A Simple Case Study for Schema Integration Tools

AutoMed Technical Report 1, Version 1

Peter M^{c}Brien

Monday 18 th August 2003

1 Introduction

The purpose of this document is to record a simple small example of database schema integration, which contains enough complexity to test all key aspects of the AutoMed achitecture [7] as it is developed. In particular, the example is used to test the representation of modelling languages in the MDR and schemas and transformations in the STR of the AutoMed repositories [2, 3]. The example is a development of that presented in [6], and is presented in Figure 1 as three extensional databases (*i.e.* are actual databases with stored data) s1, s2 and s3 in a variant of the ER modelling langauge.

The version of the ER modelling langauge we shall be using is given in Table 1. It a slight extention of the ER language used in the well-known survey paper on schema integration [1], and similar to the well known **enhanced-ER (EER)** model [4]. The key features of this ER modelling langauge are that

- only binary relationships are permitted,
- relationships may not have an attribute added,
- attributes may be key, notnull or null, and
- generalisations may be total or partial.

The following sections present a number of integration scenarios, which illustrate common techniques used in the BAV approach.

2 Integration of Two Instance Equivalent Schemas

Suppose that it can be determined that the staff entity in s1 always contains the same set of people as the person entity in s2, and that the id key of staff is the same type as the pid attribute of person. It follows the we have a synonym con¤ict between the schemas, and should rename staff to person (or *visa versa*) and rename pid to id (or *visa versa*).

Also suppose that it can be determined that the instances of the dept entity (and hence its dname key attribute) in s1 are always the same set as the values of the dname attribute of person in s2. It follows that we have a structural con¤ict between the schemas, and should transform the dname attribute in s2 into a new entity dept and key attribute dname connected to person by a relationship called worksin. In this particular example, we could alternatively have transformed the worksin relationship, dept entity and dname key attribute

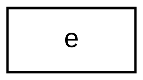
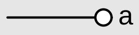
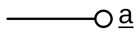
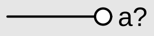
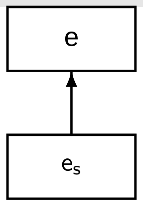
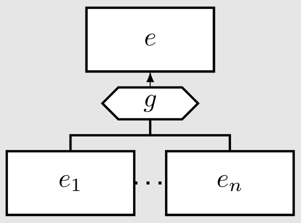
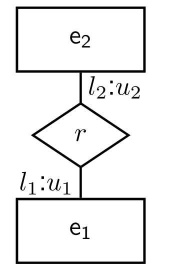
Name	Symbol	Description
entity set		Describes a set of entities in the UoD which share some common properties. We denote the entity by the scheme $\langle\!\langle \mathsf{e} \rangle\!\rangle$ and can enumerate the values of the entity in variable x by the expression $x \; \langle - \; \langle\!\langle \mathsf{e} \rangle\!\rangle$.
attribute		A set of attributes which are of a certain type, and are associated with members of an entity set. We denote the attribute by the scheme $\langle\!\langle \mathsf{e,a} \rangle\!\rangle$. Each instance of the entity must be associated an instance of the attribute, and we can emumerate. We can enumerate the values of the association be entity and attributes by the expression $\{x, y\} \; \langle - \; \langle\!\langle \mathsf{e}, \mathsf{a} \rangle\!\rangle$.
key attribute		An attribute which forms part of the key for the entity.
null attribute		An attribute for which not every instance of $\langle\!\langle \mathsf{e} \rangle\!\rangle$ appears as instance of $\langle\!\langle \mathsf{a} \rangle\!\rangle$.
subset		Denotes that any instance of the **subset entity** $\langle\!\langle \mathsf{e_s} \rangle\!\rangle$ is also a member of the **superset entity** $\langle\!\langle \mathsf{e} \rangle\!\rangle$.
generalisation		Denotes a series of subset entities $\langle\!\langle \mathsf{e_1} \rangle\!\rangle \ldots \langle\!\langle \mathsf{e_n} \rangle\!\rangle$ of superset $\langle\!\langle \mathsf{e} \rangle\!\rangle$, which have the additional property that the instances of $\langle\!\langle \mathsf{e} \rangle\!\rangle$ must be also instances of exactly one of $\langle\!\langle \mathsf{e_1} \rangle\!\rangle \ldots \langle\!\langle \mathsf{e_n} \rangle\!\rangle$. If an asterisk is put by the g, then the above is relaxed such that some instances of e may exist which are not instances of $\langle\!\langle \mathsf{e_1} \rangle\!\rangle \ldots \langle\!\langle \mathsf{e_n} \rangle\!\rangle$.
relationship	 $\mathsf{e_2}$ $l_2{:}u_2$ r $l_1{:}u_1$ $\mathsf{e_1}$	A set of binary relationships between entities, where the relationships are all of a certain type. We denote the relationship by $\langle\!\langle \mathsf{r}, \mathsf{e_1}, \mathsf{e_2} \rangle\!\rangle$. The cardinality constraint $l_1{:}u_1$ gives the minimum and maximum number of instances of e_2 that may be associated with any one instance e_1, and $l_2{:}u_2$ gives the minimum and maximum number of e_1 instances that may be associated to a particular instance of e_2. We can enumerate the values of the association be entity and attributes by the expression $\{x, y\} \; \langle - \; \langle\!\langle \mathsf{r}, \mathsf{e_1}, \mathsf{e_2} \rangle\!\rangle$.

Table 1: Constructs and declarative semantics of an entity-relationship model

into a single dname attribute of staff, but this would not be the case if the dept entity had any other associations (*i.e.* it had any other attributes, relationships, subset involvements or generalisation involvements).

Finally, suppose that it can be determined that the instances of the sex attribute of staff in s1 only takes the string values 'm' and 'f', and that always the set of staff which have the value 'm' is the same as the set of male in s2, and the set of staff which have the value 'f' is the same as the set of female. It follows that we have a structural con¤ict between the schemas, and should transform the sex attribute of staff into a total generalisation hierarchy with two subset entities male and female. In this particular case, we could alternatively have made the inverse transformation on male and female in s2 since they have no other associations.

Given these suppositions, we can use the following transformations to transformation s1 and s2 into a new global schema s4. The queries of the transformations use the AutoMed IQL language [5]. The semantics of ② are that the extent of ⟨⟨male⟩⟩ is a list of unary tuples, taken as the projection of the £rst attribute of the tuples of ⟨⟨person, sex⟩⟩ where the second attribute equates to the value 'm'. The semantics of ⑤ are that the binary tuples for the ⟨⟨person, sex⟩⟩ attribute can be covered by associating the £rst attribute of the tuple an instance of ⟨⟨male⟩⟩ and the second to the string constant 'm', concatenated (by the ++ operator) with associating the £rst attribute to an instance of ⟨⟨female⟩⟩ and the second to 'f'. Note that ⑤ has also a constraint added to it, stating that this transformation is only valid when the extent of ⟨⟨person, sex⟩⟩ is just ['m','f'].

```
     s1 → s4
①    renameEntity(⟨⟨staff⟩⟩, ⟨⟨person⟩⟩)
②    addEntity(⟨⟨male⟩⟩, [{x} | {x, y} ⟨− ⟨⟨person, sex⟩⟩; (=) y 'm'])
③    addEntity(⟨⟨female⟩⟩, [{x} | {x, y} ⟨− ⟨⟨person, sex⟩⟩; (=) y 'f'])
④    addGeneralisation(sex, total, person, male, female)
⑤    delAttribute(⟨⟨person, sex⟩⟩,
          [{x, y} | {x} ⟨− ⟨⟨male⟩⟩; (=) y 'm'] ++ [{x, y} | {x} ⟨− ⟨⟨female⟩⟩; (=) y 'f'])
          (['m', 'f'] = [{x} | {x, y} ⟨− ⟨⟨person, sex⟩⟩])

     s2 → s4
⑥    addEntity(⟨⟨dept⟩⟩, [{y} | {x, y} ⟨− ⟨⟨person, dname⟩⟩])
⑦    addAttribute(⟨⟨dept, dname, key⟩⟩, [{y, y} | {x, y} ⟨− ⟨⟨person, dname⟩⟩])
⑧    addRelationship(⟨⟨worksin, person, dept, 1:1, 1:N⟩⟩, [{x, y} | {x, y} ⟨− ⟨⟨person, dname⟩⟩])
⑨    delAttribute(⟨⟨person, dname, notnull⟩⟩, [{x, y} | {x, y} ⟨− ⟨⟨worksin, person, dept⟩⟩])
⑩    renameAttribute(⟨⟨person, pid, key⟩⟩, ⟨⟨person, id, key⟩⟩)
```

3 Integration of Overlapping Schemas

Suppose it can be determined that all schema components that appear in both s4 and s3 always share the same sets of values, *i.e.* the entity person in s4 has the same set of values as person in s3, the attribute id in s4 has the same set of values as id in s3, *etc*.

Also suppose that it can be determined that any construct that does not appear in one schema cannot be derived from the remaining components in that schema, *i.e.* the extention of the missing name attribute of person in s3 cannot be derived from any other components in s3, and the extention of the missing degree entity and associted constructs in s4 cannot be derived from any other constructs in s4.

Given these suppositions, we can the use extend transformations to build a new global schema s5 that contains all information available from the three extentional schemas s1, s2 and s3.

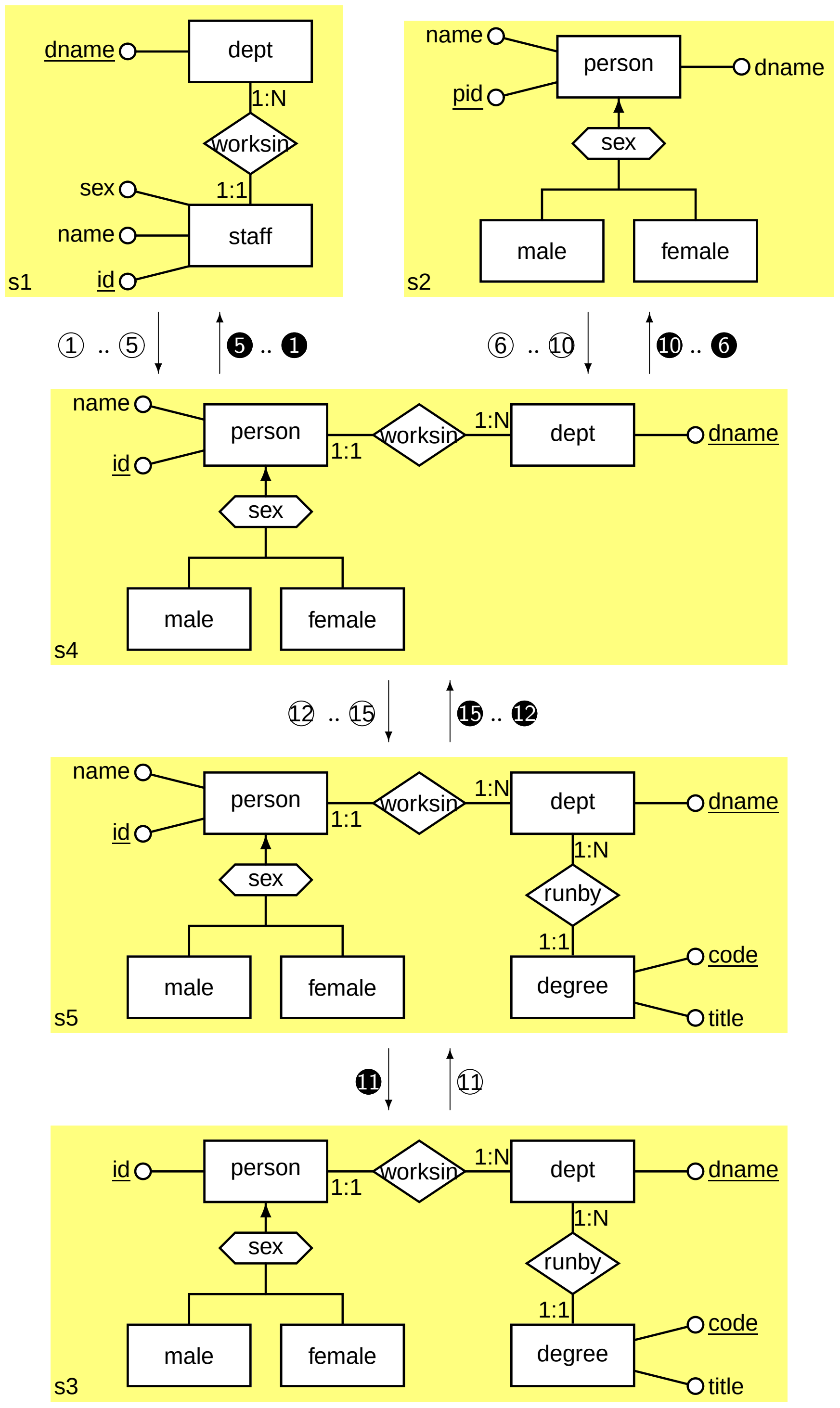


Figure 1: Example ER schemas, and the transformations that relate them

s3 → s5
⑪ extendAttribute(⟨⟨person, name, notnull⟩⟩)

s4 → s5
⑫ extendEntity(⟨⟨degree⟩⟩)
⑬ extendAttribute(⟨⟨degree, code, key⟩⟩)
⑭ extendAttribute(⟨⟨degree, title, notnull⟩⟩)
⑮ extendRelationship(⟨⟨runby, degree, dept⟩⟩)

Note that these transformations may be composed to form a direct transformation from s3 to s4 as follows (where white on black text for the transformation number indicates an inverse of the black on white transformation):

s3 → s4
⑪ extendAttribute(⟨⟨person, name, notnull⟩⟩)
⓯ contractRelationship(⟨⟨runby, degree, dept⟩⟩)
⓮ contractAttribute(⟨⟨degree, title, notnull⟩⟩)
⓭ contractAttribute(⟨⟨degree, code, key⟩⟩)
⓬ contractEntity(⟨⟨degree⟩⟩)

4 Relational Models and Sample Data

Figure 2 illustrates three relational schemas equivalent to the three EDB schemas of Figure 1, on the basis that the ER modelling language being used identi£es instances of entities by using the key attribute of the entity.

Given this relational model and data we can derive instances of the ER model. For example, s2 has the following extent:

⟨⟨person⟩⟩ $= [\{1\}, \{2\}, \{3\}, \{4\}, \{5\}]$
⟨⟨person, pid⟩⟩ $= [\{1,1\}, \{2,2\}, \{3,3\}, \{4,4\}, \{5,5\}]$
⟨⟨person, name⟩⟩ $= [\{1,$ ‘Alex’$\}, \{2,$ ‘Dimitri’$\}, \{3,$ ‘Mike’$\}, \{4,$ ‘Nerissa’$\}, \{5,$ ‘Peter’$\}]$
⟨⟨person, dept⟩⟩ $=$
$[\{1,$ ‘CS-BBK’$\}, \{2,$ ‘CS-BBK’$\}, \{3,$ ‘Comp-IC’$\}, \{4,$ ‘Comp-IC’$\}, \{5,$ ‘Comp-IC’$\}]$
⟨⟨male⟩⟩ $= [\{2\}, \{3\}, \{5\}]$
⟨⟨female⟩⟩ $= [\{1\}, \{4\}]$

References

[1] C. Batini, M. Lenzerini, and S. Navathe. A comparative analysis of methodologies for database schema integration. *ACM Computing Surveys*, 18(4):323–364, 1986.

[2] M. Boyd, P.J. McBrien, and N. Tong. The automed schema integration repository. In *Proceedings of BNCOD02*, volume 2405 of *LNCS*, pages 42–45. Springer-Verlag, 2002.

[3] M. Boyd, P.J. McBrien, and N. Tong. The AutoMed repositories and API. Technical Report AutoMed TR Number 3, Version 3, Dept. of Computing, Imperial College, 2003.

[4] R. Elmasri and S. Navathe. *Fundamentals of Database Systems*. Addison-Wesley, 3rd edition, 2000.

[5] E. Jasper, A. Poulovassilis, and L. Zamboulis. Processing IQL Queries and Migrating Data in the AutoMed toolkit. Technical report, AutoMed TR Number 20, 2003. Available from`http://www.doc.ic.ac.uk/automed/`.

staff			
id	name	sex	workins
1	Alex	F	CS-BBK
2	Dimitri	M	CS-BBK
3	Mike	M	Comp-IC
4	Nerissa	F	Comp-IC
5	Peter	M	Comp-IC

dept
dname
CS-BBK
Comp-IC

(a) s1

person		
pid	name	dname
1	Alex	CS-BBK
2	Dimitri	CS-BBK
3	Mike	Comp-IC
4	Nerissa	Comp-IC
5	Peter	Comp-IC

male
id
2
3
5

female
id
1
4

(b) s2

person	
id	worksin
1	CS-BBK
2	CS-BBK
3	Comp-IC
4	Comp-IC
5	Comp-IC

male
id
2
3
5

female
id
1
4

dept
dname
CS-BBK
Comp-IC

degree		
dcode	title	runby
1234	MSc Computing	CS-BBK
G500	BEng Computing	Comp-IC
G500	MEng Computing	Comp-IC

(c) s3

Figure 2: Relational models and data

[6] P.J. McBrien and A. Poulovassilis. Automatic migration and wrapping of database applications — a schema transformation approach. In *Proceedings of ER99*, volume 1728 of *LNCS*, pages 96–113. Springer-Verlag, 1999.

[7] P.J. McBrien and A. Poulovassilis. Automatic generation of mediator tools for heterogeneous database integration [automed]. Technical report, Birkbeck College and Imperial College, March 2000.