DRAL

Daresbury Laboratory
Rutherford Appleton Laboratory

Rutherford Appleton Laboratory
Informatics Department

Data and Hypermodels are Isomorphic: Manipulating Hyperdocuments at a Logical Level

S.A. Dobson
24 August, 1994
WWW/02/94

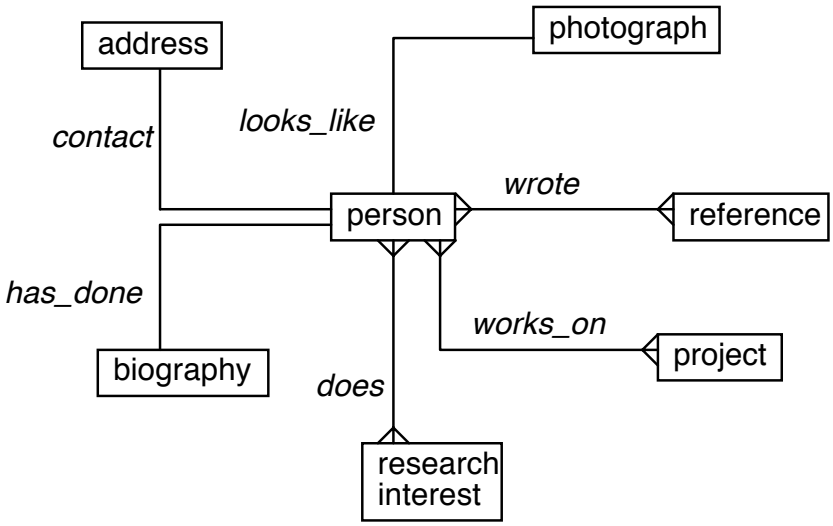
1. Introduction

This paper is a summary of a discussion between the author, Toria Burrill and Keith Jeffrey. We attempted to reconcile two views of the Informatics Department World Wide Web (WWW) which have previously been seen as in conflict: the logical, database view of the data held, and the presentation of that data. We have devised a way to unify these two views of the data in such a way that we may interact with data stored in its presentational (HTML[2]) form using database operations. In effect this formalises the structure of the web, opening up a host of new possibilities.

2. Data Models and Hypermodels

For simplicity (and without loss of generality) we consider the way in which people are described within the model.

A data model for people within the department, expressed as an entity-relationship diagramm, is as follows:



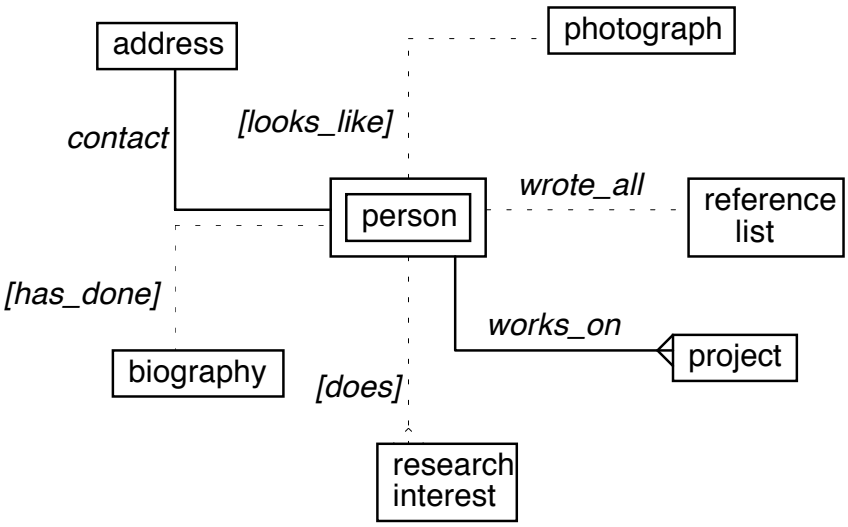
so a person works on many projects, has written many referenced works *et cetera*. (Each entity will have attributes, such as their e-mail address as part of their contact details,

which we have omitted for clarity in the diagram.) What is wanted is a way to map this data model into a hypermodel, where the relationships shown become links between objects in the hyperspace.

Creating a Well-founded Hypermodel

The simplest solution is to map each entity into a hyperobject and each relationship into a link. So a person entity becomes a person's page in the web, with links to their contact address, research interests and so forth. The problem here is that a person's page then becomes a mass of links with no content itself, which makes the page look uninteresting even though all the information is accessible from it. Furthermore it is unlikely that one would want someone's photograph down a link: it should be on the top of the page. These relationships (called *expand-in-place* in hypermedia) are not captured in the data model as they are not semantically important: they are vital for effective presentation, however.

What we need is a more flexible mapping between models. For example, the following is a "better" hypermodel (in presentational terms) for a variety of reasons:



This model applies to a single peron, so the many-to-many links at the person end have disappeared. The differences between this and the data model are quite subtle. A single object (with a double-box) is denoted as primary. Some links (shown dotted) are optional, in that a person need not supply a list of research interests; some links are expanded in-place (shown in square brackets), so that the object is included into the primary object rather than being independent.

Equally importantly a new entity – the reference list – has been created instead of a collection of individual references. It is clear that, for any person, a reference list may be created as a simple query over the set of references. This the *wrote_all* relationship is derived directly from the data model. Moreover, this derived relationship is invertible: one can re-acquire the references from the reference list (or the collection of all reference lists).

From this we can conclude the following: if care is taken in the design of hypermodels, *data model and hypermodel are isomorphic*. This means that the two views are essentially the same: one may convert a database view of the data into a presentational view or *vice versa* without loss of information. Furthermore one may reason about the data using the usual database formalisms and tool, and guarantee that any results are translated into, and maintained by, a presentation of that data.

Implementation in HTML

Can the isomorphism from data model to hypermodel be implemented using the current WWW? Yes, it can.

The essential problem is one of object identity. We wish to ensure that the entities in the data model are preserved in the presentation in a form in which they can be easily identified and extracted. An entity from an entity class in the data model should correspond to an object in the hypermedia world.

All objects in WWW are identified uniquely by Uniform Resource Locators (URLs)[3]. An object may contain zero or more anchors, which link that object to other objects iedntified by URLs. For our purposes we may distinguish two kinds of anchor:

- *HREF* anchors, which link an object to another object stored separately to it, *i.e.* in another file; and
- *NAME* anchors, where the data for the linked object is held in-line within the object.

For the first kind, a link is established by inserting an HTML anchor into the text with an HREF tag:

```
<A HREF="/people/sd.html">Address</A>
```

which links an object to an object */people/sd.html* (a person's contact address in this case). The link is marked in the source object by the word “Address”, which denotes the contact address object. Clicking this denotation will jump to the target object.

The second kind uses a similar HTML form:

```
<A NAME="sd_bio.html">
Simon Dobson received......
</A>
```

where the text of the biography object is saved in the anchor. The anchor is not a link *per se*, but still identifies a target object (the in-lined data denotes itself). Furthermore NAME anchor is itself a valid URL. If the parent object has URL *http://web/people/sd/sd_int.html* then the biography anchor may be identified by the URL *http://web/people/sd/sd_int.html#sd_bio.html*.

One may conclude that all entities in the data model are indeed represented by URL-named objects in the hypermodel: *the two kinds of link are semantically identical*, with the distinction being the location of the data and the manner of presentation.

It remains to include information about the relationships under which objects are linked together. HTML includes an optional REL tag which adds a type to the link. If we use the relationship names from the hypermodel above (which were derived from the relationships in the data model), we may re-phrase the two anchors above as

```
<A HREF="/people/sd.html" REL=contact >Address</A>
```

and

```
<A NAME="sd_bio.html" REL=has_done>
Simon Dobson received......
</A>
```

respectively, adding the type of the relationship to the anchor. Currently the HTML specification excludes the use of REL tags without HREF tags, but this is probably a mistake in the specification of HTML and does not interfere with any browsers[1]. We will contribute this impression to the HTML standardisation process.

3. Manipulating Hyperdocuments

We are now in a poaition of having a created a hypermodel which is well-founded on a data model. This means that we may apply any processing techniques applicale to databases to our hyperdocuments.

Printed Hand-outs

Of particular interest at present is converting the hypermedia representation of a project into a printed project hand-out. This process is easily specified using our new representation. We will illustrate the procedure with reference to people, generating a person's biography and contact information, as this is consistent with our previous examples.

For this (illustrative) hand-out, we wish to include the following information:

1. name of a person;
2. their biographic history;
3. their current research interests; and
4. a list of their references.

We do not include their photograph or project involvements.

---

[1]There is another slight wrinkle in that one is supposed to register all relationship names with the HTML Registration Authority which (as far as we are aware) has not yet convened. We may recover correctness in this by adopting the standard Internet convention of prefixing non-standard items with "x-" (for eXperimental), and by using commonplace terminology. Work at Stanford on knowledge sharing and ontologies may help in deciding the terminology to use.

Generating a hand-out involves extracting the necessary entities from the data model (*i.e.* objects from the hypermodel) and composing them into the required form. The procedure is simply to specify those relationships in the hypermodel which we wish to traverse, and how.

For example, for a person's biography we specify that, on encountering a link of type has_done in the source object, we include the data for that relationship in the generated object. In this case the data is immediately accessible (through a NAME anchor). For the contact address, the information required is accessible in an external object which must be acquired and parsed.

For a person's photograph, we specify that anchors or type photo are completely removed from the generated document. Similarly for works_on links.

We essentially define the generation of new objects using a simple "logic of links". If we restrict the terms of this logic to relationships then we immediately remove all the detailed formatting information contained on the source objects: prettiness falls away, and is replaced by prettiness defined expressly for the generaed object. This allows a format suitable for printing to be generated mechanically from a format suitable for browsing.

There are other possibilities, including generating index objects and different views onto the same data (project lists in alphabetical order or by area, for example).

Exactly how to implement transformation systems of this sort is an open question. Certainly it might be done in C++ with a suitable HTML parser and logic; alternatively it may be possible to encode the process into full SGML.

Additional Benefits and Research Directions

The formal hypermodel constitutes a brief description of the intended form of the web. It is derived from a data model. We may therefore check to ensure that the web is a full and faithful representation of the hypermodel (and data model) fairly easily. We may also be able to check that certain common rules of hypertext (such as length of link chains and circularity) are observed.

Since the model is a concise representation of the web, it might be used for navigation. A browser might acquire the model from a server on entry and display it to the user together with a "you are here" marker. As the user traverses the web his marker moves through the model, giving a navigational cue as the his position in the web.

There are also close relationships between hypermodels and the generalised mark-up of documents discussed in [1], since a semantic description is very close to a data model. These are areas of possible future research.

4. References

[1] Simon Dobson, Victoria Burrill and Julian Gallop, "*Semantic mark-up of generalised documents,*" WWW/01/94, Informatics Department, Rutherford Appleton Laboratory (1994).

[2] Tim Berners-Lee and Daniel Connolly, "*Hypertext markup language: a representation of textual information and metainformation for retrieval and interchange*," Draft RFC, (1993).

[3] Tim Berners-Lee, "*Uniform resource locators: a uniform syntax for the expression of names and address of objects on the network*," Draft RFC, (1993).