3 Inclusive-or Constraints on Roles Hosted by Different Types

Typically, an inclusive-or constraint applies to two more roles hosted by the same object type. Recently, we extended ORM to allow such constraints to apply to roles hosted by different types, so long as those types are compatible (and hence may share instances). For such cases, we define the inclusive-or constraint thus: for each state of the model, *each instance in the population of the minimal common supertype of the types hosting the constrained role must play at least one of the constrained roles*. Typically, the minimal common supertype will be the root of the subtype graph.

A simple example is shown in Figure 2(a), where one constrained role is hosted by Employee and the other by Manager. Employee and Manager are compatible since Manager is a subtype of Employee. Here, the inclusive-or constraint means that each employee reports to some manager, or is a manager who manages some employee. For a one-person company, this is satisfiable only if we allow self-management.

Verbalization patterns for inclusive-or constraints are presented in our earlier work [4], and we recently extended these patterns by including the implied link fact type (see Figure 2(b)) as part of the verbalization path. Assuming that Employee is declared personal, our current NORMA verbalization for this constraint is as shown below. In the longer term, we plan to optimize the verbalization simply to: **Each** Employee reports to **some** Manager **or** is **some** Manager **who** manages **some** Employee:

For each Employee
that Employee reports to **some** Manager
or some Manager **who** is **that** Employee manages **some** Employee.

Figure 2(c) includes a reading for the reporting fact type in one direction only. For this case, NORMA currently generates the following verbalization:

For each Employee,
some Manager manages **that** Employee
or some Manager **who** is **that** Employee manages **some** Employee.

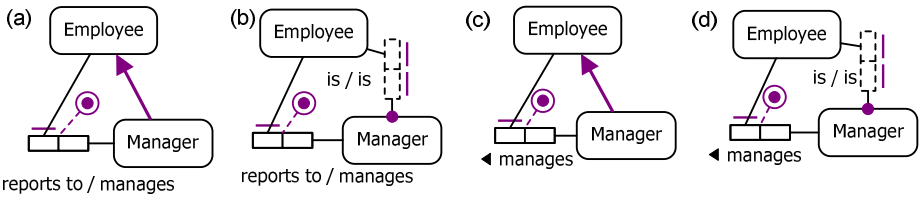

Fig. 2. An inclusive-or constraint on roles hosted by different types

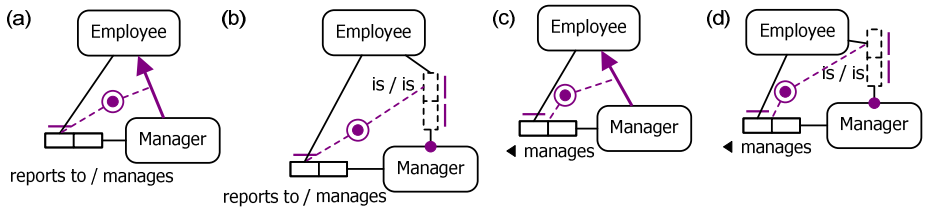

Fig. 3. A multi-type inclusive-or constraint on a supertype link role

Figure 3(a) shows an example of a more realistic inclusive-or constraint involving multiple types. In this case, one of the constrained roles is a supertype link role, as shown explicitly in Figure 3(b). Our previous verbalization patterns may be applied to this situation simply by using the link predicate reading like a normal predicate in the constraint path. Figure 3(a) has forward and inverse predicate readings available, so the inclusive-or constraint may be verbalized thus:

Each Employee reports to **some** Manager **or** is **some** Manager.

Figure 3(c) includes a reading for the reporting fact type in one direction only, so the inclusive-or constraint is verbalized as follows:

For each Employee,
some Manager manages **that** Employee
or that Employee is **some** Manager.

For all cases, if the constraint has deontic rather than alethic modality [7], an appropriate deontic operator is prepended (e.g. "**It is obligatory that**" or "**It is forbidden that**" for positive and negative forms of the verbalization respectively) [4].

4 Independent Object Types Revisited

The notion of independent object types was introduced to ORM in an earlier paper [5], where we originally described them as "lazy object types". Later, we defined an independent object type to be a primitive object type whose fact roles (if any) are collectively optional [15, p. 219]. A primitive object type is an object type that is not a subtype, and "fact role" in this context means a role in an elementary fact type. The basic idea is that an independent type can contain instances that simply exist, independently of playing in any other *elementary* facts. We now use the term "atomic fact" to include elementary facts (which simply apply a predicate to one or more objects), and *existential* facts (which are used to simply assert the existence of, or refer to, an entity instance). Existential fact types are also known as reference types.

Recently, we refined the concept of independent object types to clearly distinguish the kinds of types and roles involved, and to exclude from consideration facts that can be derived. Without the latter exclusion, situations can arise where full software tool support for determining the consistency of independent type settings could require extremely complex computations over derivation paths.

To assist in formulating our new definition for independent object types, some terminology is first provided. Our recent formalization of ORM [10] partitioned ORM object types into entity types (e.g. Employee, Country), semantic value types (e.g. EmployeeNr, CountryCode), and datatypes (e.g. integer, string). Semantic value types are now called *domain value types* (or simply *value types*).

A *referential role* (or existential role) *of an entity type* is a role that it hosts in one of its existential fact types, i.e. a fact type in its preferred reference scheme. Note that implied, objectification link fact types [15, p. 442] that are used in the reference scheme for the objectified entity type are existential fact types for that entity type.

A *non-referential role of an object type* is a role that is hosted by that object type but is not referential for that object type. All roles hosted by a value type are non-referential because value types have no reference scheme. A *non-referential role* is a role that is not a referential role of any object type. For example, Figure 4 displays with gold fill the referential roles for (a) Country, (b) Room, and (c) Enrolment. All roles displayed with white fill are non-referential.

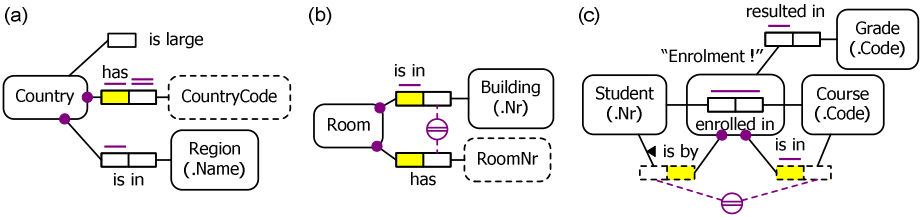

Fig. 4. Referential roles of Country, Room and Enrolment are displayed with gold fill

Implied identity link roles of subtypes are non-referential, even if the subtyping arrow connection is displayed with a solid line (showing an inheritance path to the reference scheme) [15, p. 519].For example, the identity link roles played by the subtypes in the previous figures, such as Figure 1(e), are non-referential.

An *assertable role* is a role in a fact type that is either asserted or semiderived. Hence, if an object type hosts an assertable role, *some* of its instances may be asserted to play that role. If the fact type is asserted, all instances in the object's type's population may be asserted to play the role. Roles in derived fact types are not assertable.

We now refine the definition of independent object type as follows. An *independent object type* is a primitive, domain object type that may include instances that are simply asserted to exist, without playing any assertable, non-referential roles (i.e. instances of it may exist independently of playing any such role). A *non-independent object type* is an object type that is not independent. In this case, the disjunction of its assertable, non-referential roles is non-empty and is implied to be mandatory.

For example, Enrolment in Figure 4(c) is independent (as indicated by the "!" appended to its name), allowing some enrolments to be recorded before the resulting grade of the student in that course is known. The domain object type requirement ensures that no datatype can be an independent object type. Moreover, an object type that hosts no assertable, non-referential role is implicitly independent. For example, if Room in Figure 4(b) hosts no other role in the conceptual schema, then it is implicitly independent.

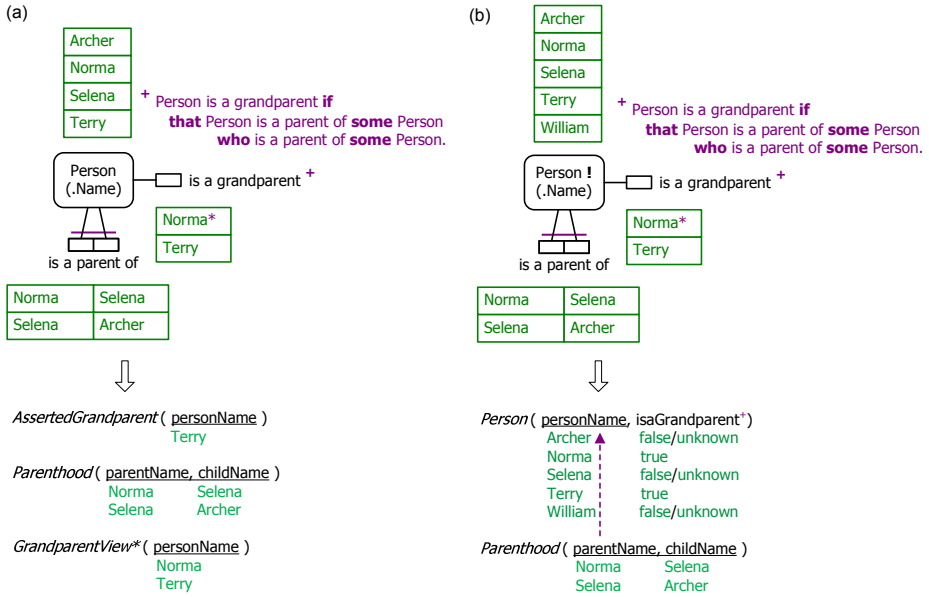

Fig. 5. Influence of independence status on relational mapping of a semiderived fact type

The different semantics for independence status involving semiderived fact types is illustrated by the different relational mapping results in Figure 5 (other mapping choices are possible). In the ORM schema of Figure 5(a), Person is non-independent, so each recorded person must be asserted to play at least one of the assertable, non-referential roles (i.e. the role of being a parent, the role of being a child, or the role of being a grandparent). If desired, one may add a view for Person as the union of the columns in the two base tables.

In the ORM schema of Figure 5(b), Person is independent, so a person may be recorded without being recorded to play any assertable, non-referential role (e.g. as illustrated by William in the sample data). The semiderived isaGrandparent column may be implemented by setting its default to false or unknown depending on whether closed world or open world semantics respectively is assumed for the semiderived fact type, and by using a trigger or stored procedure for derived grandparenthood. Alternatively, one may replace the isaGrandparent column by an asserted isAssertedGrandparent column, and include a derived GrandparentView. The key difference is that for independent types the relational schema must enable recording of instances that play no other roles.

5 Additional Uniqueness and Frequency Constraints

In ORM, an external uniqueness constraint spans two or more roles and is displayed as a circled bar connected to the constrained roles. If the constraint underlies the preferred reference scheme for an entity type at the other end of the relationships, the roles at the other end must be mandatory, and a double bar is used, as in Figure 6(a).

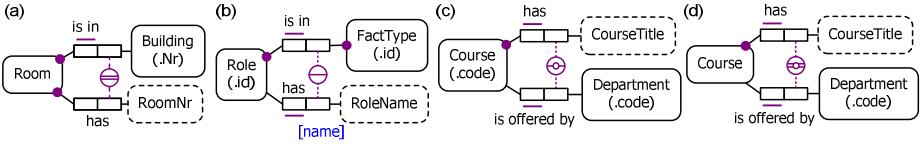

Fig. 6. External uniqueness constraints with (a), (b) inner or (c), (d) outer join semantics

The external uniqueness constraint in the ORM metamodel fragment shown in Figure 6(b) ensures that role names within a given fact type must be distinct. This constraint has *inner join semantics* so is equivalent to an internal uniqueness constraint on the relational inner join of the binary fact types: *Role*(roleId, factType, roleName). So in general a fact type may have many unnamed roles, but any named role can be identified by the combination of its name and its fact type.

External uniqueness constraints with inner join semantics can be used for the primary reference scheme of an entity type only if the coroles hosted by that entity type in the constrained relationships are mandatory, as in Figure 6(a) but not Figure 6(b).

The external uniqueness constraints in Figures 6(c) and 6(d) are depicted with an *inner "o" through the uniqueness bar*. We recently added this graphical notation to indicate that the uniqueness constraint has *outer join semantics, with the added proviso that nulls produced in the outer join are treated as actual values*. Hence each course may be referenced by exactly one of the following patterns: the combination of its course title and its department (where the course is offered by a department); its course title (where the course is offered by no department). External uniqueness constraints of this nature are verbalized in two sentences, one for the inner join semantics and one for the additional semantics of the outer join. The external uniqueness constraint in Figure 6(c) verbalizes thus:

For each CourseTitle **and** Department,
  at most one Course has **that** CourseTitle **and** is offered by **that** Department.
For each CourseTitle,
  at most one Course has **that** CourseTitle **and** is offered by **no** Department.

External uniqueness constraints with outer join semantics involve at least one optional corole but can be used for primary reference, as shown by the double uniqueness bar in Figure 6(d). If there are multiple optional roles, the order of outer joins is relevant, and is determined by the role selection order of the uniqueness constraint. The schemas in Figures 6(c) and (d) exemplify disjunctive reference, a capability introduced to ORM in [16], using a now outmoded notation. For a comparative review of reference schemes in ORM, Barker ER, UML, the Web Ontology Language (OWL) [19] and relational databases, see [11].

Though exceedingly rare in practice, it is possible to have an external uniqueness constraint over roles projected from a join path involving multiple joins some of which have inner join semantics and some of which have outer join semantics. For such mixed join semantics cases, we use "o" with a dashed line for the uniqueness bar, as in Figure 7(a), which also depicts the weakest constraint pattern allowed for a reference. The join types (inner or outer) for specific roles are not depicted graphically, but in an extended version of the NORMA tool may be specified in its join path editor, and then automatically verbalized.

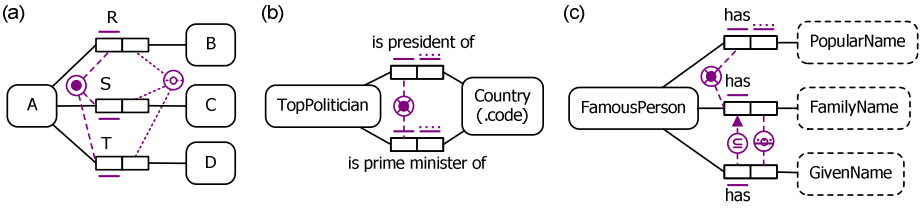

Fig. 7. Further varieties of uniqueness constraints

The schemas in Figures 7(b) and (c) show a graphical notation we recently introduced to deal with other reference cases found in practice. A *uniqueness constraint displayed with both a solid and a dotted bar* may be used to reference just some instances of the relevant entity type. A disjunctive reference scheme for the entity type may then be provided by two or more *partial, preferred reference schemes*. In Figure 7(b), some top politicians may be identified by being the president of a specified country, while other top politicians are identified by being prime ministers of a specified country. For example, the reference scheme in Figure 7(b) verbalizes as follows.

For each Country,
  at most one TopPolitician is president of **that** Country, **and**
  at most one TopPolitician is prime minister of **that** Country.
These associations with Country **provide the preferred identification scheme for** TopPolitician.

In Figure 7(c), some famous persons may be identified simply by their popular name (e.g. 'Confucius'), some others by their family name (e.g. 'Einstein'), and the rest by combining their family name and a given name (e.g. 'Marie Curie', 'Pierre Curie'). Disjunctive reference may be implemented directly in SQL by a table with no primary key, or indirectly using a simple identifier that is auto-generated. In some cases, a simple character string identifier can be derived by concatenating the reference components with suitable separator characters and recasting types as needed.

Although *external uniqueness and frequency constraints* typically apply to roles in binary fact types, we now support their *use with n-ary and unary fact types*. We have space here to discuss just a few examples. In Figure 8(a) the external uniqueness constraint restricts roles from binary and ternary and ternary fact types. This is equivalent to an internal uniqueness constraint on the corresponding roles in the inner join of the two fact types, and verbalizes as follows: **For each** Performer **and** Date, **there is at most one** Concert **such that that** Concert features **that** Performer in **some** Position **and** is on **that** Date.

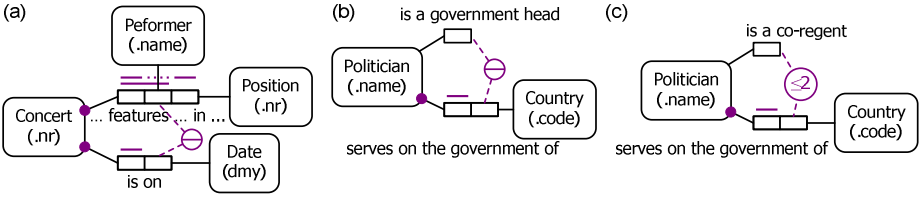

Fig. 8. Further cases of external uniqueness and frequency constraints

The external uniqueness constraint in Figure 8(b) is a *unique-where-true constraint*, where the uniqueness applies only to those politicians who are government heads (i.e. where it is true that they play the unary predicate). This kind of constraint was introduced to ORM in [11], using a now outmoded notation. We now use the usual constraint symbol for external uniqueness, and have improved the automated verbalization. For this example, the unique-where-true constraint verbalizes thus:

For each Country,
  at most one Politician is a government head
  and serves on the government of **that** Country.

Recently, we extended *external frequency constraints* to cater for such *where-true* cases. For example, NORMA verbalizes the frequency constraint in Figure 8(c) as follows:

For each Country,
  there are at most 2 **instances of** Politician **such that**
    that Politician is a co-regent
    and serves on the government of **that** Country.

6 Conclusion

This paper has provided a concise treatment of several of our recent enhancements to ORM, including: further constraints on supertype link roles and their relevance to restricted mandatory role constraints; inclusive-or constraints on roles hosted by different types; refinements to the concept of independent object types; additional kinds of reference schemes and associated uniqueness constraints; and verbalization of further constraint cases involving subtyping, additional reference scheme patterns, external uniqueness and frequency constraints involving unaries or *n*-ary fact types. It should be noted that these extensions have been added to enable modeling of practical cases that we have met in industrial modeling. Business domains can be complex, and models that reflect them faithfully need to provide the relevant expressive power.

We have implemented in NORMA all of the work described in this paper, with the sole exception of the additional flavors of external uniqueness constraints relating to disjunctive reference, which we plan to soon support. Other research plans in this area include extending NORMA's relational mapping support for such advanced cases, and refining the constraint verbalization. For example, adding pluralization support would enable more natural rendering of many constraints (e.g. compare "**at most** 2 **instances of** Politican" with "**at most** 2 Politicians". Although some research efforts have been made to provide verbalization of ORM models in languages other than English, much more work needs to be done in this regard.

References

1. Bakema, G., Zwart, J., van der Lek, H.: Fully Communication Oriented Information Modelling. Ten Hagen Stam (2000)
2. Chen, P.P.: The entity-relationship model—towards a unified view of data. ACM Transactions on Database Systems 1(1), 9–36 (1976), http://csc.lsu.edu/news/erd.pdf

3. Curland, M., Halpin, T.: The NORMA Software Tool for ORM 2. In: Soffer, P., Proper, E. (eds.) CAiSE Forum 2010. LNBIP, vol. 72, pp. 190–204. Springer, Heidelberg (2011)
4. Curland, M., Halpin, T.: Enhanced Verbalization of ORM Models. In: Herrero, P., Panetto, H., Meersman, R., Dillon, T. (eds.) OTM-WS 2012. LNCS, vol. 7567, pp. 399–408. Springer, Heidelberg (2012)
5. Halpin, T.: What is an elementary fact? In: Nijssen, G.M., Sharp, J. (eds.) Proceedings of First NIAM-ISDM Conference. Utrecht. (1993)
6. Halpin, T.: ORM 2. In: Meersman, R., Tari, Z. (eds.) OTM-WS 2005. LNCS, vol. 3762, pp. 676–687. Springer, Heidelberg (2005)
7. Halpin, T.: Modality of Business Rules. In: Siau, K. (ed.) Research Issues in Systems Analysis and Design, Databases and Software Development, pp. 206–226. IGI Publishing, Hershey (2007)
8. Halpin, T.: Object-Role Modeling: Principles and Benefits. International Journal of Information Systems Modeling and Design 1(1), 32–54 (2010)
9. Halpin, T.: Fact-Orientation and Conceptual Logic. In: Proc. 15th International EDOC Conference, pp. 14–19. IEEE Computer Society, Helsinki (2011)
10. Halpin, T.: Formalization of ORM Revisited. In: Herrero, P., Panetto, H., Meersman, R., Dillon, T. (eds.) OTM-WS 2012. LNCS, vol. 7567, pp. 348–357. Springer, Heidelberg (2012)
11. Halpin, T.: Modeling of Reference Schemes. In: Nurcan, S., Proper, H.A., Soffer, P., Krogstie, J., Schmidt, R., Halpin, T., Bider, I. (eds.) BPMDS 2013 and EMMSAD 2013. LNBIP, vol. 147, pp. 308–323. Springer, Heidelberg (2013)
12. Halpin, T., Carver, A., Owen, K.: Reduction Transformations in ORM. In: Meersman, R., Tari, Z. (eds.) OTM-WS 2007, Part I. LNCS, vol. 4805, pp. 699–708. Springer, Heidelberg (2007)
13. Halpin, T., Curland, M.: Automated Verbalization for ORM 2. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM 2006 Workshops. LNCS, vol. 4278, pp. 1181–1190. Springer, Heidelberg (2006)
14. Halpin, T., Curland, M.: Enriched Support for Ring Constraints. In: Meersman, R., Dillon, T., Herrero, P. (eds.) OTM-WS 2011. LNCS, vol. 7046, pp. 309–318. Springer, Heidelberg (2011)
15. Halpin, T., Morgan, T.: Information Modeling and Relational Databases, 2nd edn. Morgan Kaufmann, San Francisco (2008)
16. Halpin, T., Ritson, R.: Fact-Oriented Modelling and Null Values. In: Srinivasan, B., Zeleznikov, J. (eds.) Research and Practical Issues in Databases. World Scientific, Singapore (1992)
17. ter Hofstede, A., Proper, H., van der Weide, T.: Formal definition of a conceptual language for the description and manipulation of information models. Information Systems 18(7), 489–523 (1993)
18. Object Management Group: OMG Unified Modeling Language (OMG UML), version 2.5 FTF Beta 1 (2012), `http://www.omg.org/spec/UML/2.5`
19. W3C: OWL 2 Web Ontology Language: Direct Semantics, 2nd edn. (2012), `http://www.w3.org/TR/owl2-direct-semantics/`
20. Wintraecken, J.: The NIAM Information Analysis Method: Theory and Practice. Kluwer, Deventer (1990)