Northumbria Research Link

Citation: Rossiter, Nick and Heather, Michael (2005) Conditions for interoperability. In: 7th International Conference on Enterprise Information Systems, 25-28 May 2005, Florida.

URL:

This version was downloaded from Northumbria Research Link: http://nrl.northumbria.ac.uk/1020/

Northumbria University has developed Northumbria Research Link (NRL) to enable users to access the University's research output. Copyright © and moral rights for items on NRL are retained by the individual author(s) and/or other copyright owners. Single copies of full items can be reproduced, displayed or performed, and given to third parties in any format or medium for personal research or study, educational, or not-for-profit purposes without prior permission or charge, provided the authors, title and full bibliographic details are given, as well as a hyperlink and/or URL to the original metadata page. The content must not be changed in any way. Full items must not be sold commercially in any format or medium without formal permission of the copyright holder. The full policy is available online: http://nrl.northumbria.ac.uk/policies.html

This document may differ from the final, published version of the research and has been made available online in accordance with publisher policies. To read and/or cite from the published version of the research, please visit the publisher's website (a subscription may be required.)

Northumbria University NEWCASTLE

University Library

Northumbria Research Link

University Library, Sandyford Road, Newcastle-upon-Tyne, NE1 8ST

# CONDITIONS FOR INTEROPERABILITY

Nick Rossiter
*School of Informatics, Northumbria University, NE1 8ST, UK Email: nick.rossiter@unn.ac.uk*

Michael Heather
*Northumbria University, NE1 8ST, UK Email: m.heather@unn.ac.uk*

Key words: semantic interoperability, organisational interoperability, Godement calculus, commuting diagrams

Abstract: Interoperability for information systems remains a challenge both at the semantic and organisational levels. The original three-level architecture for local databases needs to be replaced by a categorical four-level one based on concepts, constructions, schema types and data together with the mappings between them. Such an architecture provides natural closure as further levels are superfluous even in a global environment. The architecture is traversed by means of the Godement calculus: arrows may be composed at any level as well as across levles. The necessary and sufficient conditions for interoperability are satisfied by composable (formal) diagrams both for intension and extension in categories that are cartesian closed and locally cartesian closed. Methods like partial categories and sketches in schema design can benefit from Freyd's punctured diagrams to identify precisely type-forcing natural transformations. Closure is better achieved in standard full categories. Global interoperability of extension can be achieved through semantic annotation but only if applied at run time.

## 1 Classical Data Structures

Classical information systems employ some suitable model to mediate between data and hardware. A database model is a representation of policies in a structured form according to some perceived view of reality.

In the ANSI/SPARC architecture (Tsichritzis 1978) a conceptual schema or model is defined as a global logical definition of the data structure. This schema relates to the internal (physical) definition by a mapping from the logical level to the physical level. The schema is protected from changes at the physical level by adjusting this mapping. Each user has a particular view (external schema) of the database which may be a restricted view. The architecture including the series of mappings shown in Figure 1 provides aspects such as security and logical data independence.

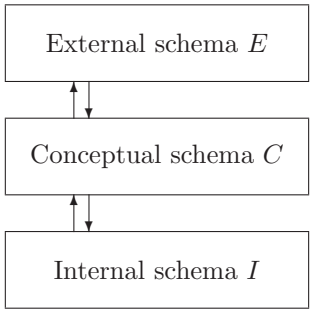


Figure 1: Classical ANSI/SPARC Architecture for Databases

The classical ANSI/SPARC architecture of Figure 1 has the disadvantage that the levels are not independent of each other. This may be compared with the natural architecture of Figure 2 (Heather & Rossiter 2002). The four levels (top-down) are categories for concepts (real-world abstractions), constructs (facilities available for schema design), schema (definition of data types available in system) and data (the data itself). In ANSI/SPARC the types of the three levels are similar to the external schema but the internal schema is composed of subcat-

egories of the conceptual schema. The top level, the external schema, is not a universal closure of types but a local closure of the conceptual schema. The four-level architecture in Figure 2 has orthogonal types with the relationships between the levels expressed as categorical adjunctions as already applied to structures in GRID data processing (Heather & Rossiter 2002). Categorical adjunctions relate one level to another. The relationship between levels is measurable by the unit of adjunction. For instance the adjunction $Policy \dashv MetaMeta$ indicates that the free functor $Policy$ is left adjoint to the underlying functor $MetaMeta$. The unit of adjunction is given by $\eta_{cpt} : 1_{cpt} \longrightarrow MetaMeta \circ Policy(cpt)$.

The ANSI/SPARC architecture was a useful way of capturing abstractions of the relational model in the 1970s and 1980s. It has proved less suitable to facilitate the techniques needed today such as interoperability where systems with different underlying models are required to work together. ANSI/SPARC can be viewed as pseudo-natural. It was developed using mathematical techniques and theories like sets. But there is a gap between classical theory and real-world performance and pragmatics. Triggers are an example of an attempt to patch the weakness of the system by providing some local strong anticipation using Event-Conditions-Actions (ECA) (Date & Darwen 2000).

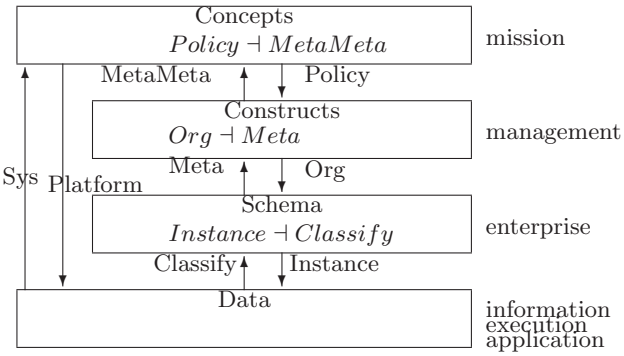


Figure 2: Interpretation of Levels: natural schema with strong anticipation

In Figure 2 the terms used have their normal meaning. Basically in the downward direction, a collection of data structuring concepts (abstractions) are mapped through policies to a collection of constructions (for example classes, tables) which are in turn mapped through organisation to a collection of types (for example, schema definitions) which are finally mapped through instantiation to named data values. In the opposite direction, the named data values are mapped through classification to types, which are in turn mapped through metadata to constructions which are finally mapped to concepts through metameta data.

## 1.1 Natural Closure

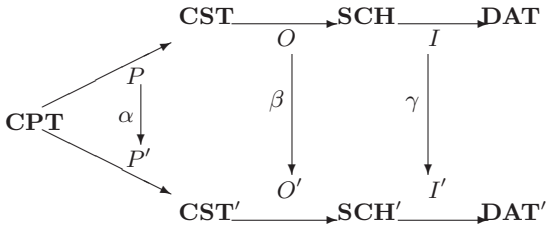


Figure 3: Comparison of Mappings in two Systems

In category theory four levels are needed to define an arrow as unique up to natural isomorphism. The four levels are: 1) object or identity arrow (within a category), 2) category (comparing objects), 3) functor (comparing categories) and 4) natural transformation (comparing functors). No more levels are required.

The relationships between one four-level architecture and another can be constructed as in Figure 3, the expanded view of Figure 2. Here for simplicity the mappings are viewed in one direction only. Two systems are compared, one involving categories **CPT**, **CST**, **SCH** and **DAT**, the other **CPT**, **CST′**, **SCH′** and **DAT′**, representing concepts (**CPT**), constructs (**CST**), schema (**SCH**) and data (**DAT**) from Figure 2. **CPT** is the same in both systems as there is one universal type for concepts. As usual the functors relate the categories. We have now though added natural transformations to relate the mapping between one functor and another. It needs to be emphasised that none of these categories are discrete: all have an internal arrow-based structure so the natural transformations are non-trivial (Rossiter 2003). The functors need to be of the same variance for a meaningful natural transformation to exist between them and this is the case for $\alpha$, $\beta$ and $\gamma$.

An arrow comparing natural transformations is itself a natural transformation. Some categorists use an older terminology with degrees of 'cell' and describe the identity arrow as 0-cell, an arrow in a category as 1-cell and an arrow between arrows

as a 2-cell (Kelly 1972). An arrow from one natural transformation to another gives a composition of the natural transformations, not a new level (((Barr & Wells 1999), 1st ed., at p.85); (Rossiter & Heather 2003)). This means that four levels are needed to give the natural closure (Heather & Rossiter 2002).

An alternative view to Figure 3, shown in Figure 4, is closer to the four levels inherent in category theory. The fundamental levels are considered to be data values, named values, classified values and contrasted representation corresponding in category theory to object, category, functor and natural transformation respectively. The natural transformations are now the duals of those shown earlier in Figure 3 as indicated by the *op* superscript. The earlier natural transformations were comparing the downward functorial mapping (towards data) while the current ones compare the upward mapping (away from data) (Rossiter & Heather 2003).

| alternative fundamental levels | category theory levels | four levels of Figure 2 |
|---|---|---|
| 1. data values | objects (identity arrows) | $id_{dat}$ |
| 2. named values | category | **DAT** |
| 3. classified values | functor | $I^{op}$ : **DAT** $\longrightarrow$ **SCH** |
| 4. contrasted representation | natural transformation | $\alpha^{op} \circ \beta^{op}$ |

Figure 4: Alternative Interpretation of Levels in the Architecture

It can be shown (Rossiter & Heather 2003) that the addition of further levels is possible but nothing is gained by it type-wise. Thus addition of an extra level to the top of a four-level architecture simply results in the top level (comparison of mapping from concepts to schema) being a composition of three arrows rather than two. Thus consider the addition of a new top level **PHI** with the mappings $F$ : **PHI** $\longrightarrow$ **CPT**, $G$ : **CPT** $\longrightarrow$ **PHI** and $\alpha^{op\prime}$ : $F \longrightarrow F'$ where $\alpha^{op\prime}$ compares the mappings $F$ and $F'$ in two different approaches. The adjunction is now $I \circ O \circ P \circ F \dashv G \circ A \circ M \circ C$. The level four of Figure 4 is now $\alpha^{op\prime} \circ \alpha^{op} \circ \beta^{op}$ and is still a natural transformation through the rules of composition. The practical consequence is that a fifth level is equivalent to an alternative fourth level. The meta-meta level gives ultimate closure of types.

## 2 Natural Calculus

We therefore have three types of mapping to consider: within a category (for instance from a name to a value), from one category to another (for instance the functor $P'$ from **CPT** to **CST**′) and from one functor to another (for instance the natural transformation $\alpha$ from $P$ to $P'$).

Following the constructive principles of category theory, the composition of these arrows is natural. This consequently gives rise to a natural calculus first expounded by (Godement 1958) and ((Barr & Wells 1999), 1st ed., pp 94-97) in the form of rules governing composition. The composition of functors and natural transformations is associative so that for instance in Figure 3:

$$(I'O')\alpha = I'(O'\alpha); \quad \gamma(OP) = (\gamma O)P$$

Natural transformations may be composed with each other:

$$\gamma\beta = (\gamma O) \circ (I'\beta); \quad \beta\alpha = (\beta P) \circ (O'\alpha)$$

Godement's five rules are given by ((Barr & Wells 1999), 1st ed., p.96-97).

Consider as in Figure 5:
five categories **A**, **B**, **C**, **D** and **E**
the following eight functors:
$E$ : **A** $\longrightarrow$ **B**, $F_1, F_2, F_3$ : **B** $\longrightarrow$ **C**, $G_1, G_2, G_3$ : **C** $\longrightarrow$ **D**, $H$ : **D** $\longrightarrow$ **E**
and the following four natural transformations:
$\alpha : F_1 \longrightarrow F_2$, $\beta : F_2 \longrightarrow F_3$, $\gamma : G_1 \longrightarrow G_2$, $\delta : G_2 \longrightarrow G_3$,

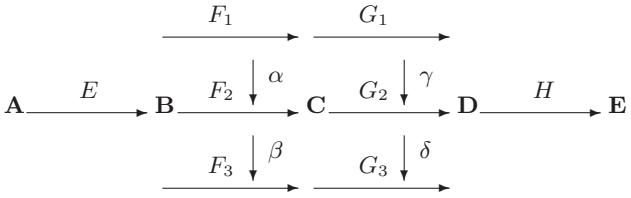


Figure 5: Godement in (Barr & Wells 1999), 1st ed., p.96

Then the following rules hold:

$$(\delta \circ \gamma)(\beta \circ \alpha) = (\delta\beta) \circ (\gamma\alpha) \quad (1)$$
$$(H \circ G_1)\alpha = H(G_1\alpha) \quad (2)$$
$$\gamma(F_1 \circ E) = (\gamma F_1)E \quad (3)$$
$$G_1(\beta \circ \alpha)E = (G_1\beta E) \circ (G_1\alpha E) \quad (4)$$
$$\gamma\alpha = (\gamma F_2) \circ (G_1\alpha) = (G_2\alpha) \circ (\gamma F_1) \quad (5)$$

Equation 5 is particularly interesting as it has different members on each side of the equation,

permitting solutions via simultaneous equations. The first four rules are concerned with interchange (commutativity), associativity and permutation.

Simmons ((Simmons, 1989) section 3.8) also deals with Godement's rules. For a simplified version of Figure 5, omitting categories $\mathbf{A}, \mathbf{E}$ and functors $E, H$, as shown in Figure 6, he derives the commuting diagram of Figure 7 to represent the composition of functors and natural transformations.

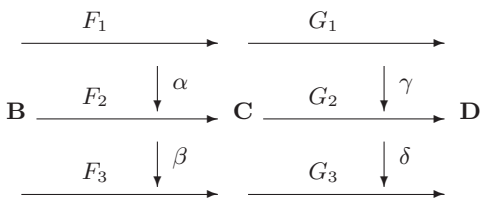


Figure 6: Godement in (Simmons, 1989) section 3.8

The commuting properties are shown in the diagram in Figure 7. A small change in Simmons' notation has been made so that pairs beginning with a natural transformation have the subsequent functor in subscript form to indicate that the object of the natural transformation is the functor ((Barr & Wells 1999), 1st ed., p.94-95). Where the pairs begin with a functor the notation is unchanged as it appropriately indicates the application of a functor to the output from a natural transformation.

The composition, say $\lambda$, of $(\delta * \gamma) \circ (\beta * \alpha)$ follows the path from $G_1F_1 \longrightarrow G_1F_2 \longrightarrow G_1F_3 \longrightarrow G_2F_3 \longrightarrow G_3F_3$; that, say $\rho$, of $(\delta \circ \beta) * (\gamma \circ \alpha)$ follows the path from $G_1F_1 \longrightarrow G_1F_2 \longrightarrow G_2F_2 \longrightarrow G_2F_3 \longrightarrow G_3F_3$. Both routes start and end with the same objects $G_1F_1$ and $G_3F_3$ respectively. Hence they are equivalent and the interchange law is demonstrated with $\lambda = \rho$.

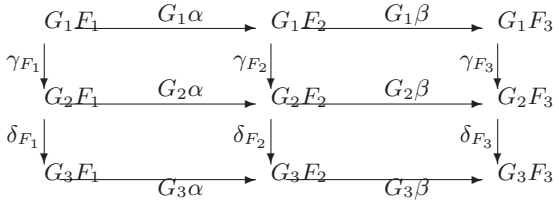


Figure 7: Commuting Diagram in (Simmons, 1989) section 3.8

## 3 Application

The consequence of natural closure is that a categorical approach ensures that the various arrows of different types can be composed with each other, irrespective of their level in the system. Equations representing an equality of paths, can be solved for unknown components that can be determined from an evaluation of the known properties. For instance in comparing methods with the path $IOP$ from **CPT** $\longrightarrow$ **CST** $\longrightarrow$ **SCH** $\longrightarrow$ **DAT** defining one approach, then the path $I'O'\alpha$ from **CPT** $\longrightarrow$ **CST′** $\longrightarrow$ **SCH′** $\longrightarrow$ **DAT′** might define an alternative approach if $P'$ maps onto constructs in the category **CST′**.

The diagram in Figure 8 shows the application of the Godement calculus to handle semantic interoperability, defined as the interoperation of one system with another at the level of meaning of the data, that is at the metadata level.

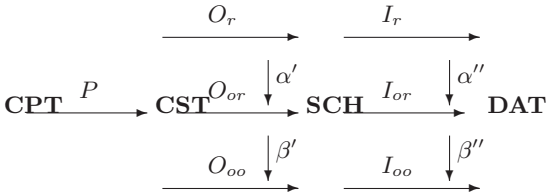


Figure 8: Semantic Interoperability in terms of Godement

The composition of the top line of functors $I_r \circ O_r \circ P$ gives the mapping from concepts to data for say a relational system $r$. The composition of the middle line of functors $I_{or} \circ O_{or} \circ P$ gives the mapping from concepts to data for say an object-relational system *or*. The composition of the bottom line of functors $I_{oo} \circ O_{oo} \circ P$ gives the mapping from concepts to data for say an object-oriented system *oo*. Comparing these compositions gives a framework for interoperability. For instance the natural transformation $\alpha'$ compares how the mapping is performed from constructions to types in a relational system $r$ with that from constructions to types in an object-relational system *or*. The natural transformation $\beta''$ compares how the mapping is performed from types to data in an object-relational system *or* with that from types to data in an object-oriented system *oo*. The advantage of the Godement approach is that arrows of any type can be composed with each other so that any route can be taken through the various mappings. The diagram in Figure 7 shows that a number of commuting equations can be derived, enabling solution of equations for unknown values. For instance $\beta'' \circ \beta'$ compares the mapping

from constructions to data in an object-relational system *or* with that in an object-oriented system *oo*.

To extend the categorical framework to handle organisational interoperability, defined as the interoperation of systems at the business process level, we need to vary the functor $P$ for each environment so that the metameta level is variable. The required diagram is shown in Figure 9.

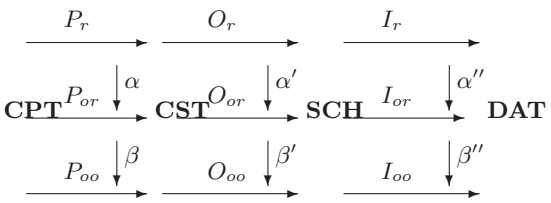


Figure 9: Organisational Interoperability in terms of Godement

The following canonical rules hold according to the Godement calculus:

$$(\beta' \circ \alpha')(\beta \circ \alpha) = (\beta'\beta) \circ (\alpha'\alpha) \tag{6}$$

$$(I_{or} \circ O_r)\alpha = I_{or}(O_r\alpha) \tag{7}$$

$$\alpha'(O_r \circ P_{or}) = (\alpha' O_r)P_{or} \tag{8}$$

$$I_r(\beta' \circ \alpha')P_{or} = (I_r\beta' P_{or}) \circ (I_r\alpha' P_{or}) \tag{9}$$

$$\alpha''\alpha' = (\alpha'' O_{or}) \circ (I_r\alpha') = (I_{or}\alpha') \circ (\alpha'' O_r) \tag{10}$$

A number of general principles in composition are shown by the equations. Equation 6 indicates that of commutativity (the interchange law); equations 7...8 indicate that of associativity; equation 9 indicates that of permutation of paths. The last equation, 10, shows the production of simultaneous equations representing different paths through the diagram. This is an important feature as it facilitates the solution for an unknown mapping. For example, in equation 10 above, if the values $\alpha', \alpha''$ and $I_{or}$ are known, then $O_r$ is the only unknown and a solution can be found for it. That is if it is known how the mapping from constructions to types and from types to data varies between a relational system $r$ and an object-relational system *or* and what the mapping is between types and data in an object relational system *or*, then the mapping between constructions and types in the relational system $r$ can be derived.

## 4 Semantic Interoperability

The foregoing indicates that semantic interoperability can be guaranteed therefore for a system that implements in full formal form the four-level categorical diagram and approach as just described. In particular all compositions of arrows (identity, function, functor, natural transformation) must be natural, that is all diagrams must commute. Semantic interoperability depends on the horizontal and vertical composition of both syntactical and semantic diagrams as well as the interaction (contravariant) between the syntactical and the semantic. The semantics involve instantiation everywhere, that is local extensionalities interconnected one with another through global intensionality. We have not explicitly mentioned the usual point (because reality is equivalent to naturality in category theory) that all the categories already referred to in this paper are cartesian closed. In formal terms of category theory, this further condition for global connectivity in interoperability means that the categories need to also be locally cartesian closed. This property connects and integrates in a coherent way slice categories [1].

The method of semantic annotation as advanced at present appears to be a local method carried out either manually or by some automated agents. This will be very reductionist but may be quite sufficient if carried out at run time.

Linked with the typing problems exhibited in punctured diagrams, there is the whole question of the capture of type information. Semantic interoperability depends on as complete a picture as possible being obtained of types in the different systems. Semantic annotation is employed in the semantic web (Hendler, Berners-Lee & Miller 2002) and in other techniques such as metadata creation (Soo *et al* 2003) where agents are used to explore the data structures for type information.

For semantic annotation we are investigating the use of natural database techniques (Rossiter & Heather 2004) to see how much of such information can be collected automatically through analysis of the data in a categorical framework. Collection of metameta data is essentially an open architecture task and we intend to employ the categorial topos and its internal intuitionistic logic i.e. Heyting (Mac Lane & Moerdijk 1991; Johnstone 2002) for this purpose.

[1] Barr & Wells provide a comprehensive definition of locally cartesian closed categories ((Barr & Wells 1999), 3rd ed.) and the significance of slice categories in computer science is extensively dealt with by (Goguen & Burstall 1984).

## 5 Composition Failure

Composition is only certain for categories with arrows as properly defined (Simmons, 1989). Figure 1 is not a formal diagram and the arrows in that figure for the classical ANSI/SPARC architecture often do not satisfy such requirements. Implementations of database schemas typically make use of partial functions in a reductionist view of real-world naturality. For instance the relational data model of Codd has had to be compromised for the various SQL standards (Date & Darwen 2000). Partial functions are in such common use in mathematical modelling that various attempts have been made to carry them over into the use of category theory. Peter Freyd, an early categorist pioneer, has proposed that composition failure be acknowledged in formal diagrams by a puncture mark.

### 5.1 Punctured Diagrams

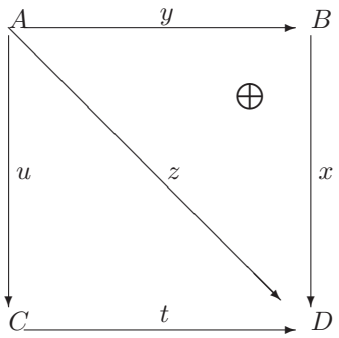


Figure 10: Punctured Commuting Diagram

A commuting diagram is itself a proof of some naturality. A weaker version of the commuting diagram may be represented by the punctured diagram promoted by ((Freyd 1990) section 1.251) as in Figure 10 where the puncture mark $\bigoplus$ removes the commutation of the right-hand triangle although it retains the commutation of the left-hand triangle and the commutation of the rectangle as a whole. That is the puncture sign $\bigoplus$ removes the equivalent of one equation. A weakness of punctured diagrams is that which diagram is punctured is not always explicitly shown. What about the outer square in Figure 10? Does it need a separate puncture mark? In databases there are examples where the puncture might be used, for instance, with the problem of representing partial functions. If in Figure 11, $STK$ is a library stock, $ISS$ is the category of books issued on loan at the current time, $CAT$ is the catalogue, $ACC$ is the accession numbers, then $t$ is a total function from $CAT$ to $ACC$, $u$ is a total function from $STK$ to $CAT$, $x$ is a total function from $ISS$ to $ACC$ and $z$ is a total function from $STK$ to $ACC$. $y$ is, however, a partial function from $STK$ to $ISS$ as not all books will be out on loan at any one time. If all the functions were total, then $z = tu = xy$ so we have three commuting equations: $z = tu$, $z = xy$ and $tu = xy$. With the partial function $y$, then we lose the commuting equation $z = xy$ but retain $z = tu$. The outer diagram will still commute if $ACC$ is derived independently of $ISS$ so we have lost one commuting equation, hence the one puncture mark.

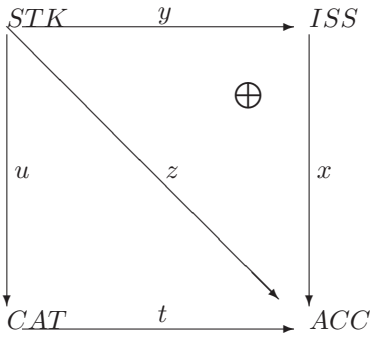


Figure 11: Punctured Commuting Diagram for Library Example
$ACC$ = accessions, $STK$ = stock, $ISS$ = issues, $CAT$ = catalogue

Punctured diagrams represent a type failure in a category: the type of an input is unexpected and an appropriate output cannot be generated. One of the causes shown here of partial functions could be avoided by making all functions total as is usual in category theory. Source objects that are unassigned by the function in its normal operation may then be assigned to the initial object ($\perp$) so that a complete assignment is made of objects in the source category. In categorical terms, type forcing is necessary to avoid punctured diagrams and maintain interoperability.

### 5.2 Lifted Categories and Sketches

There are alternative approaches to composition failure in category theory. Two of these have been the focus of database workers.

(Lellahi & Spyratos 1990), in the FIDE project, attempted to adapt category theory to partial functions by creating a new categorical type of lifted (or partial) functions. This technique has not been further developed, perhaps because of its

inherent complexity and its conflict with much of the established theory of categories.

Sketches have had more advocates. The reductionism of set theoretic methods to represent real-world activities means that corresponding categories may not commute because of the departure from naturality. The purpose of sketches was to identify, as in punctured diagrams, the departures from naturality in the internal components of a diagram like the cones and cocones. For instance (Johnson, Rosebrugh, & Wood 2002) applied sketches to entity-relationship and relational modelling and (Diskin & Cadish 1995) to object databases.

Sketches are strictly outside category theory as they permit diagrams that do not commute but they may be mapped onto categories by a model functor. Many types of sketches have been developed in the theory itself. For instance (Johnstone 2002) defines eight at D2.1.3 **2** p.863-864. They separate out four components of a diagram in order to flag the parts for which composition fails. These four components may be defined as a 4-tuple $< E, L, R, S >$ where $E$ is a finite graph for the data structure, $L$ is a set of diagrams in $E$ giving the constraints as commuting diagrams, $R$ is a finite set of discrete cones in $E$ giving the relationships and $S$ a finite set of discrete cocones in $E$ specifying the attributes. For example the omission of a diagram from $L$ means that it is not required to commute, so this diagram is effectively punctured.

Sketches lack flexibility as all structures and constraints have to be pre-specified. In difficult areas such as interoperability, sketches are inadequate as they do not offer natural closure. (Johnson & Rosebrugh 2000) attempt to adapt their sketches to achieve interoperability but the aim is to achieve only logical independence, as in the three-level architecture of Figure 1, not semantic interoperability, as in the four-level architecture of Figure 2. The difference between a natural structure and a sketch is like that between typing and labelling. A graph is richer than an entity-relationship model as its arrows are typed with identity functors. Labelling in the entity-relationship model is an informal typing whereas the identity arrow is a formal typing.

## 6 Natural Composition

Some problems with partial functions can be avoided by altering the data design so that the partial functions only operate in the assignment to the end of the chain (the terminal object). For instance an alternative design can be considered for Figure 11. Here the natural order would be to consider first accessions, which are then put into the stack and can be issued later. For this schema the composition diagram would be as in Figure 12. These are full categories without composition failure and the puncture sign can be removed. There are no punctured diagrams if $ISS$ is the codomain of each of $x', t'$ and $z'$. This is because these are all partial functions, mapping onto a category which is last in the sequence, the terminal object. There is a type change but it occurs just once, in the final step. It is when partial functions map onto intermediate categories in a chain that typing problems are likely to occur, because of the fluctuations of the types.

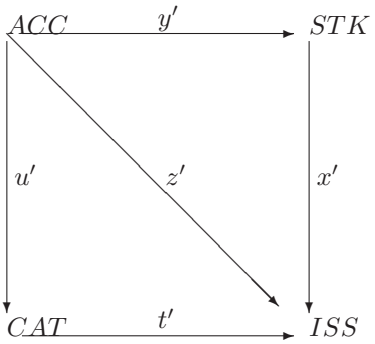


Figure 12: Non-punctured Commuting Diagram for Library Example
$ACC$ = accessions, $STK$ = stock, $ISS$ = issues, $CAT$ = catalogue

## 7 Conclusions

The use of a formal four-level architecture, based on category theory, provides an encouraging framework for tackling both semantic and organisational interoperability. The use of the Godement calculus, in particular, enables many different paths at a number of level to be compared and analysed. A number of problems remain. Failure of composition, particularly due to the existence of partial functions, needs to be identified. Punctured categorical diagrams are used for this purpose in preference to lifted categories or sketches. Semantic annotation remains a challenging area where the open Heyting logic may be of assistance.

## REFERENCES


Barr, M, & Wells, C, *Category Theory for Computing Science*, Prentice-Hall (1990, 1995), Les Publications Centre de Recherches Mathématiques, Montréal (1999).

Date, C J, & Darwen, Hugh, *Foundation for Future Database Systems: The Third Manifesto* 2nd Ed, Addison Wesley (2000).

Diskin, Z, & Cadish, B, Algebraic Graph-Based Approach to Management of Multidatabase Systems, *NGITS'95* 69-79 (1995).

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

Godement, R, *Théorie des faisceaux*, Hermann, Appendix I (1958).

Goguen, Joseph A, & Burstall, Rod M, Some Fundamental Algebraic Tools for the Semantics of Computation. Part 1: Comma Categories, Colimits, Signatures and Theories, *Theor Comp Sci* **31** 175-209 (1984)

Heather, M A, & Rossiter, B N, The Anticipatory and Systemic Adjointness of E-Science Computation on the Grid, *Computing Anticipatory Systems, Proceedings CASYS'01* Liège, Dubois, D M, (ed.), AIP Conference Proceedings **627** 565-574 (2002).

Hendler, J, Berners-Lee, T, & Miller, E, Integrating Applications on the Semantic Web *J Institute Electrical Engineers* Japan **122**(10) 676-680, (2002).

Johnson, M, & Rosebrugh, R, Database Interoperability Through State Based Logical Data Independence, *Proc 4th CSCW2000* IEEE Hong Kong 161-166 (2000).

Johnson, M, Rosebrugh, R, & Wood, R J, Entity-Relationship-Attribute Designs and Sketches, *TAC* **10** 94-111 (2002).

Johnstone, P T, Sketches of an Elephant, A Topos Theory Compendium, *Oxford Logic Guides* 43, Clarendon (2002).

Kelly, G M, & Street, R, Review on the Elements of 2-categories, Proceedings Sydney Category Theory Seminar 1972-73, ed. G M Kelly, *Lecture Notes in Mathematics*, Springer-Verlag **420** 75-103 (1974).

Lellahi, S Kazem, & Spyratos, Nicolas, Towards a Categorial Data Model Supporting Structured Objects and Inheritance, *East/West Database Workshop* 86-105 (1990).

Mac Lane, S, & Moerdijk, I, *Sheaves in Geometry and Logic*, Springer-Verlag (1991).

Rossiter, N, From Classical to Quantum Databases with Applied Pullbacks, *78th Meeting Peripatetic Seminar on Sheaves and Logic* Institut de Recherche Mathématique Avancée, Strasbourg University 15-16 February (2003).

Rossiter, N, & Heather, M, Four-level Architecture for Closure in Interoperability, *EFIS2003, Fifth International Workshop on Engineering Federated Information Systems*, Coventry, UK, 17-18 July 83-88 (2003).

Rossiter, B Nick, & Heather, M A, Data Structures in Natural Computing: Databases as Weak or Strong Anticipatory Systems, *CASYS'03, Sixth International Conference on Computing Anticipatory Systems* Liège, Belgium, AIP Conference Proceedings **718** 392-405 (2004).

Simmons, H, *Lecture Notes on Category Theory*, Logic in IT Initiative, SERC (1989).

Von-Wun Soo, Chen-Yu Lee, Chung-Cheng Li, Shu Lei Chen, Ching-chih Chen, Automated semantic annotation and retrieval based on sharable ontology and case-based learning techniques *Int Conf Digital Libraries Archive*, 61-72 (2003).

Tsichritzis, D, ANSI/X3/SPARC DBMS Framework 1978, Report of the Study Group on Database Management Systems, *Information Systems* **3** (1978).