Higher Order Logic and Interoperability in Global Enterprise

Michael Heather[1], David Livingstone[2] and Nick Rossiter[2]

[1] Ambrose Solicitors, St Bede's Chambers, Jarrow NE32 5JB, UK,
michael.heather@cantab.net
[2] School of Computing, Engineering and Information Sciences, Northumbria University, NE1 8ST, UK
{david.livingstone,nick.rossiter}@unn.ac.uk
WWW home page: http://computing.unn.ac.uk/staff/CGNR1/

Abstract. Techniques based on sets have proved useful in many types of information systems. In databases the relational model has maintained wide dominance in business data processing. However, interoperability between different databases, even when based on the relational model, is proving a major problem. Sets satisfy first-order predicate logic which as consistent and complete has many advantages in practical application. Interoperability requires higher order logic as the arguments themselves are relations and functions. Higher order logic in the context of set theory behaves less satisfactorily according to Gödel's theorems as such logic cannot satisfy all three of soundness, completeness and effectiveness. This may be a fundamental reason why interoperability is proving to be so difficult. This paper looks at underlying problems and suggests that they may be avoided by the use of categorial higher order logic. Cartesian categories are complete, consistent and decidable. They can be employed as an engineering technique to construct a general architecture of interoperability.

1 Introduction

Information systems are basic building materials for the knowledge-based economy of the 2000 Lisbon strategy. The Lisbon agenda of March 2000 set out a strategy resulting in the eEurope 2005 Action Plan to build the knowledge-based economy of the single market [Rodrigues 2003]. The fundamental basis of knowledge is information which has to be handled appropriately by both the technology and by the law. What was appropriate for the physical media of the last millennium cannot be just carried over to the new digital media.

Modern information systems operate at every level: from data held in a single purpose fixed device, through common PCs at home or mobile computing with business systems of an SME, to databases, intra-acting locally and at national level, inter-acting between nation states and even then open to wider global systems outside of Europe. This interoperability requires global coherence which in synecdoche correlates with the interoperation of the EU itself. The reported

slow progress with the Lisbon agenda is reflected in a similar tardy development of interoperable information systems. The latter is perhaps one of the causes of the former. The report of Wim Kok, quoted in [Euractiv 2005], recommended that the agenda be re-focused on growth and employment to remedy the small progress over the first five years in member states. Like national employment the focus needs to be on the details of operation on local information systems.

> Particular attention needs to be paid to the delivery of the Lisbon agenda. In order to achieve these objectives, the Union must do more to mobilise all the resources at national and Community levels so that their synergies ... can be put to more effective use ([European Commission 2005] at p.9)

There is a problem with logic. Successful local systems are first order but they need to participate in higher-order activity. The quest for synergy between levels runs into problems arising from semantic interoperability. There are two main types of data: images and text. Images can be structured, as in graphs, or unstructured, as in photographs. Text can be structured, as in relational databases, or 'unstructured', as in natural language [Probst, Raub & Romhardt, 2000]. There is also process data, represented as transactions in many current systems.

Syntactical interoperability is already achieved for digital and analogue data, either where the data is unstructured or where the data has a natural structure representable in a form that holds the informational content, for instance as ordered pixels or natural language text. These may be universally recognisable as in pictures or other interpretable materials where a common understanding readily exists. For instance an English text is interoperable throughout the world only in its native state or by translation into some other natural language.

Consequently there are not too many problems with interoperability of the natural data. We simply need operations at the syntactical level, which Google does well for text and for well-established image formats like jpeg, tiff and gif. However, these forms of information are essentially raw data. When value is added to the data through the application of analytical methods, we obtain structured data, for which inter-communication is problematical. In the past these problems were minimised because the data was used mainly in a local setting, that is intra-communication was needed where any standard was common to the locality. Interoperability itself is concerned with the inter-communication of data at different and therefore usually heterogeneous localities. Figure 1 summarises those definitions in the form of a table. The importance of exactitude in transmission of data around the single market and commercial inter-communication with the rest of the globe needs to be emphasised.

2 Exactness

The concept of exactness is important in commerce. Customers always want what is ordered, whether it is good for their services or not, and not some approxi-

Type	Structure	Examples	Applications
Images	Structured	Graphics	Business graphics
	Natural ('Unstructured')	Photographs	Publishing
Text	Structured	Relational databases	Business data
	Natural ('Unstructured')	The web	Google
Intermediate Data	Meta-structure added to unstructured data	Semi-structured data (as in XML) Semantic Web CAD/CAM Engineering Drawing with instructions	Fitting spare parts
Process Data	Dynamic	Banking Transactions Biometric identity	ATM accounting Iris data

Fig. 1. Natural and Structured Data Types

mation of the order. Of course in the real world there are always experimental errors but these can normally be controlled in a local environment and in practical terms can be minimised according to how much the customer wants to pay. Commerce has always had an international dimension but until very recently it has tended to be locally based. Communications between local bases have been point to point, with transportation by land, sea or air and conversations by mail, telephone or wireless broadcasting. The world consisted of a large number of mainly autonomous locally-based entities with simple inter-connection where the main effort is intra-activity at the local level, fit for the purpose with merchantable quality. A satisfactory theory can rely on linear models, linear logic, etc [Girard 1971]. Local equates with classical and of the same type.

Exceptionally large systems, even if of the same type, may lie outside local operating conditions. For example the large database cannot be maintained by one person and a very large database cannot be copied because in the meantime it is changed. However, distinction is usually qualitative rather than quantitative.

Non-locality on the other hand may still be composed of what approximates to a set of localities of the same type. It then operates as a classical organisation. The US legal system is able to operate this way with a federal law coordinating different local state laws. The early days of the EU imposed the same laws on the member states which at that time were few in number. But the character of Europe is diversity and variety, in language, culture, customs and style with quite different ways of working in manufacturing, commodities and providing services. When the number of member states of Europe became enlarged, imposing the same laws became impossible and there was a move to harmonisation. It is a comparable position with information systems in different states. Co-ordinating

systems of the same type is only a first-order activity but for heterogeneous systems higher-order operations are needed.

Legal systems are archetypal general information systems. Intensionally the legal systems of the member states of Europe are identical and feed into the over-arching European law in the European Court of Justice. But each local legal system is extensionally different. We shall see here how the theory of inter-operating business information follows the same practice. The topos in Figure 8 could represent the legal systems of Europe just as easily as information systems.

Exact operations with systems of the same type mean it is easy to assume that the same conditions will apply for different type systems but in reality the results can be radically different and unpredictable even leading to dangerous results and therefore a subject for rigorous risk management. These may be derived from theory or experimental results, preferably both. The theory in this respect is shown by the Austrian mathematician Kurt Gödel. Gödel was able to show in his doctoral thesis of 1929 on the completeness theorem [Gödel 1929] that first-order predicate logic is complete.

The significance is that consistent logical propositions may be applied as a model of first-order systems. Thus a digital computer operating with a von Neumann architecture gives satisfactory results with first-order limits. Most of the work in applied mathematics in the last two centuries has been gauged on clever theories to keep within the first-order limits. However, there are applications which have defied such analytical methods like turbulence where it has been necessary to resort to more qualitative techniques as found in chaos theory. These do not provide exact results. This can also be explained by Gödel's theory of undecidability which is perhaps even better known [Gödel 1931]. Gödel's theorem shows that both intensional and extensional systems which rely on axiom and number are undecidable [1].

Traditional mathematical modelling, which relies on set theory cannot therefore be applied directly to higher order behaviour. An example of undecidability in the case of a computational machine relying on the Church-Turing thesis is the halting problem. A commercial example of this can be found in the implementation of Codd's relational model [Codd 1990] as it is utilised in modified form in much of current data processing. As we shall see in the next section, in its pure form the relational model works well for atomic data because it is within Gödel's principle of first-order predicate logic and is therefore complete. That is the relational model and its corresponding calculus give exact results for atomic data. While the commercial version of the relational model SQL is a vast improvement on earlier data models, it has compromised some of the relational model features and is neither complete, nor decidable. The relational model is sometimes trumpeted as an example of the effectiveness of logic in com-

[1] There is the question of even how to define completeness, consistency, decidability, soundness and effectiveness. The literature itself is not consistent and we will therefore leave aside what these words mean in a set theoretic context and rely below on corresponding categorial concepts as definitive.

puter science [Halpern *et al* 2001]. This is only for theoretical computer science. Implementations of the model show a divergence between theory and practice.

Of course real-world data does not consist of homogeneous independent items making up atoms. This has resulted in various techniques such as normalisation with a series of normal forms: first normal form, second, third, etc, which attempt to squeeze real-world phenomena into a collection of first-order relations, to behave optimally with regard to update and search operations. Hence data, lacking any naturally regular atomic form, may be squeezed by normalisation into such a structure in a consistent manner.

3 Practical Examples of Interoperability Problems

The relational model predominates in much of commerce today as the format for structured information. Yet there are very significant interoperability problems between one relational database and another. Some of these can be attributed to problems with the underlying SQL standard as described below.

3.1 Variants of SQL

Vendors of SQL DBMS support different variants of SQL, all in varying degrees differing from the versions of the SQL International Standard [2], either having additional features and/or omitting features.

Features	Achievements	Problem in interoperability
Full facilities	Not achieved by MySQL	Very difficult between MySQL and other DBMS
Hierarchies, manipulation	Peculiar to Oracle	Difficult between DB2 and Oracle in network/hierarchical structures
Recursive union, assembling networks	Peculiar to DB2	
Implementation of integer type	Oracle treats as numeric(38)	Difficult between Oracle and other DBMS in formatting and rounding numbers
Dates	Different logical formats	Difficult between all systems in reliable data format recognition

Fig. 2. Effects of Variants of SQL on Interoperability

Some features described in the standard are labelled implementation-dependent, meaning they are independent of the standard. Others are implementation-defined, meaning the manner in which the feature is achieved is at the discretion

[2] Information Technology – Database Languages - SQL, ISO/IEC 9075:2003 (2003).

of the implementer. Therefore it is not always possible to guarantee a semantically valid transfer of data from one SQL DBMS to another, since the recipient DBMS may treat the received data in a different way to that which the sending DBMS would have treated it, had it carried out ostensibly the same operations. This situation arises because the standards are not based completely on scientific or mathematical principles. Standards are also influenced by the software vendors, who are looking for pragmatic and strategic ways in which their products can be promoted. Examples of problems at the data level are shown in Figure 2.

Therefore if two databases are to be interoperable, it is much simpler if they are both managed by the same DBMS package, because then the only problems in this context are those arising from the consistent application of one variant of SQL. For this and other reasons, such as reduced DBMS maintenance and licence fees, in practice multiple-vendor SQL DBMS are unheard of, except where they arise due to force of circumstance, for example the merger of two previously independent companies.

3.2 SQL versus the Relational Model

The relational model is based on two mathematical theories: first-order predicate calculus and relations ([Codd 1990] at p.v). Interestingly the relations permitted are not completely general (at p.467-477). In particular a collection of n-ary relations of assorted degrees is strongly encouraged where n is a positive, finite number, giving the degree of a particular relation. Both a single universal relation and a collection of binary relations are strongly discouraged, as the former loses flexibility in logical navigation and the latter is cumbersome and unnatural. Further if the collection of n-ary relations is constrained to be in first normal form with all values atomic, then the predictable regular form greatly simplifies the query language.

No version of SQL implements the full relational model, either that specified by Codd or evolved from Codd's model by others, for example see [Date & Darwen 2006]. Some differences between SQL and the relational model are summarised in Figure 3. The different default structures of set and bag respectively for the relational model and SQL are of particular interest. Sets, bags and other container-types such as sequences of tuples are specifically defined and cannot be used interchangeably. However, there are means of carrying out conversions from one kind of container type to either of the other two, in an attempt to make it trivial to achieve mathematical exactitude in defining and manipulating the different kinds of tuple containers. Current work at Northumbria in the Open Database Project shows the need for rigorous definition at the local level in prototyping languages like Raquel [Livingstone 2007] to produce an open source implementation of the features developed in the demonstration language Tutorial D [Date & Darwen 2006]. The aim is to keep as close as possible to the philosophy of the pure relational database model including object classes as data types orthogonal to relations, an open architecture satisfying this philosophy, a design for the architecture and an implementation of that design. Interoperability is

facilitated by the use of pure relational languages such as Raquel, together with conversion techniques for mapping between different container types.

Feature in SQL	Feature in relational model	Consequence for SQL
Default structure is bag	Default structure is set	Duplicate rows permitted, inconsistency in updates
Row identifier	No identifiers	Physical bias to extension
Rows may be sequenced	No sequencing of rows	Data is apparently ordered
Set operations such as union based on column position	Set operations such as union based on column name	Set operations are based on physical, not logical, ordering of columns
Duplicate column names allowed in output	No duplicates allowed	Confusing output

Fig. 3. Differences between SQL and the Relational Model

3.3 Closed World Assumption

The definition of a relation should be a logical predicate such that each tuple in a relation corresponds to a logical proposition that is true for that predicate. By the Closed World Assumption (that is the CWA) any tuple not in the relation represents a false proposition. This is an attempt to satisfy Gödel's decidability principle: that tuples in the relation are true and tuples outisde the relation are false [Date 2006]. However it is not possible for a relational DBMS to guarantee that all the tuples in a database represent true propositions, only that they all consistently meet a set of integrity constraints that partially represent the real-world logical predicate. This is because the typing system is based on set inclusion principles rather than constructive ones. Therefore decidability may be a problem with all relational systems.

Suppose there are two relations in a database, ProdCust(ProdNo, CustNo) and ProdEmp(ProdNo, EmpNo), such that CustNo and EmpNo have the same data type, and ProdCust signifies that a product was bought by a customer and ProdEmp that a product was made by an employee. While the union of both of the relations is valid, there is a question over whether it should be and on the nature of the resulting product. The answer to this question depends on whether employees can also be customers. If they can be customers, and a single tuple exists for both relations in the result of the union, what does that tuple mean in the result, and how is the fact that an employee both made and bought a certain kind of product represented?

If the data types of CustNo and EmpNo are different, then the relations cannot be subject to union, intersect and difference operations. So to intersect them to see if any employees buy products that they have made, it would be

necessary to do some type casting, perhaps faciliiated by storing predicates of a relation in the database metadata.

Nulls are another example of where database approaches based on CWA run into difficulties. ([Codd 1990] at p.383-387) suggests that the relational model should permit nulls as markers, with two interpretations: missing-but-applicable and missing-and-inapplicable. Nulls are not data values. A four-valued logic is then employed to manipulate such data with the outcome of true, false or two types of maybe.

SQL claims to have a three-valued logic since logical variables may take the values true, false or null. However, it is not clear that null can be safely equated with maybe and a number of problems arise as shown in Figure 4.

Case	Result	Problem
Creation of nulls whether value is missing-but-applicable or missing-and- inapplicable	No distinction	Semantic simplification
Use of null to represent maybe in the Boolean type	Three values for Boolean logic	Contrary to normal view of Boolean logic as binary valued
Comparing a null value with a null value	maybe with join/restrict, true in set operations	Difference in outcome between set operations and other operations such as join
Split table into a set of sub-tables using restrict; union resulting sub-Tables	No guarantee that this will be the original table	Restrict only returns rows where the comparison returns true; hence those returning null are ignored and lost
Aggregation operators applied to columns containing some nulls	Count includes them; others ignore them	Arbitrary application
Aggregation operators applied to columns containing all nulls	Count returns zero; others return null	Arbitrary application
Second order distributivity (e.g. fuzzy sets)	Logical equivalences are not true	Inconsistent treatment of nulls

Fig. 4. Problems with Handling of Nulls by SQL

From the Gödel perspective, nulls make a system undecidable. It is therefore not surprising that practical implementations such as SQL have many problems in handling nulls. Some more recent versions of the relational model do not permit nulls, for example [Date & Darwen 2006]. It is likely that the handling of nulls will be facilitated by the use of metadata to describe the reason for the null. The questionable CWA assumption also raises problems in proving that the result of a query is logically valid. This gives rise to undecidability and a lack of completeness. Compounding the problem is that many users look for plausible results, often on small volumes of data.

It is perhaps worth raising the question as to whether object-oriented databases would overcome some of the disadvantages above. The answer is no, as objects will suffer from all the problems of sets and methods are implemented

through an enriched type system, similar to sketches or perhaps 3-categories. The object-oriented approach needs to be founded in category theory to be complete and decidable. Intuitionistic logic offers a more convincing way forward. Compared to predicate logic it is more naturally applicable to open systems, is constructive and avoids the problem of impredication.

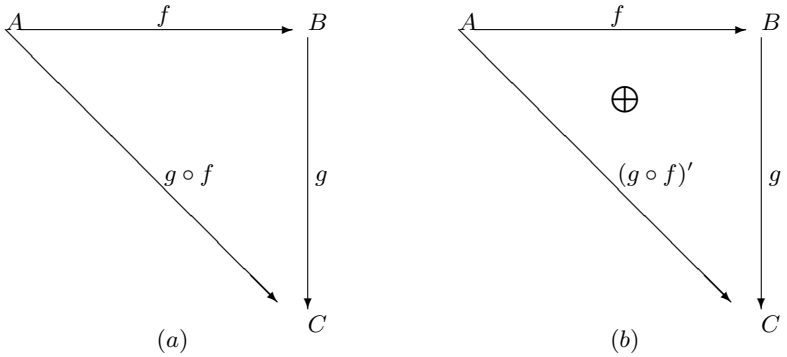

Fig. 5. Commuting Diagrams for a) Composition, b) Punctured Composition: $(g \circ f) \neq (g \circ f)'$

4 Interoperability and Categories

In category theory [Mac Lane 1998] alternative meanings of decidability, completeness, satisfiability, soundness and consistency, all used by Gödel, converge. They come together in the composition diagram in Figure 5(a). The negation of these terms or where they fail wholly or in part are all subsumed in the diagram of punctured composition [Freyd 1990] in Figure 5(b). At one level the composition diagram of Figure 5(a) is a formal categorial representation of Gödel's result that first-order predicate logic is complete. This diagram therefore satisfies a local intra-operability of a single system and first-order interoperability between simple systems. The difference is that moving to higher orders such as axiomatic number systems is undecidable. However, that limitation does not apply to a process view of the arrow. Composition is still satisfied by the diagram but where the arrows can be a different type or from different levels. We have shown in [Rossiter, Heather & Nelson 2006] that free interchange between four levels can satisfy any realisable system. The conditions for interoperability come from adjointness [Rossiter & Heather 2005] between the two composition triangles in Figure 6.

Critical details of these two triangles are the values η, ϵ, respectively the unit and counit of adjunction [Lawvere 1969] in Figure 7. F is the functor that carries the data across from the left system to the right system. G is the underlying functor giving the rules for that transmission. f, g respectively represent dynamic data in the left and right systems. L is an object in category **L** and R an object in category **R**. Note that this only defines information in one direction. For two-way communication there has to be a self-adjointness of both left and right

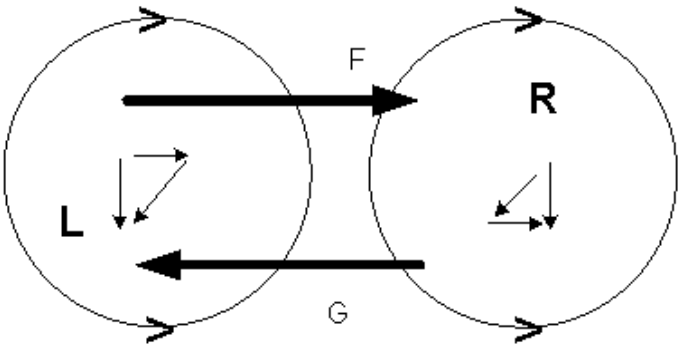

Fig. 6. Adjointness between two Composition Triangles

systems. As explained above in the example of legal systems this is a process of harmonisation. The systems do not need to be identical, that is η is other than $\perp$ and $\top$ other than ϵ.

4.1 Architecture for Interoperability

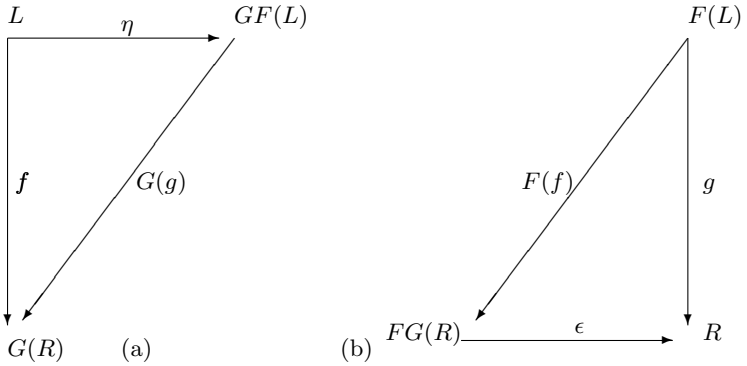

Fig. 7. Roles in Adjointness of a) η, b) ϵ

The architecture for full interoperability between more than two systems is then a composition diagram of the form shown in Figure 8. Fundamental category theory shows that for physical existence the real world operates as a cartesian closed category. All the categories drawn above are therefore cartesian closed. The theory also shows that any such operation involves only two categories $(\mathbf{L}, \mathbf{R})$ and a context category $\mathbf{C}$, that is a left system communicates with a right system in the context of all other systems. All other systems are therefore a single context category. More precisely this is a topos $\mathbf{T}$ as in Figure 8. Remember that interoperability is really a global character where everything is connected to everything else. We are not dealing with discrete systems. The context category

described above is only a view and is really the limit of the topos. So $\mathbf{C} \longrightarrow \mathbf{T}$. And of course $\mathbf{L}$ and $\mathbf{R}$ are themselves subobjects of $\mathbf{T}$. This is the essence of interoperability where category theory, or nothing less than category theory, can give the required insight to construct exact and decidable interoperating information systems.

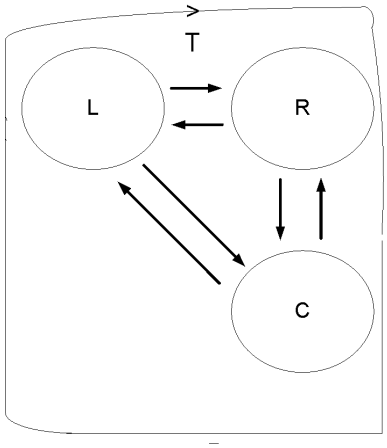

Fig. 8. Architecture for Interoperability: Topos $\mathbf{T}$ involving categories $\mathbf{L}, \mathbf{R}$ and Context Category $\mathbf{C}$

5 Conclusions

From the work of Gödel, first-order predicate systems are complete, consistent and decidable. Much of the attention in defining a relational data model has focused on keeping to a strict first-order system. The treatment of issues such as normalisation, nulls and recursion by workers developing a pure relational model [Date & Darwen 2006,Codd 1990] is designed to avoid the need to handle higher-order logic in set theory. Indeed the relational model in its proper form is classified as one of the outstanding successes of logic in computer science [Halpern *et al* 2001]. The more casual treatment of such factors in SQL has led to systems which are no longer consistent and decidable, giving many problems in interoperability.

Interoperability is essentially a higher-order problem, For higher order systems we need composability to achieve the same rigour as found in first-order predicate systems. Composability is a cornerstone of category theory and an architecture has been proposed, based on the topos, for achieving interoperability while meeting Gödel's requirements.

6 Acknowledgements

We are grateful to Hugh Darwen for his comments on the SQL standard and the relational model.

References

[Codd 1990] Codd, E F, *The Relational Model for Database Management* Addison-Wesley (1990).

[Date 2006] Date, C J, *Gödel, Russell, Codd: A Recursive Golden Crowd* http://www.dcs.warwick.ac.uk/∼hugh/TTM/goedel.pdf 6pp July 17th (2006).

[Date & Darwen 2006] Date, C J, & Darwen, Hugh, *Databases, Types and the Relational Model: The Third Manifesto* 3rd Ed, Addison Wesley (2006).

[Euractiv 2005] Euractiv Network, *Relaunch of the Lisbon Strategy* http://www.euractiv.com/en/innovation/growth-jobs-relaunch-lisbon-strategy/article-131891 (2005).

[European Commission 2005] European Commission, Internal Guidelines in: *Integrated Guidelines for Growth and Jobs 2005-2008* 2005/0057, http://ec.europa.eu/growthandjobs/pdf/COM2005_141_en.pdf (2005).

[Freyd 1990] Freyd, P, & Scedrov, A, *Categories, Allegories*, North-Holland (1990).

[Girard 1971] Girard, Jean-Yves, Une extension de l'interpretation de Gödel, à l'analyse, et son application l'élimination des coupures dans l'analyse et la théorie des types, *Studies in Logic and the Foundations of Mathematics* North-Holland 63-92 (1971).

[Gödel 1929] Gödel, Kurt, *Über die Vollständigkeit des Logikkalküls* Doctoral Thesis, D1.736 33pp, University of Vienna (1929). Reprinted in Feferman, S, ed. Gödel Collected Works, volume 1, publications 1929-1936, Oxford, p.60-100 (even numbers) original; p.61-101 (odd numbers) Translators Bauer-Mengelberg, Stefan, & Heijenoort, Jean van, (1986).

[Gödel 1931] Gödel, Kurt, Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme, I. *Monatshefte für Mathematik und Physik* **38** 173-198 (1931): translated in Jean van Heijenoort, *From Frege to Gödel: A Source Book on Mathematical Logic* Harvard 596-616 (1967).

[Halpern *et al* 2001] Halpern, Joseph Y, Harper, Robert, Immerman, Neil, Kolaitis, Phokion G, Vardi, Moshe Y, Vianu, Victor, On the unusual effectiveness of logic in computer science *Bulletin of Symbolic Logic* **7**(2) 213-236 (2001).

[Lawvere 1969] Lawvere, F W, Adjointness in Foundations, *Dialectica* **23** 281-296 (1969).

[Leinster 2004] Leinster, Tom, *Higher Operads, Higher Categories* Cambridge (2004).

[Livingstone 2007] Livingstone, David, *Open Database Project*, CEIS, Northumbria University, http://computing.unn.ac.uk/openDBproject/ (2007).

[Mac Lane 1998] Mac Lane, *Categories for the Working Mathematician*, 2nd edition, Springer (1998).

[Probst, Raub & Romhardt, 2000] Probst,G, Raub, S, & Romhardt,K, *Managing Knowledge Building-Blocks for Success* Wiley (2000).

[Rodrigues 2003] Rodrigues, Maria Joao, *European Policies for a Knowledge Economy* Edward Elgar (2003).

[Rossiter & Heather 2005] Rossiter, Nick, & Heather, Michael, Conditions for Interoperability, *7th ICEIS* Florida, USA, 25-28 May 2005, 92-99 (2005).

[Rossiter, Heather & Nelson 2006] Rossiter, Nick, Heather, Michael, & Nelson, David, A Natural Basis for Interoperability, *I-ESA'06* Preproceedings 12pp pdf (CD, W11, 22 March 2006); proceedings 10pp LNCS Springer (2006).